Simulation of random fields on random domains

Zhibao Zheng[a,*], Marcos Valdebenito[b], Michael Beer[c,d,e], Udo Nackenhorst[a]

[a]*Leibniz Universität Hannover, Institute of Mechanics and Computational Mechanics & International Research Training Group 2657, Appelstraße 9a, 30167, Hannover, Germany*
[b]*TU Dortmund University, Chair for Reliability Engineering, Leonhard-Euler-Straße 5, 44227 Dortmund, Germany*
[c]*Leibniz Universität Hannover, Institute for Risk and Reliability, Callinstraße 34, 30167 Hannover, Germany*
[d]*University of Liverpool, Institute for Risk and Uncertainty and School of Engineering, Peach Street, Liverpool L69 7ZF, UK*
[e]*Tongji University, International Joint Research Center for Resilient Infrastructure & International Joint Research Center for Engineering Reliability and Stochastic Mechanics, 1239 Siping Road, Shanghai 200092, China*

Abstract

This paper focuses on the simulation of random fields on random domains. This is an important class of problems in fields such as topology optimization and multiphase material analysis. However, there is still a lack of effective methods to simulate this kind of random fields. To this end, we extend the classical Karhunen–Loève expansion (KLE) to this class of problems, and we denote this extension as stochastic Karhunen–Loève expansion (SKLE). We present three numerical algorithms for solving the stochastic integral equations arising in the SKLE. The first algorithm is an extension of the classical Monte Carlo simulation (MCS), which is used to solve the stochastic integral equation on each sampled domain. However, such approach demands remeshing each sampled domain and solving the corresponding integral equation, which can become computationally very demanding. In the second algorithm, a domain transformation is used to map the random domain into a reference domain, and only one mesh for the reference domain is required. In this way, remeshing different sample realizations of the random domain is avoided and much computational effort is thus saved. MCS is then adopted to solve the corresponding stochastic integral equation. Further, to avoid the computational effort of MCS, the third algorithm proposed in this contribution involves a reduced-order method to solve the stochastic integral equation efficiently. In this third algorithm, stochastic eigenvectors are represented as a sum of products of unknown random variables and deterministic vectors, where the deterministic vectors are efficiently computed by solving deterministic eigenvalue problems. The random variables and stochastic eigenvalues that

appear in this third algorithm are calculated by a reduced-order stochastic eigenvalue problem constructed by the obtained deterministic vectors. Based on the obtained stochastic eigenvectors, the target random field is then simulated and reformulated as a classical KLE-like representation. Finally, three numerical examples are presented to demonstrate the performance of the proposed methods.
Keywords: Random fields; Random domains; Stochastic Karhunen–Loève expansion; Domain transformation; Stochastic eigenvalue equations;

1. Introduction

The considerable influence of uncertainties on system behavior has led to the development of dedicated numerical methods for its characterization, propagation and quantification [1, 2, 3]. Uncertainties may appear, for example, in material parameters, boundary conditions or loadings of mechanical systems. Moreover, these uncertainties may affect the physical domain associated with a given system. For example, uncertainty in manufacturing processes may lead to variability with respect to the physical dimensions of a component, leading to a random domain problem. The focus of this work lies precisely on modeling uncertainties associated with random domains as well as random quantities associated with random domains, which implies dealing with random fields defined on random domains. Such a class of uncertainty model can be found in many practical engineering problems, e.g., topology optimization, material properties with spatial variability in multiphase materials with random interfaces, etc [4, 5, 6]. In this paper, we develop effective numerical methods for simulating the random fields on random domains, which are termed as RFRD for brevity in the following. Although problems of uncertainty quantification defined on random domains have received attention in the past decades, most applications only involve random variables [4, 5, 7, 8, 9, 10] and do not capture the spatial variability. In fact, to the authors' best knowledge, there are no contributions that focus on the simulation of RFRD. The closest attempt to address this problem is reported in [11], which provides a weak form that integrates

*Corresponding author
Email address: zhibao.zheng@ibnm.uni-hannover.de (Zhibao Zheng)

spatially random material properties into stochastic finite element equations involving random domains. However, that contribution performs neither a detailed feasibility analysis nor any numerical experiments that illustrate its application to RFRD problems. Simulation of random fields on random domains can be a challenging task. Indeed, geometric uncertainties make the discretization of complex domains not easily accessible. Thus, discretization-based numerical techniques for simulating random fields may not be directly applicable. Hence, dedicated methods need to be developed for the simulation of RFRD. As an additional challenge, it should be noted that uncertainties associated with such random fields are multi-source, as they may appear as the combined effect from both the random domain and one (or more) random quantity on the random domain. The latter implies that, potentially, multiple sources of uncertainties can be coupled in the simulation. Evidently, it is not easy to capture multiple sources of uncertainty accurately and efficiently. In this paper, we first extend the classical Karhunen–Loève expansion (KLE) [12] to a stochastic Karhunen–Loève expansion (SKLE) to characterize random fields on random domains, where a key issue is to solve a homogeneous stochastic Fredholm integral equation of the second kind over the random domain. For this purpose, we present a direct Monte Carlo simulation (MCS) scheme, a domain transformation-based MCS procedure and a stochastic eigenequation-based approach to solve the stochastic integral equation (SIE).

The MCS and its extensions [9] are direct and effective methods to deal with random domains. Hence, the classical MCS is adopted as the first method to simulate RFRD in this paper. In this method, the random domain is meshed for each sample realization, and classical numerical techniques are then used to solve the corresponding integral equation associated with each sampled mesh. The method is straightforward to implement and it can be coupled, for example, with existing finite element codes without any modification. However, to generate a large number of random samples of RFRD, we have to remesh each sample realization of the random domain and solve the corresponding deterministic integral equation, which can become numerically demanding, especially for large-scale RFRD. Also, given that the random samples of RFRD are generated on different sampled domains and meshes, it is not a simple matter to perform the postprocessing and apply the simulated RFRD to subsequent stochastic analyses.

To avoid remeshing the domain for each sample realization, we develop a second approach

that couples domain transformation with MCS. In fact, domain transformation, as introduced in [7, 13, 10], allows to map the random domain into a reference domain. In our previous work [10], similar domain transformation methods are used to solve partial differential equations (PDEs) on random domains, where the coefficients of PDEs are considered as deterministic values or random variables, but not random fields. In the domain transformation method, a boundary-conforming coordinate system between the random domain and the reference domain is produced via a Laplace equation with random boundary conditions. In this way, the integral equation defined on the random domain arising in SKLE is transformed into an SIE defined on the deterministic reference domain. All SIEs can be solved on the reference domain and the computational effort for remeshing sampled domains is saved. Then, the classical MCS is used to solve the SIE on the reference domain. Further, postprocessing in this method is easily performed on the reference domain, which avoids the difficulty encountered in the classical MCS and can be readily applied to subsequent stochastic analyses. However, the computational cost of solving the SIE on the reference domain with MCS can be still very expensive, especially for large-scale problems.

To avoid the expensive computational effort of solving the SIE, we further propose a third approach which is a stochastic eigenequation-based simulation algorithm. The domain transformation approach presented above is first used to transform the random domain into a reference domain and generate the corresponding SIE on the reference domain. An efficient reduced-order stochastic eigenvalue algorithm proposed in our previous work [14] is adopted to solve the SIE. In this method, stochastic eigenvectors are approximated by a sum of the products of unknown random variables and deterministic vectors. The deterministic vectors are solved efficiently by a few deterministic eigenvalue equations that are obtained by applying the stochastic Galerkin procedure to the original stochastic problem. Based on the obtained deterministic vectors, a reduced-order stochastic eigenvalue equation is constructed and used to solve stochastic eigenvalues of the original stochastic eigenequation and the random variables in the approximation of stochastic eigenvectors. Furthermore, this method generates sample descriptions for the stochastic eigenvalues and semi-explicit representations for the stochastic eigenvectors, which thus provides a semi-explicit representation of RFRD. In this way, two computationally demanding aspects, i.e. remeshing the random domain and solving the SIE, are avoided successfully and replaced by an approach

involving very low numerical effort.

The rest of the paper is organized as follows. Section 2 presents the first algorithm of this work, which comprises an SKLE derived from the classical KLE and a direct MCS is used to solve the associated SIE arising from the SKLE. In Section 3, a domain transformation is formulated to map the random domain into a reference domain. Following that, the SIE is solved by using a domain transformation-based MCS, leading to the second algorithm that is introduced in this work. Further, a third simulation algorithm combining the domain transformation and a reduced-order method for solving stochastic eigenequations is presented in Section 4. To demonstrate the performance of the proposed methods, three numerical examples, including a one-dimensional beam with random length, two-dimensional multiphase material with a random interface and three-dimensional femur implant with random prosthesis position, are presented and analyzed in Section 5. Conclusions and discussions follow in Section 6.

2. Simulating RFRD via stochastic Karhunen–Loève expansion

In this paper, we consider the random field $\omega\left(\mathbf{x}\left(\theta_{\mathcal{D}}\right),\theta_q\right)$, $\mathbf{x}\left(\theta_{\mathcal{D}}\right)\in\mathbb{R}^d$, $\theta_{\mathcal{D}}\in\Theta_{\mathcal{D}}$, $\theta_q\in\Theta_q$ defined on a random domain $\mathcal{D}\left(\theta_{\mathcal{D}}\right)\subset\mathbb{R}^d$, where $d=1,2,3$ is the spatial dimension, $\Theta_{\mathcal{D}}$ and Θ_q denote the spaces of elementary events that are used to describe the geometrical randomness associated with the random domain and the randomness associated with quantities acting on the random domain, e.g. material properties and external forces, etc. We assume that $\theta_{\mathcal{D}}$ and θ_q are mutually independent random events. Thus, $\omega\left(\mathbf{x}\left(\theta_{\mathcal{D}}\right),\theta_q\right)$ can be thought of as a composite random field or a random parameterized random field, i.e. involving multiple sources of randomness, which is different from classical random fields that usually involve a single source of randomness. The mean function and the covariance function of $\omega\left(\mathbf{x}\left(\theta_{\mathcal{D}}\right),\theta_q\right)$ are given by $\overline{\omega}\left(\mathbf{x}\left(\theta_{\mathcal{D}}\right)\right)$ and $C_{\omega\omega}\left(\mathbf{x}_1\left(\theta_{\mathcal{D}}\right),\mathbf{x}_2\left(\theta_{\mathcal{D}}\right)\right)$, $\mathbf{x}_1\left(\theta_{\mathcal{D}}\right),\mathbf{x}_2\left(\theta_{\mathcal{D}}\right)\in\mathcal{D}\left(\theta_{\mathcal{D}}\right)$, respectively. In this work, we only consider Gaussian random fields, i.e. the marginal distribution of $\omega\left(\mathbf{x}\left(\theta_{\mathcal{D}}\right),\theta_q\right)$ is Gaussian. It is noted that the mean function $\overline{\omega}\left(\mathbf{x}\left(\theta_{\mathcal{D}}\right)\right)$ is still a stochastic function related to $\theta_{\mathcal{D}}$, which represents the mean function of the quantities related to θ_q with respect to a given domain realization $\mathcal{D}\left(\theta_{\mathcal{D}}\right)$. The domain mean function, i.e. the mean function with respect to $\theta_{\mathcal{D}}$, is independent of the random event θ_q and will be discussed in detail in the subsequent section.

To simulate the random field $\omega\left(\mathbf{x}\left(\theta_{\mathcal{D}}\right), \theta_q\right)$, we extend the classical KLE to a stochastic case comprising a random domain. Specifically, we represent the random field $\omega\left(\mathbf{x}\left(\theta_{\mathcal{D}}\right), \theta_q\right)$ by the following SKLE

$$\omega\left(\mathbf{x}\left(\theta_{\mathcal{D}}\right), \theta_q\right) = \overline{\omega}\left(\mathbf{x}\left(\theta_{\mathcal{D}}\right)\right) + \sum_{i=1}^{r} \xi_i\left(\theta_q\right) \sqrt{\lambda_i\left(\theta_{\mathcal{D}}\right)} f_i\left(\mathbf{x}\left(\theta_{\mathcal{D}}\right)\right), \quad \mathbf{x}\left(\theta_{\mathcal{D}}\right) \in \mathcal{D}\left(\theta_{\mathcal{D}}\right), \tag{1}$$

where r is the number of retained terms in the KLE expansion, $\left\{\xi_i\left(\theta_q\right)\right\}_{i=1}^{r}$ are independent standard Gaussian random variables, and $\left\{\lambda_i\left(\theta_{\mathcal{D}}\right), f_i\left(\mathbf{x}\left(\theta_{\mathcal{D}}\right)\right)\right\}_{i=1}^{r}$ are stochastic eigenvalues and stochastic eigenvectors of the covariance function $C_{\omega\omega}\left(\mathbf{x}_1\left(\theta_{\mathcal{D}}\right), \mathbf{x}_2\left(\theta_{\mathcal{D}}\right)\right)$. They are solved by the following homogeneous stochastic Fredholm integral equation of the second kind

$$\int_{\mathcal{D}(\theta_{\mathcal{D}})} C_{\omega\omega}\left(\mathbf{x}_1\left(\theta_{\mathcal{D}}\right), \mathbf{x}_2\left(\theta_{\mathcal{D}}\right)\right) f_i\left(\mathbf{x}_1\left(\theta_{\mathcal{D}}\right)\right) \mathrm{d}\mathbf{x}_1\left(\theta_{\mathcal{D}}\right) = \lambda_i\left(\theta_{\mathcal{D}}\right) f_i\left(\mathbf{x}_2\left(\theta_{\mathcal{D}}\right)\right). \tag{2}$$

The stochastic solution of Eq. (2) is crucial to the numerical implementation of the SKLE. However, classical methods may not work well on solving Eq. (2) since both the integral kernel $C_{\omega\omega}\left(\mathbf{x}_1\left(\theta_{\mathcal{D}}\right), \mathbf{x}_2\left(\theta_{\mathcal{D}}\right)\right)$ and the integral domain $\mathcal{D}\left(\theta_{\mathcal{D}}\right)$ are random. In this paper, we develop three numerical strategies to solve Eq. (2) and then implement Eq. (1) based on the stochastic solution of Eq. (2). Without loss of generality, we let the mean function $\overline{\omega}\left(\mathbf{x}\left(\theta_{\mathcal{D}}\right)\right) = 0$. Otherwise, we can use existing numerical methods to simulate the nonzero random field $\overline{\omega}\left(\mathbf{x}\left(\theta_{\mathcal{D}}\right)\right)$, such as the classical KLE and the Polynomial Chaos expansion [12, 15].

2.1. Direct Monte Carlo simulation

A straightforward method to solve Eq. (2) is the classical MCS, which is denoted here as direct MCS (DMCS). To implement the method, we solve Eq. (2) on the deterministic domain $\mathcal{D}\left(\theta_{\mathcal{D}}^{(j)}\right)$ for each sample realization $\theta_{\mathcal{D}}^{(j)}$, $j = 1, \cdots, n_s$, where n_s is the number of sample realizations. The discretization of each sampled domain $\mathcal{D}\left(\theta_{\mathcal{D}}^{(j)}\right)$ can be performed using classical numerical techniques, e.g. the finite element method and the finite volume method, which usually depends on the methods used for subsequent uncertainty propagation. For instance, the finite element discretization should be adopted if the stochastic finite element method is used to solve the subsequent stochastic problem. The discretized covariance matrix of the covariance function $C_{\omega\omega}\left(\mathbf{x}_1\left(\theta_{\mathcal{D}}\right), \mathbf{x}_2\left(\theta_{\mathcal{D}}\right)\right)$ is then given as $\mathbf{C}_{\mathrm{DMC}}\left(\theta_{\mathcal{D}}^{(j)}\right) \in \mathbb{R}^{n_j \times n_j}$ based on the domain discretization,

where n_j is the number of discretized nodes. Eq. (2) can thus be solved by the following eigenvalue equation

$$\mathbf{C}_{\mathrm{DMC}}\left(\theta_{\mathcal{D}}^{(j)}\right)\mathbf{f}_{\mathrm{DMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right)=\lambda_{\mathrm{DMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right)\mathbf{f}_{\mathrm{DMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right)\quad\text{on}\quad\mathcal{D}\left(\theta_{\mathcal{D}}^{(j)}\right),\tag{3}$$

where $\lambda_{\mathrm{DMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right)$ and $\mathbf{f}_{\mathrm{DMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right)$ are the eigenvalues and the eigenvectors of the sampled covariance matrix $\mathbf{C}_{\mathrm{DMC}}\left(\theta_{\mathcal{D}}^{(j)}\right)$. Thus, Eq. (1) is implemented as

$$\boldsymbol{\omega}_{\mathrm{DMC}}\left(\theta_{\mathcal{D}}^{(j)},\theta_q\right)=\sum_{i=1}^{r}\xi_i\left(\theta_q\right)\sqrt{\lambda_{\mathrm{DMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right)}\mathbf{f}_{\mathrm{DMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right)\quad\text{on}\quad\mathcal{D}\left(\theta_{\mathcal{D}}^{(j)}\right),\tag{4}$$

which actually corresponds to a classical random field representation of the random event θ_q conditional on the sample realization $\theta_{\mathcal{D}}^{(j)}$. A sample realization of the conditional random field $\omega_{\mathrm{DMC}}\left(\mathbf{x}\left(\theta_{\mathcal{D}}^{(j)}\right),\theta_q\right)$ is given by

$$\boldsymbol{\omega}_{\mathrm{DMC}}\left(\theta_{\mathcal{D}}^{(j)},\theta_q^{(j)}\right)=\sum_{i=1}^{r}\xi_i\left(\theta_q^{(j)}\right)\sqrt{\lambda_{\mathrm{DMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right)}\mathbf{f}_{\mathrm{DMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right),\tag{5}$$

where $\theta_q^{(j)}$ is the sample realization of θ_q and the samples $\theta_{\mathcal{D}}^{(j)}$ and $\theta_q^{(j)}$ are also independent due to the independence of $\theta_{\mathcal{D}}$ and θ_q.

Algorithm 1 DMCS algorithm for simulating RFRD

1: **for** $j=1,\cdots,n_s$ **do**
2: Generate the sampled domain $\mathcal{D}\left(\theta_{\mathcal{D}}^{(j)}\right)$
3: Mesh the domain $\mathcal{D}\left(\theta_{\mathcal{D}}^{(j)}\right)$
4: Calculate $\left\{\lambda_{\mathrm{DMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right),\mathbf{f}_{\mathrm{DMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right)\right\}_{i=1}^{r}$ by solving Eq. (3) on $\mathcal{D}\left(\theta_{\mathcal{D}}^{(j)}\right)$
5: **end**
6: Generate random sample vectors $\widehat{\xi_i}\left(\widehat{\boldsymbol{\theta}_q}\right)\in\mathbb{R}^{n_s}$, $i=1,\cdots,r$
7: Generate sample realizations $\boldsymbol{\omega}_{\mathrm{DMC}}\left(\theta_{\mathcal{D}}^{(j)},\theta_q^{(j)}\right)$, $j=1,\cdots,n_s$ of the random field using Eq. (5)

A step-by-step implementation of the DMCS is detailed in Algorithm 1. A loop from step 1 to step 5 is used to perform MCS on each sampled domain $\mathcal{D}\left(\theta_{\mathcal{D}}^{(j)}\right)$. For each sample realization $\theta_{\mathcal{D}}^{(j)}$, the deterministic domain $\mathcal{D}\left(\theta_{\mathcal{D}}^{(j)}\right)$ is generated and meshed in step 2 and step 3, respectively. In this work, we adopt the finite element method for the mesh discretization, but other discretization can also be used for this purpose. The sample realization $\left\{\lambda_{\mathrm{DMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right),\mathbf{f}_{\mathrm{DMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right)\right\}_{i=1}^{r}$ of the

stochastic eigenvalues and the stochastic eigenvectors are then calculated in step 4. After the loop, the random sample vector $\widehat{\xi_i}\left(\widehat{\boldsymbol{\theta}}_q\right) = \left[\xi_i\left(\theta_q^{(1)}\right), \cdot, \xi_i\left(\theta_q^{(n_s)}\right)\right]^{\mathrm{T}} \in \mathbb{R}^{n_s}$ of the random variable $\xi_i\left(\theta_q\right)$ is generated in step 6. Combining the sample realizations $\left\{\lambda_{\mathrm{DMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right), \mathbf{f}_{\mathrm{DMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right)\right\}$ and $\xi_i\left(\theta_q^{(j)}\right)$ we can generate the sample realizations $\boldsymbol{\omega}_{\mathrm{DMC}}\left(\theta_{\mathcal{D}}^{(j)}, \theta_q^{(j)}\right)$, $j = 1, \cdots, n_s$ in step 7.

The above DMCS is straightforward to implement and can take advantage of existing codes. However, DMCS possesses drawbacks regarding its associated numerical effort. On one hand, it is necessary to remesh each sampled domain $\mathcal{D}\left(\theta_{\mathcal{D}}^{(j)}\right)$ and solve the corresponding eigenvalue equation (3). The computational burden associated with the latter step may be considerable, particularly to achieve a highly accurate stochastic solution. On the other hand, it is not a simple matter to perform postprocessing of the random field due to different meshes generated for different sampled domains.

3. Domain transformation-based simulation of RFRD

The computational burden of Algorithm 1 stems from two aspects: domain remeshing in step 3 and solution of stochastic eigenvalue equations in step 4. In this section, we focus on the first issue and discuss an efficient approach to avoid remeshing the random domain.

3.1. Transformation of random domain into a reference domain

To avoid remeshing the sampled domain $\mathcal{D}\left(\theta_{\mathcal{D}}^{(j)}\right)$ for each sample realization $\theta_{\mathcal{D}}^{(j)}$, we transform the random domain $\mathcal{D}\left(\theta_{\mathcal{D}}\right) \subset \mathbb{R}^d$ into a reference domain $\overline{\mathcal{D}} \subset \mathbb{R}^d$. To this end, we map each random point $\mathbf{x}\left(\theta_{\mathcal{D}}\right) = \left(x_1\left(\theta_{\mathcal{D}}\right), \cdots, x_d\left(\theta_{\mathcal{D}}\right)\right) \in \mathcal{D}\left(\theta_{\mathcal{D}}\right)$ to a deterministic point $\overline{\mathbf{x}} = \left(\overline{x}_1, \cdots, \overline{x}_d\right) \in \overline{\mathcal{D}}$ via the transformation $\overline{\mathbf{x}} = \mathscr{M}^{-1}\left(\mathbf{x}\left(\theta_{\mathcal{D}}\right), \theta_{\mathcal{D}}\right)$ and represent the random coordinate $\mathbf{x}\left(\theta_{\mathcal{D}}\right)$ using the deterministic coordinate $\overline{\mathbf{x}}$ via the inverse transformation $\mathbf{x}\left(\theta_{\mathcal{D}}\right) = \mathscr{M}\left(\overline{\mathbf{x}}, \theta_{\mathcal{D}}\right)$, where $\mathscr{M}\left(\cdot, \theta_{\mathcal{D}}\right)$ represents a mapping operator and $\mathscr{M}^{-1}\left(\cdot, \theta_{\mathcal{D}}\right)$ is its inverse operator. The finite element mesh associated with the random domain is thus calculated by applying the aforementioned inverse transformation with respect to the reference domain. In practice, the mean of the random domain, i.e. the mean function associated with the random event $\theta_{\mathcal{D}}$, is chosen as the reference domain. However, any reference domain that ensures the well-posedness of the transformation can be used. Note that several different methods can be used to construct the mapping operator $\mathscr{M}\left(\cdot, \theta_{\mathcal{D}}\right)$ [13].

In this work, we adopt a transformation based on the Laplace equation [7, 10, 13], which is expressed as

$$\Delta x_i\left(\theta_{\mathcal{D}}\right) = 0 \quad \text{in} \quad \overline{\mathcal{D}} \tag{6}$$

for $i = 1, \cdots, d$, with the following boundary constraints

$$x_i\left(\theta_{\mathcal{D}}\right)|_{\Gamma_j} = \mathscr{M}_{i,j}\left(\overline{\mathbf{x}}\big|_{\Gamma_j}, \theta_{\mathcal{D}}\right) \tag{7}$$

for $j = 1, \cdots, b$, where $\Delta = \sum_{j=1}^{d} \frac{\partial^2}{\partial \overline{x}_j^2}$ denotes the Laplace operator, $\overline{\mathbf{x}}\big|_{\Gamma_j} = \left[\overline{x}_1|_{\Gamma_j}, \cdots, \overline{x}_d|_{\Gamma_j}\right] \in \mathbb{R}^d$ and $\mathbf{x}\left(\theta_{\mathcal{D}}\right)|_{\Gamma_j} = \left[x_1\left(\theta_{\mathcal{D}}\right)|_{\Gamma_j}, \cdots, x_d\left(\theta_{\mathcal{D}}\right)|_{\Gamma_j}\right] \in \mathbb{R}^d$ represent the reference (deterministic) and random coordinates on the boundary Γ_j, respectively, $\mathscr{M}_{i,j}\left(\cdot, \theta_{\mathcal{D}}\right)$ represents the mapping of the i-th coordinate on the boundary Γ_j, and b is the total number of domain boundaries.

To solve Eq. (6), we discretize the reference domain $\overline{\mathcal{D}}$ by the finite element method. Discretized nodal coordinates corresponding to the i-th spatial dimension $\overline{x}_i$, $i = 1, \cdots, d$ in the reference domain are denoted as $\overline{\mathbf{X}}_i \in \mathbb{R}^n$, where n denotes the number of nodes associated with the finite element discretization. Similarly, the corresponding discretized nodal coordinates associated with the i-th spatial dimension $x_i\left(\theta_{\mathcal{D}}\right)$, $i = 1, \cdots, d$ in the random domain are denoted as $\mathbf{X}_i\left(\theta_{\mathcal{D}}\right) \in \mathbb{R}^n$, which is also the stochastic solution with respect to the nodal coordinates of the reference domain. Thus, the discrete counterpart of Eq. (6) is

$$\begin{bmatrix} \mathbf{K}_{\mathcal{D},0,0} & \mathbf{K}_{\mathcal{D},0,1} & \cdots & \mathbf{K}_{\mathcal{D},0,b} \\ \mathbf{K}_{\mathcal{D},1,0} & \mathbf{K}_{\mathcal{D},1,1} & \cdots & \mathbf{K}_{\mathcal{D},1,b} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{K}_{\mathcal{D},b,0} & \mathbf{K}_{\mathcal{D},b,1} & \cdots & \mathbf{K}_{\mathcal{D},b,b} \end{bmatrix} \begin{bmatrix} \mathbf{X}_i\left(\theta_{\mathcal{D}}\right)|_{\mathcal{D}(\theta_{\mathcal{D}})} \\ \mathscr{M}_{i,1}\left(\overline{\mathbf{X}}\big|_{\Gamma_1}, \theta_{\mathcal{D}}\right) \\ \vdots \\ \mathscr{M}_{i,b}\left(\overline{\mathbf{X}}\big|_{\Gamma_b}, \theta_{\mathcal{D}}\right) \end{bmatrix} = 0 \tag{8}$$

for $i = 1, \cdots d$, where $\mathscr{M}_{i,j}\left(\overline{\mathbf{X}}\big|_{\Gamma_j}, \theta_{\mathcal{D}}\right) \in \mathbb{R}^{n_j}$, $j = 1, \cdots, b$ are calculated using Eq. (7) and represent the stochastic coordinates of nodes on the boundary Γ_j, n_j is the number of nodes on the boundary Γ_j, $\mathbf{X}_i\left(\theta_{\mathcal{D}}\right)|_{\mathcal{D}(\theta_{\mathcal{D}})} \in \mathbb{R}^{n_0}$ are the unknown stochastic coordinates of nodes which are not located on the constraint boundaries, n_0 is the number of nodes which are not located on the constraint boundaries, $\mathbf{K}_{\mathcal{D},i,j} \in \mathbb{R}^{n_i \times n_j}$ are submatrices obtained by the finite element discretization of the Laplace operator. The total number n of nodes of the finite element mesh, the number n_0 of nodes not located on the constraint boundaries and the number n_j of nodes located on the constraint boundaries

fulfill the relationship $n = n_0 + \sum_{j=1}^{b} n_j$. Note that Eq. (8) can be interpreted as a deterministic finite element equation with stochastic Dirichlet boundary conditions. The explicit form of the stochastic solution $\mathbf{X}_i(\theta_{\mathcal{D}})|_{\mathcal{D}(\theta_{\mathcal{D}})}$ is obtained by solving the first set of equations contained in Eq. (8), that is

$$\mathbf{X}_i(\theta_{\mathcal{D}})|_{\mathcal{D}(\theta_{\mathcal{D}})} = \sum_{j=1}^{b} \widetilde{\mathbf{K}}_{\mathcal{D},j} \mathscr{M}_{i,j}\left(\overline{\mathbf{X}}\big|_{\Gamma_j}, \theta_{\mathcal{D}}\right) \tag{9}$$

for $i = 1, \cdots d$, where the matrices appearing in the last equation are defined as $\widetilde{\mathbf{K}}_{\mathcal{D},j} = -\mathbf{K}_{\mathcal{D},0,0}^{-1}\mathbf{K}_{\mathcal{D},0,j} \in \mathbb{R}^{n_0 \times n_j}$.

In this way, the random domain can be described based on the reference domain while taking into account the random event $\theta_{\mathcal{D}}$. We only need to mesh the domain once and Eq. (1) and Eq. (2) are solved on the reference domain only. The validation on the well-posedness of the coordinate transformation as cast in Eq. (6) to Eq. (9) can be found in [5, 13]. Further, if the above transformation does not work for some cases because, e.g. one or more elements flip because of large mesh deformations, more advanced methods can be adopted to overcome these difficulties [16, 17]. In addition, it may be beneficial to consider advanced finite element methods to deal with large deformations, such as the arbitrary Lagrangian–Eulerian approach [18, 19]. However, these methods are beyond the scope of this paper and will be analyzed in follow-up studies.

Algorithm 2 Algorithm for generating samples of the random domain

1: Choose a reference domain and generate its finite element mesh
2: Assemble the deterministic matrices $\mathbf{K}_{\mathcal{D},0,j}$, $j = 0, 1, \cdots, n_b$ in Eq. (8)
3: Initialize the stochastic solutions $\widehat{\mathbf{X}}_i\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\Big|_{\mathcal{D}(\theta_{\mathcal{D}})} = \mathbf{0} \in \mathbb{R}^{n_0 \times n_s}$, $i = 1, \cdots, d$
4: **for** The domain boundary $j = 1, \cdots, b$ **do**
5: Calculate the matrices $\widetilde{\mathbf{K}}_{\mathcal{D},j} = -\mathbf{K}_{\mathcal{D},0,0}^{-1}\mathbf{K}_{\mathcal{D},0,j} \in \mathbb{R}^{n_0 \times n_j}$
6: **for** The spatial dimension $i = 1, \cdots, d$ **do**
7: Calculate the coordinate transformation $\mathscr{M}_{i,j}\left(\overline{\mathbf{X}}\big|_{\Gamma_j}, \widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right) \in \mathbb{R}^{n_j \times n_s}$ on the boundary Γ_j
8: Update the stochastic solution $\widehat{\mathbf{X}}_i\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\Big|_{\mathcal{D}(\theta_{\mathcal{D}})} = \widehat{\mathbf{X}}_i\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\Big|_{\mathcal{D}(\theta_{\mathcal{D}})} + \widetilde{\mathbf{K}}_{\mathcal{D},j}\mathscr{M}_{i,j}\left(\overline{\mathbf{X}}\big|_{\Gamma_j}, \widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)$
9: **end**
10: **end**

The above domain transformation described in Eq. (6) to Eq. (9) is summarized in Algorithm 2. A reference domain is chosen and meshed in step 1. The submatrices $\mathbf{K}_{\mathcal{D},0,j}$ are assembled in step 2. It is noted that only $b+1$ submatrices need to be assembled, which saves computational effort and storage memory compared to a full assembly. Following that, we introduce a loop from step 3 to step 10 to implement Eq. (9). The stochastic solutions $\widehat{\mathbf{X}}_i\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\Big|_{\mathcal{D}(\theta_{\mathcal{D}})} = \mathbf{0} \in \mathbb{R}^{n_0 \times n_s}$, $i = 1, \cdots, d$ are initialized in step 3, where $\widehat{\mathbf{X}}_i\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\Big|_{\mathcal{D}(\theta_{\mathcal{D}})}$ represents the sample realization of the random vector $\mathbf{X}_i(\theta_{\mathcal{D}})|_{\mathcal{D}(\theta_{\mathcal{D}})} \in \mathbb{R}^{n_0}$, i.e. each column of $\widehat{\mathbf{X}}_i\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\Big|_{\mathcal{D}(\theta_{\mathcal{D}})}$ represents the sample realizations of the i-th nodal coordinate. Note that in subsequent sections of this paper, we use $\widehat{\square}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)$ to denote a set of sample realizations of the random quantity $\square(\theta_{\mathcal{D}})$, where $\square(\theta_{\mathcal{D}})$ is an abstract symbol, which can be a random variable, a random vector or a random matrix in practice. In step 5, the matrices $\widetilde{\mathbf{K}}_{\mathcal{D},j}$ are calculated by solving a set of linear equations $\mathbf{K}_{\mathcal{D},0,0}\widetilde{\mathbf{K}}_{\mathcal{D},j} = -\mathbf{K}_{\mathcal{D},0,j}$, which can be implemented efficiently by use of existing numerical solvers. The boundary transformation $\mathscr{M}_{i,j}\left(\overline{\mathbf{X}}\big|_{\Gamma_j}, \boldsymbol{\theta}_{\mathcal{D}}\right) \in \mathbb{R}^{n_j \times n_s}$ is also executed in a non-intrusive way in step 7. Combining the above steps, the stochastic solutions $\widehat{\mathbf{X}}_i\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\Big|_{\mathcal{D}(\theta_{\mathcal{D}})}$ are updated in step 8 until all domain boundaries and spatial dimensions are traversed.

3.2. Random domain transformation-based covariance matrix assembly

Based on the random coordinates $\mathbf{X}(\theta_{\mathcal{D}}) = [\mathbf{X}_1(\theta_{\mathcal{D}}), \cdots, \mathbf{X}_d(\theta_{\mathcal{D}})] \in \mathbb{R}^{n \times d}$ obtained by Algorithm 2, we can assemble the covariance matrix $\mathbf{C}(\theta_{\mathcal{D}})$ without remeshing. To this end, we first calculate the element at row i and column j of the matrix $\mathbf{C}(\theta_{\mathcal{D}})$

$$\mathbf{C}_{ij}(\theta_{\mathcal{D}}) = C_{\omega\omega}\left(\mathbf{X}^{(i)}(\theta_{\mathcal{D}}), \mathbf{X}^{(j)}(\theta_{\mathcal{D}})\right), \tag{10}$$

where $\mathbf{X}^{(i)}(\theta_{\mathcal{D}})$ and $\mathbf{X}^{(j)}(\theta_{\mathcal{D}})$ are the random coordinates of the discretized nodes i and j, respectively. The covariance matrix is then given by $\mathbf{C}(\theta_{\mathcal{D}}) = \left[\mathbf{C}_{ij}(\theta_{\mathcal{D}})\right]_{i,j=1,\cdots,n} \in \mathbb{R}^{n \times n}$. Eq. (2) is thus discretized and solved by the following stochastic eigenvalue equation

$$\mathbf{C}(\theta_{\mathcal{D}})\,\mathbf{f}_i(\theta_{\mathcal{D}}) = \lambda_i(\theta_{\mathcal{D}})\,\mathbf{f}_i(\theta_{\mathcal{D}}). \tag{11}$$

We can reformulate Eq. (1) using $\{\lambda_i(\theta_{\mathcal{D}}), \mathbf{f}_i(\theta_{\mathcal{D}})\}_{i=1}^{r}$ obtained by Eq. (11). However, it may not be straightforward to solve Eq. (11) efficiently and accurately. We will propose two methods for this purpose in sections 3.3 and 4.

3.3. Domain transformation-based MCS

In this section, we develop a domain transformation-based MCS (TMCS) to simulate RFRD. Specifically, we adopt Algorithm 2 to calculate the random domain transformation and assemble the covariance matrix $\mathbf{C}_{\mathrm{TMC}}(\theta_{\mathcal{D}})$. The MCS is then used to solve the stochastic eigenvalue equation (11), which corresponds to

$$\mathbf{C}_{\mathrm{TMC}}\left(\theta_{\mathcal{D}}^{(j)}\right)\mathbf{f}_{\mathrm{TMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right)=\lambda_{\mathrm{TMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right)\mathbf{f}_{\mathrm{TMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right) \tag{12}$$

for each random sample realization $\theta_{\mathcal{D}}^{(j)}$, $j=1,\cdots,n_s$. Thus, the sample realization of the random field $\omega_{\mathrm{TMC}}\left(\mathbf{x}(\theta_{\mathcal{D}}),\theta_q\right)$ is given by

$$\boldsymbol{\omega}_{\mathrm{TMC}}\left(\theta_{\mathcal{D}}^{(j)},\theta_q^{(j)}\right)=\sum_{i=1}^{r}\xi_i\left(\theta_q^{(j)}\right)\sqrt{\lambda_{\mathrm{TMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right)}\mathbf{f}_{\mathrm{TMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right). \tag{13}$$

Algorithm 3 TMCS algorithm for simulating random fields on random domains

1: Calculate n_s sample realizations of the random coordinates $\widehat{\mathbf{X}}\left(\widehat{\boldsymbol{\theta}_{\mathcal{D}}}\right)$ by Algorithm 2
2: **for** $j=1,\cdots,n_s$ **do**
3: Assemble the covariance matrix $\mathbf{C}_{\mathrm{TMC}}\left(\theta_{\mathcal{D}}^{(j)}\right)\in\mathbb{R}^{n\times n}$
4: Calculate the pairs $\left\{\lambda_{\mathrm{TMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right),\mathbf{f}_{\mathrm{TMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right)\right\}_{i=1}^{r}$ by solving Eq. (12)
5: **end**
6: Generate the random sample vectors $\widehat{\xi_i}\left(\widehat{\boldsymbol{\theta}_q}\right)\in\mathbb{R}^{n_s}$, $i=1,\cdots,r$
7: Generate the random samples $\boldsymbol{\omega}_{\mathrm{TMC}}\left(\theta_{\mathcal{D}}^{(j)},\theta_q^{(j)}\right)$, $j=1,\cdots,n_s$ using Eq. (13)

The above domain transformation-based MCS for simulating RFRD is described in Algorithm 3. Algorithm 2 is adopted to calculate the random coordinates $\mathbf{X}(\theta_{\mathcal{D}})$ of the random domain in step 1. A loop of MCS is implemented from step 2 to step 5. For each sample realization $\theta_{\mathcal{D}}^{(j)}$, the covariance matrix $\mathbf{C}_{\mathrm{TMC}}\left(\theta_{\mathcal{D}}^{(j)}\right)$ is assembled in step 3 and the corresponding eigenequation (12) is solved in step 4. Finally, the sample realization $\boldsymbol{\omega}_{\mathrm{DMC}}\left(\theta_{\mathcal{D}}^{(j)},\theta_q^{(j)}\right)$ of the random field is generated in step 7 by combining the random samples $\left\{\lambda_{\mathrm{TMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right),\mathbf{f}_{\mathrm{TMC},i}\left(\theta_{\mathcal{D}}^{(j)}\right)\right\}$ obtained in step 4 and $\xi_i\left(\theta_q^{(j)}\right)$ obtained in step 6.

4. Stochastic eigenequation-based simulation of RFRD

Algorithm 3 provides an effective method to avoid remeshing random domains. However, it may be demanding to solve Eq. (11) for all n_s random sample realizations, especially for large-scale stochastic problems. In this section, we present an efficient method to solve Eq. (11) and a novel algorithm for simulating RFRD is also proposed.

4.1. Solution of stochastic eigenvalue equation

In this section, we adopt an efficient numerical method proposed in [14] to efficiently solve stochastic eigenvalue equation (11). In this method, the stochastic eigenvector $\mathbf{f}_i(\theta_{\mathcal{D}})$ is approximated as

$$\mathbf{f}_i(\theta_{\mathcal{D}}) \approx \sum_{k=1}^{q} \phi_{ik}(\theta_{\mathcal{D}})\,\mathbf{g}_k = \mathbf{G}\boldsymbol{\phi}_i(\theta_{\mathcal{D}}), \tag{14}$$

where $\{\mathbf{g}_k \in \mathbb{R}^n\}_{k=1}^q$ are deterministic vectors, $\mathbf{G} = \left[\mathbf{g}_1, \cdots, \mathbf{g}_q\right] \in \mathbb{R}^{n\times q}$ is a deterministic matrix, $\{\phi_{ik}(\theta_{\mathcal{D}}) \in \mathbb{R}\}_{k=1}^q$ are scalar random variables and $\boldsymbol{\phi}_i(\theta_{\mathcal{D}}) = \left[\phi_{i1}(\theta_{\mathcal{D}}), \cdots, \phi_{iq}(\theta_{\mathcal{D}})\right]^{\mathrm{T}} \in \mathbb{R}^q$ is a random variable vector. All of them are unknown a priori and need to be solved. To this end, we introduce the following stochastic residual

$$\mathcal{R}_j(\theta_{\mathcal{D}}) = \mathbf{C}(\theta_{\mathcal{D}})\,\mathbf{g}_j - \lambda_j(\theta_{\mathcal{D}})\,\mathbf{g}_j, \tag{15}$$

where the random variables $\{\phi_{ik}(\theta_{\mathcal{D}})\}_{k=1}^q$ given in Eq. (14) are not involved and will be calculated in the subsequent iterative process. More details of this setting can be found in [14], e.g. convergence and optimal approximation, etc. Applying the stochastic Galerkin procedure (that is, $\mathbb{E}\left\{\lambda_j(\theta_{\mathcal{D}})\,\mathcal{R}_j(\theta_{\mathcal{D}})\right\} = 0$, where $\mathbb{E}\{\cdot\}$ is the expectation operator) and the deterministic Galerkin procedure (that is, $\mathbf{g}_j^{\mathrm{T}}\mathcal{R}_j(\theta_{\mathcal{D}}) = 0$) to Eq. (15) one after the other, we have the following alternating iteration

$$\mathbf{C}_{d,j}\mathbf{g}_j = \lambda_{d,j}\mathbf{g}_j, \tag{16}$$

$$\mathbf{g}_j^{\mathrm{T}}\mathbf{C}(\theta_{\mathcal{D}})\,\mathbf{g}_j = \lambda_j(\theta_{\mathcal{D}})\,\mathbf{g}_j^{\mathrm{T}}\mathbf{g}_j, \tag{17}$$

where the deterministic matrix $\mathbf{C}_{d,j} \in \mathbb{R}^{n\times n}$ and the scalar quantity $\lambda_{d,j}$ are

$$\mathbf{C}_{d,j} = \mathbb{E}\left\{\lambda_j(\theta_{\mathcal{D}})\,\mathbf{C}(\theta_{\mathcal{D}})\right\}, \quad \lambda_{d,j} = \mathbb{E}\left\{\lambda_j^2(\theta_{\mathcal{D}})\right\}. \tag{18}$$

For a known random variable $\lambda_j(\theta_{\mathcal{D}})$ (or given an initial value), Eq. (16) is a deterministic eigenvalue equation that can be efficiently solved by existing numerical techniques. According to the theory of the classical KLE, only the first few largest eigenvectors need to be solved. In this paper, the power iteration [20] is adopted for this purpose. In practical numerical implementations, $\lambda_{d,j}$ is not calculated since its value does not affect the calculation of the vector $\mathbf{g}_j$. Further, we enforce the condition that the vector $\mathbf{g}_j$ must be orthogonal with respect to the obtained vectors $\{\mathbf{g}_i\}_{i=1}^{j-1}$. The Gram-Schmidt orthonormalization is adopted here, which corresponds to

$$\mathbf{g}_j = \mathbf{g}_j - \sum_{i=1}^{j-1} \frac{\mathbf{g}_j^{\mathrm{T}}\mathbf{g}_i}{\mathbf{g}_i^{\mathrm{T}}\mathbf{g}_i}\mathbf{g}_i, \quad \mathbf{g}_j^{\mathrm{T}}\mathbf{g}_j = 1. \tag{19}$$

where the last equality imposes that the vector $\mathbf{g}_j$ possesses unit L_2 norm. Based on the known vector $\mathbf{g}_j$, the random variable $\lambda_j(\theta_{\mathcal{D}})$ can be easily solved by Eq. (17). In this paper, we use a non-intrusive method to solve Eq. (17)

$$\widehat{\lambda}_j\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right) = \mathbf{g}_j^{\mathrm{T}}\widehat{\mathbf{C}}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\mathbf{g}_j \in \mathbb{R}^{n_s}, \tag{20}$$

where $\widehat{\mathbf{C}}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right) \in \mathbb{R}^{n\times n\times n_s}$ is a third-order tensor. In practice, Eq. (20) is implemented via $\lambda_j\left(\theta_{\mathcal{D}}^{(i)}\right) = \mathbf{g}_j^{\mathrm{T}}\mathbf{C}\left(\theta_{\mathcal{D}}^{(i)}\right)\mathbf{g}_j$ for each sample realization $\theta_{\mathcal{D}}^{(i)}$, $i = 1, \cdots, n_s$. In this way, Eq. (20) provides sample realizations of the random variable $\lambda_j(\theta_{\mathcal{D}})$. Statistical methods can be used to calculate probability characteristics of the random variable $\lambda_j(\theta_{\mathcal{D}})$ from the random samples $\widehat{\lambda}_j\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)$. Repeated solution of Eq. (16) and Eq. (17) is performed until both the deterministic vector $\mathbf{g}_j$ and the random variable $\lambda_j(\theta_{\mathcal{D}})$ converge. Afterwards, the same iteration as described previously is used to calculate the next pair $\left\{\lambda_{j+1}(\theta_{\mathcal{D}}), \mathbf{g}_{j+1}\right\}$.

Further, we use the deterministic matrix $\mathbf{G} = \left[\mathbf{g}_1, \cdots, \mathbf{g}_q\right] \in \mathbb{R}^{n\times q}$ to solve the random variable vector $\boldsymbol{\phi}_i(\theta_{\mathcal{D}}) = \left[\phi_{i1}(\theta_{\mathcal{D}}), \cdots, \phi_{iq}(\theta_{\mathcal{D}})\right]^{\mathrm{T}}$. It is noted that the calculation of the components $\mathbf{g}_j$, $j = 1, \cdots, q$ of the matrix $\mathbf{G}$ is carried out only once, thus the proposed method is efficient. Substituting the matrix $\mathbf{G}$ into Eq. (11) we have

$$\mathbf{C}(\theta_{\mathcal{D}})\,\mathbf{G}\boldsymbol{\phi}_i(\theta_{\mathcal{D}}) = \lambda_i(\theta_{\mathcal{D}})\,\mathbf{G}\boldsymbol{\phi}_i(\theta_{\mathcal{D}}), \tag{21}$$

which is transformed into a reduced-order stochastic eigenvalue equation by multiplying the matrix $\mathbf{G}^{\mathrm{T}}$ on both sides

$$\mathbf{c}_r\left(\theta_{\mathcal{D}}\right)\boldsymbol{\phi}_i\left(\theta_{\mathcal{D}}\right)=\lambda_i\left(\theta_{\mathcal{D}}\right)\boldsymbol{\phi}_i\left(\theta_{\mathcal{D}}\right), \tag{22}$$

where the reduced-order stochastic matrix $\mathbf{c}_r\left(\theta_{\mathcal{D}}\right)$ is given by

$$\mathbf{c}_r\left(\theta_{\mathcal{D}}\right)=\mathbf{G}^{\mathrm{T}}\mathbf{C}\left(\theta_{\mathcal{D}}\right)\mathbf{G}\in\mathbb{R}^{q\times q}, \tag{23}$$

which has a small size $q \ll n$. Eq. (22) can thus be cheaply solved by existing numerical methods. In this paper, we adopt the classical MCS due to its dimensionality independence, high accuracy and easy implementation, which corresponds to solving the following deterministic reduced-order eigenvalue equation

$$\mathbf{c}_r\left(\theta_{\mathcal{D}}^{(j)}\right)\boldsymbol{\phi}_i\left(\theta_{\mathcal{D}}^{(j)}\right)=\lambda_i\left(\theta_{\mathcal{D}}^{(j)}\right)\boldsymbol{\phi}_i\left(\theta_{\mathcal{D}}^{(j)}\right) \tag{24}$$

for all random sample realizations $\theta_{\mathcal{D}}^{(j)}$, $j=1,\cdots,n_s$.

4.2. Representation of RFRD

Substituting the expansion Eq. (14) of stochastic eigenvectors into Eq. (1), we can obtain a new stochastic eigenvalue-based SKLE for the random field $\boldsymbol{\omega}\left(\theta_{\mathcal{D}},\theta_q\right)$

$$\boldsymbol{\omega}_{\mathrm{SE}}\left(\theta_{\mathcal{D}},\theta_q\right)=\sum_{i=1}^{r}\sum_{k=1}^{q}\xi_i\left(\theta_q\right)\sqrt{\lambda_i\left(\theta_{\mathcal{D}}\right)}\phi_{ik}\left(\theta_{\mathcal{D}}\right)\mathbf{g}_k, \tag{25}$$

$$=\sum_{k=1}^{q}\eta_k\left(\theta_{\mathcal{D}},\theta_q\right)\mathbf{g}_k=\mathbf{G}\boldsymbol{\eta}\left(\theta_{\mathcal{D}},\theta_q\right), \tag{26}$$

where the random variable vector $\boldsymbol{\eta}\left(\theta_{\mathcal{D}},\theta_q\right)=\left[\eta_1\left(\theta_{\mathcal{D}},\theta_q\right),\cdots,\eta_q\left(\theta_{\mathcal{D}},\theta_q\right)\right]^{\mathrm{T}}\in\mathbb{R}^q$, and the random variables $\eta_k\left(\theta_{\mathcal{D}},\theta_q\right)=\sum_{i=1}^{r}\xi_i\left(\theta_q\right)\sqrt{\lambda_i\left(\theta_{\mathcal{D}}\right)}\phi_{ik}\left(\theta_{\mathcal{D}}\right)$ are related to both $\theta_{\mathcal{D}}$ and θ_q. Thus, they are considered as composite random variables. In this way, the proposed method provides a classical KLE-like representation through Eq. (26), i.e. decomposing random fields into deterministic and stochastic spaces. All random sources related to $\theta_{\mathcal{D}}$ and θ_q are embedded into the random variables $\eta_k\left(\theta_{\mathcal{D}},\theta_q\right)$. We discuss several probabilistic properties of $\eta_k\left(\theta_{\mathcal{D}},\theta_q\right)$ in the following.

The mean values of $\eta_k\left(\theta_{\mathcal{D}},\theta_q\right)$ with respect to $\theta_{\mathcal{D}}$ and θ_q are given by

$$\overline{\eta}_k\left(\theta_{\mathcal{D}}\right) = \mathbb{E}_{\theta_q}\left\{\eta_k\left(\theta_{\mathcal{D}}, \theta_q\right)\right\} = 0, \tag{27}$$

$$\overline{\eta}_k\left(\theta_q\right) = \mathbb{E}_{\theta_{\mathcal{D}}}\left\{\eta_k\left(\theta_{\mathcal{D}}, \theta_q\right)\right\} = \sum_{i=1}^{r} h_{ik}\xi_i\left(\theta_q\right) \sim \mathcal{G}\left(0, \sum_{i=1}^{r} h_{ik}^2\right), \tag{28}$$

$$\overline{\eta}_k = \mathbb{E}_{\theta_{\mathcal{D}}\theta_q}\left\{\eta_k\left(\theta_{\mathcal{D}}, \theta_q\right)\right\} = \mathbb{E}_{\theta_q\theta_{\mathcal{D}}}\left\{\eta_k\left(\theta_{\mathcal{D}}, \theta_q\right)\right\} = 0, \tag{29}$$

where the expectation operators are defined such that $\mathbb{E}_{\theta_i}\{\cdot\} = \int \cdot \mathrm{d}\mathcal{P}\left(\theta_i\right)$, $\mathbb{E}_{\theta_i\theta_j}\{\cdot\} = \int \cdot \mathrm{d}\mathcal{P}\left(\theta_i\right)\mathrm{d}\mathcal{P}\left(\theta_j\right)$, $\mathcal{P}\left(\theta_i\right)$ denotes the probability measure of $\theta_i = \theta_q, \theta_{\mathcal{D}}$, and the coefficients h_{ik} are constants. $\mathcal{G}\left(0, \sum_{i=1}^{r} h_{ik}^2\right)$ denotes that the random variable obeys the Gaussian distribution with mean value 0 and variance $\sum_{i=1}^{r} h_{ik}^2$. Further, the variances of $\eta_k\left(\theta_{\mathcal{D}}, \theta_q\right)$ with respect to $\theta_{\mathcal{D}}$ and θ_q are given by

$$\sigma_{\eta,k}^2\left(\theta_{\mathcal{D}}\right) = \mathbb{E}_{\theta_q}\left\{\eta_k^2\left(\theta_{\mathcal{D}}, \theta_q\right)\right\} = \sum_{i=1}^{r} s_{ik}^2\left(\theta_{\mathcal{D}}\right), \tag{30}$$

$$\begin{aligned}\sigma_{\eta,k}^2\left(\theta_q\right) &= \mathbb{E}_{\theta_{\mathcal{D}}}\left\{\left[\eta_k\left(\theta_{\mathcal{D}}, \theta_q\right) - \overline{\eta}_k\left(\theta_q\right)\right]^2\right\} \\ &= \sum_{i,j=1}^{r}\left[\mathbb{E}_{\theta_{\mathcal{D}}}\left\{s_{ik}\left(\theta_{\mathcal{D}}\right) s_{jk}\left(\theta_{\mathcal{D}}\right)\right\} - h_{ik}h_{jk}\right]\xi_i\left(\theta_q\right)\xi_j\left(\theta_q\right),\end{aligned} \tag{31}$$

$$\sigma_{\eta,k}^2 = \mathbb{E}_{\theta_{\mathcal{D}}\theta_q}\left\{\eta_k^2\left(\theta_{\mathcal{D}}, \theta_q\right)\right\} = \mathbb{E}_{\theta_q\theta_{\mathcal{D}}}\left\{\eta_k^2\left(\theta_{\mathcal{D}}, \theta_q\right)\right\} = \sum_{i=1}^{r} \mathbb{E}_{\theta_{\mathcal{D}}}\left\{s_{ik}^2\left(\theta_{\mathcal{D}}\right)\right\}, \tag{32}$$

where it is considered that $s_{ik}\left(\theta_{\mathcal{D}}\right) = \sqrt{\lambda_i\left(\theta_{\mathcal{D}}\right)}\phi_{ik}\left(\theta_{\mathcal{D}}\right)$. The above equations (27) to (32) are proved as follows.

Proof. For Eq. (27), we have

$$\overline{\eta}_k\left(\theta_{\mathcal{D}}\right) = \mathbb{E}_{\theta_q}\left\{\eta_k\left(\theta_{\mathcal{D}}, \theta_q\right)\right\} = \sum_{i=1}^{r} \underbrace{\mathbb{E}_{\theta_q}\left\{\xi_i\left(\theta_q\right)\right\}}_{=0} \sqrt{\lambda_i\left(\theta_{\mathcal{D}}\right)}\phi_{ik}\left(\theta_{\mathcal{D}}\right) = 0. \tag{33}$$

For Eq. (28), we have

$$\overline{\eta}_k\left(\theta_q\right) = \mathbb{E}_{\theta_{\mathcal{D}}}\left\{\eta_k\left(\theta_{\mathcal{D}}, \theta_q\right)\right\} = \sum_{i=1}^{r} \mathbb{E}_{\theta_{\mathcal{D}}}\left\{\sqrt{\lambda_i\left(\theta_{\mathcal{D}}\right)}\phi_{ik}\left(\theta_{\mathcal{D}}\right)\right\}\xi_i\left(\theta_q\right). \tag{34}$$

Thus, the coefficients h_{ik} in Eq. (28) are given by $h_{ik} = \mathbb{E}_{\theta_{\mathcal{D}}}\left\{\sqrt{\lambda_i\left(\theta_{\mathcal{D}}\right)}\phi_{ik}\left(\theta_{\mathcal{D}}\right)\right\}$. The mean value of $\overline{\eta}_k\left(\theta_q\right)$ is

$$\mathbb{E}_{\theta_q}\left\{\overline{\eta}_k\left(\theta_q\right)\right\} = \sum_{i=1}^{r} \mathbb{E}_{\theta_{\mathcal{D}}}\left\{\sqrt{\lambda_i\left(\theta_{\mathcal{D}}\right)}\phi_{ik}\left(\theta_{\mathcal{D}}\right)\right\} \underbrace{\mathbb{E}_{\theta_q}\left\{\xi_i\left(\theta_q\right)\right\}}_{=0} = 0. \tag{35}$$

Further, the variance of $\overline{\eta}_k\left(\theta_q\right)$ is calculated as $\sum_{i=1}^{r} h_{ik}^2$ since $\left\{\xi_i\left(\theta_q\right)\right\}_{i=1}^{r}$ are Gaussian random variables with unit variances. Further, for Eq. (29), we have

$$\overline{\eta}_k = \mathbb{E}_{\theta_{\mathcal{D}}\theta_q}\left\{\eta_k\left(\theta_{\mathcal{D}},\theta_q\right)\right\} = \mathbb{E}_{\theta_q\theta_{\mathcal{D}}}\left\{\eta_k\left(\theta_{\mathcal{D}},\theta_q\right)\right\} = \sum_{i=1}^{r} \mathbb{E}_{\theta_{\mathcal{D}}}\left\{\sqrt{\lambda_i\left(\theta_{\mathcal{D}}\right)}\phi_{ik}\left(\theta_{\mathcal{D}}\right)\right\} \underbrace{\mathbb{E}_{\theta_q}\left\{\xi_i\left(\theta_q\right)\right\}}_{=0} = 0. \tag{36}$$

To prove Eqs. (30), (31) and (32), let us consider

$$\eta_k^2\left(\theta_{\mathcal{D}},\theta_q\right) = \sum_{i,j=1}^{r} \xi_i\left(\theta_q\right)\xi_j\left(\theta_q\right) s_{ik}\left(\theta_{\mathcal{D}}\right) s_{jk}\left(\theta_{\mathcal{D}}\right). \tag{37}$$

Thus, we have

$$\sigma_{\eta,k}^2\left(\theta_{\mathcal{D}}\right) = \sum_{i,j=1}^{r} \underbrace{\mathbb{E}_{\theta_q}\left\{\xi_i\left(\theta_q\right)\xi_j\left(\theta_q\right)\right\}}_{=\delta_{ij}} s_{ik}\left(\theta_{\mathcal{D}}\right) s_{jk}\left(\theta_{\mathcal{D}}\right) = \sum_{i=1}^{r} s_{ik}^2\left(\theta_{\mathcal{D}}\right), \tag{38}$$

$$\begin{aligned}
\sigma_{\eta,k}^2\left(\theta_q\right) &= \sum_{i,j=1}^{r} \mathbb{E}_{\theta_{\mathcal{D}}}\left\{\left[s_{ik}\left(\theta_{\mathcal{D}}\right) - h_{ik}\right]\left[s_{jk}\left(\theta_{\mathcal{D}}\right) - h_{jk}\right]\right\}\xi_i\left(\theta_q\right)\xi_j\left(\theta_q\right) \\
&= \sum_{i,j=1}^{r} \left[\mathbb{E}_{\theta_{\mathcal{D}}}\left\{s_{ik}\left(\theta_{\mathcal{D}}\right) s_{jk}\left(\theta_{\mathcal{D}}\right)\right\} - \mathbb{E}_{\theta_{\mathcal{D}}}\left\{s_{ik}\left(\theta_{\mathcal{D}}\right)\right\} h_{jk} - \mathbb{E}_{\theta_{\mathcal{D}}}\left\{s_{jk}\left(\theta_{\mathcal{D}}\right)\right\} h_{ik} + h_{ik}h_{jk}\right]\xi_i\left(\theta_q\right)\xi_j\left(\theta_q\right) \\
&= \sum_{i,j=1}^{r} \left[\mathbb{E}_{\theta_{\mathcal{D}}}\left\{s_{ik}\left(\theta_{\mathcal{D}}\right) s_{jk}\left(\theta_{\mathcal{D}}\right)\right\} - h_{ik}h_{jk}\right]\xi_i\left(\theta_q\right)\xi_j\left(\theta_q\right),
\end{aligned} \tag{39}$$

$$\sigma_{\eta,k}^2 = \sum_{i,j=1}^{r} \underbrace{\mathbb{E}_{\theta_q}\left\{\xi_i\left(\theta_q\right)\xi_j\left(\theta_q\right)\right\}}_{=\delta_{ij}} \mathbb{E}_{\theta_{\mathcal{D}}}\left\{s_{ik}\left(\theta_{\mathcal{D}}\right) s_{jk}\left(\theta_{\mathcal{D}}\right)\right\} = \sum_{i=1}^{r} \mathbb{E}_{\theta_{\mathcal{D}}}\left\{s_{ik}^2\left(\theta_{\mathcal{D}}\right)\right\}. \tag{40}$$

where δ_{ij} denotes a Kronecker delta, which is equal to 1 in case $i = j$ and zero, otherwise. □

4.3. Algorithm implementation

The proposed stochastic eigenequation (SE)-based algorithm for simulating RFRD is summarized in Algorithm 4, which consists of three parts. The first part is to calculate the random coordinates $\mathbf{X}\left(\theta_{\mathcal{D}}\right)$ by Algorithm 2 in step 1. Note that the second part involving steps 2 to 12 solves the stochastic eigenvalue equation. The SKLE is then implemented in the third part from

Algorithm 4 SE-based algorithm for simulating RFRD

1: Calculate n_s sample realizations of the random coordinates $\widehat{\mathbf{X}}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)$ by Algorithm 2
2: **while** $j \leq q$ **do**
3: Initialize the random sample vector $\widehat{\lambda}_j^{(0)}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right) \in \mathbb{R}^{n_s}$
4: **while** $\varepsilon_{\mathbf{g},k} > \varepsilon_{\mathbf{g}}$ **do**
5: Assemble the matrix $\mathbf{C}_{d,j}^{(k)}$ by looping over all elements $\widehat{\mathbb{E}}\left\{\widehat{\lambda}_j^{(k)}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\widehat{\mathbf{C}}_{lm}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\right\}$
6: Compute the deterministic eigenvector $\mathbf{g}_j^{(k)}$ via the power iteration
7: Update the random sample vector $\widehat{\lambda}_j^{(k)}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right) = \mathbf{g}_j^{(k)\mathrm{T}}\widehat{\mathbf{C}}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\mathbf{g}_j^{(k)} \in \mathbb{R}^{n_s}$
8: Calculate the locally iterative error $\varepsilon_{\mathbf{g},k}$, $k \leftarrow k+1$
9: **end**
10: Update the deterministic matrix $\mathbf{G} = \left[\mathbf{G}, \mathbf{g}_j\right] \in \mathbb{R}^{n\times j}$, $j \leftarrow j+1$
11: **end**
12: Calculate the sample realizations $\left\{\lambda_i\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right), \boldsymbol{\phi}_i\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\right\}_{i=1}^{q}$ of reduced-order stochastic eigenpairs by Eq. (22)
13: Generate random sample vectors $\widehat{\xi}_i\left(\widehat{\boldsymbol{\theta}}_q\right) \in \mathbb{R}^{n_s}$, $i = 1, \cdots, r$
14: Calculate random sample vectors $\left\{\widehat{\eta}_k\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}, \widehat{\boldsymbol{\theta}}_q\right) \in \mathbb{R}^{n_s}\right\}_{k=1}^{q}$
15: Simulate the random field $\boldsymbol{\omega}_{\mathrm{SE}}\left(\theta_{\mathcal{D}}, \theta_q\right)$ by Eq. (26)

step 13 to 15. The second part includes two loops, where the outer loop from step 2 to step 11 is used to solve all pairs $\left\{\lambda_j\left(\theta_{\mathcal{D}}\right), \mathbf{g}_j\right\}_{j=1}^{q}$ and the inner loop from step 3 to step 9 is used to calculate each pair $\left\{\lambda_j\left(\theta_{\mathcal{D}}\right), \mathbf{g}_j\right\}$. For the inner loop, the random sample vector $\widehat{\lambda}_j^{(0)}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right) \in \mathbb{R}^{n_s}$ is initialized in step 3. Numerical experience indicates that the value of the random sample vector $\widehat{\lambda}_j^{(0)}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)$ has little influence on the proposed algorithm. Thus, any nonzero vectors of size n_s can be used as the initial random sample vector. With the initial random sample vector, the deterministic matrix $\mathbf{C}_{d,j}^{(k)}$ is assembled in step 5. A direct way is to assemble $\mathbf{C}\left(\theta_{\mathcal{D}}\right)$ by looping all random samples and store them using a third order tensor $\widehat{\mathbf{C}}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right) \in \mathbb{R}^{n\times n\times n_s}$. The matrix $\mathbf{C}_{d,j}^{(k)}$ is then calculated by Eq. (18). However, this approach requires considerable memory storage, especially for large-scale problems. To avoid this problem, we adopt a point-wise strategy to assemble $\mathbf{C}_{d,j}^{(k)}$. That is, the

random element in row l and column m of the matrix $\mathbf{C}\left(\theta_{\mathcal{D}}\right)$ is calculated by

$$\widehat{\mathbf{C}}_{lm}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right) = C_{\omega\omega}\left(\widehat{\mathbf{X}}^{(l)}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right), \widehat{\mathbf{X}}^{(m)}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\right) \in \mathbb{R}^{n_s}. \tag{41}$$

The element in row l and column m of the matrix $\mathbf{C}_{d,j}^{(k)}$ in step 5 is thus calculated by

$$\mathbf{C}_{d,j,lm}^{(k)} = \widehat{\mathbb{E}}\left\{\widehat{\lambda}_j^{(k)}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\widehat{\mathbf{C}}_{lm}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\right\} \approx \frac{1}{n_s}\widehat{\lambda}_j^{(k)\mathrm{T}}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\widehat{\mathbf{C}}_{lm}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right) \in \mathbb{R}. \tag{42}$$

Assembling all elements $\mathbf{C}_{d,j,lm}^{(k)}$, $l, m = 1, \cdots, n$ we can obtain the deterministic matrix $\mathbf{C}_{d,j}^{(k)}$. Following that, the deterministic eigenvector $\mathbf{g}_j^{(k)}$ is solved in step 6 by the power iteration [20], which corresponds to

$$\mathbf{g}_j^{(k,s+1)} = \mathbf{C}_{d,j}^{(k)}\mathbf{g}_j^{(k,s)}, \tag{43}$$

where $s = 1, 2, \cdots$ is the iterative step. The iteration is stopped when $\left\|\mathbf{g}_j^{(k,s+1)} - \mathbf{g}_j^{(k,s)}\right\|_2^2$ achieves a specified precision, where $\|\cdot\|_2$ is the L_2 norm. It is noted that the Gram-Schmidt orthonormalization as shown in Eq. (19) is used to keep the vector $\mathbf{g}_j^{(k,s)}$ orthogonal to the vectors $\{\mathbf{g}_i\}_{i=1}^{j-1}$ along the whole iterative process. With the obtained vector $\mathbf{g}_j^{(k)}$, the random sample vector $\widehat{\lambda}_j^{(k)}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)$ is also calculated in a point-wise way in step 7

$$\widehat{\lambda}_j^{(k)}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right) = \mathbf{g}_j^{(k)\mathrm{T}}\widehat{\mathbf{C}}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right)\mathbf{g}_j^{(k)} = \sum_{l,m=1}^{n} g_{j,l}^{(k)} g_{j,m}^{(k)}\widehat{\mathbf{C}}_{lm}\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}\right) \in \mathbb{R}^{n_s}. \tag{44}$$

Also, the stopping criterion of the inner loop in step 8 is given by $\varepsilon_{\mathbf{g},k} = \left\|\mathbf{g}_j^{(k)} - \mathbf{g}_j^{(k-1)}\right\|_2^2$, whose choice can be inherited from the classical power iteration. Based on the reduced-order matrix $\mathbf{G}$ obtained in step 10, the reduced-order stochastic eigenvalue equation is solved in step 12. Combining the random samples from step 12 and step 13, random sample vectors $\left\{\widehat{\eta}_k\left(\widehat{\boldsymbol{\theta}}_{\mathcal{D}}, \widehat{\boldsymbol{\theta}}_q\right)\right\}_{k=1}^{q}$ of the composite random variables $\left\{\eta_k\left(\theta_{\mathcal{D}}, \theta_q\right)\right\}_{k=1}^{q}$ are calculated in step 14 and the random field $\boldsymbol{\omega}_{\mathrm{SE}}\left(\theta_{\mathcal{D}}, \theta_q\right)$ is simulated in step 15. However, in the practical implementation, the preselection of the number q of retained terms in step 2 is still an open problem and requires further research.

5. Numerical examples

In this section, we test the proposed methods with the aid of three numerical examples. In Algorithm 4, the convergence errors of the inner loop (i.e. $\varepsilon_{\mathbf{g}}$ in step 4) and the power iteration are both set as 1×10^{-3}. For all algorithms, $n_s = 1 \times 10^4$ random sample realizations are used for

the simulations. All examples are executed on a desktop computer (sixteen cores, Intel Core i7, 2.50GHz), but only one core is used for the numerical implementation. Whenever using a random field to describe a quantity that must remain within a certain range due to physical reasons, we truncate it to obtain meaningful results.

5.1. Example 1: one-dimensional beam with random length

In this example, we consider a one-dimensional beam with random length $l(\theta_{\mathcal{D}})$, as shown in Fig. 1. Its Young's modulus $E\left(x(\theta_{\mathcal{D}}), \theta_q\right)$ is modeled as a Gaussian random field with the mean value $E_0(x(\theta_{\mathcal{D}})) = 10$ MPa and the covariance function

$$C_{\omega\omega}(x_1(\theta_{\mathcal{D}}), x_2(\theta_{\mathcal{D}})) = \min(x_1(\theta_{\mathcal{D}}), x_2(\theta_{\mathcal{D}})), \tag{45}$$

where $x_1(\theta_{\mathcal{D}}), x_2(\theta_{\mathcal{D}}) \in [0, l(\theta_{\mathcal{D}})]$ and $l(\theta_{\mathcal{D}})$ is a uniform random variable on $[0.9, 1.1]$ m.

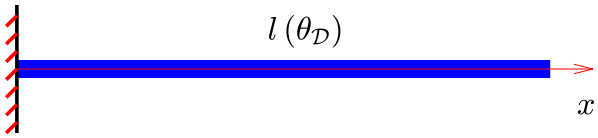


Figure 1: Model of the beam with random length $l(\theta_{\mathcal{D}})$.

The analytical stochastic eigenfunctions and stochastic eigenvalues of the stochastic covariance function $C_{\omega\omega}(x_1(\theta_{\mathcal{D}}), x_2(\theta_{\mathcal{D}}))$ are given by

$$f_i(x(\theta_{\mathcal{D}})) = \sqrt{\frac{2}{l(\theta_{\mathcal{D}})}} \sin\left(\frac{2i-1}{2l(\theta_{\mathcal{D}})}\pi x(\theta_{\mathcal{D}})\right), \quad \lambda_i(\theta_{\mathcal{D}}) = \left(\frac{l(\theta_{\mathcal{D}})}{\left(i-\frac{1}{2}\right)\pi}\right)^2 \tag{46}$$

for $i = 1, 2, \cdots$. More details can be found in Appendix A. The SKLE Eq. (1) of the random field $E\left(x(\theta_{\mathcal{D}}), \theta_q\right)$ is thus given by

$$E\left(x(\theta_{\mathcal{D}}), \theta_q\right) = 10 + \sum_{i=1}^{r} \xi_i\left(\theta_q\right) \frac{2\sqrt{2l(\theta_{\mathcal{D}})}}{(2i-1)\pi} \sin\left(\frac{2i-1}{2l(\theta_{\mathcal{D}})}\pi x(\theta_{\mathcal{D}})\right), \tag{47}$$

which can be transformed into a random field defined on a deterministic domain by introducing a normalizing process $\overline{x} = \frac{x(\theta_{\mathcal{D}})}{l(\theta_{\mathcal{D}})} \in [0, 1]$,

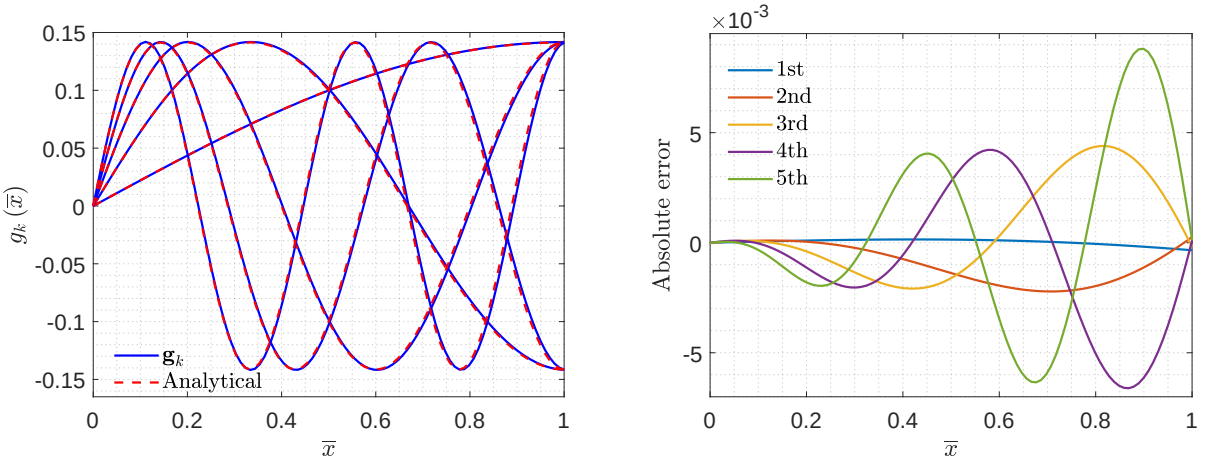
$$E\left(\overline{x}, \theta_{\mathcal{D}}, \theta_q\right) = 10 + \sum_{i=1}^{r} \xi_i\left(\theta_q\right) \frac{2\sqrt{2l(\theta_{\mathcal{D}})}}{(2i-1)\pi} \sin\left(\left(i-\frac{1}{2}\right)\pi\overline{x}\right). \tag{48}$$

In this way, the random field $E\left(x\left(\theta_{\mathcal{D}}\right), \theta_q\right)$ can be simulated and postprocessed on the deterministic domain $[0, 1]$.

We implement the three proposed algorithms for this example. For the DMCS Algorithm 1, we adopt the linear line element to discretize each sample realization $l\left(\theta_{\mathcal{D}}^{(i)}\right)$ of the random length, and 100 elements are generated for all sample realizations. For the TMCS Algorithm 3 and the SE-based Algorithm 4, the reference domain $[0, 1]$ is selected in the domain transformation Algorithm 2. It is also meshed into 100 linear line elements. Random coordinates of the original random domain $[0, l\left(\theta_{\mathcal{D}}\right)]$ are thus discretized as $\mathbf{X}\left(\theta_{\mathcal{D}}\right) = [X_1\left(\theta_{\mathcal{D}}\right), \cdots, X_{101}\left(\theta_{\mathcal{D}}\right)]$ and the boundary conditions in Eq. (7) are given by

$$X_1\left(\theta_{\mathcal{D}}\right) = 0, \quad X_{101}\left(\theta_{\mathcal{D}}\right) = l\left(\theta_{\mathcal{D}}\right). \tag{49}$$

The unknown random coordinates $[X_2\left(\theta_{\mathcal{D}}\right), \cdots, X_{100}\left(\theta_{\mathcal{D}}\right)]$ are then solved by Eq. (9). For the SE-based Algorithm 4, $q = 10$ approximate terms in Eq. (14) are adopted. For all three algorithms, $r = 6$ terms are retained for the KLE in Eq. (1). In practice, we choose the number of retained terms r such that the truncation error $\mathbb{E}\left\{\lambda_r^2\left(\theta_{\mathcal{D}}\right)\right\} \Big/ \sum_{i=1}^{r} \mathbb{E}\left\{\lambda_i^2\left(\theta_{\mathcal{D}}\right)\right\} < 1 \times 10^{-3}$. The comparison between the first five reduced bases $\{\mathbf{g}_k\}_{k=1}^{5}$ solved by the SE-based Algorithm 4 and the normalized analytical functions $\left\{\sqrt{2}\sin\left(\left(k - \frac{1}{2}\right)\pi\overline{x}\right)\right\}_{k=1}^{5}$ from Eq. (48) is shown in Fig. 2a and



(a) The first five deterministic eigenvectors.

(b) Absolute errors.

Figure 2: The first five deterministic eigenvectors obtained by the analytical representation and the proposed SE-based Algorithm 4 (left) and their absolute errors (right).

their absolute errors are seen from Fig. 2b, which indicates that the reduced bases $\{\mathbf{g}_k\}_{k=1}^5$ are very close to the analytical functions. Thus, the reduced bases can provide a good approximation to the stochastic eigenvectors. The absolute error of high-order functions is increasing, but it has small influence since high-order functions only have small contributions to the approximation of stochastic eigenvectors, which can be verified via the rapidly decreasing variances of the random variables $\left\{\eta_k\left(\theta_{\mathcal{D}},\theta_q\right)\right\}_{k=1}^5$, as shown in Table 1. Probability density functions (PDFs) of the first five random variables $\left\{\eta_k\left(\theta_{\mathcal{D}},\theta_q\right)\right\}_{k=1}^5$ in Eq. (26) are calculated by using 1×10^4 random samples, and they are plotted in the first row of Fig. 3. PDFs of the corresponding standardized random variables $\eta_k^*\left(\theta_{\mathcal{D}},\theta_q\right)=\eta_k\left(\theta_{\mathcal{D}},\theta_q\right)\Big/\sqrt{\mathbb{E}_{\theta_{\mathcal{D}}\theta_q}\left\{\eta_k^2\left(\theta_{\mathcal{D}},\theta_q\right)\right\}}$ are shown in the second row of Fig. 3, which demonstrates that the first five random variables $\left\{\eta_k\left(\theta_{\mathcal{D}},\theta_q\right)\right\}_{k=1}^5$ are very close to the Gaussian distribution. It is also seen from Table 1 that there is less variability of the random variable $\eta_k\left(\theta_{\mathcal{D}},\theta_q\right)$ as the index k increases.

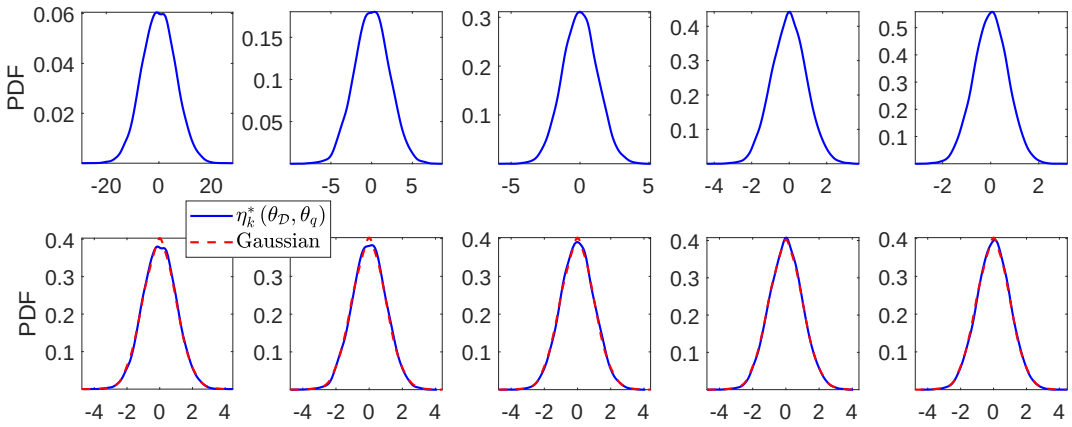


Figure 3: PDFs of the first five random variables $\left\{\eta_k\left(\theta_{\mathcal{D}},\theta_q\right)\right\}_{k=1}^5$ (the first line) and the PDF comparison between the standardized random variables $\left\{\eta_k^*\left(\theta_{\mathcal{D}},\theta_q\right)\right\}_{k=1}^5$ and the standard Gaussian random variable.

Table 1: Variances of the first five random variables $\left\{\eta_k\left(\theta_{\mathcal{D}},\theta_q\right)\right\}_{k=1}^5$.

Index k	1	2	3	4	5
$\mathbb{E}_{\theta_{\mathcal{D}}\theta_q}\left\{\eta_k^2\left(\theta_{\mathcal{D}},\theta_q\right)\right\}$	39.55	4.50	1.58	0.85	0.50

Table 2: Computational times of the three proposed methods.

Time for	Domain transformation	Reduced bases	SE/Reduced-order SE	Total cost (s)
DMCS	–	–	–	17.66
TMCS	0.26	–	10.17	10.43
SE-based method	0.26	2.33	1.15	3.74

The computational times associated with the execution of each of the three algorithms proposed in this work are listed in Table 2. The times for the domain transformation, the reduced bases and the SE/reduced-order SE represent the computational times for meshing the reference domain and solving Eq. (9), calculating the reduced bases $\mathbf{g}_k$ by step 2 to step 11 of Algorithm 4 and solving the stochastic eigenequation/the reduced-order stochastic eigenequation by the MCS (i.e. the stochastic eigenequation solved by Algorithm 3 and 4), respectively. The total cost is the summation of all computational times described above. Both the TMCS method and the SE-based method are cheaper than the DMCS since the domain transformation is much less computationally expensive than remeshing the random domain. The computational time of the TMCS is mainly concentrated on solving the stochastic eigenvalue equation using MCS, while the SE-

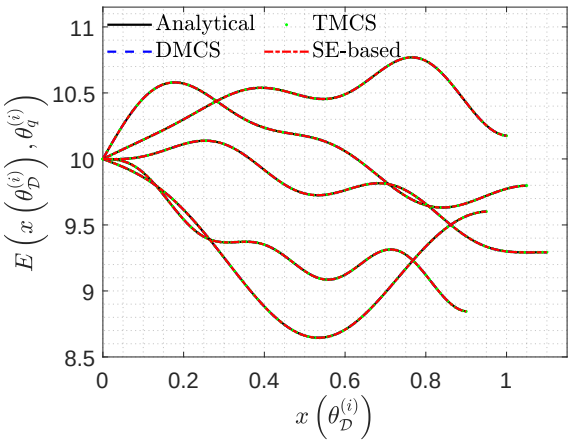


Figure 4: Five sample realizations $E\left(x\left(\theta_{\mathcal{D}}^{(i)}\right),\theta_q^{(i)}\right)$, $i = 1,\cdots,5$ obtained by the analytical, DMCS, TMCS and SE-based methods, respectively.

based method is cheap enough to solve stochastic eigenequations. Thus, the SE-based method has the lowest total computational cost among the three proposed methods. Five sample realizations $E\left(x\left(\theta_{\mathcal{D}}^{(i)}\right), \theta_q^{(i)}\right), i = 1, \cdots, 5$ of the random field are depicted in Fig. 4. They are generated by using the analytical representation Eq. (47), the DMCS-based approximation Eq. (5), the TMCS-based approximation Eq. (13) and the SE-based approximation Eq. (26), respectively. The sample realizations generated by all methods are highly consistent, which indicates the high accuracy of the three proposed methods. It is also easily seen that the randomness of both Young's modulus and length are described in the random sample realizations.

5.2. *Example 2: two-dimensional multiphase material with random interface*

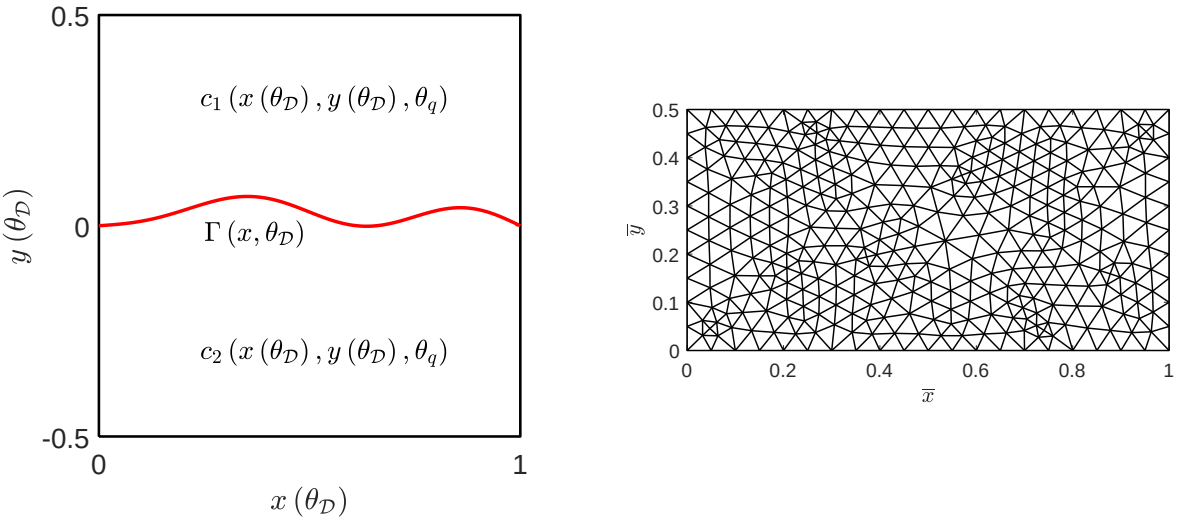


Figure 5: The model with a random interface (left) and the corresponding finite element mesh of the reference domain of the upper part (right).

In this example, we consider a two-dimensional problem with a random interface, as shown in Fig. 5 (left), which typically occurs in multiphase materials with random interfaces [4, 10]. The random interface is modeled as a Gaussian random field $\Gamma\left(x, \theta_{\mathcal{D}}\right)$ with the mean function $\overline{\Gamma}\left(x\right) = 0$ and the covariance function

$$C_{\Gamma\Gamma}\left(x_1, x_2\right) = \sigma_\Gamma^2 \left(\min\left(x_1, x_2\right) - x_1 x_2\right), \tag{50}$$

where the standard deviation is $\sigma_\Gamma = 0.1$. In the numerical implementation, the random field $\Gamma(x, \theta_\mathcal{D})$ is approximated by using the classical KLE [12, 15]

$$\Gamma(x, \theta_\mathcal{D}) = \sigma_\Gamma \sum_{i=1}^{5} \zeta_i(\theta_\mathcal{D}) \sqrt{\kappa_i} \Gamma_i(x), \tag{51}$$

where $\{\zeta_i(\theta_\mathcal{D})\}_{i=1}^5$ are mutually independent standard Gaussian random variables, and $\{\kappa_i, \Gamma_i(x)\}_{i=1}^5$ are eigenvalues and eigenfunctions of the covariance function $C_{\Gamma\Gamma}(x_1, x_2)$. Their analytical solutions are given by $\Gamma_i(x) = \sqrt{2}\sin(i\pi x)$ and $\kappa_i = (i\pi)^{-2}$, $i = 1, 2, \cdots$.

The coefficients $c_1\left(x(\theta_\mathcal{D}), y(\theta_\mathcal{D}), \theta_q\right)$ and $c_2\left(x(\theta_\mathcal{D}), y(\theta_\mathcal{D}), \theta_q\right)$ of the upper and lower parts are modeled as Gaussian random fields. Here we only focus on simulating the random field $c_1\left(x(\theta_\mathcal{D}), y(\theta_\mathcal{D}), \theta_q\right)$. The random field $c_2\left(x(\theta_\mathcal{D}), y(\theta_\mathcal{D}), \theta_q\right)$ can be simulated using an analogous approach. The random field $c_1\left(x(\theta_\mathcal{D}), y(\theta_\mathcal{D}), \theta_q\right)$ has the mean function $\overline{c}_1(x, y) = 5$ and the modified exponential covariance function [21, 22]

$$\begin{aligned} C_{\omega\omega}(x_1(\theta_\mathcal{D}), y_1(\theta_\mathcal{D}); x_2(\theta_\mathcal{D}), y_2(\theta_\mathcal{D})) = \sigma_\omega^2 \exp\left(-\frac{|x_1(\theta_\mathcal{D}) - x_2(\theta_\mathcal{D})|}{l_x} - \frac{|y_1(\theta_\mathcal{D}) - y_2(\theta_\mathcal{D})|}{l_y}\right) \\ \times \left(1 + \frac{|x_1(\theta_\mathcal{D}) - x_2(\theta_\mathcal{D})|}{l_x}\right)\left(1 + \frac{|y_1(\theta_\mathcal{D}) - y_2(\theta_\mathcal{D})|}{l_y}\right), \end{aligned} \tag{52}$$

where the standard deviation $\sigma_\omega = 0.5$ and the correlation lengths $l_x = l_y = 1$ are considered.

In this example, we only adopt the SE-based Algorithm 4 for the simulation. $q = 10$ approximate terms in Eq. (14) are adopted and $r = 4$ terms are retained in Eq. (1). The reference domain of the upper part is shown in Fig. 5 (right) and its finite element mesh includes 367 nodes and 672 linear triangular elements. The left, right and upper boundaries of the reference domain are the

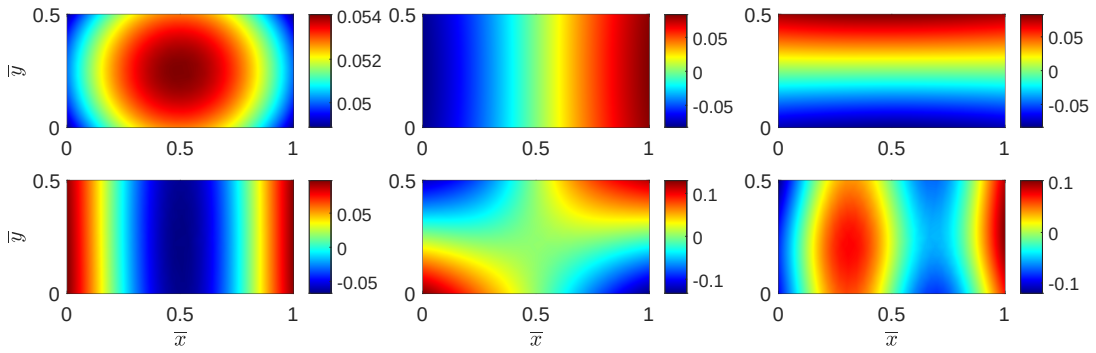


Figure 6: The first six deterministic vectors $\{\mathbf{g}_k\}_{k=1}^6$ obtained by the proposed SE-based Algorithm 4.

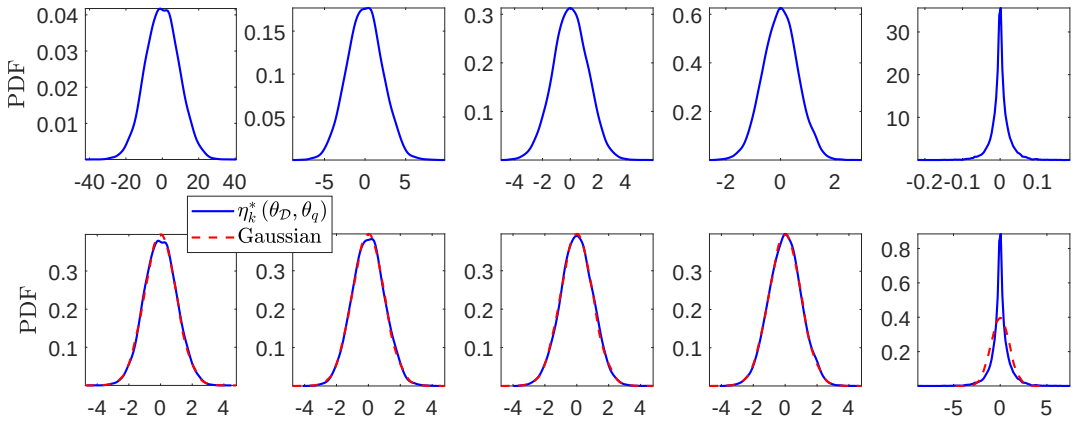


Figure 7: PDFs of the first five random variables $\left\{\eta_k\left(\theta_\mathcal{D},\theta_q\right)\right\}_{k=1}^5$ (the first line) and the PDF comparison between the standardized random variables $\left\{\eta_k^*\left(\theta_\mathcal{D},\theta_q\right)\right\}_{k=1}^5$ and the standard Gaussian random variable.

same as that of the original random domain. The lower boundary of the reference domain is set as the mean function $\overline{\Gamma}(x) = 0$. The random coordinates of the points on the random interface are thus represented as

$$x(\theta_\mathcal{D}) = \overline{x}, \quad y(\theta_\mathcal{D}) = \Gamma(\overline{x}, \theta_\mathcal{D}). \tag{53}$$

In this way, the horizontal coordinate $x(\theta_\mathcal{D})$ is deterministic and its solution at discretized finite element nodes is given by $\mathbf{X}(\theta_\mathcal{D}) = \left[\overline{X}_1, \cdots, \overline{X}_{367}\right]$. Further, substituting Eq. (53) into Eq. (8) and solving Eq. (9) we can obtain the solution $\mathbf{Y}(\theta_\mathcal{D}) = [Y_1(\theta_\mathcal{D}), \cdots, Y_{367}(\theta_\mathcal{D})]$ of the random coordinate $y(\theta_\mathcal{D})$. The first six reduced bases $\{\mathbf{g}_k\}_{k=1}^6$ are shown in Fig. 6. Similar to Fig. 3, PDFs of the first five random variables $\left\{\eta_k\left(\theta_\mathcal{D},\theta_q\right)\right\}_{k=1}^5$ in Eq. (26) are calculated by using 1×10^4 random samples and the corresponding standardized random variables $\eta_k^*\left(\theta_\mathcal{D},\theta_q\right) = \eta_k\left(\theta_\mathcal{D},\theta_q\right)\Big/\sqrt{\mathbb{E}_{\theta_\mathcal{D}\theta_q}\left\{\eta_k^2\left(\theta_\mathcal{D},\theta_q\right)\right\}}$, $k = 1, \cdots, 5$ are shown in the first and second rows of Fig. 7, respectively. In this example, the first four random variables $\left\{\eta_k\left(\theta_\mathcal{D},\theta_q\right)\right\}_{k=1}^4$ are quite close to the

Table 3: Variances of the first five random variables $\left\{\eta_k\left(\theta_\mathcal{D},\theta_q\right)\right\}_{k=1}^5$.

Index k	1	2	3	4	5
$\mathbb{E}_{\theta_\mathcal{D}\theta_q}\left\{\eta_k^2\left(\theta_\mathcal{D},\theta_q\right)\right\}$	82.66	4.68	1.56	0.40	6.17×10^{-4}

Gaussian distribution, but the distribution of the fifth random variable $\eta_5\left(\theta_{\mathcal{D}},\theta_q\right)$ is non-Gaussian. It is seen from Table 3 that the variances of the random variables $\left\{\eta_k\left(\theta_{\mathcal{D}},\theta_q\right)\right\}_{k=1}^{5}$ still rapidly decrease as the index k increases.

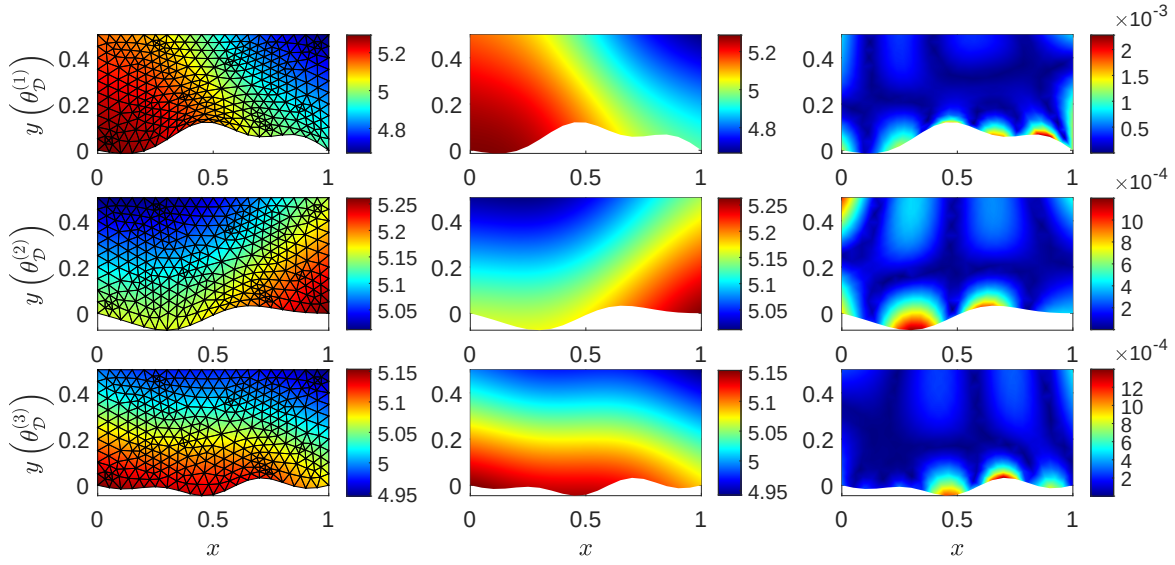
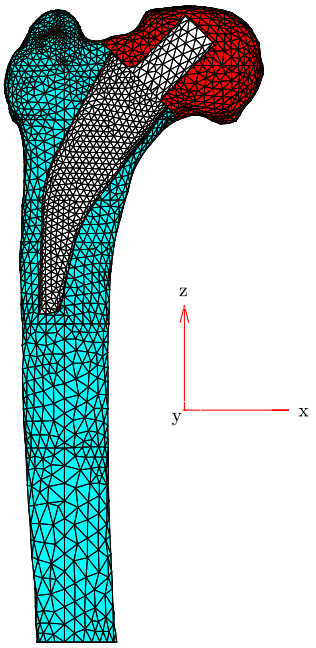


Figure 8: Three sample realizations $c_1\left(x\left(\theta_{\mathcal{D}}^{(i)}\right),y\left(\theta_{\mathcal{D}}^{(i)}\right),\theta_q^{(i)}\right)$, $i=1,2,3$ obtained by the SE-based method (the first column) and the DMCS (the second column), and their relative errors (the third column).

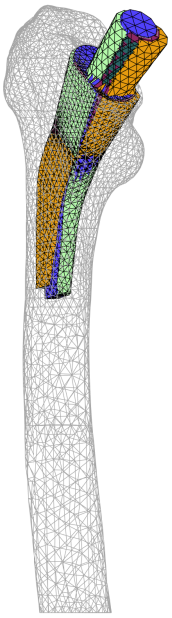
The total computational time for the simulation is 13.19s, including 0.45s for executing the domain transformation, 6.58s for calculating the reduced bases $\{\mathbf{g}_k\}_{k=1}^{10}$ and 6.16s for solving the reduced-order stochastic eigenvalue equation, while the computational time for the DMCS is 94.61s, which indicates that the proposed SE-based method is still efficient in this case. Three sample realizations $c_1\left(x\left(\theta_{\mathcal{D}}^{(i)}\right),y\left(\theta_{\mathcal{D}}^{(i)}\right),\theta_q^{(i)}\right)$, $i=1,2,3$ obtained by the SE-based method and the DMCS are depicted in the first and second columns of Fig. 8, respectively. Further, their relative errors $\left|\frac{c_{1,\mathrm{SE}}\left(x\left(\theta_{\mathcal{D}}^{(i)}\right),y\left(\theta_{\mathcal{D}}^{(i)}\right),\theta_q^{(i)}\right)-c_{1,\mathrm{DMCS}}\left(x\left(\theta_{\mathcal{D}}^{(i)}\right),y\left(\theta_{\mathcal{D}}^{(i)}\right),\theta_q^{(i)}\right)}{c_{1,\mathrm{DMCS}}\left(x\left(\theta_{\mathcal{D}}^{(i)}\right),y\left(\theta_{\mathcal{D}}^{(i)}\right),\theta_q^{(i)}\right)}\right|$ are shown in the third column of Fig. 8. It is seen that all relative errors are less than 0.2%, which demonstrates the high accuracy of the SE-based simulation algorithm.

5.3. Example 3: three-dimensional bone implant with random prosthesis position

In this example, we consider a three-dimensional femur implant problem [23, 24] shown in Fig. 9a, where the prosthesis (i.e. the white part shown in Fig. 9a) has a random position. Our goal



(a) Model of the femur and its finite element mesh.



(b) Instantiations of the random position of the prosthesis.

Figure 9: The femur model with a prosthesis and its finite element mesh (left) and the sample realizations of the random position of the prosthesis (right).

is to simulate the random field $\rho\left(x\left(\theta_{\mathcal{D}}\right), y\left(\theta_{\mathcal{D}}\right), z\left(\theta_{\mathcal{D}}\right), \theta_q\right)$ of the bone density that takes into account spatial variability and the effect of random position of the prosthesis. It is noted that as shown in Fig. 9b, the femoral head (i.e. the red part shown in Fig. 9a) needs to be removed before implanting the prosthesis in practice. Therefore, the simulation of bone density for the femoral head is of less significance. In this example, we still consider the full geometry since the femoral head has less influence on the simulation. We consider that the bone density $\rho\left(x\left(\theta_{\mathcal{D}}\right), y\left(\theta_{\mathcal{D}}\right), z\left(\theta_{\mathcal{D}}\right), \theta_q\right)$ has the mean value $\overline{\rho}_0(x, y, z) = 1.5$ g/cm^3 and the exponential covariance function

$$\begin{aligned} &C_{\omega\omega}\left(x_1\left(\theta_{\mathcal{D}}\right), y_1\left(\theta_{\mathcal{D}}\right), z_1\left(\theta_{\mathcal{D}}\right); x_2\left(\theta_{\mathcal{D}}\right), y_2\left(\theta_{\mathcal{D}}\right), z_2\left(\theta_{\mathcal{D}}\right)\right) \\ &\quad = \sigma_\omega^2 \exp\left(-\frac{\left|x_1\left(\theta_{\mathcal{D}}\right) - x_2\left(\theta_{\mathcal{D}}\right)\right|}{l_x} - \frac{\left|y_1\left(\theta_{\mathcal{D}}\right) - y_2\left(\theta_{\mathcal{D}}\right)\right|}{l_y} - \frac{\left|z_1\left(\theta_{\mathcal{D}}\right) - z_2\left(\theta_{\mathcal{D}}\right)\right|}{l_z}\right), \end{aligned} \tag{54}$$

where the standard deviation $\sigma_\omega = 0.1$ and the correlation lengths $l_x = \max(x) - \min(x)$, $l_y = \max(y) - \min(y)$ and $l_z = \max(z) - \min(z)$. Nine random parameters $\{d_x\left(\theta_{\mathcal{D}}\right), d_y\left(\theta_{\mathcal{D}}\right), d_z\left(\theta_{\mathcal{D}}\right), s_x\left(\theta_{\mathcal{D}}\right),$

$s_y(\theta_{\mathcal{D}}), s_z(\theta_{\mathcal{D}}), \alpha_x(\theta_{\mathcal{D}}), \alpha_y(\theta_{\mathcal{D}}), \alpha_z(\theta_{\mathcal{D}})\Big\}$ are used to position the prosthesis relative to its nominal position, where $d_x(\theta_{\mathcal{D}}), d_y(\theta_{\mathcal{D}}), d_z(\theta_{\mathcal{D}}) \in [-0.1, 0.1]$ mm are translation parameters along the x, y and z axes, $s_x(\theta_{\mathcal{D}}), s_y(\theta_{\mathcal{D}}), s_z(\theta_{\mathcal{D}}) \in [0.95, 1.05]$ are the scaling parameters in the x, y and z directions and used to adjust the size of the prosthesis, and $\alpha_x(\theta_{\mathcal{D}}), \alpha_y(\theta_{\mathcal{D}}), \alpha_z(\theta_{\mathcal{D}}) \in [-2^\circ, 2^\circ]$ are the rotation angles around the x, y and z axes relative to the center point of the prosthesis. Several realizations of the random position of the prosthesis are depicted in Fig. 9b.

To execute the SE-based Algorithm 4, $q = 20$ approximate terms in Eq. (14) are adopted and $r = 15$ terms are retained in Eq. (1) to achieve the specified precision. The nominal position of the prosthesis is chosen as the reference domain and its finite element mesh includes 11046 nodes and 56711 linear tetrahedron elements. The outer boundary of the reference domain is the same as that of the original random domain. The random coordinates of the points on the interface between the femur and the prosthesis are represented as

$$\begin{bmatrix} x(\theta_{\mathcal{D}}) \\ y(\theta_{\mathcal{D}}) \\ z(\theta_{\mathcal{D}}) \end{bmatrix} = \mathbf{D}(\theta_{\mathcal{D}})\,\mathbf{S}(\theta_{\mathcal{D}})\,\mathbf{R}_x(\theta_{\mathcal{D}})\,\mathbf{R}_y(\theta_{\mathcal{D}})\,\mathbf{R}_z(\theta_{\mathcal{D}}) \begin{bmatrix} \overline{x} \\ \overline{y} \\ \overline{z} \\ 1 \end{bmatrix}, \tag{55}$$

where the stochastic translation transformation matrix $\mathbf{D}(\theta_{\mathcal{D}})$ and the stochastic scaling transformation matrix $\mathbf{S}(\theta_{\mathcal{D}})$ are given by

$$\mathbf{D}(\theta_{\mathcal{D}}) = \begin{bmatrix} 1 & 0 & 0 & d_x(\theta_{\mathcal{D}}) \\ 0 & 1 & 0 & d_y(\theta_{\mathcal{D}}) \\ 0 & 0 & 1 & d_z(\theta_{\mathcal{D}}) \end{bmatrix} \in \mathbb{R}^{3\times 4}, \quad \mathbf{S}(\theta_{\mathcal{D}}) = \begin{bmatrix} s_x(\theta_{\mathcal{D}}) & 0 & 0 & 0 \\ 0 & s_y(\theta_{\mathcal{D}}) & 0 & 0 \\ 0 & 0 & s_z(\theta_{\mathcal{D}}) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4} \tag{56}$$

and $\mathbf{R}_x(\theta_{\mathcal{D}})$, $\mathbf{R}_y(\theta_{\mathcal{D}})$ and $\mathbf{R}_z(\theta_{\mathcal{D}})$ are the stochastic rotation transformation matrices around the x, y and z axes, respectively. They are given by

$$\mathbf{R}_x(\theta_{\mathcal{D}}) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_x(\theta_{\mathcal{D}}) & -\sin\alpha_x(\theta_{\mathcal{D}}) & 0 \\ 0 & \sin\alpha_x(\theta_{\mathcal{D}}) & \cos\alpha_x(\theta_{\mathcal{D}}) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4},$$

$$\mathbf{R}_y(\theta_{\mathcal{D}}) = \begin{bmatrix} \cos\alpha_y(\theta_{\mathcal{D}}) & 0 & \sin\alpha_y(\theta_{\mathcal{D}}) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\alpha_y(\theta_{\mathcal{D}}) & 0 & \cos\alpha_y(\theta_{\mathcal{D}}) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times4},$$

$$\mathbf{R}_z(\theta_{\mathcal{D}}) = \begin{bmatrix} \cos\alpha_z(\theta_{\mathcal{D}}) & \sin\alpha_z(\theta_{\mathcal{D}}) & 0 & 0 \\ -\sin\alpha_z(\theta_{\mathcal{D}}) & \cos\alpha_z(\theta_{\mathcal{D}}) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times4}. \tag{57}$$

The total computational time for the simulation is 463.26s, including 3.07s for executing the domain transformation, 413.938s for calculating the reduced bases $\{\mathbf{g}_k\}_{k=1}^{20}$ and 46.26s for solving the reduced-order stochastic eigenvalue equation (22), which is much less than the cost of 1.69×10^4s to execute the DMCS. Most of the cost of executing the proposed SE-based method is used to solve the reduced bases, thus more efficient eigenvalue solvers can speed up the proposed SE-

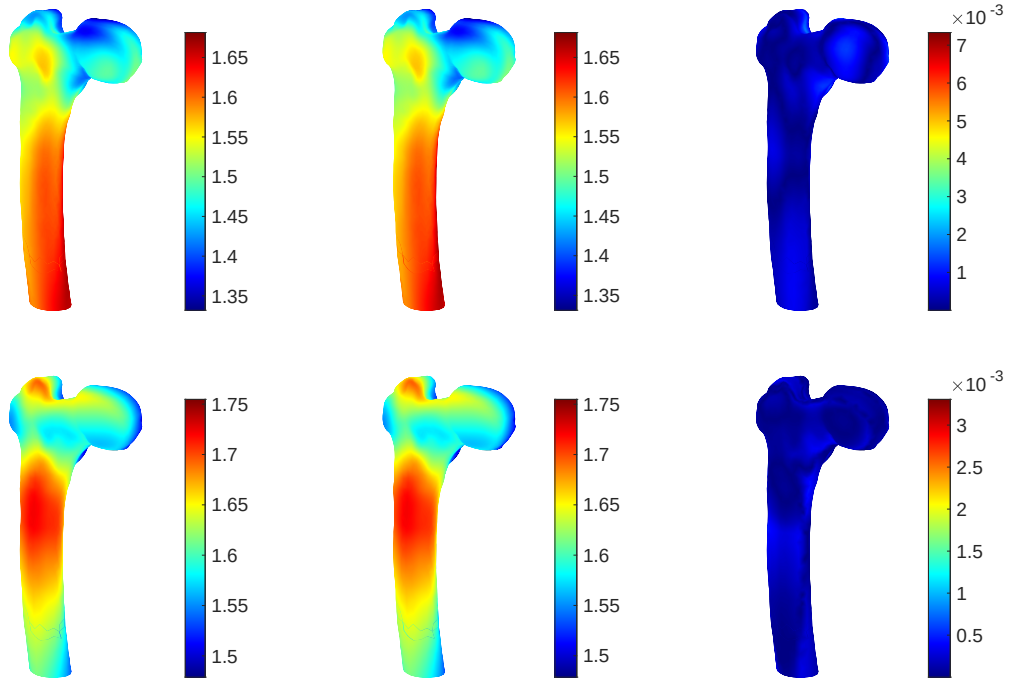


Figure 10: Two sample realizations $\rho\left(x\left(\theta_{\mathcal{D}}^{(i)}\right), y\left(\theta_{\mathcal{D}}^{(i)}\right), z\left(\theta_{\mathcal{D}}^{(i)}\right), \theta_q^{(i)}\right)$, $i = 1, 2$ obtained by the SE-based method (the first column) and the DMCS (the second column), and their relative errors (the third column).

based method. Two sample realizations $\rho\left(x\left(\theta_{\mathcal{D}}^{(i)}\right), y\left(\theta_{\mathcal{D}}^{(i)}\right), z\left(\theta_{\mathcal{D}}^{(i)}\right), \theta_q^{(i)}\right)$, $i = 1, 2$ obtained by the proposed SE-based method and the DMCS are shown in the first and second columns of Fig. 10, respectively. Their relative errors are shown in the third column of Fig. 10. It is seen that the maximum relative error is less than 0.7% and the proposed SE-based simulation algorithm is still accurate enough for this three-dimensional example.

6. Conclusions

We presented three algorithms for simulating the random fields defined on random domains, including the direct MCS, the domain transformation-based MCS and the stochastic eigenequation-based method. A stochastic KLE is first proposed as an extension of the classical KLE to stochastic cases. All three numerical methods are used to solve the stochastic integral equations arising in the stochastic KLE. The direct MCS is computationally expensive since all sampled domains are remeshed and the corresponding integral equations must be solved. The domain transformation-based MCS combines the domain transformation for random domains and the classical MCS for solving stochastic integral equations. This method saves the computational effort for remeshing the sampled domains, but it is still expensive in solving stochastic integral equations. To overcome this problem, we propose the stochastic eigenequation-based method, which combines the domain transformation for random domains and a reduced-order method for cheaply solving stochastic eigenvalue equations. All three proposed methods have been verified with numerical examples. They can be readily applied to the stochastic problems defined on random domains. However, more effective numerical strategies are still needed to simulate the random domains with large uncertainties, which remains a challenge and will be investigated in follow-up studies. Further, only Gaussian random fields are studied in this paper. By combining the proposed methods in this paper and the iterative methods in [25], we can simulate the non-Gaussian random fields defined on random domains.

Acknowledgments

The authors are grateful to the Alexander von Humboldt Foundation and the International Research Training Group 2657 (IRTG 2657) funded by the German Research Foundation (DFG)

(Grant number 433082294).

Appendix A. Analytical stochastic eigenvalues and eigenfunctions of Example 5.1

In this section, we give the analytical stochastic eigenvalues and stochastic eigenfunctions of the covariance function given in Eq. (45) in Section 5.1. To this end, the following stochastic Fredholm integral equation of the second kind is solved

$$\int_0^{l(\theta_{\mathcal{D}})} \min\left(x_1\left(\theta_{\mathcal{D}}\right), x_2\left(\theta_{\mathcal{D}}\right)\right) f\left(x_1\left(\theta_{\mathcal{D}}\right)\right) \mathrm{d}x_1\left(\theta_{\mathcal{D}}\right) = \lambda\left(\theta_{\mathcal{D}}\right) f\left(x_2\left(\theta_{\mathcal{D}}\right)\right). \tag{A.1}$$

Dividing the random domain into $x_1\left(\theta_{\mathcal{D}}\right) \in [0, l\left(\theta_{\mathcal{D}}\right)] = [0, x_2\left(\theta_{\mathcal{D}}\right)] \oplus \left(x_2\left(\theta_{\mathcal{D}}\right), l\left(\theta_{\mathcal{D}}\right)\right]$ and substituting it into Eq. (A.1) we have

$$\int_0^{x_2(\theta_{\mathcal{D}})} x_1\left(\theta_{\mathcal{D}}\right) f\left(x_1\left(\theta_{\mathcal{D}}\right)\right) \mathrm{d}x_1\left(\theta_{\mathcal{D}}\right) + x_2\left(\theta_{\mathcal{D}}\right) \int_{x_2(\theta_{\mathcal{D}})}^{l(\theta_{\mathcal{D}})} f\left(x_1\left(\theta_{\mathcal{D}}\right)\right) \mathrm{d}x_1\left(\theta_{\mathcal{D}}\right) = \lambda\left(\theta_{\mathcal{D}}\right) f\left(x_2\left(\theta_{\mathcal{D}}\right)\right). \tag{A.2}$$

Taking the derivative with respect to $x_2\left(\theta_{\mathcal{D}}\right)$ on both sides we get

$$\int_{x_2(\theta_{\mathcal{D}})}^{l(\theta_{\mathcal{D}})} f\left(x_1\left(\theta_{\mathcal{D}}\right)\right) \mathrm{d}x_1\left(\theta_{\mathcal{D}}\right) = \lambda\left(\theta_{\mathcal{D}}\right) \frac{\mathrm{d}f\left(x_2\left(\theta_{\mathcal{D}}\right)\right)}{\mathrm{d}x_2\left(\theta_{\mathcal{D}}\right)}. \tag{A.3}$$

Differentiating $x_2\left(\theta_{\mathcal{D}}\right)$ on both sides again we have

$$-f\left(x_2\left(\theta_{\mathcal{D}}\right)\right) = \lambda\left(\theta_{\mathcal{D}}\right) \frac{\mathrm{d}^2 f\left(x_2\left(\theta_{\mathcal{D}}\right)\right)}{\mathrm{d}x_2^2\left(\theta_{\mathcal{D}}\right)}, \tag{A.4}$$

whose solution has the form (the subscript is omitted)

$$f\left(x\left(\theta_{\mathcal{D}}\right)\right) = c_1\left(\theta_{\mathcal{D}}\right) \sin\left(\frac{x\left(\theta_{\mathcal{D}}\right)}{\sqrt{\lambda\left(\theta_{\mathcal{D}}\right)}}\right) + c_2\left(\theta_{\mathcal{D}}\right) \cos\left(\frac{x\left(\theta_{\mathcal{D}}\right)}{\sqrt{\lambda\left(\theta_{\mathcal{D}}\right)}}\right), \tag{A.5}$$

where $c_1\left(\theta_{\mathcal{D}}\right)$ and $c_2\left(\theta_{\mathcal{D}}\right)$ are unknown stochastic coefficients that need to be determined. Furthermore, we have

$$f(0) = 0, \quad \left.\frac{\mathrm{d}f\left(x\left(\theta_{\mathcal{D}}\right)\right)}{\mathrm{d}x\left(\theta_{\mathcal{D}}\right)}\right|_{x(\theta_{\mathcal{D}})=l(\theta_{\mathcal{D}})} = 0 \tag{A.6}$$

by letting $x_2\left(\theta_{\mathcal{D}}\right) = 0$ in Eq. (A.2) and letting $x_2\left(\theta_{\mathcal{D}}\right) = l\left(\theta_{\mathcal{D}}\right)$ in Eq. (A.3), respectively. Substituting them into Eq. (A.5) we get

$$c_2\left(\theta_{\mathcal{D}}\right) = 0, \quad \cos\left(\frac{l\left(\theta_{\mathcal{D}}\right)}{\sqrt{\lambda\left(\theta_{\mathcal{D}}\right)}}\right) = 0. \tag{A.7}$$

Thus, the stochastic eigenvalues are given by

$$\lambda_i(\theta_{\mathcal{D}}) = \left(\frac{l(\theta_{\mathcal{D}})}{\left(i - \frac{1}{2}\right)\pi} \right)^2, \quad i = 1, 2, \cdots \tag{A.8}$$

and the stochastic eigenfunctions are given by

$$f_i(x(\theta_{\mathcal{D}})) = c_1(\theta_{\mathcal{D}}) \sin\left(\frac{2i-1}{2l(\theta_{\mathcal{D}})} \pi x(\theta_{\mathcal{D}}) \right), \quad i = 1, 2, \cdots, \tag{A.9}$$

where the stochastic coefficient $c_1(\theta_{\mathcal{D}}) = \sqrt{\frac{2}{l(\theta_{\mathcal{D}})}}$ is obtained by normalizing the eigenfunctions $f_i(x(\theta_{\mathcal{D}}))$,

$$\int_0^{l(\theta_{\mathcal{D}})} f_i^2(x(\theta_{\mathcal{D}}))\, \mathrm{d}x(\theta_{\mathcal{D}}) = 1. \tag{A.10}$$

References

[1] R. C. Smith, Uncertainty quantification: theory, implementation, and applications, volume 12, SIAM, 2013.

[2] H. Dai, R. Zhang, M. Beer, A new method for stochastic analysis of structures under limited observations, Mechanical Systems and Signal Processing 185 (2023) 109730.

[3] Y. Li, J. Xu, A pdf discretization scheme in wavenumber–frequency joint spectrum for simulating multivariate random fluctuating wind fields, Probabilistic Engineering Mechanics 72 (2023) 103422.

[4] R. Ghanem, W. Brzakala, Stochastic finite-element analysis of soil layers with random interface, Journal of Engineering Mechanics 122 (1996) 361–369.

[5] P. S. Mohan, P. B. Nair, A. J. Keane, Stochastic projection schemes for deterministic linear elliptic partial differential equations on random domains, International Journal for Numerical Methods in Engineering 85 (2011) 874–895.

[6] W. Zhang, Z. Kang, Robust shape and topology optimization considering geometric uncertainties with stochastic level set perturbation, International Journal for Numerical Methods in Engineering 110 (2017) 31–56.

[7] D. Xiu, D. M. Tartakovsky, Numerical methods for differential equations in random domains, SIAM Journal on Scientific Computing 28 (2006) 1167–1185.

[8] A. Nouy, A. Clement, eXtended Stochastic Finite Element Method for the numerical simulation of heterogeneous materials with random material interfaces, International Journal for Numerical Methods in Engineering 83 (2010) 1312–1344.

[9] S. Badia, J. Hampton, J. Principe, Embedded multilevel Monte Carlo for uncertainty quantification in random domains, International Journal for Uncertainty Quantification 11 (2021).

[10] Z. Zheng, M. Valdebenito, M. Beer, U. Nackenhorst, A stochastic finite element scheme for solving partial differential equations defined on random domains, Computer Methods in Applied Mechanics and Engineering 405 (2023) 115860.

[11] A. Kundu, S. Adhikari, M. Friswell, Stochastic finite elements of discretely parameterized random systems on domains with boundary uncertainty, International Journal for Numerical Methods in Engineering 100 (2014) 183–221.

[12] R. G. Ghanem, P. D. Spanos, Stochastic finite elements: a spectral approach, Courier Corporation, 2003.

[13] V. D. Liseikin, Grid generation methods, Springer, 2017.

[14] Z. Zheng, M. Beer, U. Nackenhorst, An efficient reduced-order method for stochastic eigenvalue analysis, International Journal for Numerical Methods in Engineering 123 (2022) 1–23.

[15] Z. Zheng, H. Dai, Simulation of multi-dimensional random fields by Karhunen–Loève expansion, Computer Methods in Applied Mechanics and Engineering 324 (2017) 221–247.

[16] B. T. Helenbrook, Mesh deformation using the biharmonic operator, International Journal for Numerical Methods in Engineering 56 (2003) 1007–1021.

[17] M. Selim, R. Koomullil, Mesh deformation approaches – a survey, Journal of Physical Mathematics 7 (2016) 1–9.

[18] J. Donea, A. Huerta, J.-P. Ponthot, A. Rodríguez-Ferran, Arbitrary Lagrangian–Eulerian methods, Encyclopedia of Computational Mechanics (2004).

[19] U. Nackenhorst, The ALE-formulation of bodies in rolling contact: Theoretical foundations and finite element approach, Computer Methods in Applied Mechanics and Engineering 193 (2004) 4299–4322.

[20] Y. Saad, Numerical methods for large eigenvalue problems, SIAM, 2011.

[21] P. D. Spanos, M. Beer, J. Red-Horse, Karhunen–Loève expansion of stochastic processes with a modified exponential covariance kernel, Journal of Engineering Mechanics 133 (2007) 773–779.

[22] M. G. Faes, M. Broggi, P. D. Spanos, M. Beer, Elucidating appealing features of differentiable auto-correlation functions: A study on the modified exponential kernel, Probabilistic Engineering Mechanics (2022) 103269.

[23] U. Nackenhorst, M. Bittens, How to push computational bio-mechanics to clinical application?, in: Current Trends and Open Problems in Computational Mechanics, Springer, 2022, pp. 367–374.

[24] A. Lutz, U. Nackenhorst, Numerical investigations on the osseointegration of uncemented endoprostheses based on bio-active interface theory, Computational Mechanics 50 (2012) 367–381.

[25] Z. Zheng, H. Dai, Y. Wang, W. Wang, A sample-based iterative scheme for simulating non-stationary non-Gaussian stochastic processes, Mechanical Systems and Signal Processing 151 (2021) 107420.