A multiscale multilevel Monte Carlo method for multiscale elliptic PDEs with random coefficients

Junlong Lyu[a], Zhiwen Zhang[a,*]

[a]*Department of Mathematics, The University of Hong Kong, Pokfulam Road, Hong Kong SAR.*

Abstract

We propose a multiscale multilevel Monte Carlo (MsMLMC) method to solve multiscale elliptic PDEs with random coefficients in the multi-query setting. Our method consists of offline and online stages. In the offline stage, we construct a small number of reduced basis functions within each coarse grid block, which can then be used to approximate the multiscale finite element basis functions. In the online stage, we can obtain the multiscale finite element basis very efficiently on a coarse grid by using the pre-computed multiscale basis. The MsMLMC method can be applied to multiscale RPDE starting with a relatively coarse grid, without requiring the coarsest grid to resolve the smallest-scale of the solution. We have performed complexity analysis and shown that the MsMLMC offers considerable savings in solving multiscale elliptic PDEs with random coefficients. Moreover, we provide convergence analysis of the proposed method. Numerical results are presented to demonstrate the accuracy and efficiency of the proposed method for several multiscale stochastic problems without scale separation.

AMS subject classification: 35J15, 65C05, 65N12, 65N30, 65Y20.

Keywords: Random partial differential equations (RPDEs); uncertainty quantification (UQ); multiscale finite element method (MsFEM); multilevel Monte Carlo (MLMC); reduced basis; convergence analysis.

1. Introduction

Many physical and engineering applications involving uncertainty quantification (UQ) can be described by stochastic partial differential equations (SPDEs, i.e., PDEs driven by Brownian motion) or partial differential equations with random coefficients (RPDEs). In recent years, there has been an increased interest in the simulation of systems with uncertainties, and several numerical methods have been developed in the literature to solve SPDEs and RPDEs; see e.g. [15, 25, 38, 2, 27, 21, 35, 37, 28, 31, 33, 14]. These methods can be effective when the dimension of stochastic input variables is low. However, their performance deteriorates dramatically when the dimension of stochastic input variables is high because of the curse of dimensionality.

*Corresponding author

Email addresses: u3005480@connect.hku.hk (Junlong Lyu), zhangzw@hku.hk (Zhiwen Zhang)

There are some attempts in developing sparsity or data-driven basis to attack these challenging problems. Most of them take advantage of the fact that even though the stochastic input has high dimension, the solution actually lives in a relatively low dimensional space. Therefore, one can develop certain sparsity or data-driven basis functions to solve the SPDEs and RPDEs efficiently. In [8, 9, 7, 40, 39, 20], Hou et al. explored the Karhunen-Loève expansion of the stochastic solution, and constructed problem-dependent stochastic basis functions to solve these SPDEs and RPDEs. In [11, 26], the compressive sensing technique is employed to identify a sparse representation of the solution in the stochastic direction. In [5, 6], Schwab et al. studied the sparse tensor discretization of elliptic RPDEs.

In this paper, we consider another challenge in UQ, i.e., solving multiscale elliptic PDEs with random coefficients. Due to the large range of scales in these solutions, it requires tremendous computational resources to resolve the small scales of the solution. We propose a multiscale multilevel Monte Carlo method (MsMLMC) to significantly reduce the computational cost in solving multiscale elliptic PDEs with random coefficients. We use the following elliptic equation with multiscale random coefficient as an example to illustrate the main idea of our approach:

$$-\nabla \cdot (a^\varepsilon(x,\omega)\nabla u^\varepsilon(x,\omega)) = f(x), \quad x \in D, \omega \in \Omega, \tag{1}$$

$$u^\varepsilon(x,\omega) = 0, \quad x \in \partial D, \tag{2}$$

where $D \in R^d$ is a bounded spatial domain, Ω is a sample space, and $f(x) \in L^2(D)$. The multiscale information is described by the multiscale coefficient $a^\varepsilon(x,\omega)$. The precise definition of the $a^\varepsilon(x,\omega)$ will be given in Section 3.1.

Our MsMLMC method consists of two steps. In the first step, we apply the non-intrusive method (Monte Carlo or stochastic collocation method) to discretize the random coefficient space and obtain a set of multiscale finite element basis realizations. The multiscale finite element basis method has the advantage of requiring less memory consumption which is essential for multiscale problems. Then, we extract a set of multiscale reduced basis functions from these realizations using the proper orthogonal decomposition (POD) method. In the second step of our method, we apply the MLMC to solve multiscale RPDE with the multiscale reduced basis functions. We can regard the first step as an offline computation. Once we obtain the multiscale reduced basis functions, we can apply the MLMC to the multiscale RPDEs with different force functions in the online step, which is very efficient. We also study the complexity of the proposed method. We find that the MsMLMC can offer considerable saving over the original MLMC. The level of saving is problem dependent. The saving is more significant when we work on high space dimension and when the ratio between the largest and the smallest scales is large; see Fig.12.

The rest of the paper is organized as follows. In Section 2, we give a brief introduction of the MLMC method, and present a theorem that estimates the cost of the algorithm under certain problem-dependent assumptions. In Sections 3, we present our derivations of MsMLMC method. Some numerical implementation issues are also discussed. In Section 4, we prove that the asymptotic variance reduction requirement is satisfied in the MsMLMC under certain mild assumptions. In Section 5, we present several numerical results to demonstrate the

accuracy and effectiveness of our method. We also study the computational complexity of the MsMLMC. Finally, some concluding remarks are given in Section 6.

2. Multilevel Monte Carlo method

The MLMC was first introduced by Giles to solve SDEs arising from mathematical finance [16], which was inspired by multigrid iterative solution methods. This work is related to the earlier work of Heinrich for solving finite-dimensional parametric integration and integral equations [18]. Later on, the MLMC method was extended to solve elliptic PDEs with random coefficients; see e.g. [3, 10, 1, 12]. To make this paper self-contained, we give a brief review of the basic ideas and the main result of the MLMC method.

In many applications, the quantity of interest is the expected value of a functional of the solution $u^\varepsilon(x,\omega)$ of RPDE (1), which can be the mean or a high-order moment. Let $v^\varepsilon \in L^2(\Omega, H^1(D))$ denote this functional. In general, the expected value of $v^\varepsilon(x,\omega)$ can be obtained by the standard Monte Carlo (MC) method. The standard MC estimator for the mean, $\mathbb{E}[v^\varepsilon(x,\omega)]$, is given as follows:

$$Y_h = \frac{1}{N}\sum_{i=1}^{N} v_h^\varepsilon(x,\omega_i), \tag{3}$$

where ω_i is the i-th sample, h is the mesh size, v_h^ε is a numerical approximation of $v^\varepsilon(x,\omega)$, and N is the total number of MC samples. We define the mean square error (MSE) of Y as $e(Y_h) = \big|\big|Y_h - \mathbb{E}[v^\varepsilon(x,\omega)]\big|\big|^2_{L^2(\Omega,H^1(D))}$. Simple calculations show that $e(Y_h)$ satisfies

$$\begin{aligned} e(Y_h) &= \mathbb{E}\left[\left|\left|Y_h(x,\omega) - \mathbb{E}[v^\varepsilon(x,\omega)]\right|\right|^2_{H^1(D)}\right] \\ &= \left|\left|\mathbb{E}[v_h^\varepsilon(x,\omega)] - \mathbb{E}[v^\varepsilon(x,\omega)]\right|\right|^2_{H^1(D)} + \frac{1}{N} Var\left[\left|\left|v_h^\varepsilon(x,\omega)\right|\right|_{H^1(D)}\right]. \end{aligned} \tag{4}$$

The first term in Eq.(4) gives the error introduced by the numerical discretization in physical space, while the second term represents the sampling errors, which decays inversely proportional to the number of samples. In this paper, we choose the mesh size h fine enough so that the error from space discretization is negligible. Thus, the convergence rate of the root mean square error (RMSE) of the MC method is merely $O(\frac{1}{\sqrt{N}})$, which means that many such realizations are required to obtain accurate results. For the multiscale RPDE (1), it is very time-consuming to obtain each realization. As a result, it is very expensive to solve (1) by using the MC method.

The MC method can be accelerated by different variation reduction techniques, such as the importance sampling method and the quasi Monte Carlo method. The MLMC method [16] is a novel variance reduction technique for the standard MC method. It exploits the linearity of expectation by expressing the quantity of interest on the finest spatial grid in terms of the same quantity on a relatively coarse grid and a set of correction terms.

First, the domain D is partitioned into a number of nested mesh grids, i.e., $D_0 \subset \ldots \subset D_{j-1} \subset D_j \ldots \subset D_L$. Here the mesh size of the l-th level is $h_l = h_0 2^{-l}$, $l = 0, 1, \ldots, L$, and h_0 and $h_L = h_0 2^{-L}$ are the coarsest and finest level mesh size, respectively. The main idea of MLMC method can be described easily. Linearity of the expectation operator implies that

$$\mathbb{E}[v_L^\varepsilon(x,\omega)] = \mathbb{E}[v_0^\varepsilon(x,\omega)] + \sum_{l=1}^{L} \mathbb{E}[v_l^\varepsilon(x,\omega) - v_{l-1}^\varepsilon(x,\omega)], \tag{5}$$

where $v_l^\varepsilon(x,\omega)$ denotes the numerical approximation of the functional of the solution $u^\varepsilon(x,\omega)$ obtained on the mesh grids D_l.

The key point is to avoid estimating $\mathbb{E}[v_L^\varepsilon(x,\omega)]$ on the finest level, but instead to estimate the expectation on the coarsest level, plus a sum of corrections adding the difference in expectation between simulations on consecutive levels. The multilevel idea is to independently estimate each of these expectations such that the overall variance is minimized for a fixed computational cost. We rewrite the estimate of the expected value of $v^\varepsilon(x,\omega)$ as

$$Y = \sum_{l=0}^{L} Y_l, \tag{6}$$

where

$$\begin{aligned} Y_l &= \frac{1}{N_l} \sum_{i=1}^{N_l} [v_l^\varepsilon(x,\omega_i^{(l)}) - v_{l-1}^\varepsilon(x,\omega_i^{(l)})], \quad l = 1, \ldots, L; \\ Y_0 &= \frac{1}{N_0} \sum_{i=1}^{N_0} v_0(x,\omega_i^{(0)}). \end{aligned}$$

Here $L+1$ is the number of levels used in the MLMC, and N_l is the number of MC simulations at the l-th level. Y_0 is the coarsest level estimate, while Y_l ($l = 1, \ldots, L$) measures the fluctuations of l-th and $(l-1)$-th level. It is important to note that we have used the same random sample $\omega_i^{(l)}$ to estimate both v_l and v_{l-1} in the quantity Y_l.

Simple calculations show that the $e(Y)$ satisfies the following equation:

$$\begin{aligned} e(Y) &= \mathbb{E}\left[\left|\left| \sum_{k=0}^{L} Y_k(x,\omega) - \mathbb{E}[v^\varepsilon(x,\omega)] \right|\right|_{H^1(D)}^2 \right] \\ &= \left|\left| \mathbb{E}[v_L^\varepsilon(x,\omega)] - \mathbb{E}[v^\varepsilon(x,\omega)] \right|\right|_{H^1(D)}^2 \\ &\quad + \sum_{l=1}^{L} \frac{1}{N_l} Var\left[\left|\left| v_l^\varepsilon(x,\omega) - v_{l-1}^\varepsilon(x,\omega) \right|\right|_{H^1(D)} \right] + \frac{1}{N_0} Var\left[\left|\left| v_0^\varepsilon(x,\omega) \right|\right|_{H^1(D)} \right]. \end{aligned} \tag{7}$$

The first term in Eq.(7) gives the error introduced by the numerical discretization at the finest level L, while the second and third terms represent the sampling errors and decay inversely proportional to the number of samples. Recall that h_L is the finest mesh used in solving the

multiscale RPDE (1). Again, we assume the mesh size h_L fine enough so that the error from space discretization is negligible.

The computational cost of the MLMC estimator is

$$C = \sum_{l=0}^{L} N_l C_l, \tag{8}$$

where C_l represents the cost of a single sample of $v_l^\varepsilon(x,\omega) - v_{l-1}^\varepsilon(x,\omega)$. Let us treat the MC sample number N_l as continuous variables. Then the variance of the MLMC estimator is minimized for a fixed computational cost by choosing the sample number N_l be proportional to $\sqrt{Var(v_l^\varepsilon - v_{l-1}^\varepsilon)/C_l}$, and the constant of proportionality is chosen to achieve a certain MSE error; see [16].

We claim that to achieve an MSE of $O(\delta^2)$, the MLMC method is cheaper than the MC method. This analysis is made more precise in the following theorem. Its proof can be found in Appendix A.

Theorem 2.1. *Let Y be the MLMC estimator of the mean function $\mathbb{E}[v^\varepsilon(x,\omega)]$ and $h_l = h_0 2^{-l}$ be the mesh size of l-th level, with $l = 0, 1, \ldots, L$. Let $v_{-1}^\varepsilon = 0$. We assume the following estimate:*

1. $\int \mathrm{Var}\left(v_0^\varepsilon\right) dx = c_1$;
2. $\int \mathrm{Var}\left(v_l^\varepsilon - v_{l-1}^\varepsilon\right) dx = c_2 h_l^\beta$;
3. *the computational complexity of the l-th level MC simulation is $c_3 h_l^{-\gamma}$.*

Then the MLMC estimator Y has the δ^2 MSE with a computational cost $\mathcal{C}_2$

$$\mathcal{C}_2 \approx \begin{cases} c_5\delta^{-2} & = \tilde{c}_5 h_L^\gamma \mathcal{C}_1, \quad \text{if} \quad \beta > \gamma; \\ c_6\delta^{-2}|\ln h_L|^2 & = \tilde{c}_6 h_L^\gamma |\ln h_L|^2 \mathcal{C}_1, \quad \text{if} \quad \beta = \gamma; \\ c_7\delta^{-2} h_L^{-(\gamma-\beta)} & = \tilde{c}_7 h_L^\beta \mathcal{C}_1, \quad \text{if} \quad 0 \le \beta < \gamma. \end{cases}$$

Here c_i's are constants that do not depend on h, ε, and δ, and $\mathcal{C}_1$ is the cost of the MC method achieving the same δ^2 MSE at level h_L, which is,

$$\mathcal{C}_1 = c_3 N h_L^{-\gamma} = c_1 c_3 h_L^{-\gamma} \delta^{-2}.$$

The reduction in the computational cost associated with the MLMC method over the MC method is due to the fact that most of the uncertainty can be captured on the coarse grids and so the number of realizations needed on the finest grid is greatly reduced. However, we have the observation that $\int \mathrm{Var}\left(v_l^\varepsilon - v_{l-1}^\varepsilon\right) dx = c_2 h_l^\beta$ is achieved only if $h_{l-1} < \varepsilon$; see Figure 5. In other words, for multiscale problems the MLMC method becomes effective if the coarsest level mesh size h_0 is smaller than ε. Therefore, it could still be quite expensive to solve RPDE (1) using the MLMC method, when the multiscale parameter ε is small. To alleviate the difficulty of requiring $h_0 < \varepsilon$, we propose a MsMLMC method to effectively solve RPDEs with multiscale features.

Finally, we remark that the idea of MLMC can be combined with other well developed methods, such as the stochastic collocation method, to further reduced the computational cost in solving RPDEs. We refer the reader to the papers [17, 34] for details.

3. Multiscale multilevel Monte Carlo method

3.1. Multiscale finite element basis functions

We consider the following multiscale elliptic PDE with a random coefficient:

$$-\nabla \cdot (a^{\varepsilon}(x,\omega)\nabla u^{\varepsilon}(x,\omega)) = f(x), \quad x \in D, \omega \in \Omega, \tag{9}$$

$$u^{\varepsilon}(x,\omega) = 0, \quad x \in \partial D, \tag{10}$$

where $D = [0,1] \times [0,1]$ is a bounded spatial domain, Ω is a sample space, and $f(x) \in L^2(D)$. To simplify our notations, we assume $a^{\varepsilon}(x,\omega)$ is a scalar function. The same approach can be applied for general cases where $a^{\varepsilon}(x,\omega)$ is a multiscale coefficient matrix. Besides, we assume $a(x,w)$ satisfies uniform ellipticity almost surely, i.e. there exist $a_{\min}, a_{\max} > 0$, such that

$$\mathbb{P}\big(\omega \in \Omega : a^{\varepsilon}(x,\omega) \in [a_{\min}, a_{\max}], \forall x \in D\big) = 1. \tag{11}$$

Under the assumption (11), we can easily verify that

$$||u^{\varepsilon}(x,\omega)||_{H_0^1(D)} \leq C_D \frac{||f||_{L^2(D)}}{a_{\min}}, \quad \text{a.s.}, \tag{12}$$

where C_D is the constant given by the Poincaré inequality. Moreover, we have

$$||u^{\varepsilon}(x,\omega)||_{L^q(\Omega, H_0^1(D))} \leq C_D \frac{||f||_{L^2(D)}}{a_{\min}}, \quad \forall q \geq 1. \tag{13}$$

In the MsMLMC method, we partition the domain D into a number of nested mesh grids, i.e., $D_0 \subset \ldots \subset D_{j-1} \subset D_j \ldots \subset D_L$. However, the coarsest mesh size h_0 does not necessarily well resolve the multiscale feature of the Eq.(9). In fact, we choose $h_0 \gg \varepsilon$. This is achieved with the help of the upscaling through the multiscale finite element method (MsFEM) [22].

Specifically, we assume that the j-th level grid D_j is partitioned into a finite set of compact triangles or quadrilaterals $\{D_j^k, 1 \leq k \leq K\}$. The union of all triangles or quadrilaterals covers the closure of D, and the intersection of different triangles or quadrilaterals is either empty, a common node, or a common edge.

Within each block $D_j^k, 1 \leq k \leq K$, we solve the following cell problems to obtain the multiscale finite element basis $\phi^{kl}(x,\omega)$,

$$-\nabla \cdot (a^{\varepsilon}(x,\omega)\nabla \phi^{kl}(x,\omega)) = 0, \quad x \in D_j^k, \quad \omega \in \Omega, \tag{14}$$

$$\phi^{kl}(x,\omega) = p^{kl}(x), \quad x \in \partial D_j^k, \quad l = 1, ..., d, \tag{15}$$

where d is the number of nodes on D_j^k, and $p^{kl}(x)$ is the Dirichlet boundary condition defined on the boundary of D_j^k. In practice, one can choose $p^{kl}(x)$ to be the standard bilinear or linear basis functions.

Let $\phi^{kl}(x,\omega) \in H^1(D_j^k)$ be the solution of Eq.(14) satisfying the boundary condition in Eq.(15). It can be decomposed as

$$\phi^{kl}(x,\omega) = \phi_0^{kl}(x,\omega) + p^{kl}(x). \tag{16}$$

The unknown homogeneous part $\phi_0^{kl}(x,\omega) \in H_0^1(D_j^k)$ is the solution of the following variational formulation

$$a^\varepsilon(\phi_0^{kl}(x,\omega), v; \omega) = f^\varepsilon(v;\omega), \quad \forall v \in H_0^1(D_j^k), \tag{17}$$

where

$$a^\varepsilon(w, v; \omega) = \int_{D_j^k} a^\varepsilon(x,\omega)\nabla w \cdot \nabla v dx, \tag{18}$$

$$f^\varepsilon(v;\omega) = -\int_{D_j^k} a^\varepsilon(x,\omega)\nabla p^{kl}(x) \cdot \nabla v dx. \tag{19}$$

When the boundary conditions of the basis functions are linear, we can obtain the following convergence results.

Proposition 3.1. *Assume $a^\varepsilon(x,\omega) = a(\frac{x}{\varepsilon},\omega)$ and satisfy assumption (11). Let $u^\varepsilon(x,\omega)$ be the solution of (9) and $u_{h_j}^\varepsilon(x,\omega)$ be the numerical approximation obtained from the space spanned by the multiscale basis with linear boundary conditions; see Eqns.(14)-(15). Then, we have*

$$||u^\varepsilon - u_{h_j}^\varepsilon||_{H^1(D)} \leq C(h_j + \varepsilon)||f||_{L^2(D)} + C\sqrt{\frac{\varepsilon}{h_j}}, \quad a.s. \quad \omega \in \Omega, \tag{20}$$

where $h_j \gg \varepsilon$ is the mesh size of the j-th level mesh grids and C does not depend on h_j and ε. Moreover, we have

$$||u^\varepsilon - u_{h_j}^\varepsilon||_{L^q(\Omega,H^1(D))} \leq C(h_j + \varepsilon)||f||_{L^2(D)} + C\sqrt{\frac{\varepsilon}{h_j}}, \quad \forall q \geq 1. \tag{21}$$

The convergence of a multiscale finite element method for deterministic multiscale elliptic problems was proved in [23]. The proposition 3.1 can be easily proved since for each realization of $a^\varepsilon(x,\omega)$, the problem (9)-(10) is a deterministic problem.

One can use oversampling technique [22, 13] to improve the convergence result in proposition 3.1, i.e. replacing the error term $C\sqrt{\frac{\varepsilon}{h_j}}$ by $C\frac{\varepsilon}{h_j}$. The main idea is to solve Eqns.(14)-(15) in a larger domain and use only the interior information to construct the basis, which can significantly reduce the effect of the boundary layer on the basis functions.

Specifically, let ψ^{kl} be solution of the following problem in a larger domain $S_j^k \supset D_j^k$ (with $\text{dist}(\partial S_j^k, D_j^k) > \varepsilon$), $1 \leq k \leq K$,

$$-\nabla \cdot (a^\varepsilon(x,\omega)\nabla\psi^{kl}(x,\omega)) = 0, \quad x \in S_j^k, \quad \omega \in \Omega, \tag{22}$$

$$\psi^{kl}(x,\omega) = p^{kl}(x), \quad x \in \partial S_j^k, \quad l = 1, ..., d, \tag{23}$$

where d is the number of nodes on S_j^k and $p^{kl}(x)$ is a linear basis function defined on the boundary of S_j^k. Then, we form the actual MsFEM basis $\phi^{ki}(x,\omega)$ by linear combination of $\psi^{kl}(x,\omega)$, i.e.,

$$\phi^{ki}(x,\omega) = \sum_{l=1}^{d} c_{il}\psi^{kl}(x,\omega)|_{D_j^k}, \quad i = 1, ..., d. \tag{24}$$

The coefficients c_{il}'s are determined by condition $\phi^{ki}(x_l,\omega)=\delta_{il}$, where $x_l, l=1,...,d$ are the nodes of domain D_j^k. The over-sampling technique results in a non-conforming MsFEM method. We refer the interested reader to [13] for the convergence analysis of the oversampling MsFEM. Later on, a Petrov-Galerkin MsFEM formulation with nonconforming multiscale trial functions and linear test functions was proposed in [24], which further improves the convergence result in proposition 3.1.

3.2. Construction of multiscale reduced basis functions

We can solve Eq.(14) (or its corresponding weak form Eq.(17)) by partitioning the coarse block D_j^k into a fine grid with mesh size h, where $h \ll \varepsilon$. Let N_{dof} denote the degree of freedom (DOF) of the fine grid inside D_j^k. If we solve Eqns. (14)-(15) for each sample ω using the standard FEM, the computational cost will be prohibitively expensive. To reduce the computational cost in computing the mulsticale basis function $\phi^{kl}(x,\omega)$, we will construct a small number of multiscale reduced basis $\{\zeta^{kl,i}(x)\}_{i=1}^m$ within each block D_j^k of each level using the proper orthogonal decomposition (POD) method [32, 4, 36].

First, we use MC method or stochastic collocation method to compute Q solution samples $\{\phi^{kl}(x,\omega_q)\}_{q=1}^Q$ of the multiscale FEM basis functions within each D_j^k. Given Q snapshots of solution samples, we subtract $p^{kl}(x)$ and obtain a linear space V_{snap}, denoted as

$$V_{snap}=\{\phi_0^{kl}(x,\omega_1),\phi_0^{kl}(x,\omega_2),...,\phi_0^{kl}(x,\omega_Q)\}. \tag{25}$$

Then, we apply POD method to build a set of multiscale reduced basis $\{\zeta^{kl,1}(x),...,\zeta^{kl,m}(x)\}$ with $m \ll \min(Q, N_{dof})$ that optimally approximates the input solution snapshots. Given any integer m, the POD basis functions minimize the following error

$$\frac{1}{Q}\sum_{q=1}^{Q}\Big|\Big|\phi_0^{kl}(\cdot,\omega_q)-\sum_{s=1}^{m}\big(\phi_0^{kl}(\cdot,\omega_q),\zeta^{kl,s}(\cdot)\big)_X\zeta^{kl,s}(\cdot)\Big|\Big|_X^2, \tag{26}$$

subject to the constraints that $\big(\zeta^{kl,s_1}(\cdot),\zeta^{kl,s_2}(\cdot)\big)_X=\delta_{s_1s_2}$, $1\le s_1,s_2\le m$, where $\delta_{s_1s_2}=1$ if $s_1=s_2$, otherwise $\delta_{s_1s_2}=0$, where $X=H^1(D)$.

Using the method of snapshot proposed by Sirovich [32], we know that the optimization problem (26) can be reduced to an eigenvalue problem

$$Kv=\lambda v, \tag{27}$$

where $K\in\mathbb{R}^{Q\times Q}$ is the correlation matrix with (i,j)-element $K_{ij}=\frac{1}{Q}\big(\phi_0^{kl}(\cdot,\omega_i),\phi_0^{kl}(\cdot,\omega_j)\big)_X$ and is symmetric and semi-positive definite. We sort the eigenvalues in a decreasing order as $\lambda_1\ge\lambda_2\ge...\ge\lambda_Q>0$ and the corresponding eigenvectors are denoted by v_s, $s=1,...,Q$. It can be shown that the POD basis functions are constructed by

$$\zeta^{kl,s}(\cdot)=\frac{1}{\sqrt{\lambda_s}}\sum_{q=1}^{Q}(v_s)_q\phi_0^{kl}(\cdot,\omega_q),\quad 1\le s\le Q, \tag{28}$$

where $(v_s)_q$ is the q-th component of the eigenvector v_s. The basis functions $\{\zeta^{kl,s}(\cdot)\}_{s=1}^m$ minimizes the error (26). This result as well as the error formula were proved in [19].

Proposition 3.2 (Sec. 3.3.2, [19]). *Let $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_Q > 0$ denote the positive eigenvalues of K in (27). Then $\{\zeta^{kl,s}(\cdot)\}_{s=1}^m$ constructed according to (28) is the set of POD basis functions, and we have the following error formula:*

$$\frac{\sum_{q=1}^{Q} \left|\left|\phi_0^{kl}(\cdot,\omega_q) - \sum_{s=1}^{m} \left(\phi_0^{kl}(\cdot,\omega_q), \zeta^{kl,s}(\cdot)\right)_X \zeta^{kl,s}(\cdot)\right|\right|_X^2}{\sum_{q=1}^{Q} \left|\left|\phi_0^{kl}(\cdot,\omega_q)\right|\right|_X^2} = \frac{\sum_{k=m+1}^{Q} \lambda_k}{\sum_{k=1}^{Q} \lambda_k}, \tag{29}$$

where $X = H^1(D)$ and the number m is determined according to the ratio $\rho = \frac{\sum_{k=m+1}^{Q} \lambda_k}{\sum_{k=1}^{Q} \lambda_k}$.

In practice, we shall make use of the decay property of eigenvalues in λ_k and choose the first m dominant eigenvalues such that the ratio ρ is small enough to achieve an expected accuracy, for instance $\rho = 1\%$. To compute the multiscale reduced basis requires a certain amount of computational cost in the offline stage. However the multiscale reduced basis can be repeatedly used in the online stage for different realizations of the coefficient $a^\varepsilon(x,\omega)$ and deterministic forcing function $f(x)$, which gives considerable savings if we need to solve the Eq.(9) with many different forcing functions. We will demonstrate this through numerical examples in Section 5.

With the multiscale reduced basis $\{\zeta^{kl,i}(x)\}_{i=1}^m$, we can approximate $\phi^{kl}(x,\omega)$ by

$$\tilde{\phi}^{kl}(x,\omega) = p^{kl}(x) + \Sigma_{i=1}^m c_i(\omega)\zeta^{kl,i}(x). \tag{30}$$

Then, we substitute (30) into Eqns. (14)-(15) and use Galerkin method to obtain $\{c_i\}_{i=1}^m$. The multiscale reduced basis $\{\zeta^{kl,i}(x)\}_{i=1}^m$ will be efficient since $m \ll N_{dof}$. Finally, we use $\{\tilde{\phi}^{kl}\}$ to assemble the stiffness matrix of the Eq.(9) for each sample ω, just as we did in MsFEM.

3.3. The affine parameterization to speedup

We discuss the offline-online computational procedure [29, 30] that allows us to efficiently compute $\{c_i\}_{i=1}^m$ in Eq.(30). We assume that $a^\varepsilon(x,\omega)$ in Eq.(14) satisfies the affine parameter dependence property. With this assumption, we can pre-compute the main part of the stiffness matrix in the offline stage. This leads to considerable saving in assembling the stiffness matrix for each ω in the online stage. Specifically, we assume $a^\varepsilon(x,\omega)$ can be expressed as follows

$$a^\varepsilon(x,\omega) = \sum_{r=1}^{R} \xi_r(\omega) a_r^\varepsilon(x), \tag{31}$$

where $\{\xi_r(\omega)\}_{r=1}^R$ are R i.i.d. random variables that are L^2-intergrable and $a_r^\varepsilon(x) \in L^\infty(D)$ depends only on x. Then the bilinear forms and linear functionals defined in Eqns.(18) and (19) can be written as

$$a^\varepsilon(w,v;\omega) = \sum_{r=1}^{R} \xi_r(\omega) a_r^\varepsilon(w,v), \tag{32}$$

$$f^\varepsilon(v;\omega) = \sum_{r=1}^{R} \xi_r(\omega) f_r^\varepsilon(w,v), \tag{33}$$

where

$$a_r^{\varepsilon}(w,v) = \int_{D^k} a_r^{\varepsilon}(x)\nabla w \cdot \nabla v dx, \tag{34}$$

$$f_r^{\varepsilon}(v) = -\int_{D^k} a_r^{\varepsilon}(x)\nabla p^{kl}(x) \cdot \nabla v dx. \tag{35}$$

One can see that $a_r^{\varepsilon}(w,v)$ and $f_r^{\varepsilon}(v)$ are independent of the random sample ω. Thus, they can be pre-computed and saved in the offline stage.

We now describe the computational procedure for constructing the multiscale finite element basis. Recall that in Eq.(16), we assume that $\phi_0^{kl}(x,\omega) \in span\{\zeta^{kl,i}(x)\}_{i=1}^m$, where $\{c_i\}_{i=1}^m$ are expansion coefficients which can be determined by using the Galerkin method. Substituting the ansatz (30) into Eq.(17) and choosing the test function $v = \zeta^{kl,i}(x), i = 1, ..., m$, we get the linear equation system for $\mathbf{c}^{kl}$ as

$$\mathbf{A}^{kl}(\omega)\mathbf{c}^{kl}(\omega) = \mathbf{F}^{kl}(\omega), \tag{36}$$

where $\mathbf{A}^{kl}(\omega)$ is an $m \times m$ symmetric positive definite matrix with entries $\mathbf{A}_{pq}^{kl}(\omega) = a^{\varepsilon}(\zeta^{kl,p}, \zeta^{kl,q}; \omega)$, $1 \le p, q \le m$, and $\mathbf{F}^{kl}(\omega)$ is the load vector with entries $\mathbf{F}_q^{kl}(\omega) = f^{\varepsilon}(\zeta^{kl,q}; \omega)$, $1 \le q \le m$. Since $m \ll N_h$ (DOF of reduced basis is far less than that of the linear FEM basis), Eq. (36) can be solved very efficiently.

Next, we apply the affine decomposition (32) and (33) and obtain

$$\mathbf{A}^{kl}(\omega) = \sum_{r=1}^{R} \xi_r(\omega)\mathbf{A}_r^{kl}, \tag{37}$$

$$\mathbf{F}^{kl}(\omega) = \sum_{r=1}^{R} \xi_r(\omega)\mathbf{F}_r^{kl}, \tag{38}$$

where $\mathbf{A}_r^{kl}$ and $\mathbf{F}_r^{kl}$ are given by $\mathbf{A}_{r,pq}^{kl}(\omega) = a_r^{\varepsilon}(\zeta^{kl,p}, \zeta^{kl,q})$, $1 \le p, q \le m$, and $\mathbf{F}_{r,q}^{kl}(\omega) = f_r^{\varepsilon}(\zeta^{kq})$, $1 \le q \le m$, respectively. We can efficiently assemble the stiffness matrix $\mathbf{A}^{kl}(\omega)$ and load vector $\mathbf{F}^{kl}(\omega)$ for each sample ω, since $\mathbf{A}_r^k$ and $\mathbf{F}_r^{kl}$ have been pre-computed and saved in the offline stage.

3.4. Online computation

The online computation of our MsMLMC is essentially the application of the standard MLMC method. Since the multiscale finite element basis obtained from Eq.(14) have contains the multiscale information of the Eq.(9), the coarsest level mesh size h_0 does not necessarily well resolve the smallest-scale of the Eq.(9). Thus, we choose $h_0 \gg \varepsilon$.

On the level j, we use the precomputed multiscale reduced basis $\{\zeta^{kl,i}\}_{i=1}^m$ to compute the multiscale finite basis within each block $D_j^k, 1 \le k \le K$, and use these multiscale finite basis elements to obtain the multiscale solution for each sample ω.

We briefly discuss how to assemble the global stiffness matrix and the right hand side for the weak form of Eq. (9). On each block $D_j^k, 1 \le k \le K$, we solve Eqns. (14) and (15) using

the multiscale reduced basis to obtain the MsFEM basis, and form the local stiffness matrix S^k (an 3×3 or 4×4 matrix for triangular or quadrilateral element, respectively) and vector b^k (an 3×1 or 4×1 vector accordingly). In this paper, we choose the quadrilateral element. Then S^k and b^k have entries

$$S^k_{ij}(\omega) = \int_{D^k} a^\varepsilon(x,\omega)\nabla(\phi_0^{ki}(x,\omega) + p^{ki}(x)) \cdot \nabla(\phi_0^{kj}(x,\omega) + p^{kj}(x))dx, \quad 1 \le i,j \le 4, \tag{39}$$

$$b^k_j(\omega) = \int_{D^k} f(x)(\phi_0^{kj}(x,\omega) + p^{kj}(x))dx, \quad 1 \le j \le 4. \tag{40}$$

Substituting the ansatzs (30) and (31) into (39), (40), we get

$$S^k_{ij}(\omega) = \sum_{r=1}^{R} \xi_r(\omega)(\mathbf{c}^{ki}(\omega)^T \mathbf{A}_r^{k,ij} \mathbf{c}^{kj}(\omega) + 2\mathbf{c}^{ki}(\omega)^T \mathbf{B}_r^{k,ij} + \mathbf{C}_r^{k,ij}), \quad 1 \le i,j \le 4, \tag{41}$$

$$b^k_j(\omega) = \mathbf{c}^{kj}(\omega)^T \mathbf{D}^k + \mathbf{E}^{kj}, \quad 1 \le j \le 4, \tag{42}$$

where

$$\begin{aligned} \mathbf{A}^{k,ij}_{r,mn} &= \int_{D^k} a_r^\varepsilon(x)\nabla\zeta^{ki,m} \cdot \nabla\zeta^{kj,n}dx, \quad \mathbf{B}^{k,ij}_{rn} = \int_{D^k} a_r^\varepsilon(x)\nabla\zeta^{ki,n} \cdot \nabla p^{kj}(x)dx, \\ \mathbf{C}^{k,ij}_{r} &= \int_{D^k} a_r^\varepsilon(x)\nabla p^{ki}(x) \cdot \nabla p^{kj}(x)dx, \quad \mathbf{D}^{k,j}_{n} = \int_{D^k} f(x)\zeta^{kj,n}dx, \\ \mathbf{E}^{k,j} &= \int_{D^k} f(x)p^{kj}(x)dx. \end{aligned} \tag{43}$$

Recall that in (41) and (42), the quantities $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{D}$, and $\mathbf{E}$ do not depend on the sample ω and can be pre-computed and saved in the offline stage. In the online computing, for each sample ω, we can solve Eq. (36) to obtain $\mathbf{c}^{ki}(\omega)$ and assemble the local stiffness matrix S^k and vector b^k by using (41) and (42). Therefore, we can efficiently solve an algebraic system and obtain the multiscale solution of Eq.(9).

In practice, we can also choose the Petrov-Galerkin MsFEM formulation with nonconforming multiscale trial functions (obtained by the over-sampling technique) and linear test functions. Through numerical experiments, we find that this does reduce the numerical error significantly. In the Petrov-Galerkin MsFEM framework, we can still pre-compute and save all the data that do not depend on the sample ω, and can be used to form the stiffness matrix and load vector in the online stage. This can be done by using the similar idea as we did in (39)-(43). Thus, we omit details here.

3.5. The MsMLMC algorithm

In this subsection, we give the algorithm of the MsMLMC to solve the multiscale stochastic equation in a multiquery setting. Our method consists of the offline and online stages. They are summarized in Algorithm 1 and 2, respectively. The implementation of the Petrov-Galerkin MsFEM with over-sampling technique can be developed similarly.

One important issue of the Algorithm 2 is how to decide the sample numbers at different levels. In [16], Giles proposed an effective way to estimate the variance and cost on each level,

and used this to determine the optimal number of samples on each level. In the MsMLMC, the online computational cost consists of several parts, such as the solving local cell problem to decide multiscale basis functions, assembling the global stiffness matrix, and solving a linear equation to obtain the global multiscale solution. Thus, we cannot directly apply the strategy used in [10] to estimate the cost per sample. We propose an idea based on the variance decay property and run-time execution time to obtain a sub-optimal number of samples on each level.

We first use Q_0 Monte Carlo samples to compute the expectation on the coarsest grid. Meanwhile, we estimate the computational cost of per sample, denoted by t_0, on the coarsest grid level. Roughly speaking, the computational cost of per sample on j-th level can be estimated as $4^j t_0$. In addition, we generate certain number of Monte Carlo samples to estimate the empirical variance of the variance decay. Based on the run-time information about empirical variance and cost per sample, we decide how many Monte Carlo samples, denoted by Q_j, are used in computing the expectation of the difference of solutions between levels j and $j-1$, $j = 1, ..., L$.

Algorithm 1 The offline stage of MsMLMC

1: Partition the domain D into a number of nested mesh grid blocks, i.e., $D_0 \subset \ldots \subset D_{j-1} \subset D_j \ldots \subset D_L$. The mesh size of j-th level is $h_j = h_0 2^{-j}$ and h_0 is the coarsest level mesh size.
2: **for** $j = 0 \to L$ **do**
3: **for** $k = 1 \to K_j$ **do**
4: On each element E^k, we do only once:
5: Solve Eqns.(14) and (15) with MC or SC method to obtain samples $\{\phi^{kl}(x, \omega_q)\}_{q=1}^{Q}$, $l = 1, ..., 4$.
6: Apply POD to compute the reduced basis $\{\zeta^{kl,i}(x)\}_{i=1}^{m}$.
7: Compute and save all the relevant quantities **A**, **B**, **C**, **D**, and **E** from Eq.(43).
8: **end for**
9: **end for**

4. Some analyses

One essential requirement in the MLMC method is the asymptotic variance reduction between two consecutive coarse grid levels. We will prove that this requirement is satisfied in the MsMLMC method. Specifically, let $\tilde{u}_j^{\varepsilon}(x, \omega)$ and $\tilde{u}_{j-1}^{\varepsilon}(x, \omega)$ denote numerical solutions of the MsMLMC method obtained at the j-th level and $(j-1)$-th level, respectively. We will prove that $\int \mathrm{Var}\left(v_j^{\varepsilon} - v_{j-1}^{\varepsilon}\right) dx = c_2 h_j^{\beta}$, where v_j^{ε} is a numerical approximation of the functional of the solution $\tilde{u}_j^{\varepsilon}(x, \omega)$ (the definition for v_{j-1}^{ε} is the same) and h_j is the mesh size of the grids of the j-th level. Recall that $\tilde{u}_j^{\varepsilon}(x, \omega)$ is the numerical solution obtained in the space $span\{\tilde{\phi}^{kl}(x, \omega)\}$ (see (30)) at the j-th level. We first estimate the error between $u^{\varepsilon}(x, \omega)$ and $\tilde{u}_j^{\varepsilon}(x, \omega)$. Before that, we need two assumptions.

Algorithm 2 The online stage of MsMLMC

For each query $f(x)$ in Eq.(9), we calculate expected values of a functional of $u^{\varepsilon}(x,\omega)$.
Calculate the coarsest level solution, $u_0^{\varepsilon}(x,\omega)$ with Q_0 samples $\{\omega_q^0\}_{q=1}^{Q_0}$.
for $q = 1 \to Q_0$ **do**
    for $k = 1 \to K_0$ **do**
        Assemble $\mathbf{A}^{kl}(\omega_q^{(0})$ and $\mathbf{F}^{kl}(\omega_q^0)$ from (37) and (38), and solve (36) to obtain $\mathbf{c}^{kl}(\omega_q^0)$.
        Assemble $S^k(\omega_q^0)$ and $b^k(\omega_q^0)$ from (41) and (42).
        Form and solve the global linear equation system to obtain $u_0^{\varepsilon}(x,\omega_q^0)$.
    end for
end for
Calculate the solution differences on consecutive levels.
for $j = 1 \to L$ **do**
    Generate Q_j samples $\{\omega_q^j\}_{q=1}^{Q_j}$ used for both levels.
    for $q = 1 \to Q_j$ **do**
        Compute on level j
        for $k = 1 \to K_j$ **do**
            Assemble $\mathbf{A}^{kl}(\omega_q^j)$ and $\mathbf{F}^{kl}(\omega_q^j)$ from (37) and (38), and solve (36) to obtain $\mathbf{c}^{kl}(\omega_q^j)$.
            Assemble $S^k(\omega_q^j)$ and $b^k(\omega_q^j)$ from (41) and (42).
            Form and solve the global linear equation system to obtain $u_j^{\varepsilon}(x,\omega_q^j)$.
        end for
        Compute on level $j-1$
        for $k = 1 \to K_{j-1}$ **do**
            Assemble $\mathbf{A}^{kl}(\omega_q^j)$ and $\mathbf{F}^{kl}(\omega_q^j)$ from (37) and (38), and solve (36) to obtain $\mathbf{c}^{kl}(\omega_q^j)$.
            Assemble $S^k(\omega_q^j)$ and $b^k(\omega_q^j)$ from (41) and (42).
            Form and solve the global linear equation system to obtain $u_{j-1}^{\varepsilon}(x,\omega_q^j)$.
        end for
    end for
end for
Adopt the MLMC formulation (6) to calculate expected values of a functional of $u^{\varepsilon}(x,\omega)$.

Assumption 4.1. *Suppose $a^\varepsilon(x,\omega)$ has following property: $\forall \delta_1 > 0$, there exists a Q_{δ_1} and a choice of snapshots $\{a^\varepsilon(x,\omega_q)\}$ such that*

$$\mathbb{E}\left[\inf_{1\le q\le Q_{\delta_1}} \left|\left|a^\varepsilon(x,\omega)-a^\varepsilon(x,\omega_q)\right|\right|_{L^\infty(D)}\right] \le \delta_1, \tag{44}$$

Remark 4.1. Under the affine parameterization assumption (31), we can easily verify assumption 4.1 and even give a way to choose snapshots $\{a^\varepsilon(x,\omega_q)\}$ if we know the probability density functions of all random variables $\xi_r(\omega)$, $r = 1, ..., R$.

Assumption 4.2. *For any $\delta_2 > 0$, we can choose m large enough such that the m-term approximation $\tilde{\phi}^{kl}$ satisfies*

$$||\tilde{\phi}_0^{kl}(x,\omega_q)-\phi_0^{kl}(x,\omega_q)||_{H^1(D_j^k)} \le \delta_2, \quad \forall 1 \le q \le Q \tag{45}$$

The Asm.4.2 can be verified according to the approximating property of the reduced basis stated in Prop.3.2.

For each $\omega \in \Omega$, the MsFEM basis function $\phi^{kl}(x,\omega) = p^{kl}(x)+\phi_0^{kl}(x,\omega)$ is obtained by solving Eqns.(14)-(15), where $p^{kl}(x)$ is the boundary condition. Let $\tilde{\phi}^{kl}(x,\omega) = p^{kl}(x) + \tilde{\phi}_0^{kl}(x,\omega)$ denote the approximated MsFEM basis functions, where $\tilde{\phi}_0^{kl}(x,\omega) = \sum_{i=1}^m c_i(\omega)\zeta^{kl,i}(x)$. We now estimate the error between $\phi^{kl}(x,\omega)$ and $\tilde{\phi}^{kl}(x,\omega)$.

Theorem 4.3. *Under assumptions 4.1 and 4.2 and the assumption for the coefficient $a^\varepsilon(x,\omega)$ (11), for any $\delta > 0$, we can choose coefficient samples $\{a^\varepsilon(x,\omega_q)\}_{1\le q\le Q}$ and the number of the mulsticale reduced basis m such that*

$$\mathbb{E}\left[\max_{kl} \left|\left|\phi^{kl}(x,\omega)-\tilde{\phi}^{kl}(x,\omega)\right|\right|^p_{H^1(D_j^k)}\right] \le C\delta_3, \quad \textit{for} \quad p = 1, 2, \tag{46}$$

where C is a generic constant depending on $a_{\min}$, $a_{\max}$ and the size of the coarse sub-domain, and $\delta_3 = \delta_1 h_j^{-1} + \delta_2$ with δ_1 and δ_2 being defined in Asm.4.1 and Asm.4.2.

Proof. Let $\phi_0^{kl}(x,\omega) = \phi^{kl}(x,\omega) - p^{kl}(x)$, $\tilde{\phi}_0^{kl}(x,\omega) = \tilde{\phi}^{kl}(x,\omega) - p^{kl}(x)$, where $p^{kl}(x)$ is the boundary condition defined in (15). We have the estimate,

$$\begin{aligned}
&\left|\left|\phi^{kl}(x,\omega)-\tilde{\phi}^{kl}(x,\omega)\right|\right|_{H^1(D_j^k)} = \left|\left|\phi_0^{kl}(x,\omega)-\tilde{\phi}_0^{kl}(x,\omega)\right|\right|_{H^1(D_j^k)} \\
\le &\left|\left|\phi_0^{kl}(x,\omega)-\phi_0^{kl}(x,\omega_q)\right|\right|_{H^1(D_j^k)} + \left|\left|\phi_0^{kl}(x,\omega_q)-\tilde{\phi}_0^{kl}(x,\omega_q)\right|\right|_{H^1(D_j^k)} + \left|\left|\tilde{\phi}_0^{kl}(x,\omega_q)-\tilde{\phi}_0^{kl}(x,\omega)\right|\right|_{H^1(D_j^k)} \\
:= &I_1 + I_2 + I_3
\end{aligned} \tag{47}$$

We estimate the term I_1 first. Let $V(D_j^k)$ be a piecewise linear finite element space over D_j^k, where the grid size is much smaller than ε. Let $V_0(D_j^k) = \{f|_{\partial D_j^k} = 0 \big| f \in V(D_j^k)\}$. Then $\phi_0^{kl}(x,\omega) \in V_0(D_j^k)$ is the solution to the equation

$$a^\varepsilon(\phi_0^{kl}, v; \omega) = -a^\varepsilon(p^{kl}, v; \omega), \ \forall v \in V_0(D_j^k), \tag{48}$$

where the bilinear form $a^\varepsilon(\cdot,\cdot;\omega)$ is defined in (18) and $\phi_0^{kl}(x,\omega_i) \in V_0(D_j^k)$ is the solution to the equation

$$a^\varepsilon(\phi_0^{kl}, v; \omega_q) = -a^\varepsilon(p^{kl}, v; \omega_q), \quad \forall v \in V_0(D_j^k). \tag{49}$$

We choose $\phi_0^{kl}(x,\omega_q)$ as the test function in (48) and (49) and obtain

$$\begin{aligned}
&a_{\min}\big|\big|\phi_0^{kl}(x,\omega) - \phi_0^{kl}(x,\omega_q)\big|\big|^2_{H^1(D_j^k)},\\
\leq &\int_{D_j^k} a^\varepsilon(x,\omega)\big(\nabla\phi_0^{kl}(x,\omega) - \nabla\phi_0^{kl}(x,\omega_q)\big)\cdot\big(\nabla\phi_0^{kl}(x,\omega) - \nabla\phi_0^{kl}(x,\omega_q)\big)dx,\\
= &\int_{D_j^k} \big(a^\varepsilon(x,\omega_q) - a^\varepsilon(x,\omega)\big)\nabla(\phi_0^{kl}(x,\omega_q) + p^{kl}(x))\cdot\nabla\big(\phi_0^{kl}(x,\omega) - \phi_0^{kl}(x,\omega_q)\big)dx,\\
\leq &\big|\big|a^\varepsilon(x,\omega_q) - a^\varepsilon(x,\omega)\big|\big|_{L^\infty(D_j^k)}\big|\big|\phi^{kl}(x,\omega_q)\big|\big|_{H^1(D_j^k)}\big|\big|\phi_0^{kl}(x,\omega) - \phi_0^{kl}(x,\omega_q)\big|\big|_{H^1(D_j^k)}.
\end{aligned} \tag{50}$$

Dividing the term $\big|\big|\phi_0^{kl}(x,\omega) - \phi_0^{kl}(x,\omega_q)\big|\big|_{H^1(D_j^k)}$ on both sides of (50), we get

$$\big|\big|\phi_0^{kl}(x,\omega) - \phi_0^{kl}(x,\omega_q)\big|\big|_{H^1(D_j^k)} \leq C\big|\big|a^\varepsilon(x,\omega_q) - a^\varepsilon(x,\omega)\big|\big|_{L^\infty(D_j^k)}\big|\big|\phi^{kl}(x,\omega_q)\big|\big|_{H^1(D_j^k)}. \tag{51}$$

Additionally, we choose $\phi_0^{kl}(x,\omega_q)$ as the test function in (49) and obtain

$$\begin{aligned}
&a_{\min}\big|\big|\phi_0^{kl}(x,\omega_q)\big|\big|^2_{H^1(D_j^k)} \leq \int_{D_j^k} a^\varepsilon(x,\omega)\nabla\phi_0^{kl}(x,\omega_q)\cdot\nabla\phi_0^{kl}(x,\omega_q)dx\\
= &-\int_{D_j^k} a^\varepsilon(x,\omega)\nabla p^{kl}(x)\cdot\nabla\phi_0^{kl}(x,\omega_q)dx \leq C\big|\big|\phi_0^{kl}(x,\omega_q)\big|\big|_{H^1(D_j^k)}\big|\big|p^{kl}(x)\big|\big|_{H^1(D_j^k)}.
\end{aligned} \tag{52}$$

Therefore, we have

$$||\phi_0^{kl}(x,\omega_q)||_{H^1(D_j^k)} \leq \frac{C}{a_{min}}||p^{kl}(x)||_{H^1(D_j^k)}.$$

Since p^{kl} is a linear function in the MsFEM, we know that $||p^{kl}(x)||_{H^1(D_j^k)} \leq h_j^{-1}$, where C is a constant related to the domain. Therefore,

$$||\phi^{kl}(x,\omega_q)||_{H^1(D_j^k)} \leq ||\phi_0^{kl}(x,\omega_q)||_{H^1(D_j^k)} + ||p^{kl}(x)||_{H^1(D_j^k)} \leq (\frac{C}{a_{min}} + 1)h_j^{-1} \tag{53}$$

Substituting (53) into (51), we obtain the estimate for the term I_1

$$\big|\big|\phi_0^{kl}(x,\omega) - \phi_0^{kl}(x,\omega_q)\big|\big|_{H^1(D_j^k)} \leq C\big|\big|a^\varepsilon(x,\omega_q) - a^\varepsilon(x,\omega)\big|\big|_{L^\infty(D_j^k)}(\frac{C}{a_{min}} + 1)h_j^{-1}. \tag{54}$$

For the term I_3 in Eq.(47), we can similarly get that

$$\big|\big|\tilde{\phi}_0^{kl}(x,\omega) - \tilde{\phi}_0^{kl}(x,\omega_q)\big|\big|_{H^1(D_j^k)} \leq C\big|\big|a^\varepsilon(x,\omega_q) - a^\varepsilon(x,\omega)\big|\big|_{L^\infty(D_j^k)}(\frac{C}{a_{min}} + 1)h_j^{-1}. \tag{55}$$

The term I_2 in Eq.(47) can be controlled according to the POD method in 4.2. Combining the estimates for terms I_1, I_2 and I_3, we can choose the number of coefficient samples $\{a^\varepsilon(x,\omega_q)\}_{1\le q\le Q}$ in Asm.4.1 and number of the multiscale reduced basis in Asm.4.2, such that

$$\big|\big|\phi^{kl}(x,\omega)-\tilde{\phi}^{kl}(x,\omega)\big|\big|_{H^1(D_j^k)}\le 2C\big|\big|a^\varepsilon(x,\omega_q)-a^\varepsilon(x,\omega)\big|\big|_{L^\infty(D_j^k)}(\frac{C}{a_{min}}+1)h_j^{-1}+\delta_2\le\delta_3. \tag{56}$$

Since this estimate (56) holds almost surely for each realization $\omega\in\Omega$, we integrate over the random space and prove the theorem.

□

Theorem 4.4. *For each ω, let $u^\varepsilon_{h_j}(x,\omega)$ be the solution obtained by using the MsFEM basis on coarse mesh with size h_j, and $\tilde{u}^\varepsilon_{h_j}(x,\omega)$ be the solution obtained by using the multiscale reduced basis. We have the estimate for the error between $u^\varepsilon_{h_j}(x,\omega)$ and $\tilde{u}^\varepsilon_{h_j}(x,\omega)$*

$$\mathbb{E}\left[||u^\varepsilon_{h_j}(x,\omega)-\tilde{u}^\varepsilon_{h_j}(x,\omega)||^p_{H^1(D)}\right]\le C\delta_3,\quad a.s.,\quad \omega\in\Omega,\quad p=1,2, \tag{57}$$

where C is a generic constant, $\delta_3=\delta_1h_j^{-1}+\delta_2$, and the parameters δ_1 and δ_2 are defined in Asm.4.1 and Asm.4.2.

Proof. For each ω, we have $u^\varepsilon_{h_j}(x,\omega)=\sum_{k=1}^K u_k(\omega)\phi^k(x)$ and $\tilde{u}^\varepsilon_{h_j}(x,\omega)=\sum_{k=1}^K \tilde{u}_k(\omega)\tilde{\phi}^k(x)$, where $\phi^k(x)$ denotes the MsFEM basis associated with the coarse grids on level j, $\tilde{\phi}^k(x)$ denotes the multiscale reduced basis associated with the coarse grids on level j, and K is the degree of freedom of the coarse grids.

Let $\mathbf{B}=(b_{ij})$ and $\tilde{\mathbf{B}}=(\tilde{b}_{ij})$ denote the stiffness matrices associated with the MsFEM and our method, where the entries in the matrices are defined as

$$b_{ij}=\int\nabla\phi^i(x,\omega)a^\varepsilon(x,\omega)\cdot\nabla\phi^j(x,\omega)dx, \tag{58}$$

$$\tilde{b}_{ij}=\int\nabla\tilde{\phi}^i(x,\omega)a^\varepsilon(x,\omega)\cdot\nabla\tilde{\phi}^j(x,\omega)dx. \tag{59}$$

We estimate the difference between the entries b_{ij} and $\tilde{b}_{ij}$ by

$$\begin{aligned}
|b_{ij}-\tilde{b}_{ij}| &= \Big|\int\nabla\phi^i(x,\omega)a^\varepsilon(x,\omega)\cdot\nabla\phi^j(x,\omega)dx-\int\nabla\tilde{\phi}^i(x,\omega)a^\varepsilon(x,\omega)\cdot\nabla\tilde{\phi}^j(x,\omega)dx\Big|,\\
&\le \Big|\int\nabla\phi^i a^\varepsilon\cdot\nabla\phi^j dx-\int\nabla\tilde{\phi}^i a^\varepsilon\cdot\nabla\phi^j dx\Big|+\Big|\int\nabla\tilde{\phi}^i a^\varepsilon\cdot\nabla\phi^j dx-\int\nabla\tilde{\phi}^i a^\varepsilon\cdot\nabla\tilde{\phi}^j dx\Big|,\\
&\le ||a^\varepsilon||_{L^\infty(D)}\Big(||\phi^j||_{H^1(D)}||\nabla\phi^i-\nabla\tilde{\phi}^i||_{L^2(D)}+||\tilde{\phi}^i||_{H^1(D)}||\nabla\phi^j-\nabla\tilde{\phi}^j||_{L^2(D)}\Big).
\end{aligned} \tag{60}$$

Notice that $a^\varepsilon(x,\omega)$ is bounded almost surely, $||\phi^j(x,\omega)||_{H^1(D)}\le Ch_j^{-1}$, and $||\tilde{\phi}^i(x,\omega)||_{H^1(D)}\le Ch_j^{-1}$, where C depends on $a_{\min}$. Using the same argument in Theorem 4.3, we can prove that

$$\mathbb{E}\left[|b_{ij}-\tilde{b}_{ij}|\right]\le C(\delta_1h_j^{-2}+\delta_2h_j^{-1}), \tag{61}$$

where C depends on $a_{\min}$. When the mesh size of the coarse grid h_j is given, one can choose δ_1 and δ_2 so that $|b_{ij} - \tilde{b}_{ij}|$ is small.

Let $\mathbf{E}_a = \mathbf{E}_a(\delta_1, \delta_2, H)$ denote the error between $\mathbf{B}$ and $\tilde{\mathbf{B}}$, i.e., $\tilde{\mathbf{B}} = \mathbf{B}+\mathbf{E}_a$. Let $\mathbf{f} = (f_i)$ and $\tilde{\mathbf{f}} = (\tilde{f}_i)$ denote the right-hand side vectors with $f_i = \int \phi^i(x) f(x) dx$ and $\tilde{f}_i = \int \tilde{\phi}^i(x) f(x) dx$, respectively. Using a similar argument as (60) and the Poincaré inequality, we get

$$\mathbb{E}\left[|f_i - \tilde{f}_i|\right] \leq C(\delta_1 h_j^{-1} + \delta_2), \tag{62}$$

where C depends on $a_{\min}$. Let $\mathbf{e}_f = \mathbf{e}_f(\delta_1, \delta_2, H)$ denote the error between $\mathbf{f}$ and $\tilde{\mathbf{f}}$, i.e., $\tilde{\mathbf{f}} = \mathbf{f} + \mathbf{e}_f$.

Recall that $u^{\varepsilon}_{h_j}(x,\omega) = \sum_{k=1}^{K} u_k(\omega)\phi^k(x)$ and $\tilde{u}^{\varepsilon}_{h_j}(x,\omega) = \sum_{k=1}^{K} \tilde{u}_k(\omega)\tilde{\phi}^k(x)$. Let $\mathbf{u} = (u_1, ..., u_K)^T$ and $\tilde{\mathbf{u}} = (\tilde{u}_1, ..., \tilde{u}_K)^T$. They satisfy $\mathbf{B}\mathbf{u} = \mathbf{f}$ and $\tilde{\mathbf{B}}\tilde{\mathbf{u}} = \tilde{\mathbf{f}}$, respectively. After simple calculations, we get $\mathbf{u} - \tilde{\mathbf{u}} = \mathbf{B}^{-1}(\mathbf{E}_a\tilde{\mathbf{u}} - \mathbf{e}_f)$ and the estimate

$$\begin{aligned} ||\mathbf{u} - \tilde{\mathbf{u}}||_2 &\leq ||\mathbf{B}^{-1}||_2\Big(||\mathbf{E}_a||_2||\tilde{\mathbf{u}}||_2 + ||\mathbf{e}_f||_2\Big), \\ &\leq \frac{1}{a_{\min}}\Big(\sqrt{K}||\mathbf{E}_a||_\infty||\tilde{\mathbf{u}}||_2 + \sqrt{K}||\mathbf{e}_f||_\infty\Big), \end{aligned} \tag{63}$$

where C depends on K, h_j, $a_{\min}$ and $f(x)$. In addition, we have used the fact that $||\tilde{\mathbf{u}}||_2$ is bounded since there is an isomorphism between $\tilde{\mathbf{u}}$ and $\tilde{u}^{\varepsilon}_{h_j}(x,\omega)$.

Let $\mathbf{\Phi} = \big(\phi^1(x), ..., \phi^K(x)\big)^T$ and $\tilde{\mathbf{\Phi}} = \big(\tilde{\phi}^1(x), ..., \tilde{\phi}^K(x)\big)^T$. We have the estimate

$$\begin{aligned} &||u^{\varepsilon}_h(x,\omega) - \tilde{u}^{\varepsilon}_h(x,\omega)||_{H^1(D)} = ||\mathbf{\Phi}^T\mathbf{u} - \tilde{\mathbf{\Phi}}^T\tilde{\mathbf{u}}||_{H^1(D)}, \\ \leq &||\mathbf{\Phi}^T\mathbf{u} - \tilde{\mathbf{\Phi}}^T\mathbf{u}||_{H^1(D)} + ||\tilde{\mathbf{\Phi}}^T\mathbf{u} - \tilde{\mathbf{\Phi}}^T\tilde{\mathbf{u}}||_{H^1(D)}, \\ \leq &||\mathbf{u}||_2 \max_{1\leq k\leq K} ||\phi^k(x) - \tilde{\phi}^k(x)||_{H^1(D)} + \sqrt{\sum_{k=1}^{K} ||\tilde{\phi}^k||^2_{H^1(D)}}||\mathbf{u} - \tilde{\mathbf{u}}||_2 \end{aligned} \tag{64}$$

Integrating over the random space, we got that

$$\mathbb{E}\left[||u^{\varepsilon}_h(x,\omega) - \tilde{u}^{\varepsilon}_h(x,\omega)||^p_{H^1(D)}\right] \leq C\delta_3^p \tag{65}$$

where C a generic constant depending on K, $a_{\min}$, $a_{\max}$, and $f(x)$. Thus we complete the proof. □

According to Theorem 4.4, we can choose δ_1 and δ_2 (that are controlled by the number of coefficient samples and the number of the POD basis) so that

$$\mathbb{E}\left[||u^{\varepsilon}_{h_j}(x,\omega) - \tilde{u}^{\varepsilon}_{h_j}(x,\omega)||^p_{H^1(D)}\right] \leq Ch_j^p, \quad p = 1, 2. \tag{66}$$

Corollary 4.5. *Suppose $\varepsilon \ll h_j < 1$ and $\varepsilon < ch_j^{3+\eta}$ for some $c > 0$ and $\eta > 0$. We can get the error between the exact solution $u^{\varepsilon}(x,\omega)$ and multiscale reduced basis solution $\tilde{u}^{\varepsilon}_{h_j}(x,\omega)$ as*

$$\mathbb{E}\left[||u^{\varepsilon} - \tilde{u}^{\varepsilon}_h||^p_{H^1(D)}\right] \leq Ch_j^p, \quad p = 1, 2 \tag{67}$$

for some constant C.

Proof. According to (20) and (66) and the fact that when $\varepsilon < ch_j^{3+\eta}$ for some $c > 0$ and $\eta > 0$, $\sqrt{\varepsilon/h_j} = o(h_j)$, we can easily get the result. □

Finally, we can prove the required property that the asymptotic variance reduction between two consecutive coarse grid levels.

Theorem 4.6. *The difference between* $\tilde{u}^{\varepsilon}_{h_j}(x,\omega)$ *and* $\tilde{u}^{\varepsilon}_{h_{j-1}}(x,\omega)$ *can be bounded by*

$$\mathbb{E}\left[||\tilde{u}^{\varepsilon}_{h_j}(x,\omega) - \tilde{u}^{\varepsilon}_{h_{j-1}}(x,\omega)||\right] \leq Ch_j^2. \tag{68}$$

The proof is a simply application of the triangle inequality and Corollary 4.5. The asymptotic variance reduction between two consecutive coarse grid levels can be obtained immediately since $\mathrm{Var}\left(\tilde{u}^{\varepsilon}_{h_j}(x,\omega) - \tilde{u}^{\varepsilon}_{h_{j-1}}(x,\omega)\right) \leq \mathbb{E}\left[||\tilde{u}^{\varepsilon}_{h_j}(x,\omega) - \tilde{u}^{\varepsilon}_{h_{j-1}}(x,\omega)||^2\right]$.

5. Numerical examples

In this section we will present various numerical results to demonstrate the accuracy and efficiency of our proposed MsMLMC method.

5.1. An example with an oscillatory coefficient

We consider the following multiscale elliptic PDE with random coefficient on $D = [0,1] \times [0,1]$:

$$-\nabla \cdot (a^{\varepsilon}(x,y,\omega)\nabla u^{\varepsilon}(x,y)) = f(x,y,\theta), \quad (x,y) \in D, \omega \in \Omega, \tag{69}$$

$$u^{\varepsilon}(x,y,\omega) = 0, \quad (x,y) \in \partial D. \tag{70}$$

The multiscale and random coefficient is given by

$$\begin{aligned} a^{\varepsilon}(x,y,\omega) = 0.1 &+ \frac{\xi_1(\omega)}{2 + P\sin(2\pi(x-y)/\epsilon_1)} + \frac{\xi_2(\omega)}{4 + P(\sin(2\pi x/\epsilon_2) + \sin(2\pi y/\epsilon_2))} \\ &+ \frac{\xi_3(\omega)}{10(2 + P\sin(2\pi(x-0.5)/\epsilon_3))(2 + P\sin(2\pi(y-0.5)/\epsilon_3))}, \end{aligned} \tag{71}$$

where $P = 1.9$, $\epsilon_1 = \frac{1}{3}$, $\epsilon_2 = \frac{1}{27}$ and $\epsilon_3 = \frac{1}{51}$, and $\{\xi_i\}_{i=1}^3$ are independent uniform random variables in $[0,1]$.

In our computations, we use the FEM to discretize the spatial dimension. We choose a 512×512 fine mesh grid to well resolve the spatial dimension of the stochastic solution $u^{\varepsilon}(x,y,\omega)$. Due to the high computational cost, we use the Monte Carlo method with 10^4 samples to calculate the reference solution.

To implement the MsMLMC, the coarsest mesh grid is chosen as 4×4, i.e., $h_0 = \frac{1}{4}$, and $h_j = \frac{h_0}{2^j}$, $j = 1, ..., L$ is the jth level grid size with the coarsening factor 2. The expectation solution on the coarsest grid is obtained by 10^4 Monte Carlo samples, and the expectation of the difference of solutions between levels j and $j-1$, $j = 1, ..., 3$ are obtained by the run-time information. In this example, they are computed by Monte Carlo method with 4029, 1705,

and 861 samples, respectively. When we construct the reduced basis in the offline stage, we use 500 Monte Carlo samples. Because the error in the solution can be corrected in subsequent level, the reduced basis in the current level does not necessarily need to be accurate enough.

The coarsest mesh grid of MLMC is chosen as 52×52, i.e., $h_0 = \frac{1}{52}$, and $h_j = \frac{h_0}{2^j}$, $j = 1, ..., L$ is the jth level grid size with the coarsening factor 2. The Monte Carlo samples on different levels are chosen in the same way as the MsMLMC. In addition, we implement the MLMC with the coarsest mesh grid chosen as 4×4, i.e., $h_0 = \frac{1}{4}$, and $h_j = \frac{h_0}{2^j}$, $j = 1, ..., L$ is the jth level grid size with the coarsening factor 2.

Multiquery results in the online stage. We randomly generate 20 force functions of the form $f(x, y) \in \{\sin(k_i\pi x + \phi_i)\cos(l_i\pi y + \varphi_i)\}_{i=1}^{20}$, where k_i and l_i are uniformly distributed over the interval $[0, 2]$, while ϕ_i and φ_i are uniformly distributed over the interval $[0, 1]$. In Fig.1 and Fig.2, we show the relative errors of the mean and the STD of the solution in L^2 norm and H^1 norm, respectively. The MsMLMC method gives very accurate results. The MLMC method that uses very coarse mesh grids generates a relatively large error. We show only the error in L^2 norm, since the accuracy in the H^1 norm is quite large. If we set the coarsest mesh grids to well resolve the multiscale feature, the MLMC method improves the accuracy, but the computational cost will increase dramatically.

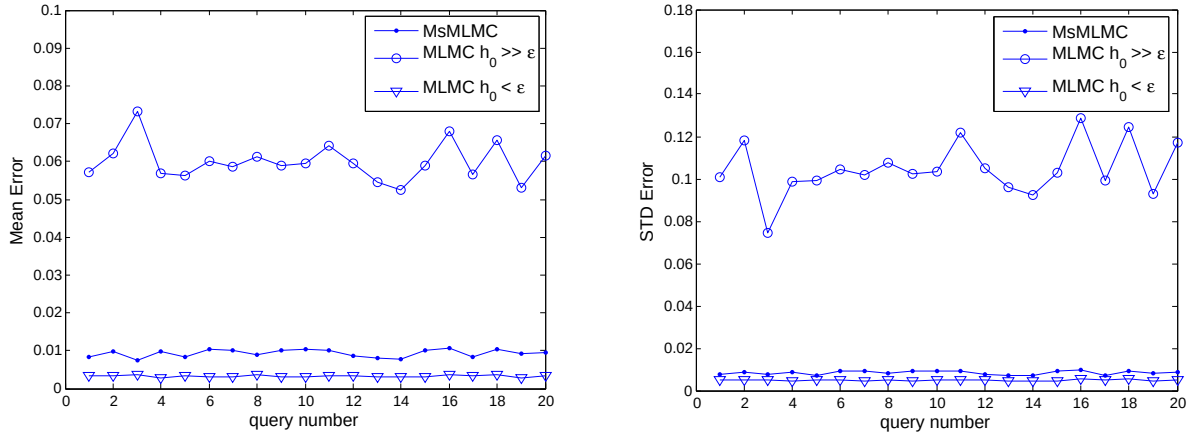

Figure 1: The mean and STD error in L^2 norm.

In Fig.3, we show the mean and STD of the exact and MsMLMC solutions corresponding to $f(x, y) = \sin(1.75\pi x + 0.6)\cos(0.75\pi y + 0.5)$. It can be seen that the mean and STD of the MsMLMC solution match those of the exact solution very well. We also show the contour plot of the mean of the solution in Fig.8. One can see the heterogeneous structures of the multiscale solution. Thanks to the multiscale of our MsMLMC, we can obtain the multiscale feature of the solution on a relative coarse mesh grid, i.e., we can apply the MLMC method without assuming the coarsest grid should resolve the multiscale features.

Variance decay results in the online stage. Fig.5 shows the variance decay result for the solution corresponding to $f(x, y) = \sin(1.75\pi x + 0.6)\cos(0.75\pi y + 0.5)$ by using different methods. The blue line with cross denotes the solution obtained on single level, which indicates

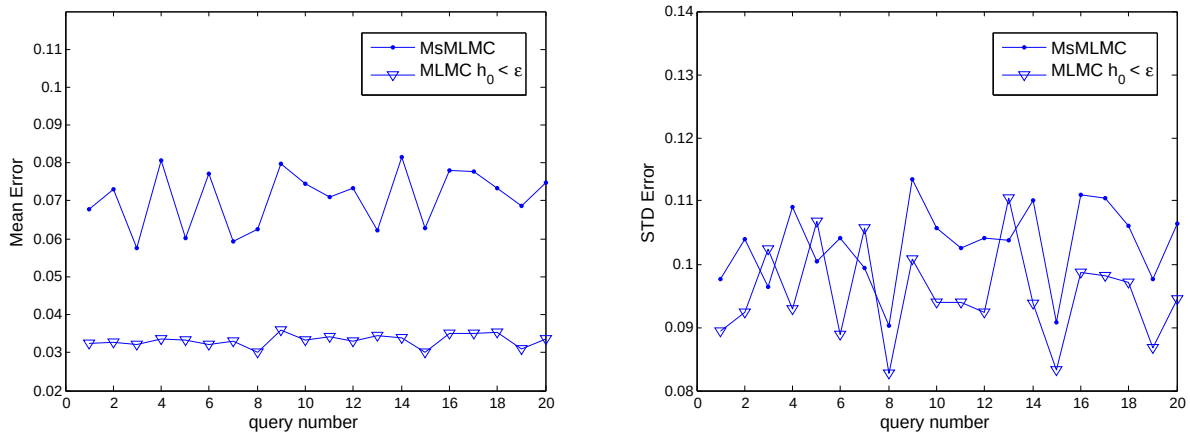

Figure 2: The mean and STD error in H^1 norm.

that variance does not decay. The red line with circle denotes the $\int Var(u_l - u_{l-1})dx$ obtained by MsMLMC. The slope of the line is approximately equal to 3.74, indicating that $\int Var(u_l - u_{l-1})dx \approx Ch^{3.74}$. If we implement the MLMC with the coarsest mesh grid $h_0 \gg \varepsilon$, as shown in the green line with square, the variance does not decay. When we choose the coarsest mesh grid $h_0 < \varepsilon$ (in this case $h_0 = \frac{1}{52}$), as shown in the black line with triangle, the variance decays. However, the computational cost increases dramatically. The slope of the line is approximately equal to 4.35, indicating that $\int Var(u_l - u_{l-1})dx \approx Ch^{4.35}$.

5.2. An example with a high-dimensional random and multiscale coefficient

We consider the RPDE (69) with a high-dimensional random input and multiscale features on $D = [0,1] \times [0,1]$. The coefficient is given by

$$a^{\varepsilon}(x,y,\omega) = \sum_{i=1}^{15} \xi_i(\omega) \frac{2 + P_i \sin(2\pi x/\epsilon_i)}{2 + Q_i \cos(2\pi y/\epsilon_i)}, \tag{72}$$

where $\{\xi_i\}$ are independent uniform random variables in $[0,1]$, P_i, $Q_i \in (1.8, 1.9)$ are randomly generated, and $(\epsilon_1, ..., \epsilon_{15}) = (\frac{1}{n_1}, ..., \frac{1}{n_{15}})$, where the integer $3 \le n_j \le 31$ are randomly generated.

The stochastic collocation method are computationally prohibitive due to the *curse of dimensionality*. We implement our method in the Monte Carlo method setting. We use the standard FEM to discretize the spatial dimension. We choose a 384×384 fine mesh grid to well resolve the spatial dimension of the stochastic solution $u^{\varepsilon}(x,y,\omega)$. Due to the high computational cost, we use the Monte Carlo method with 10^4 samples to calculate the "reference" solution.

To implement the MsMLMC, the coarsest mesh grid of MsMLMC is chosen as 4×4, i.e., $h_0 = \frac{1}{4}$, and $h_j = \frac{h_0}{2^j}$, $j = 1, ..., 3$. Similarly, the coarsest mesh grid of MLMC is chosen as 48×48, i.e., $h_0 = \frac{1}{48}$, and $h_j = \frac{h_0}{2^j}$, $j = 1, ..., 3$. The coarsest grid expectation solution is obtained by 10^4 Monte Carlo samples, and the difference of the expectation solutions between

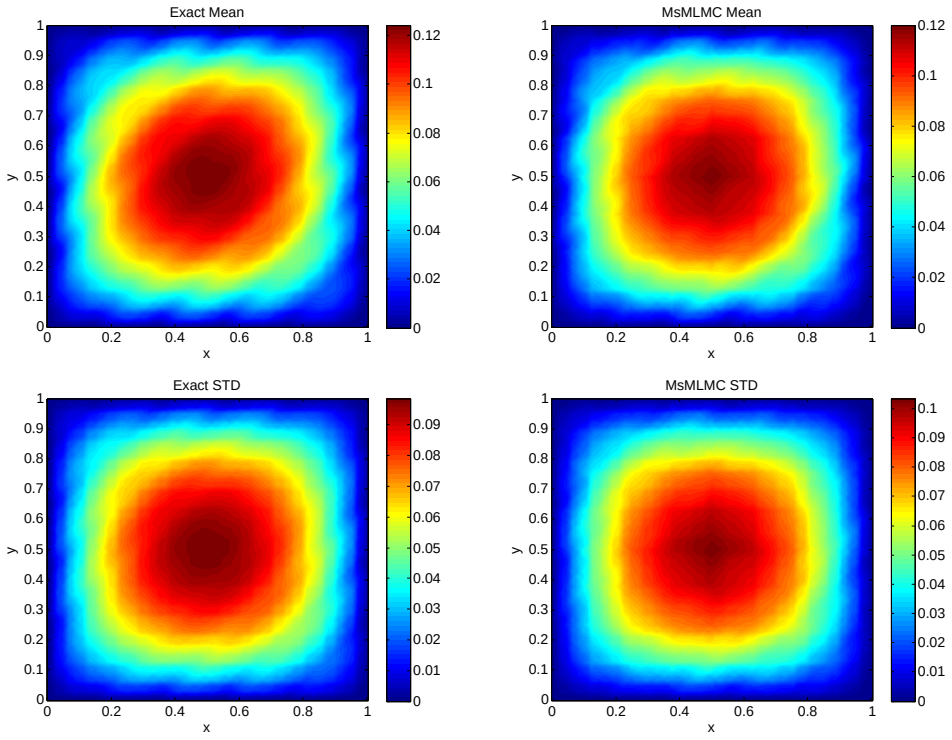

Figure 3: Profiles of the mean and STD solution.

j and $j-1$, $j = 1, ..., 3$ grids are obtained by the run-time information. In this example, they are chosen as 4428, 1960, and 968 respectively.

Multiquery results in the online stage. We randomly generate 20 force functions of the form $f(x, y) \in \{\sin(k_i \pi x + \phi_i) \cos(l_i \pi y + \varphi_i)\}_{i=1}^{20}$, where k_i and l_i are uniformly distributed over the interval $[0, 2]$, while ϕ_i and φ_i are uniformly distributed over the interval $[0, 1]$. In Fig.6, we show the relative errors of the mean and the STD of the solution in L^2 norm. The MsMLMC method gives accurate results. The MLMC method that uses very coarse mesh grids generates very poor results (not shown here). If we set the coarsest mesh grids to well resolve the multiscale feature, the MLMC method improves the accuracy in the mean solution, but the computational cost for the MC method will increase dramatically. In Fig.7, we show the mean of the exact and MsMLMC solutions corresponding to $f(x, y) = \sin(2\pi x + 0.25) \cos(2\pi y + 0.50)$. It can be seen that the mean of the MsMLMC solution match the exact solution very well. We also show the contour plot of the mean of the solution in Fig.8. One can see the heterogeneous structures of the multiscale solution.

Comparison of the computational costs of MsMLMC and MLMC. We follow the same strategy in comparing the computational costs of MC, MLMC and MsMLMC in the multiquery setting. It takes 451.30 and 504.84 seconds for the MC and MLMC method respectively in the offline preparation stage. For the MC method, it will take 33942.57 seconds to solve Eq.(69) with one specific forcing term $f(x, y)$. Thus in a multiquery problem, if we need to solve Eq.(69) with n different forcing term $f(x, y)$, the total computational cost will be $t_1 = 451.30 + 33942.57n$. In the online stage of the MLMC, it takes 3514.94 seconds to

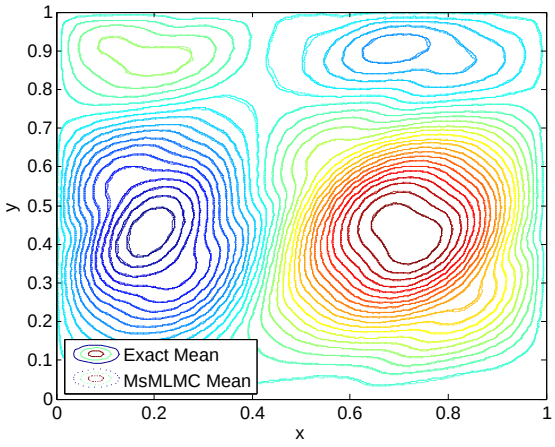

Figure 4: Contour plot of the mean solution.

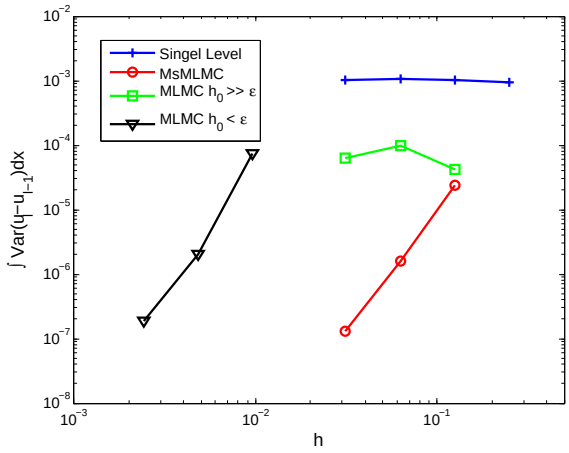

Figure 5: Variance decay of MsMLMC solutions.

compute each query, thus, the total computational cost will be $t_2 = 504.84 + 3514.94n$. In our MsMLMC method, the offline computation costs 5719.13 seconds, which includes the computational time of solving cell problem, constructing the reduced basis, and computing the fixed data structure for the global stiffness matrix. In the online stage of the MsMLMC, it takes 693.91 seconds to compute each query, thus, the total computational cost will be $t_3 = 4036.21 + 693.91n$. We achieve $5.1X$ speedup in the MsMLMC over MLMC in the online stage. We plot the total computational time in Fig.9. One can see that the MsMLMC offers considerable computational savings over the MLMC if we need to solve the original RPDE with more than two different forcing functions. In addition, we can see that MLMC is superior to the MC method. Higher speedup ratio can be achieved for Eq.(69) with smaller multiscale parameters.

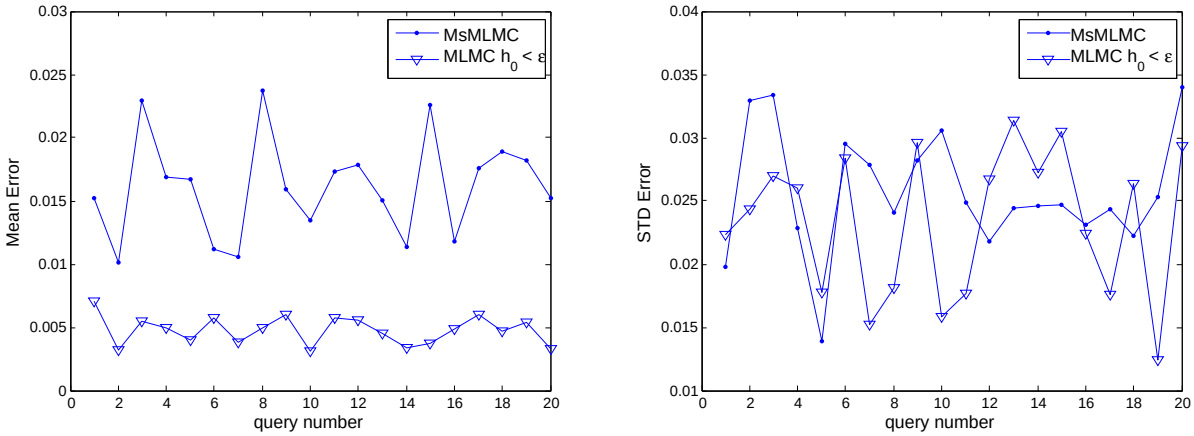

Figure 6: The mean and STD error in L^2 norm.

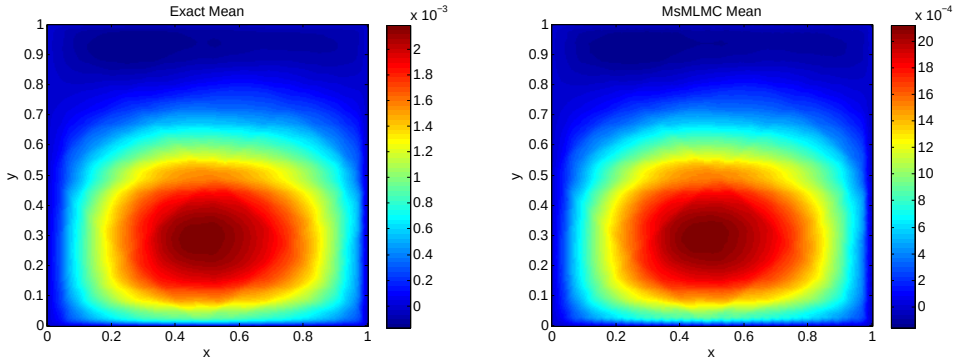

Figure 7: Profiles of the mean solution.

5.3. An example to investigate the computational complexity

Choosing the reduced basis number m and coarsest levels of the MsMLMC and MLMC to obtain an accurate solution is problem dependent. In this subsection, we test a model problem to understand this issue. In addition, we numerically study the computational complexity of the MsMLMC method.

We consider the Eq.(9) on $D = [0,1] \times [0,1]$ with the coefficient given by

$$a^{\varepsilon}(x,y,\omega) = 0.1 + \xi_1(\omega)\frac{2+1.8\sin(\frac{2\pi x}{\epsilon_1})}{2+1.8\sin(\frac{2\pi y}{\epsilon_1})} + \xi_2(\omega)\frac{2+1.8\sin(\frac{2\pi y}{\epsilon_2})}{2+1.8\cos(\frac{2\pi x}{\epsilon_2})} + \xi_3(\omega)\frac{2+1.8\cos(\frac{2\pi x}{\epsilon_3})}{2+1.8\sin(\frac{2\pi y}{\epsilon_3})}, \tag{73}$$

where ϵ_1, ϵ_2, and ϵ_3 are multiscale parameters, and $\{\xi_i\}_{i=1}^3$ are independent uniform random variables in $[0,1]$.

In our computations, we use 2000 Monte Carlo samples to discretize the random space. We use the standard finite element method to discretize the spatial space. The fine mesh grid is chosen as 512×512 to well resolve the spatial dimension of the stochastic solution $u^{\varepsilon}(x,y,\omega)$.

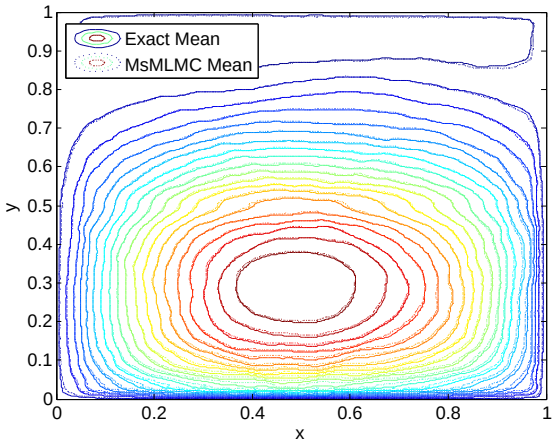

Figure 8: Contour plot of the mean solution.

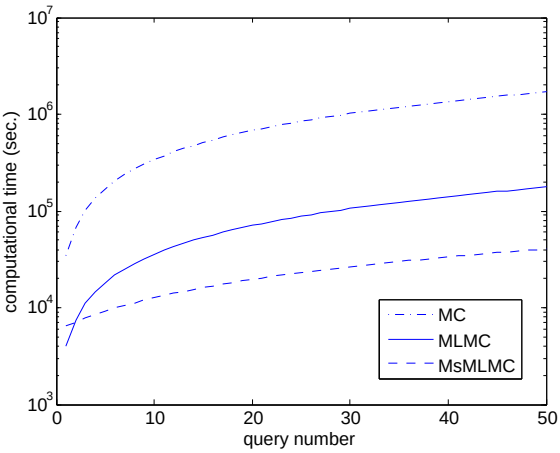

Figure 9: The computation time comparison.

Assume the MsMLMC has four levels, with the mesh size given by $h_c = \frac{1}{4}$, $\frac{1}{8}$, $\frac{1}{16}$, and $\frac{1}{32}$. Then the degrees of freedom (DOF) of the cell problem (14) on each level are approximately given by $N = 16384$, 4096, 1024, and 256, respectively. In Fig.10, we plot the computational time of the linear equation solver on each level, and we find that the computational complexity of solving the linear equation with N unknowns is approximately given by $O(N^{1.3})$. Then in Fig.11, we plot the POD eigenvalues in decreasing order as a function of their index, for different choices of the mesh size. We use the decaying property of eigenvalues to select parameter m, i.e., to select m such that $\frac{\theta_{m+1}}{\theta_1}$ is smaller than some predefined threshold, say, 10^{-4}. We choose $m = 6$ in our numerical test.

We choose four different kinds of multiscale parameters in (73), i.e.,

$$\{(\epsilon_1, \epsilon_2, \epsilon_3) | (\frac{1}{3}, \frac{1}{12}, \frac{1}{21}), (\frac{1}{3}, \frac{1}{17}, \frac{1}{31}), (\frac{1}{3}, \frac{1}{22}, \frac{1}{41}), (\frac{1}{3}, \frac{1}{27}, \frac{1}{51})\}. \tag{74}$$

The coarsest mesh of MsMLMC is chosen as $h_c = \frac{1}{4}$, while the coarsest mesh of MLMC is chosen as $h_c = \frac{1}{N_c}$, with $N_c = [\frac{1}{\epsilon_3}] + 1$. We calculate the computational time of obtaining

one sample solution on the coarsest mesh by the MsMLMC and MLMC, respectively. In Fig.12, we plot the speedup ratio of MsMLMC over MLMC for the four kinds of multiscale parameters defined in (74). In addition, we use these data to interpolate a model for the speedup ratio as a function of the smallest-scale parameter. One can see that small multiscale parameter results in a significant speedup in the MsMLMC over the MLMC, which reveals the power of the multiscale reduced basis. As we see in the plot of Fig.13, the asymptotic variance reduction between two consecutive levels starts when $h_c = \frac{1}{4}$ in our MsMLMC, since the multiscale finite element basis functions already capture the multiscale information of the RPDE solution.

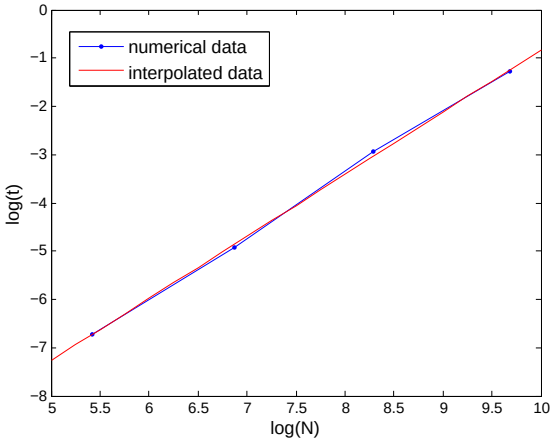

Figure 10: Computational cost of the linear equation solver in the offline stage of MsMLMC. The slope of the interpolated line is $\gamma = 1.3$.

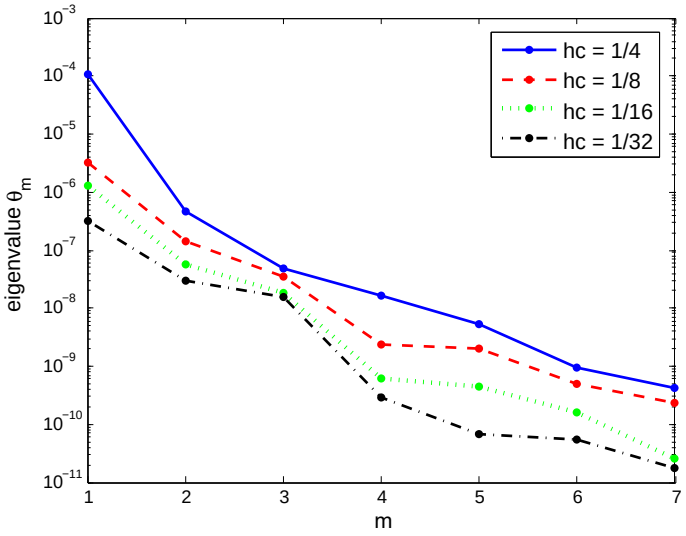

Figure 11: The POD eigenvalues in decreasing order.

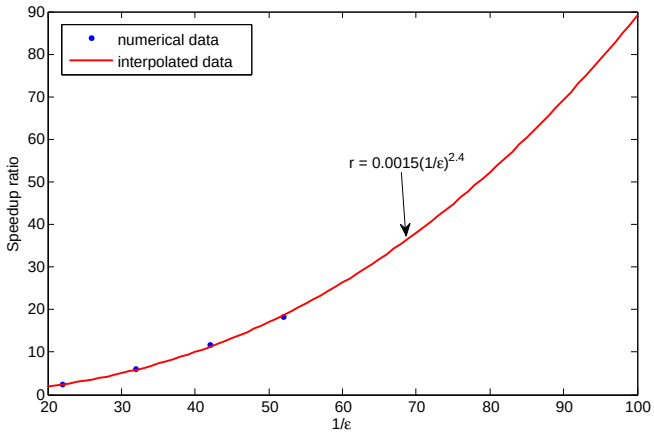

Figure 12: Speedup ratio of MsMLMC over MLMC for different ϵ.

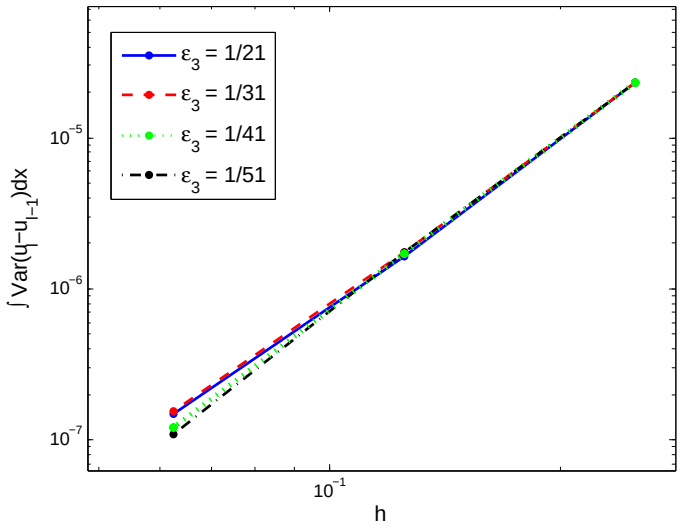

Figure 13: Variance decay of MsMLMC solutions.

6. Conclusion remarks

In this paper, we developed a novel multiscale multilevel Monte Carlo method (MsMLMC) to solve multiscale elliptic PDEs with random coefficients in a multiquery setting. These model problems arise from various applications, such as the heterogeneous porous media flow problem in the water aquifer and oil reservoirs simulation. Our method consists of offline and online stages. In the offline stage, we construct a small number of reduced basis elements within each coarse grid block, which can be used to approximate the multiscale finite element basis. In the online stage, for each new realization of the random coefficient, we can efficiently obtain the corresponding multiscale finite element basis with the help of these reduced basis and solve the multiscale RPDEs on coarse mesh grid. By applying the MLMC method, we can solve the RPDEs efficiently without requiring the coarsest mesh size to be smaller than the

smallest-scale parameter in the coefficient.

We presented numerical examples to demonstrate the accuracy and efficiency of the proposed method. These numerical examples indicate the following advantages of the proposed MsMLMC: (1) By combining the upscaling technique with the MLMC, the MsMLMC provides an effective alternative to solve multiscale RPDEs with desirable accuracy on a coarse grid; (2) The construction of the reduced basis is independent of the forcing functions, thus the MsMLMC can be used for the same multiscale RPDEs with a large number of deterministic forcing functions; (3) Comparing to the other existing numerical solvers, such as the MC and MLMC method, the MsMLMC offers considerable computational savings in solving multiscale RPDEs in a multiquery setting.

Acknowledgements

The research of J. Lyu is partially supported by the Hong Kong PhD Fellowship Scheme. The research of Z. Zhang is supported by the Hong Kong RGC General Research Funds (Projects 27300616, 17300817, and 17300318), National Natural Science Foundation of China (Project 11601457), Seed Funding Programme for Basic Research (HKU), and Basic Research Programme (JCYJ20180307151603959) of The Science, Technology and Innovation Commission of Shenzhen Municipality.

Appendix A. Computational cost analysis of MLMC

Proof. Without loss of generality, we choose $|D| = 1$. For single level MC (SLMC) method, we have the averaged-mean-square error

$$e(Y) = \frac{1}{N} c_1 = \delta^2. \tag{A.1}$$

Hence the number of SLMC simulations is $N = \frac{c_1}{\delta^2}$. Then the computational cost is

$$\mathcal{C}_1 = c_3 N h_L^{-\gamma} = c_1 c_3 h_L^{-\gamma} \delta^{-2}.$$

In practice, the number of simulations N is chosen as integer, which should be the smallest integer greater or equal to N. So without confusion, we treat N as a continuous quantity throughout the proof. The same applies to the implementation of N_k in MLMC.

Now we turn to MLMC and minimize the cost

$$\mathcal{C}_2 = c_3 N_L h_L^{-\gamma} + \sum_{k=0}^{L-1} c_3 (N_{k+1} + N_k) h_k^{-\gamma}$$

subjected to the δ^2-MSE restriction

$$e(Y) = c_2 \sum_{k=1}^{L} \frac{1}{N_k} h_k^{\beta} + \frac{1}{N_0} c_1 = \delta^2. \tag{A.2}$$

For simply notation, we denote

$$V_k = \begin{cases} c_2 h_k^{\beta}, & k \neq 0; \\ c_1, & k = 0, \end{cases} \tag{A.3}$$

and computational cost

$$\tilde{V}_k = \begin{cases} c_3 h_k^{-\gamma} + c_3 h_{k-1}^{-\gamma}, & k \neq 0; \\ c_3 h_0^{-\gamma}, & k = 0. \end{cases} \tag{A.4}$$

Then the δ^2-MSE (A.2) become

$$\sum_{k=0}^{L} \frac{1}{N_k} V_k = \delta^2, \tag{A.5}$$

and the total computational cost of MLMC is

$$\mathcal{C}_2 = \sum_{k=0}^{L} N_k \tilde{V}_k.$$

By the method of Lagrange multiplier, we can minimize the cost above for a fixed (A.5) and conclude that N_k is proportional to $\sqrt{V_k/\tilde{V}_k}$, i.e., $N_k = d\sqrt{V_k/\tilde{V}_k}$, where the constant

$$d = \delta^{-2} \sum_{k=0}^{L} \sqrt{V_k \tilde{V}_k}. \tag{A.6}$$

Hence the computational cost $\mathcal{C}_2$ is

$$\begin{aligned} \mathcal{C}_2 &= \epsilon^{-2} \Big(\sum_{k=0}^{L} \sqrt{V_k \tilde{V}_k} \Big)^2 \\ &= \epsilon^{-2} \Big(\sum_{k=1}^{L} h_k^{\frac{\beta-\gamma}{2}} \sqrt{c_2 c_3 (1 + m^{-\gamma})} + \sqrt{c_1 c_3} h_0^{-\frac{\gamma}{2}} \Big)^2 \\ &= c_3 \epsilon^{-2} \Big(\sum_{k=1}^{L} h_k^{\frac{\beta-\gamma}{2}} \sqrt{c_2 (1 + m^{-\gamma})} + \sqrt{c_1} h_0^{-\frac{\gamma}{2}} \Big)^2. \end{aligned} \tag{A.7}$$

Then the ratio of the computational complexity of MLMC to that of SLMC is

$$\frac{\mathcal{C}_2}{\mathcal{C}_1} = \frac{h_L^{\gamma}}{c_1} \Big(\sum_{k=1}^{L} h_k^{\frac{\beta-\gamma}{2}} \sqrt{c_2 (1 + m^{-\gamma})} + \sqrt{c_1} h_0^{-\frac{\gamma}{2}} \Big)^2. \tag{A.8}$$

Now we separate the discussion. When $\beta = \gamma$, we have

$$\frac{\mathcal{C}_2}{\mathcal{C}_1} = \frac{h_L^{\gamma}}{c_1} \Big(L \sqrt{c_2 (1 + m^{-\gamma})} + \sqrt{c_1} m^{-\frac{\gamma L}{2}} h_L^{-\frac{\gamma}{2}} \Big)^2 \tag{A.9}$$

$$= \Big(L \sqrt{\frac{c_2 (1 + m^{-\gamma})}{c_1}} h_L^{\frac{\gamma}{2}} + m^{-\frac{\gamma L}{2}} \Big)^2 \tag{A.10}$$

Then the optimal level number L, treated as a continuous quantity, is

$$L = -\frac{2}{\gamma \ln m} \ln \Big\{ 2 \sqrt{\frac{c_2(1+m^{-\gamma})}{c_1}} \frac{h_L^{\frac{\gamma}{2}}}{\gamma \ln m} \Big\},$$

and the minimized ratio (A.8) becomes

$$\begin{aligned} \frac{\mathcal{C}_2}{\mathcal{C}_1} &= \frac{c_2(1+m^{-\gamma})}{\gamma^2 c_1 \ln^2 m} h_L^{\gamma} \Big\{ \ln \frac{e^2 c_1 \gamma^2 \ln^2 m}{4 c_2 (1+m^{-\gamma})} - \gamma \ln h_L \Big\}^2 \\ &= \tilde{c}_6 h_L^{\gamma} |\ln h_L|^2 + O(h_L^{\gamma} |\ln h_L|), \end{aligned} \tag{A.11}$$

with

$$\tilde{c}_6 = \frac{c_2(1+m^{-\gamma})}{c_1 \ln^2 m}.$$

When $\beta \neq \gamma$, the ratio becomes:

$$\frac{\mathcal{C}_2}{\mathcal{C}_1} = (c_0 h_L^{\beta/2} - c_0 h_L^{\beta/2} m^{L(\beta-\gamma)/2} + m^{-\gamma L/2})^2. \tag{A.12}$$

The optimal choices of L and the computational cost are

$$L = \frac{2}{\beta \ln m} \ln \Big(\frac{-\gamma h_L^{-\beta/2}}{c_0(\beta-\gamma)} \Big), \tag{A.13}$$

$$\begin{aligned} \mathcal{C}_2 &= \mathcal{C}_1 \Big\{ c_0 h_L^{\beta/2} - \frac{\beta}{\gamma-\beta} \Big(\frac{c_0(\gamma-\beta)}{\gamma} \Big)^{\gamma/\beta} h_L^{\gamma/2} \Big\}^2, \\ &= \mathcal{C}_1 \Big\{ \sqrt{\tilde{c}_7} h_L^{\beta/2} - \sqrt{\tilde{c}_5} h_L^{\gamma/2} \Big\}^2 \end{aligned} \tag{A.14}$$

where

$$c_0 = \sqrt{\frac{c_2(1+m^{-\gamma}) m^{(\gamma-\beta)}}{c_1}} (m^{(\gamma-\beta)/2} - 1)^{-1}, \tag{A.15}$$

$$\tilde{c}_5 = \frac{\beta^2}{(\gamma-\beta)^2} \Big(\frac{c_0(\gamma-\beta)}{\gamma} \Big)^{2\gamma/\beta}, \tag{A.16}$$

$$\tilde{c}_7 = c_0^2. \tag{A.17}$$

In the case of $\beta > \gamma$, the second term of right hand side (A.14) is dominant for $h_L \ll 1$ and the cost of MLMC is

$$\mathcal{C}_2 = \tilde{c}_5 h_L^{\gamma} \mathcal{C}_1 (1 - \sqrt{\frac{\tilde{c}_7}{\tilde{c}_5}} h_L^{(\beta-\gamma)/2})^2 = c_1 c_3 \tilde{c}_5 \delta^{-2} + O(\delta^{-2} h_L^{(\beta-\gamma)/2}).$$

In the case of $\beta < \gamma$, the first term of right hand side (A.14) is dominant for $\epsilon \ll 1$ and the cost of MLMC is

$$\mathcal{C}_2 = \tilde{c}_7 h_L^{\beta} \mathcal{C}_1 (1 - \sqrt{\frac{\tilde{c}_5}{\tilde{c}_7}} h_L^{(\gamma-\beta)/2})^2 = c_1 c_3 \tilde{c}_7 \delta^{-2} h_L^{-(\gamma-\beta)} + O(\delta^{-2} h_L^{-(\gamma-\beta)/2}).$$

□

References

[1] A. Abdulle, A. Barth, and C. Schwab. Multilevel Monte Carlo methods for stochastic elliptic multiscale PDEs. *Multiscale Modeling & Simulation*, 11(4):1033–1070, 2013.

[2] I. Babuska, R. Tempone, and G. Zouraris. Galerkin finite element approximations of stochastic elliptic partial differential equations. *SIAM J. Numer. Anal.*, 42:800–825, 2004.

[3] A. Barth, C. Schwab, and N. Zollinger. Multi-level Monte Carlo finite element method for elliptic PDEs with stochastic coefficients. *Numerische Mathematik*, 119(1):123–161, 2011.

[4] G. Berkooz, P. Holmes, and J. L. Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual review of fluid mechanics*, 25(1):539–575, 1993.

[5] M. Bieri, R. Andreev, and C. Schwab. Sparse tensor discretization of elliptic SPDEs. *SIAM Journal on Scientific Computing*, 31(6):4281–4304, 2010.

[6] M. Bieri and C. Schwab. Sparse high order fem for elliptic SPDEs. *Computer Methods in Applied Mechanics and Engineering*, 198(13-14):1149–1170, 2009.

[7] M. Cheng, T. Y. Hou, M. Yan, and Z. Zhang. A data-driven stochastic method for elliptic PDEs with random coefficients. *SIAM J. UQ*, 1:452–493, 2013.

[8] M. Cheng, T. Y. Hou, and Z. Zhang. A dynamically bi-orthogonal method for stochastic partial differential equations I: derivation and algorithms. *J. Comput. Phys.*, 242:843–868, 2013.

[9] M. Cheng, T. Y. Hou, and Z. Zhang. A dynamically bi-orthogonal method for stochastic partial differential equations II: adaptivity and generalizations. *J. Comput. Phys.*, 242:753–776, 2013.

[10] A. Cliffe, M. Giles, R. Scheichl, and A. Teckentrup. Multilevel Monte Carlo methods and applications to elliptic PDEs with random coefficients. *Computing and Visualization in Science*, 14(1):3, 2011.

[11] A. Doostan and H. Owhadi. A non-adapted sparse approximation of PDEs with stochastic inputs. *Journal of Computational Physics*, 230(8):3015–3034, 2011.

[12] Y. Efendiev, C. Kronsbein, and F. Legoll. Multilevel Monte Carlo approaches for numerical homogenization. *Multiscale Modeling & Simulation*, 13(4):1107–1135, 2015.

[13] Y. R. Efendiev, T. Y. Hou, and X. Wu. Convergence of a nonconforming multiscale finite element method. *SIAM Journal on Numerical Analysis*, 37(3):888–910, 2000.

[14] H. Elman and Q. Liao. Reduced basis collocation methods for partial differential equations with random coefficients. *SIAM/ASA J. on Uncertainty Quantification*, 1(1):192–217, 2013.

[15] R. Ghanem and P. Spanos. *Stochastic finite elements: a spectral approach.* Springer-Verlag, New York, 1991.

[16] M. B. Giles. Multilevel Monte Carlo path simulation. *Operations Research*, 56:607–617, 2008.

[17] A. Haji-Ali, F. Nobile, and R. Tempone. Multi-index Monte Carlo: when sparsity meets sampling. *Numerische Mathematik*, 132(4):767–806, 2016.

[18] S. Heinrich. Multilevel monte carlo methods. *LSSC*, 1:58–67, 2001.

[19] P. Holmes, J. Lumley, and G. Berkooz. *Turbulence, coherent structures, dynamical systems and symmetry.* Cambridge University Press, 1998.

[20] T. Hou, D. Ma, and Z. Zhang. A model reduction method for multiscale elliptic PDEs with random coefficients using an optimization approach. *Multiscale Modeling & Simulation*, 17(2):826–853, 2019.

[21] T. Y. Hou, W. Luo, B. Rozovskii, and H. M. Zhou. Wiener chaos expansions and numerical solutions of randomly forced equations of fluid mechanics. *J. Comput. Phys.*, 216:687–706, 2006.

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

[23] T. Y. Hou, X. Wu, and Z. Cai. Convergence of a multiscale finite element method for elliptic problems with rapidly oscillating coefficients. *Mathematics of Computation*, 68(227):913–943, 1999.

[24] T. Y. Hou, X. Wu, and Y. Zhang. Removing the cell resonance error in the multiscale finite element method via a Petrov-Galerkin formulation. *Communications in Mathematical Sciences*, 2(2):185–205, 2004.

[25] O. Maıtre, O. M. Knio, H. Najm, and R. G. Ghanem. A stochastic projection method for fluid flow: I. basic formulation. *J. Comput. Phys.*, 173:481–511, 2001.

[26] L. Mathelin and K. Gallivan. A compressed sensing approach for partial differential equations with random input data. *Communications in computational physics*, 12(4):919–954, 2012.

[27] H. G. Matthies and A. Keese. Galerkin methods for linear and nonlinear elliptic stochastic partial differential equations. *Comput. Method Appl. Mech. Eng.*, 194:1295–1331, 2005.

[28] H. N. Najm. Uncertainty quantification and polynomial chaos techniques in computational fluid dynamics. *Annual Review of Fluid Mechanics*, 41:35–52, 2009.

[29] N. Nguyen. A multiscale reduced-basis method for parametrized elliptic partial differential equations with multiple scales. *Journal of Computational Physics*, 227(23):9807–9822, 2008.

[30] G. Rozza, D. B. Huynh, and A. T. Patera. Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations. *Archives of Computational Methods in Engineering*, 15(3):1–47, 2007.

[31] T. Sapsis and P. Lermusiaux. Dynamically orthogonal field equations for continuous stochastic dynamical systems. *Physica D: Nonlinear Phenomena*, 238:2347–2360, 2009.

[32] L. Sirovich. Turbulence and the dynamics of coherent structures. I. Coherent structures. *Quarterly of applied mathematics*, 45(3):561–571, 1987.

[33] T. Tang and T. Zhou. Convergence analysis for stochastic collocation methods to scalar hyperbolic equations with a random wave speed. *Commun. Comput. Phys*, 8(1):226–248, 2010.

[34] A. Teckentrup, P. Jantsch, C. Webster, and M. Gunzburger. A multilevel stochastic collocation method for partial differential equations with random input data. *SIAM/ASA Journal on Uncertainty Quantification*, 3(1):1046–1074, 2015.

[35] X. L. Wan and G. Karniadakis. Multi-element generalized polynomial chaos for arbitrary probability measures. *SIAM J. Sci. Comp.*, 28:901–928, 2006.

[36] K. Willcox and J. Peraire. Balanced model reduction via the proper orthogonal decomposition. *AIAA journal*, 40(11):2323–2330, 2002.

[37] D. Xiu. Fast numerical methods for stochastic computations: a review. *Commun. Comput. Phys.*, 5:242–272, 2009.

[38] D. Xiu and G. Karniadakis. Modeling uncertainty in flow simulations via generalized polynomial chaos. *J. Comput. Phys.*, 187:137–167, 2003.

[39] Z. Zhang, M. Ci, and T. Y. Hou. A multiscale data-driven stochastic method for elliptic PDEs with random coefficients. *SIAM Multiscale Model. Simul.*, 13:173–204, 2015.

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