Model Reduction and Domain Decomposition
Methods for Uncertainty Quantification

by

Andres A. Contreras

Department of Civil and Environmental Engineering
Duke University

Date: ________________
Approved: 

___________________________
Omar M. Knio, Co-Supervisor

___________________________
Wilkins Aquino, Co-Supervisor

___________________________
Olivier P. Le Maître

___________________________
Guglielmo Scovazzi

Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Civil and Environmental Engineering in the Graduate School of Duke University

2017
ABSTRACT

Model Reduction and Domain Decomposition Methods for Uncertainty Quantification

by

Andres A. Contreras

Department of Civil and Environmental Engineering
Duke University

Date: ____________________
Approved:

________________________
Omar M. Knio, Co-Supervisor

________________________
Wilkins Aquino, Co-Supervisor

________________________
Olivier P. Le Maitre

________________________
Guglielmo Scovazzi

An abstract of a dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Civil and Environmental Engineering in the Graduate School of Duke University 2017
Abstract

This dissertation focuses on acceleration techniques for Uncertainty Quantification (UQ). The manuscript is divided into five chapters. Chapter 1 provides an introduction and a brief summary of Chapters 2, 3, and 4. Chapter 2 introduces a model reduction strategy that is used in the context of elasticity imaging to infer the presence of an inclusion embedded in a soft matrix, mimicking tumors in soft tissues. The method relies on Polynomial Chaos (PC) expansions to build a dictionary of surrogates models, where each surrogate is constructed using a different geometrical configuration of the potential inclusion. A model selection approach is used to discriminate against the different models and eventually select the most appropriate to estimate the likelihood that an inclusion is present in the domain. In Chapter 3, we use a Domain Decomposition (DD) approach to compute the Karhunen-Loève (KL) modes of a random process through the use of local KL expansions at the subdomain level. Furthermore, we analyze the relationship between the local random variables associated to the local KL expansions and the global random variables associated to the global KL expansions. In Chapter 4, we take advantage of these local random variables and use DD techniques to reduce the computational cost of solving a Stochastic Elliptic Equation (SEE) via a Monte Carlo sampling method. The approach takes advantage of a lower stochastic dimension at the subdomain
level to construct a PC expansion of a reduced linear system that is later used to compute samples of the solution. Thus, the approach consists of two main stages: 1) a preprocessing stage in which PC expansions of a condensed problem are computed and 2) a Monte Carlo sampling stage where samples of the solution are computed in order to solve the SEE. Finally, in Chapter 5 some brief concluding remarks are provided.
## Contents

Abstract iv

List of Tables ix

List of Figures x

Acknowledgements xv

1 Introduction 1

2 Multi-Model Polynomial Chaos Surrogate Dictionary for Bayesian Inference in Elasticity Problems 5

2.1 Introduction ................................................. 5

2.2 Physical Model and Polynomial Chaos Expansion ................. 10

2.2.1 Physical Model ........................................... 10

2.2.2 Polynomial Chaos Surrogate Model ......................... 12

2.2.3 Error in the PC surrogate model ............................ 16

2.3 Bayesian Inference and Model Selection .......................... 17

2.3.1 Bayesian Inference ........................................ 18

2.3.2 Evidence-Based Model Selection ............................ 21

2.4 Inference of Elastic Properties ............................... 23

2.4.1 PC Surrogate .............................................. 24
## List of Tables

2.1 Default parameters corresponding to the Young’s modulus prior [10, 69] and the Poisson’s ratio prior [15, 31, 33]. . . . . . . . . . . . . . 21

2.2 Comparison of inferred estimates of $\rho = E_I/E_B$ for different noise levels. The values $\mu$ and $\sigma$ correspond to the mean and standard deviation of the posterior density of $\rho$. . . . . . . . . . . . . . 29

3.1 Progression of $n_t$ for different values of $D$ with $\hat{\sigma}^2 = 2 \times 10^{-3}$ and $L = 0.1$. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 68

4.1 Evolutions with the number of subdomains $D$ of the dimension of the condensed problem ($N_\Gamma$), (averaged) numbers of local unknowns ($N_\Gamma^{(d)}$ and $N_{in}^{(d)}$), local random variables ($N_{\kappa}^{(d)}$) and local PC basis dimension $P^{(d)}$ for $N_o = 2$ and 6. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 126
List of Figures

2.1 Schematic of the elasticity problem in which a given domain $\Omega$ has an inclusion $\Omega_I$ with different material properties. The part of the domain which excludes the inclusion is denoted by $\Omega_B$. Figure (a) represents the geometry corresponding to the physical model. Figures (b) and (c) correspond to two different approximations of physical geometry used to build surrogate models. ............................................. 18

2.2 Schematic illustration of the computational domain, showing coordinate axes and geometry of inclusion. .............................. 23

2.3 Error versus polynomial order for $(E_B^{\min}, E_B^{\max}) = (1e4, 9e4), (\nu_B^{\min}, \nu_B^{\max}) = (0.45, 0.49)$, with curves generated for different values of $\kappa^{\max}$, as indicated. Estimates are obtained using (2.22) with $M_s = 10,000$. .... 25

2.4 Posterior distributions. In the plots, the red dashed-line corresponds to a Gaussian fit of the distribution and the purple vertical line indicates the location of the true values of $\xi_1$, $\xi_2$, $\xi_3$, and $\epsilon$, respectively. ................................. 27

2.5 Posterior distribution of $\rho = E_I/E_B$ (blue curves) for different values of $E_I$, with $E_B = 60$ kPa and 1% noise. The dashed red curve corresponds to the Gaussian fit of the distributions, and the purple vertical line indicates the true value of $\rho$. .......................................................... 28

2.6 Schematic of the location of the observations. New points are selected following a regular pattern in which the distant to existing locations is maximized. ......................................................... 30

2.7 Mean and standard deviation of the posterior density of $\rho = E_I/E_B$ as a function of the number of observations, $m_o$. ................. 31
2.8 Schematics of 12 different geometrical configurations. In the figures the area enclosed by yellow dotted-line correspond to the actual geometry of the inclusion, and the area enclosed by the blue solid-line corresponds to the location and shape of the inclusion assumed by the surrogate. The geometries are shown to scale. The models are ordered from left to right, top to bottom according to the value of the evidence.

2.9 The left plot shows the normalized evidence of the models. The plot on the right show the MAP estimate of $\rho$ for each model. The index of the model corresponds to the order of the geometries shown in figure 2.8 (index zero correspond to the true geometry). The purple line in the right plot corresponds to the true value of $\rho$.

2.10 MAP estimate of the ratio $\rho$ for the case of a non-homogeneous Young’s modulus for different combinations of $l$ and $\bar{\alpha}$. The MAP corresponds to the model with the highest Evidence. Each blue marker represent a different realization of the field. The purple line corresponds to a value of $\rho = 1$ that would be expected in the presence of a homogeneous field.

2.11 Selected realizations of the non-homogeneous Young’s modulus field for different values of $l$ and $\bar{\alpha}$ as depicted.

3.1 Partitioning of a square domain $\Omega$ into $D = 8$ non-overlapping subdomains.

3.2 Spectra of the global decomposition for the squared exponential covariance in (3.37) with different values of the correlation length $L$ as indicated.

3.3 Spectra of local decompositions for the squared Gaussian covariance (3.37), with $L = 0.1$, and different $D$ as indicated. The plot in (a) uses a linear scale for the $x$-axis, whereas the plot in (b) uses a logarithmic scale.

3.4 Relative spectrum error $\epsilon_{\text{spec}}$, between $U_N$ and $\hat{U}_N$, as a function of the local tolerance $\delta^*$ (see (3.40)) for different subspace dimensions $N$ as indicated. Computations use $D = 80$ and $L = 0.1$. The dashed line has unit slope.
3.5 Subspace error $\epsilon_{\text{sub}}^2(U_N)$ in (3.39) as a function of the local tolerance $\delta^*$ (see (3.40)) for different subspace dimensions $N$ as indicated. Computations use $D = 80$ and $L = 0.1$. The dashed line has unit slope. 62

3.6 Projection error $\epsilon_{\text{sub}}^2(U)$ with respect to the untruncated process, as a function of tolerance $\delta^*$ defining the local truncation through (3.40). Computations are conducted for $L = 0.1$ using $D = 80$ subdomains and different values of $N$ as indicated. The dashed black line is a reference of unit slope. 64

3.7 Projection error $\epsilon_{\text{sub}}^2(U)$ as a function of the number of subdomains $D$ and for different targeted relative accuracy $\delta$ as indicated. 65

3.8 Partitioning of $\Omega = (0,1)^2$ into $D = 20$ subdomains with the $k$-means clustering method, using different parameter values of $\rho$ and $R$ as indicated. 69

3.9 $\epsilon_{\text{sub}}^2(U)$ versus $\rho$ for a target accuracy $\delta^2 = 2 \times 10^{-3}$. The domain is partitioned into $D = 80$ subdomains using the $k$-means clustering with parameters $\rho$ and $R$ as indicated. 70

3.10 $n_t$ versus $\rho$ for a target accuracy $\delta^2 = 2 \times 10^{-3}$. Curves are generated for different values of $D$ and $R$ as indicated. 71

3.11 Computational times of local solves and reduced problem assembly versus $D$. Computations are performed on a fixed mesh with $N_e = 16,441$ elements. The correlation length is $L = 0.1$ and the target accuracy is $\delta^2 = 10^{-3}$. 74

3.12 Example of distribution among MPI processes of the subdomains (Fig. 3.12a) and block computation for the reduced problem assembly (Fig. 3.12b). Case of $D = 4$ subdomains and $N_{\text{MPI}} = 3$ MPI processes. The processes $p = 1,2,3$ are colored in blue, orange and green respectively. 78

3.13 Speedup (left) and efficiency (right) versus the number of MPI processes. Plotted are curves for different values of $D$ and $N_e$ as indicated. The ideal scaling law is shown on the left plot using a solid black line. The correlation length is $L = 0.1$ and the energy criterion for the local modes is $\delta^2/2 = 10^{-3}$. 80
4.1 Realizations of $\kappa$ (top row), deviation to the mean, $u - \mathbb{E}[u]$, of the direct solution (middle row) and differences between DD-PC and direct solutions (bottom row). The columns correspond to different realizations of $\kappa$ drawn at random using Gaussian fields with increasing variance: $\sigma^2 = 0.05$, 0.2 and 0.5 from left to right. The DD-PC solutions use $N_o = 2$, and $D = 480$ (the subdomains partition is shown in all the figures).

4.2 Mean fields $\mathbb{E}[u(x, \cdot)]$ (top row), and DD-PC error on the mean $\mathbb{E}[\hat{u}(x, \cdot)] - \mathbb{E}[u(x, \cdot)]$ for $N_o = 3$ (middle row) and $N_o = 3$ (bottom row), and three values of $\sigma^2$, as indicated, from left to right. The computations use $M = 500,000$ Monte Carlo samples to estimate the expectations and $D = 480$ subdomains.

4.3 Monte Carlo estimates of the norm of the error on the mean $\epsilon_{\text{mean}}$ as a function of the number of MC samples $M$ for different $\sigma^2$ as indicated and PC order $N_o = 2$ (left) and $N_o = 6$ (right).

4.4 Standard deviation fields $\text{Std}[u(x, \cdot)]$ (top row), and DD-PC error in the standard deviation $\text{Std}[\hat{u}(x, \cdot)] - \text{Std}[u(x, \cdot)]$ for $N_o = 3$ (middle row) and $N_o = 3$ (bottom row), and three values of $\sigma^2$, as indicated, from left to right. The computations use $M = 500,000$ Monte Carlo samples to estimate the expectations and $D = 480$ subdomains.

4.5 Relative stochastic error $\epsilon_u$ and relative error on the mean $\epsilon_m$ as functions of the PC order $N_o$, for different values of $D$ as indicated and $\sigma^2 = 0.05$ (left plot) and 0.5 (right plot).

4.6 Relative stochastic error $\epsilon_u$ and relative norm of error on the mean $\epsilon_m$ as functions of the PC order $N_o$, for different values of $D$ as indicated. The random field $G$ uses $\sigma^2 = 0.5$ with $L = 1$.

4.7 Log-Histograms of the error norm $\|u - \hat{u}\|_{L^2(\Omega)}$ (left), of the approximation error on condensed operator $\|\hat{A} - \hat{A}\|_F$ (center), and of the condition number of the approximate system $\text{cond}(\hat{A})$ (right) for PC orders $N_o = 2, 3, 9$. Case of $G$ with $\sigma^2 = 0.5$ and $L = 1$. 
4.8 Samples of the error in the solution $\|u - \hat{u}\|_{L^2(\Omega)}$ as a function of the condition number $\text{cond}[\hat{A}]$. The samples are colored according to the sign of the smallest eigenvalue of $[\hat{A}]$. Different PC orders as indicated. 124

4.9 Partitions of the computational mesh into different numbers of subdomains $D$ as indicated. 125

4.10 Local complexity (left plot) and local and global memory requirements (right plot), as a function of the number of subdomains $D$ and for two PC degree $N_o = 2$ and $N_o = 6$. Note that both plots use a log-log scale. 127

4.11 Parallel efficiency $E(N_{\text{MPI}})$ of the preprocessing stage, see (4.53), for different meshes (Fig. 4.11a) and different PC degrees (Fig. 4.11b). 135

4.12 Scaled CPU times, to generate one sample, as a function of the number of MPI processes $N_{\text{MPI}}$. The dashed lines represent ideal parallel scaling. 136
Acknowledgements

As an engineer, something that I learned early on was the concept of perpetual motion machine; I right away understood that, simply put, this is an idea that only the uninformed would entertain. However, years later I have come across something that without a doubt violates the laws of thermodynamics: human kindness. The concept of “paying it forward” is a mechanism that not only “transfers energy” from one person to another “without losses”, but more often than not, the output is significantly greater than the sum of the inputs. To my surprise, I have learned that human kindness can effectively create something out of nothing and, without question, is the greatest perpetual motion machine ever conceived. I have benefited tremendously from this mechanism, and I have received so much from so many, that it would be virtually impossible for me to list all of their names here. I will, however, attempt to list those that occupy a more recent position in memory.

First, I would like to thank my adviser, Dr. Omar Knio, for providing an invaluable opportunity of working under his supervision and guidance. In addition of benefiting from his vast technical knowledge, I learned a lot from the kind and friendly way he treats others. I also want to thank Dr. Olivier Le Maître, who was like a second adviser to me, for taking the time to answer my many questions and helping me develop a deeper understanding of the Uncertainty Quantification field –
I truly enjoyed our weekly Skype meetings. I am also very grateful to Dr. Wilkins Aquino, who opened the doors for me to come to Duke University and has offered his support and guidance since the very beginning. I also want to thank Dr. Guglielmo Scovazzi for his support as part of my committee. Moreover, I appreciate the support and collaboration of the REXSSS group at Sandia National Laboratories, Livermore. In particular, I would like to thank my friend Dr. Paul Mycek, with whom I had a very close collaboration and countless interesting discussions.

I would also like to thank my mentors at Utah State University. Specially, I am grateful to Dr. James Powell (who was the one who initiated me on this path), Dr. Reyhan Baktur, and Dr. Todd Moon, for their support when I applied to graduate school. I also appreciate the support of my friends and family both in the United States and the Dominican Republic; in particular Manuel Diaz, who was always available to hear me talk about my research and very frequently offered quite helpful advice.

Finally, I would like to thank my family for always standing by my decisions and having faith in me. Especially, my dear wife Lori Diaz, who has offered her unconditional support and love for almost a decade (and counting).

This work was supported by the US Department of Energy (DOE), Office of Science, Office of Advanced Scientific Computing Research, under Award Numbers DE-SC0008789 and DE-SC0010540. Support from the SRI Center for Uncertainty Quantification in Computational Science and Engineering at King Abdullah University of Science and Technology is also acknowledged. Moreover, the parallel experiments were carried out on Turing, the IBM Blue Gene/Q machine of IDRIS (Institute for Development and Resources in Intensive Scientific Computing).
The world we live in is intrinsically uncertain and, as such, when we seek to model and understand it, it is necessary to account for this uncertainty. The field of Uncertainty Quantification (UQ) is a rapidly growing area of research that provides tools to both characterize the uncertainty in our models and to understand its propagation through the system. A challenge inherent to UQ problems is the increased computational cost. Significant research has been devoted to develop techniques that reduce the computation cost of performing UQ. This dissertation contributes another step in this same direction by focusing on model reduction techniques and domain decomposition methods to accelerate UQ.

This document covers two acceleration methods that are explained in three chapters. The first method (presented in Chapter 2) deals with model reduction techniques to reduce the computational cost of inferring stiff inclusions embedded in a soft matrix. The material in this Chapter is based on a paper that has been
published in the Probabilistic Engineering Mechanics Journal (see [19]). The other method (presented in Chapters 3 and 4) use domain decomposition techniques with the ultimate goal of accelerating the solution of a Stochastic Elliptic PDE solver. These two Chapters are based on two papers that are currently in preparation and will be soon submitted for publication. All three chapters have their own detailed introduction and motivation, as well as their own concluding remarks. A detailed outline of the dissertation is provided below.

In Chapter 2, we present a method for inferring the presence of an inclusion inside a domain; the proposed approach is suitable to be used in a diagnostic device with low computational power. Specifically, a Bayesian framework is used for the inference of stiff inclusions embedded in a soft matrix, mimicking tumors in soft tissues. We rely on a Polynomial Chaos (PC) surrogate to accelerate the inference process. The PC surrogate represents the dependence of the displacements field with the random elastic moduli of the materials, and are computed by means of the Stochastic Galerkin (SG) projection method. Moreover, the inclusion’s geometry is assumed to be unknown, and this is addressed by using a dictionary consisting of several geometrical models with different configurations. A model selection approach based on the evidence provided by the data (Bayes factors) is used to discriminate among the different geometrical models and select the most suitable one. The idea of using a dictionary of pre-computed geometrical models helps to maintain the computational cost of the inference process very low, as most of the computational burden is carried out off-line for the resolution of the SG problems. Finally, numerical tests are used to validate the methodology, assess its performance, and analyze the robustness to model errors.
In Chapter 3, we present a method to efficiently determine the dominant Karhunen-Loève (KL) modes of a random process; more generally, the method solves Fredholm equations of the second kind. The truncated KL expansion is one of the most common techniques for the approximation of random processes, primarily because it is an optimal representation, in the mean squared error sense, with respect to the number of random variables in the representation. However, finding the KL expansion involves solving integral problems, which tends to be computationally demanding. This work addresses this issue by means of a divide-and-conquer strategy based on a domain decomposition approach, enabling the efficient computation of a possibly large number of dominant KL modes. Specifically, the computational domain is partitioned into smaller non-overlapping subdomains, over which independent local KL decompositions are performed to generate local bases which are subsequently used to discretize the global modes over the entire domain. The latter are determined by means of a Galerkin projection. The procedure leads to a reduced Galerkin problem, whose size is not related to the dimension of the underlying discretization space, but is actually determined by the desired accuracy and the number of subdomains. It can also be easily implemented in parallel. Another key aspect of the work presented in this chapter is that we analyze in detail the relationship between local random variables at the subdomain level and global random variables that characterize the random process over the entire domain (this relationship is exploited in Chapter 4 where we use the local random variables to accelerate a Monte Carlo sampling-based Stochastic Elliptic PDE solver). Finally, extensive numerical tests are used to validate the methodology and assess its serial and parallel performance.

In Chapter 4, we present an acceleration strategy for a Monte Carlo sampling-
based Stochastic Elliptic PDE solver. The proposed strategy relies on domain decomposition techniques to partition the global domain into smaller non-overlapping subdomains, and over each subdomain Polynomial Chaos (PC) expansions of local boundary-to-boundary (BtoB) maps are constructed. The construction of the PC expansions is done at a reduced cost by taking advantage of the notion of local random variables discussed in Chapter 3. The PC expansions of the local BtoB maps are combined together to construct a PC expansion of the global condensed problem. Basically, our approach is divided into two main stages: 1) a preprocessing stage in which PC expansions of a condensed problem are computed and 2) a Monte Carlo sampling stage where samples of the solution are computed in order to solve the SPDE. Having a PC expansion of the condensed problem significantly reduces the cost of solving the SPDE via MC sampling, since constructing the system amounts to simple polynomial evaluations. Both the preprocessing and the sampling stage can be implemented in parallel and in particular, the preprocessing stage can be implemented with minimal communication. A numerical example is used to showcase the accuracy of the approach, its convergence properties, and parallel efficiency.

Finally, in Chapter 5 some concluding remarks are provided.
2

Multi-Model Polynomial Chaos Surrogate Dictionary for Bayesian Inference in Elasticity Problems


2.1 Introduction

The nondestructive characterization of the parameters describing a physical system is a task of great importance and interest in various disciplines within science and engineering. Examples of such tasks include seismic imaging [43, 94], health monitoring of infrastructure [80, 1, 39], and more recently elasticity imaging [2, 20]. Elasticity imaging is a very promising branch of medical diagnosis which applies inverse problems techniques to compute the elasticity modulus given a set of measurements of
a displacement or velocity field that is the result of some excitation force [67]. The idea is inspired by the palpation technique used by doctors to determine the presence of abnormal tissue through the sense of touch [79, 83]. Palpation, however, is limited in detecting anomalies that lie deep in the body or which are too small [26]; moreover, it tends to be qualitative as opposed to quantitative. Elasticity imaging takes palpation to the next level by extending its range and effectiveness, all in a more quantitative manner. The general goal of this work is to use a collection of models within a Bayesian framework to estimate the contrast between the elastic properties of different regions in a given domain.

The elasticity imaging technique encompasses three basic steps: first, the body is deformed through an applied external load, then the deformation field is measured (e.g. using ultrasound techniques), and finally the elastic properties are estimated by solving an inverse problem. To approach this problem, Oberai et al. [67] assume that the displacements are governed by the equations of equilibrium of an incompressible, linear-elastic solid undergoing small, quasi-static deformation, and cast the problem as a non-linear optimization problem; the objective is to find a shear modulus field that minimizes the discrepancy between the measured and predicted displacement fields. Another optimization approach is based on minimizing the Modified Error in Constitutive Equation Functional [8], which measures the discrepancy in the constitutive equations that connect kinematically admissible strains and dynamically admissible stresses in addition to measuring the discrepancy between the measured and predicted displacement fields. Other approaches include direct inversion methods [75, 89, 72], but these methods, although computationally less expensive, tend to be more sensitive to noise in measurement data. All these approaches are deter-
ministic, and consequently result in a single estimate of the elastic modulus, which doesn’t accommodate for the quantification of uncertainty.

Important insights emerge by approaching inverse problems using a probabilistic framework. Some of the methods introduced to deal with this problem include the extended maximum likelihood method [25], the spectral stochastic method [65, 41], the sparse grid collocation approach [101, 5], stochastic reduced order models [98], and the Bayesian inference approach [91, 64]. In the Bayesian formalism, one obtains additional insight by computing a probability distribution that summarizes all available information about the elastic moduli (e.g., we can estimate moments, marginal distributions, quantiles), as opposed to the single value obtained in the deterministic setting. Specifically, in the context of elasticity imaging, Koutsourelakis [48] use a Bayesian framework to obtain probabilistic estimates of the material properties that account for various possible sources of uncertainty; this work deals with simplified geometries and large contrast ratio of the elastic properties. Another interesting approach is proposed by Iglesias [38], where the Bayesian framework is applied under an infinite dimensional setting; this work, however, is limited to deterministic (known) elastic properties and it requires an appropriate prior model.

For complex forward models, extracting information from the posterior distribution can be very computationally expensive. Several techniques are applied to address this computational challenge, such as the use of a two-stage MCMC to increase the acceptance rate of the algorithm by using an inexpensive approximation of the posterior distribution [24, 23, 17]; the use of proper orthogonal decomposition (POD) to construct a reduced-order model for the direct simulations [97, 40]; the use of adaptive hierarchical sparse grid collocation (ASGC) to obtain an approximate
stochastic solution to the forward problem using piecewise linear interpolation [55]; and the use of Polynomial Chaos (PC) to approximate the solution of the stochastic forward model either through collocation [7, 57] or through the stochastic Galerkin method [59]. A related application of PC representations in the context of inverse acoustic scattering problems is found in [27], where PC expansions are integrated with optimization methods for the probabilistic characterization of hidden obstacles and inclusions in acoustic media.

The objective of this work is to develop a method that can be used in a diagnostic device with a low computational power to quickly assess the presence of an inclusion in a given domain. To achieve this, the proposed approach breaks the process in two steps: (1) an offline or pre-processing step where surrogate models are constructed for different geometrical models, and (2) an online step where a model selection and inference are performed on the basis of observations to assess the presence of an inclusion. This is advantageous, since the main computational cost is carried by the construction of the surrogate models, which is something that can be done offline with a dedicated computer. Thus, once the surrogate models have been constructed, the computational cost of the model selection and inference problem is relatively low and can be effectively handled by the diagnostic device with limited computational power. In more details, we extend the Bayesian approach proposed by Marzouk et al. [59] to the case of multiple geometrical models as follows. First, a dictionary of inclusion geometries is considered and for each of these geometries a suitable Polynomial Chaos expansion of the displacement field is computed, in terms of the unknown parameters (in our case the Young’s modulus and the Poisson’s ratio in soft matrix and inclusion) by means of the Stochastic Galerkin (SG) method [28, 52].
The SG allows for a fine control approximation error. When observations are made available, the PC surrogates can be used to derive corresponding approximations of the posterior distribution for the elastic properties given a geometry. Then, these posteriors can be compared by computing the evidences or Bayes factors of the geometrical models, in order to rank them and select the best one (or few best ones). The posterior distribution(s) of the elastic properties for the best model (or best ones) can then be used to reach a decision confirming or refuting the presence of an inclusion, analyzing for instance the ratios between the mean properties in the inclusion and soft matrix domains.

The outline of the Chapter is as follows. In section 2.2, we introduce the mechanical model of the elasticity problem and derive the Polynomial Chaos expansions of the displacement field for a given geometry. In section 2.3, we describe the use of the Bayesian framework to solve both inverse problem and the model selection problem. In section 2.4 we present some numerical results showing the behavior of the approach when the exact geometry of the model is known. Then, in section 2.5, we look at the case when the exact model geometry is unknown and construct a dictionary of surrogate models and rank them based on the evidence provided by the data; also we test the robustness of the approach with respect to errors in the mechanical model. Finally, in section 2.6 we provide concluding remarks.
2.2 Physical Model and Polynomial Chaos Expansion

2.2.1 Physical Model

Continued problem

The strong form of the equilibrium equations of a linear-elastic solid undergoing static deformation due to boundary loads and displacements can be expressed as:

\[ \nabla \cdot \sigma = 0 \quad \text{in } \Omega \quad (2.1) \]

with boundary conditions:

\[ \sigma \cdot n = \tau \text{ on } \Gamma_{\tau}, \quad u = u_0 \text{ on } \Gamma_{u}, \quad (2.2) \]

where \( \sigma = C : \varepsilon \equiv C_{ijkl} \varepsilon_{kl} \) is the stress tensor; \( \varepsilon(u) = (\nabla u + \nabla u^T)/2 \) is the linearized strain tensor; \( u \) is the displacement field; \( n \) the unit normal to the boundary; \( \tau \) is the traction vector; \( \Omega \) is the spatial domain; \( \Gamma_{\tau} \) and \( \Gamma_{u} \) form a partition of the boundary \( \Gamma \) of \( \Omega \); \( u_0 \) is the essential boundary condition; and \( C \) is the fourth-order constitutive tensor of linear elasticity. Under the assumption of an isotropic medium, the constitutive tensor has only two independent elastic constants and can be written as:

\[ C_{ijkl} = \lambda \delta_{ij} \delta_{kl} + \mu (\delta_{ik} \delta_{jl} + \delta_{il} \delta_{jk}) \quad (2.3) \]

where \( \delta_{ij} \) is the Kronecker delta and \( \lambda \) and \( \mu \) are the Lamé constants [84]. This decomposition of \( C \) is very advantageous for the computation of the PC coefficients described in section 2.2.2.

The forward problem consists in finding the displacement field \( u \) that satisfies (2.1) for a given constitutive tensor \( C \) (i.e. known material properties). The weak formulation of the forward problem is obtained after defining the space of
trial functions, $S = \{ u \mid u_i \in H^1(\Omega), u = u_0 \text{ on } \Gamma_u \}$, and the space of test functions, $V = \{ v \mid v_i \in H^1(\Omega), v = 0 \text{ on } \Gamma_u \}$. Multiplying (2.1) by an arbitrary $v \in V$, integrating over the spatial domain, using the divergence theorem, and the symmetry of $C$ we get:

$$a(u, v) = (\tau, v), \quad \forall \ v \in V; \tag{2.4}$$

where

$$a(u, v) = \int_\Omega \varepsilon(v) : C : \varepsilon(u) d\Omega, \quad (\tau, v) = \int_{\Gamma_\tau} \tau \cdot v d\Gamma_\tau. \tag{2.5}$$

The function $u \in S$ that satisfies (2.4) is the equivalent weak solution of (2.1).

\textit{Finite element formulation}

Using standard Voigt notation [76], the displacement fields, test functions, and their derivatives are expressed as:

$$u^h = [N]u, \quad v^h = [N]v, \quad \varepsilon^h(u) = [B]u, \quad \varepsilon^h(v) = [B]v, \tag{2.6}$$

where $\{u\}$ and $\{v\}$ belong to $\mathbb{R}^m$, $m$ depends on the resolution of the finite element basis, whereas $[N]$ and $[B]$ represent matrices of finite element shape functions and their derivatives with respect to spatial coordinates, respectively. Substituting the above approximations into the variational problem in (2.4) we get:

$$[A]u = \{F\}, \quad \tag{2.7}$$

where

$$[A] = \sum_{\text{elements}} \int_{\Omega^e} [B]^T [C] [B] d\Omega, \quad \{F\} = \sum_{\text{elements}} \int_{\Gamma^e} [N]^T \tau d\Gamma. \tag{2.8}$$
and $[C]$ is the matrix representation of the fourth-order tensor $C$. Thus, the solution of the discretized forward problem reduces to inverting the linear system given by (2.7).

Notice that through (2.7) we can define a map

$$\mathcal{T} : [C] \to \{u\},$$

(2.9)

that takes a constitutive matrix as its input, and outputs a discretized displacement field. This mapping, of course, is determined by the geometry of the problem and its boundary conditions. We will refer to this map as the Deterministic Forward Map.

### 2.2.2 Polynomial Chaos Surrogate Model

From now on we will study the situation where the Lamé constants, $\lambda$ and $\mu$, are stochastic quantities (i.e. $[C]$ is random) and will layout the framework to accommodate this situation. As an initial step we seek to construct a stochastic map $\hat{\mathcal{T}}$ that approximates (2.9) when $[C]$ is random and that can be efficiently sampled. This map will be constructed using polynomial chaos expansion [28, 52], and we will refer to it as the surrogate model.

In this work, we focus on the specific class of random processes that are in $L^2$, the space of second-order quantities. Following the notation in [52], let $(\Theta, \Sigma_\Theta, P_\Theta)$ be a probability space and $\theta$ a random event belonging to $\Theta$. We denote $L^2(\Theta, P_\Theta)$ the space of second-order random variables defined on $(\Theta, \Sigma_\Theta, P_\Theta)$ equipped with the inner product $\langle \cdot, \cdot \rangle$ and associated norm $\| \cdot \|_\Theta$:

$$\langle X, Y \rangle = \int_\Theta X(\theta)Y(\theta) dP_\Theta(\theta) = \mathbb{E}[XY] \quad \forall X, Y \in L^2(\Theta, P_\Theta)$$

(2.10)

$$X \in L^2(\Theta, P_\Theta) \iff \langle X, X \rangle = \| X \|^2_\Theta < \infty,$$

(2.11)
where $\mathbb{E} [\cdot]$ is the expectation operator.

The discrete counterpart of (2.3) is

$$[C] = \lambda [C^\lambda] + \mu [C^\mu],$$  \hspace{1cm} (2.12)

where $[C^\lambda]$ and $[C^\mu]$ are constant matrices independent of both $\lambda$ and $\mu$. The parameters $\lambda$ and $\mu$ are bounded quantities and we will model them as a random processes $\lambda(x, \theta), \mu(x, \theta) \in L^\infty(\Omega) \times L^2(\Theta, P_\Theta)$, and represent them in terms of the truncated PC expansions:

$$\lambda(x, \theta) \approx \sum_{\gamma=0}^{P_\lambda} \lambda_\gamma(x) \Psi_\gamma(\xi(\theta)),$$

$$\mu(x, \theta) \approx \sum_{\gamma=0}^{P_\mu} \mu_\gamma(x) \Psi_\gamma(\xi(\theta)),$$  \hspace{1cm} (2.13)

where $\xi(\theta) = \{\xi_1(\theta), \xi_2(\theta), \cdots, \xi_N(\theta)\}$ is a $N$-dimensional random vector, with independent components having joint density $p_{\xi}$ and $P_\lambda + 1$ and $P_\mu + 1$ are the number of terms in each corresponding expansion. The $\Psi_\gamma$ form an orthonormal basis, and are indexed such that $\Psi_0 = 1$. For simplicity we will take the number of terms in both expansions to be the same, i.e. $P_\lambda = P_\mu = P$. Furthermore, the number of terms in each expansion is determined by the number of random variables, $N$, and the polynomial degree of the PC expansion, $p$ (again, we take the same polynomial degree for both expansions); for a total order truncation, it is given by [52]:

$$P + 1 = \frac{(N + p)!}{N! p!}.$$  \hspace{1cm} (2.14)

We also represent the stochastic discrete displacement field as a random field using
a PC expansion of the form:

\[ \{ \hat{u}(\theta) \} \simeq \sum_{\alpha=0}^{P_u} \{ u_\alpha \} \Psi_\alpha(\xi(\theta)), \]  

(2.15)

where \( \xi(\theta) \) is the same random vector introduced above, \( \{ u_\alpha \} \in \mathbb{R}^m \) are the PC coefficients, and \( P + 1 \) is the number of terms in the PC expansion of \( u \). \( P_u \) is determined from \( N \) and \( p_u \) using (2.14). Without loss of generality we take \( P_u = P \).

**Computation of PC coefficients**

In this section we will describe the method to compute the coefficients in (2.15). We start by substituting (2.13) into (2.12) to get the following PC expansion for \( r_C \):

\[ r_C \simeq \sum_{\gamma=0}^{P} [C_\gamma] \Psi_\gamma(\xi) = \sum_{\gamma=0}^{P} (\lambda_\gamma [C^\lambda] + \mu_\gamma [C^\mu]) \Psi_\gamma(\xi). \]  

(2.16)

Notice that the decomposition of \( [C] \) as given in (2.12) provides an explicit way to compute the \( [C_\gamma] \)'s from the coefficients in the PC expansions of \( \lambda \) and \( \mu \). Substituting (2.16) into (2.8) and using the Galerkin projection on (2.7) accounting for (2.15), we get the following coupled system of \( P + 1 \) equations for the PC coefficients:

\[ [A_0] \{ u_\beta \} + \sum_{\alpha=0}^{P} \sum_{\gamma=1}^{P} [A_\gamma] \{ u_\alpha \} \{ \Psi_\gamma \Psi_\alpha, \Psi_\beta \} = \{ F_\beta \}, \quad \beta = 0, \ldots, P, \]  

(2.17)

where

\[ [A_\gamma] = \sum_{\text{elements}} \int_{\Omega^e} [B]^T [C_\gamma] [B] d\Omega \quad \gamma = 0, \ldots, P. \]

Equation (2.17) can be written more compactly as

\[ [A] \{ U \} = \{ F \}, \]  

(2.18)
where \([\mathcal{A}] = [\mathcal{A}] + [\mathcal{A}]; \quad [\mathcal{A}]_{\alpha,\beta} = [A_0] \delta_{\alpha,\beta}; \quad [\tilde{\mathcal{A}}]_{\alpha,\beta} = \sum_{\gamma=1}^{P} [A_{\gamma}] \langle \Psi_{\gamma}, \Psi_{\alpha}, \Psi_{\beta} \rangle; \quad \{\mathcal{U}\}_{\beta} = \{u_{\beta}\}; \quad \text{and } \{\mathcal{F}\}_{\beta} = \{F_{\beta}\}; \quad 0 \leq \alpha, \beta \leq P.

We notice that the matrix \([\mathcal{A}]\) in the deterministic case, (2.7), is \(m \times m\), whereas the matrix \([\mathcal{A}]\) in (2.18) is \((m \cdot (P + 1)) \times (m \cdot (P + 1))\). Thus, if \(m\) is large to start with, which is typically the case, solving for \(\{\mathcal{U}\}\) might incur a significant computational cost. However, effectively exploiting the structure of \([\mathcal{A}]\) could offer great computational savings. A simple approach \([70, 49, 52]\) is to take advantage of the decomposition \([\mathcal{A}] = [\mathcal{A}] + [\tilde{\mathcal{A}}]\), which allows us to solve for \(\{\mathcal{U}\}\) iteratively according to:

\[
\{\mathcal{U}\}_{k+1} = [\tilde{\mathcal{A}}]^{-1} \left( \{\mathcal{F}\} - [\tilde{\mathcal{A}}]\{\mathcal{U}\}_k \right).
\] (2.19)

Since \([\tilde{\mathcal{A}}]\) is block-diagonal, the iterations in (2.19) can be decoupled into \((P + 1)\) independent updates for the \(\{u_{\beta}\}\)’s, requiring the unique factorization of \([A_0] \in \mathbb{R}^{m \times m}\). Thus, the computation can be parallelized, and the same deterministic solver can be used in the iterations.

So far we have expressed the constitutive matrix \([C]\) in term of the Lamé constants \(\lambda\) and \(\mu\) because they facilitate the computation of the PC expansion of \(u\). The proposed approach can be immediately adapted to work in terms of alternative pairs of elastic constants. For instance, it might be desirable to think in terms of the Young’s modulus (\(E\)) and the Poisson’s ratio (\(\nu\)). When this is the case, we can first determine the corresponding PC expansions for \(E\) and \(\nu\) and substitute them in the following relations

\[
\lambda(x, \theta) = \frac{E(x, \theta) \nu(x, \theta)}{(1 + \nu(x, \theta))(1 - 2\nu(x, \theta))}, \quad \mu(x, \theta) = \frac{E(x, \theta)}{2(1 + \nu(x, \theta))}.
\] (2.20)
using the Galerkin product and division [52]. The polynomial order in the expansions is selected such that the error introduced in this step is sufficiently small. The same can be done for other pairs of elastic constants such as the shear modulus and the bulk modulus. In the present work, we considered model based in \((E, \nu)\) variabilities, though we take the Poisson’s ratio in a range \(\nu \approx 0.5\) where the material is nearly incompressible and in agreement or consistent with other properties variability reported in the literature (see e.g. [29, 15, 31, 33]).

2.2.3 Error in the PC surrogate model

In order to assess the fidelity of the surrogate, we rely on the following error measure,

\[
\hat{\vartheta}^2 \equiv \frac{\mathbb{E} \left[ \| u(\xi) - \hat{u}(\xi) \|_{L^2(\Omega)}^2 \right]}{\mathbb{E} \left[ \| u(\xi) \|_{L^2(\Omega)}^2 \right]},
\]  

(2.21)

where \(u\) is the displacement computed using the full forward model and \(\hat{u} = \sum_{\alpha=0}^{P} u_{\alpha} \Psi_{\alpha}\) is the displacement computed using the surrogate model.

We approximate (2.21) by Monte Carlo sampling, computing the sample mean and integrating over the discretized finite-element domain, i.e. we estimate \(\hat{\vartheta}\) using

\[
\hat{\vartheta}^2 \approx \hat{\vartheta}^2 \equiv \frac{\sum_{i=1}^{M_S} \| u(\xi^{(i)}) - \hat{u}(\xi^{(i)}) \|_{L^2(S)}^2}{\sum_{i=1}^{M_S} \| u(\xi^{(i)}) \|_{L^2(S)}^2},
\]

(2.22)

where \(M_S\) is the number of Monte Carlo samples and \(\xi^{(i)}\) are i.i.d. random realizations of \(\xi(\theta)\). In practice, a low number of samples \(M_S\) can be used to get a sense of how well the surrogate approximates the full model over the entire range of \(\xi\). Further, we recall that this surrogate construction is performed offline, and not during the inference step.
One parameter that impacts $\vartheta$ is $P$ (or $p$). Recall that this parameter represents the number of terms in the approximations of $[C]$ and $\hat{u}$. In section 2.4, we will analyze how the error $\vartheta$ changes as a function of $P$. This will allow us to characterize how the PC expansion of the displacement (i.e. the surrogate model) converges as we increase the number of terms in the approximation.

2.3 Bayesian Inference and Model Selection

This section focuses on applying the Bayesian framework to solve two tasks: 1) the inference problem of estimating $[C]$ for a given geometrical model and 2) for a given collection of models, how to rank them in order to select the ones that are more suitable.

Figure 2.1 shows the schematic of the physical model and also two different approximations of the physical model that can be used for the construction of a surrogate model. In the first part of this section, we discuss the case in which the physical geometry is completely known and we describe how to estimate the material properties of the domain using Bayesian inference. However, in practice, the actual shape and/or location of the inclusion is unlikely to be completely known, and in the second part of this section we present an approach that consists in creating a dictionary of surrogate models each built with a different geometrical configuration for the size and/or location of the inclusion. We apply a model selection analysis based on the evidence provided by the data to discriminate among the different models.
2.3.1 Bayesian Inference

We now focus on estimating $[C]$ using Bayesian inference. We assume knowledge of a geometry $G$ and the availability of $m_o$ noisy observations of the displacement $u$ at points $x_i \in \Omega$. Since $[C]$ is parametrized by $\xi$, the Bayesian inference is recast in terms of the parameter. We rely on a Metropolis-Hastings algorithm to determine the posterior. The covariance matrix of the proposal distribution is determined using information from the Hessian as outlined in A.

Using Bayes’ formula, the posterior distribution of $\xi$ is given by

$$p(\xi, \epsilon | \{Y_i\}, G) \propto p_Y(\{Y_i\}|\xi, \epsilon, G)p_\xi(\xi)p_\epsilon(\epsilon),$$

(2.23)

where $p_Y(\{Y_i\}|\xi, \epsilon, G)$ is the likelihood of the observations, $p_\xi(\xi)$ is the prior, $p_\epsilon(\epsilon)$ is the prior distribution of the noise hyper-parameter $\epsilon$. The prior could be made dependent on the geometry $G$, but for the sake of simplicity we will take the same prior for all the geometries.
We assume that the noise in the observations follows a Gaussian distribution, *i.e.* $Y_i = u(x_i) + \eta_i$, $i = 1, \ldots, m_o$, where the $\eta_i$’s are i.i.d centered Gaussian RVs, with variance $\epsilon^2$. Since the $\eta_i$’s are independent and Gaussian, it follows that

$$p_Y(Y_i|\xi, \epsilon, G) = \prod_{i=1}^{m_o} \frac{1}{\sqrt{2\pi\epsilon^2}} \exp\left(-\frac{(\hat{u}(x_i, \xi; G) - Y_i)^2}{2\epsilon^2}\right). \quad (2.24)$$

Notice that in (2.24) we have replaced $u$ by $\hat{u}$ (the surrogate model approximation). This substitution allows us to sample the posterior distribution at low computational cost [59, 37]. Of course, success of this approximation depends on the availability of a faithful surrogate, as further analyzed below.

We rely on an uninformative Jeffreys prior [62, 63] for the hyper-parameter $\epsilon > 0$, *i.e.* we set:

$$p_\epsilon(\epsilon) \sim \frac{1}{\epsilon}. \quad (2.25)$$

As for the prior, we assume that material properties of the background and the inclusion are homogeneous, *i.e.* they are constant along the spatial dimension in each respective domain. Furthermore, we work in terms of Poisson’s ratio and Young’s modulus, and assume both of them to be uncertain. The background and the inclusion are assumed to have the same, but uncertain, Poisson’s ratio ($\nu(x, \theta) \equiv \nu(\theta)$) and a potentially different Young’s modulus ($E(x, \theta)$). Consequently, we express the Young’s modulus as:

$$E(x, \theta) = E_B(\theta) (1 - \mathbb{I}(x)) + E_I(\theta)\mathbb{I}(x) \quad (2.26)$$

where $E_B$ and $E_I$ are the Young’s moduli of the background and the inclusion, respectively; $\mathbb{I}(x)$ is an indicator function that takes a value of one inside the inclusion and
zero outside. Thus, the prior distribution is parameterized using a three-dimensional germ, \( \xi(\theta) = \{\xi_1(\theta), \xi_2(\theta), \xi_3(\theta)\} \); one dimension is used to characterize the uncertainty in Poisson’s ratio and the other two to characterize the uncertainty in the Young’s modulus of the background and the inclusion.

The Poisson ratio (\( \nu \)) and the background Young’s modulus (\( E_B \)) are modeled as uniform random variables on the intervals \( (\nu^{\text{min}}, \nu^{\text{max}}) \) and \( (E_B^{\text{min}}, E_B^{\text{max}}) \), respectively. The inclusion Young’s modulus (\( E_I \)) is modeled as \( E_I = \kappa E_B \), where \( \kappa \) is uniformly distributed on \((1, \kappa^{\text{max}})\). Parameterizing in terms of the \( \xi_i \)'s we have:

\[
\begin{align*}
\nu(\theta) &= \nu(\xi_1(\theta)) = \nu^{\text{min}} + (\nu^{\text{max}} - \nu^{\text{min}})\xi_1(\theta), \quad \xi_1 \sim U(0, 1) \\
E_B(\theta) &= E_B(\xi_1(\theta)) = E_B^{\text{min}} + (E_B^{\text{max}} - E_B^{\text{min}})\xi_2(\theta), \quad \xi_2 \sim U(0, 1) \\
E_I(\theta) &= E_I(\xi_1(\theta)) = [1 + (\kappa^{\text{max}} - 1)\xi_3(\theta)]E_B(\theta), \quad \xi_3 \sim U(0, 1)
\end{align*}
\]  

where the parameters \( \nu^{\text{min}}, \nu^{\text{max}}, E_B^{\text{min}}, E_B^{\text{max}}, \) and \( \kappa^{\text{max}} \) are variables that encode the prior knowledge. In particular, \( \mu^{\text{max}} \) characterizes the amount by which \( E_I \) surpasses \( E_B \). For this parametrization we have almost surely \( E_I > E_B \). Note that following the definitions above, the \( \xi_i \)'s are uniformly distributed over \([0,1]\). Therefore, the PC basis functions \( \Psi_\alpha(\xi_1, \xi_2, \xi_3) \) consist in products of univariate shifted-Legendre polynomials.

Table 2.1 shows the values for the prior parameters that are used, unless otherwise specified, throughout this Chapter. The values are based on the ranges used in \([15, 31, 33, 10, 69]\). Note that the Poisson’s ratio has been treated as random to show the generality of the method, although in imaging application it is usually considered known and fixed, assuming incompressible tissues. The prior distribution is however selected so that the soft tissue is modeled as nearly incompressible for almost any
Table 2.1: Default parameters corresponding to the Young’s modulus prior [10, 69] and the Poisson’s ratio prior [15, 31, 33].

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E_B^{\text{min}}$</td>
<td>$1 \times 10^4\text{Pa}$</td>
</tr>
<tr>
<td>$E_B^{\text{max}}$</td>
<td>$9 \times 10^4\text{Pa}$</td>
</tr>
<tr>
<td>$\nu^{\text{min}}$</td>
<td>0.45</td>
</tr>
<tr>
<td>$\nu^{\text{max}}$</td>
<td>0.49</td>
</tr>
<tr>
<td>$\kappa^{\text{max}}$</td>
<td>6</td>
</tr>
</tbody>
</table>

2.3.2 Evidence-Based Model Selection

As stated before, complete knowledge of the geometry might not be available. However, in order to build the surrogate model that speeds up the MCMC algorithm, we need the specifics of the geometry. To address this we propose to build not one, but many surrogates using different geometrical models. So, even if we don’t know the true geometrical model, we can find a good approximation if our dictionary is extensive enough. For instance, for the physical model shown in figure 2.1a, a surrogate built with the geometry shown in figure 2.1b would allow us to properly infer the presence of an inclusion in the domain; this will be further illustrated in section 2.5.

A remaining question, however, is how to discriminate among the different models in our dictionary to select a suitable one. To address this, we consider the probability of observing the data given the geometrical model $\mathcal{G}_k$ (commonly referred as the evidence)

$$I_k \doteq p_Y(Y_i|\mathcal{G}_k) = \int p_Y(Y_i|\xi, \epsilon, \mathcal{G}_k)p_\epsilon(\epsilon)p_\xi(\xi)d\xi d\epsilon.$$  (2.30)
Evaluating the integrals $I_k = \int p_Y(\{Y_i\}|\xi, \epsilon, G_k)p_\xi(\xi)p_\epsilon(\epsilon)d\xi d\epsilon$ can be computationally difficult and expensive. However, when the posterior density is highly peaked about its maximum a posteriori (MAP) $(\tilde{\xi}, \tilde{\epsilon})$, we can use Laplace’s method to approximate the integral $I$ as follows [46]:

$$I_k \approx \hat{I}_k = (2\pi)^{(N+1)/2} [-H_{LP}]^{-1/2} p_Y(\{Y_i\}|\tilde{\xi}, \tilde{\epsilon}, G_k)p_\xi(\tilde{\xi})p_\epsilon(\tilde{\epsilon}),$$

(2.31)

where $N$ is the dimension of $\xi$ and $[H_{LP}]$ is the Hessian matrix of the log-posterior distribution of model $G_k$. The computation of the Hessian is made relatively straightforward thanks to the fact that we are using PC representations. Additional details regarding the computation of the Hessian are provided in A. Furthermore, the MAP point, $(\tilde{\xi}, \tilde{\epsilon})$, can be obtained either by using an optimization approach or by using the MCMC algorithm to sample the posterior. In both cases, having a surrogate model provides significant computational savings. Moreover, if an optimization approach were to be used, one could take advantage of the availability of the exact PC Hessian to speed up the process. In this Chapter we use the sampled posterior to compute the MAP. Since several models are to be compared, we have to sample several posteriors which are expected to differ from a model to another. The sampling of the different models is however inexpensive, thanks to the availability of the surrogates, and it can be performed in parallel. More advanced sampling techniques could be considered, including the exploration of the models space in addition of the parameters space. This has not been considered in the present work as it was found easier to fit proposal parameters adapted to each model, than determining more general proposal parameters.

We have stressed the importance of evaluating the evidence at a low compu-
tational cost because this step would be performed by a diagnosis device with a relatively low computational power. Conversely, building an extensive dictionary of surrogates for different geometries can be costly, but this process is part of a pre-processing stage and can be done “offline” with a dedicated, more capable computer. Whence the evidences $I_k$ have been approximated a Bayes factor analysis can be applied to rank the geometrical models, with the first rank corresponding to the model with the highest evidence.

2.4 Inference of Elastic Properties

In this section, we apply the methodology above to a simple example consisting of a rectangular solid with an inclusion. A uniform static pressure (magnitude $F = 1\text{Pa}$) is applied at the top of the solid while the bottom part is restrained from motion.
A schematic representation of the setup is shown to scale in figure 2.2. We assume that the background and the inclusion are homogeneous materials, i.e. the Poisson ratio and Young’s modulus are constant in each respective domain. Furthermore, we assume that the inclusion has a potentially higher Young’s modulus than the background, whereas the Poisson ratio is the same in both regions. In addition, we set $L/H = 1$, $a_{I}/L = 1/2$, $b_{I}/H = 1/4$, and assume the inclusion is centered at $(x_c, y_c) = (0.65L, 0.65H)$. The problem is solved on a finite-element mesh with $T6$ elements, generated using the Triangle mesh generator [82] with the parameter area = 0.001 (which yields approximately 1,600 elements).

2.4.1 PC Surrogate

As stated earlier, it is essential to assess the fidelity of the PC surrogate model. To this end, we analyze the behavior of the error $\vartheta$ (defined in section 2.2.3) as function of $p$, the degree of the PC expansions of $u, E$, and $\nu$. Specifically, we use (2.22) to analyze the convergence of the approximation as the polynomial orders $p$ is increased. The error is estimated using a Monte Carlo approach with $M_s = 10,000$; a large number of samples was used to assure a high accuracy in the computation of the error.

Figure 2.3 shows plots of $\vartheta$ versus $p$, for $\nu^{\text{min}} = 0.45$, $\nu^{\text{max}} = 0.49$, $E_B^{\text{min}} = 10$ kPa, $E_B^{\text{max}} = 90$ kPa, and three different values of $\kappa^{\text{max}}$. As expected, the relative error decreases as $p$ increases. We notice that increasing $\kappa^{\text{max}}$ results in a higher error for $p < 6$; this is because higher values of $\kappa^{\text{max}}$ imply higher levels of uncertainty. Similarly, reducing (resp. increasing) the range for the Poisson’s ratio and/or the Young’s modulus would reduce (resp. increase) the error. For the present situation...
Figure 2.3: Error versus polynomial order for \((E_B^{min}, E_B^{max}) = (1e4, 9e4), (\nu^{min}, \nu^{max}) = (0.45, 0.49)\), with curves generated for different values of \(\kappa^{max}\), as indicated. Estimates are obtained using (2.22) with \(M_s = 10,000\).

where only 3 parameters are involved in the PC expansion, an isotropic order is not an issue. So, unless otherwise specified, we shall use in the following \(p = 6\), a value that ensures a relative error of approximately 1%.

2.4.2 Numerical Results

To assess the performance of the inference method, we generated a synthetic set of \(m_o\) observations of the local displacement field, \(\{Y_i\} \equiv u(\mathbf{x}_i) + \eta_i, i = 1, \ldots, m_o\), where the displacement \(u(x)\) is computed by solving (2.7) with \(\nu = 0.48, E_B = 60\) kPa, \(E_I = 240\) kPa, and the \(\eta_i's\) are independent, identically distributed, centered, Gaussian random variables with standard deviation \(\epsilon = 0.03 \times u_{ref}\), where \(u_{ref} = 2 \times F \times H/(E_B + E_I)\) is the deformation of a bar in uniaxial compression that behaves according to Hooke’s law with modulus \((E_B + E_I)/2\). We scale the noise variance by the reference displacement so that the error is relative to the displacement’s magnitude. Meanwhile, we constructed a surrogate model for \(u(x, \xi)\) using \(E_B^{min} = 10\) kPa, \(E_B^{max} = 90\) kPa, \(\nu^{min} = 0.45, \nu^{max} = 0.49, \kappa^{max} = 6\), and \(p = 6\).
Figure 2.4 shows the marginal posterior distributions of \( \xi_1, \xi_2, \xi_3 \) and the hyper-parameter, \( \epsilon \). These posteriors were sampled using a Markov Chain Monte Carlo (MCMC) algorithm. Posterior distributions were then obtained from the chain samples using kernel density estimation [13]. We used a total of \( m_o = 545 \) observations (distributed according to the regular pattern described in section 2.4.2). We notice that the marginal posterior distributions are highly peaked and allow for very good estimates of the true values of the Poisson ration \( \nu \), Young moduli \( E_B, E_I \) and noise level \( \epsilon \) to be obtained. Specifically, the calculated Maximum a Posteriori (MAP) estimates \( \nu^{MAP} = 0.482, E_B^{MAP} = 60.1 \) kPa, \( E_I^{MAP} = 239.4 \) kPa, and \( \epsilon^{MAP} = 0.01 \), are in close agreement with the true values.

From a diagnostic perspective, our main interest is to decide whether or not the value of the Young’s modulus at the suspected inclusion is different from the value of the Young’s modulus of the background, i.e. whether an anomaly is present. To this end, it is more effective to analyze, directly, the posterior distribution of \( \rho = E_I/E_B \). We expect for this ratio to be one if the Young’s modulus of the inclusion coincides with that of the background, and greater than one if they are different (because we assume that \( E_I \geq E_B \)). Consequently, the results below will focus on the posterior distribution of the contrast ratio \( \rho = E_I/E_B \).

Note that the formulation of the proposed method accommodates naturally other priors, depending on the inference problem. For instance, one could relax the constraint \( E_I \geq E_B \), using \( E_I(\xi_1(\theta)) = [1/k_{max} + (k_{max} - 1/k_{max})\xi_3(\theta)]E_B(\theta) \) with \( \xi_3 \sim U(0,1) \), and define \( \rho = \max\{E_I/E_B, E_B/E_I\} \) as the diagnostic quantity. However, since the present work focuses on inferring inclusions stiffer than the background, we found it preferable to incorporate this information in the prior definition.
Figure 2.4: Posterior distributions. In the plots, the red dashed-line corresponds to a Gaussian fit of the distribution and the purple vertical line indicates the location of the true values of $\xi_1$, $\xi_2$, $\xi_3$, and $\epsilon$, respectively.

**Effect of contrast**

In the analysis above, we used a PC surrogate obtained for $\kappa^{max} = 6$, i.e. the prior anticipates that $E_I$ falls in the range $E_B \leq E_I \leq 6E_B$. Using this PC surrogate,
a synthetic inference exercise was conducted for $E_B = 60$ kPa and $E_I = 240$ kPa, showing satisfactory predictions. However, we expect the two extreme cases $E_I = E_B$ and $E_I = \kappa^{max} E_B$ to provide more challenging tests. The former is of great interest because it represents the case where there is no anomaly, and we want to be able to correctly infer this to avoid false-positives. The case where $E_I/E_B$ is significantly larger than 1 is of lesser concern, because the presence of the inclusion will be easily ascertained; it is included nonetheless.

Figure 2.5 shows the inferred marginal distributions of $\rho = E_I/E_B$, for three different contrast values, generated with fixed $\nu = 0.48$ and $E_B = 60$ kPa, and $E_I = 60, 240,$ and $360$ kPa. Synthetic data are generated using the same procedure described above, and the PC surrogate constructed earlier is used for the purpose of sampling the posterior. In all cases, the MAP estimate of the contrast ratio is in good agreement with the true value used to generate the observations. Specifically, for the present tests, the situation involving absence of the anomaly is correctly detected.
with an essentially zero posterior probability for ratios $E_I/E_B \gtrsim 1.1$. Situations involving the presence of the anomaly are also correctly diagnosed, even for extreme values of the contrast ratio.

*Effect of noise amplitude*

One expects that the higher the noise level in the observations, the more challenging it would be to obtain accurate inference of the contrast ratio. To analyze the impact of the noise amplitude, we solved the inverse problem with $E_I/E_B = 4$ and different noise levels. Results are presented in Table 2.2, which provides estimates of the mean and standard deviation of the marginal posterior for the contrast ratio. We notice that, as expected, the standard deviation of the marginal distributions increases appreciably, roughly linearly, with the noise amplitude. On the other hand, the mean values (and the MAP estimates) are only weakly affected by the noise amplitude. Thus, for the present case and the noise range considered, the noise amplitude has insignificant impact on the diagnostic result.

Table 2.2: Comparison of inferred estimates of $\rho = E_I/E_B$ for different noise levels. The values $\mu$ and $\sigma$ correspond to the mean and standard deviation of the posterior density of $\rho$.

<table>
<thead>
<tr>
<th>$\epsilon$</th>
<th>$\mu$</th>
<th>$\sigma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5%</td>
<td>3.955</td>
<td>0.0169</td>
</tr>
<tr>
<td>1%</td>
<td>3.965</td>
<td>0.0333</td>
</tr>
<tr>
<td>3%</td>
<td>4.003</td>
<td>0.1021</td>
</tr>
<tr>
<td>5%</td>
<td>4.057</td>
<td>0.1773</td>
</tr>
<tr>
<td>10%</td>
<td>4.206</td>
<td>0.3837</td>
</tr>
<tr>
<td>20%</td>
<td>4.482</td>
<td>0.7048</td>
</tr>
</tbody>
</table>

29
To analyze how the inference is affected by the number and location of observations, we investigated the posterior distributions of the contrast ratio for different values of $m_o$. In addition, we obtained results for observations made at locations distributed according a regular pattern that is illustrated in figure 2.6. The pattern starts with an observation mesh with five points and the observation density is then systematically increased by introducing new locations such that the minimum distance between observation points is maximized. The figure depicts the observation mesh corresponding to $m_o = 13, 25$ and 41.

Figure 2.7 shows the mean and standard deviation of the posterior density of $E_I/E_B$ as a function of $m_o$. We notice that the mean fluctuates for small values of $m_o$, but that as we increase $m_o$ the inferred mean starts to converge to a value close to 4 (the true value of $E_I/E_B$). Furthermore, we see that, as expected, the standard deviation rapidly decreases as $m_o$ increases. It is very satisfactory that even with a
Figure 2.7: Mean and standard deviation of the posterior density of $\rho = E_I/E_B$ as a function of the number of observations, $m_o$.

very limited number of observations (e.g. 5), we can satisfactorily infer the presence of the inclusion.

2.5 Model Selection

In this section we analyze the case when the actual geometry of the physical problem is not completely known (which is expected in most practical applications). To address this issue, we propose to compute not only one, but several surrogates using different geometrical models. As described in section 2.3.2, we use $p_Y(\{Y_i\}|G_k)$, the evidence provided by the data, to rank the different models.

In the present work, a dictionary composed of 12 geometrical models is used. Schematics of the models are shown in figure 2.8. The models are ordered from left-to-right and top-to-bottom based on the value of the evidence; with the first model corresponding to the one with the highest evidence. In the figures, the area enclosed
by the blue solid-line corresponds to the geometry of the inclusion assumed by the
model, and the area enclosed by the yellow dotted-line corresponds to the actual
geometry of the physical model. Visually, we notice that, as expected, the highest
ranked models are the ones in which the surrogate geometries agrees the most with
the true geometry.

Figure 2.9 shows the normalized log-evidence (left plot) and also the MAP esti-
mate of $\rho$ (right plot) for each model. The evidence is normalized as follows:

$$
\mathcal{E}_k = \frac{\log (p_Y(\{Y_i\}|G_k)) - \min_j \log (p_Y(\{Y_i\}|G_j))}{\log (p_Y(\{Y_i\}|G_0)) - \min_j \log (p_Y(\{Y_i\}|G_j))},
$$

(2.32)

where $p_Y(\{Y_i\}|G_0)$ is the evidence given the true geometry. Since the true geometry
is expected to provide the highest value for the evidence, the normalized log-evidence
effectively falls in the range $[0,1]$. We appreciate that the Evidence heavily favors
models 1 through 7 (which is what one would expect from figure 2.8). From the
MAP estimates of $\rho$ shown in figure 2.9b, we notice that for model $G_1$ the estimated
value is very close to the true value; this is not surprising, since $G_1$ approximates well
$G_0$. For other models, we see some deviation in the estimate from the true value.
The estimates provided by models $G_3$ and $G_5$ are higher than the true value because
these geometries have a good overlap with the true inclusion, but they enclose a
smaller area in the domain. Because of the smaller area the surrogate believes that
the Young’s modulus of the inclusion in higher than what it actually is. The converse
is true for models $G_2$, $G_4$, $G_6$, and $G_7$. Using nested geometries might provide some
means to get a closer estimate of $\rho$ in this kind of setting.

Even though the inferred MAP of the contrast ratio are quite sensitive to geo-
metrical errors, the predictions are quite robust in the sense that in all cases one
Figure 2.8: Schematics of 12 different geometrical configurations. In the figures the area enclosed by yellow dotted-line correspond to the actual geometry of the inclusion, and the area enclosed by the blue solid-line corresponds to the location and shape of the inclusion assumed by the surrogate. The geometries are shown to scale. The models are ordered from left to right, top to bottom according to the value of the evidence.

correctly concludes that the domain is likely to contain an inclusion. In a diagnostic framework, this equates to having a significant probability of encountering an anomaly inside the inspected sample. In addition, the sensitivity to geometri-
Figure 2.9: The left plot shows the normalized evidence of the models. The plot on the right show the MAP estimate of $\rho$ for each model. The index of the model corresponds to the order of the geometries shown in figure 2.8 (index zero correspond to the true geometry). The purple line in the right plot corresponds to the true value of $\rho$.

Note that even though our dictionary in this specific example consists exclusively of ellipses, the method actually allows for different types of geometries to be considered simultaneously. This provides a great deal of flexibility because one is not restricted to a single family of shapes, which is a limitation of some approaches where the geometry is parametrized by a small set of parameters. In fact, the geometries in the dictionary can be generated based on any prior information available regarding the shape and size of the inclusion (e.g. images of known existing tumors). Moreover, the method has shown to be robust to model errors, meaning that we can still detect the presence of an inclusion despite discrepancies between the true geometry...
and the geometries considered in the dictionary.

2.5.1 Non-homogeneous Young’s modulus

Now, we analyze the impact of model errors on the robustness of the inference method. Specifically, we focus on the mechanical model errors arising due to a non-homogeneous distribution of the Young’s modulus. So far, we have assumed that the background and the inclusion are both homogeneous. However, we now analyze how a surrogate model built assuming a homogeneous Young’s moduli would perform when the observations correspond to a non-homogeneous distribution of the Young’s modulus.

The aim of this analysis is to understand how tissue inhomogeneity would be interpreted by the model. In particular, in the case when there is not an inclusion, we want to know if inhomogeneities would be interpreted as the presence of an inclusion (i.e. a false-positive diagnostic). To this end, we consider observations generated with a non-homogeneous Young modulus field, \( \tilde{E}(\mathbf{x}, \theta) \), that is constructed by perturbing the homogeneous field \( E(\mathbf{x}) = \bar{E} \) with a realization of a Gaussian process, \( K(\mathbf{x}, \theta) \), namely according to:

\[
\tilde{E}(\mathbf{x}, \theta) = \bar{E} \exp\left[ K(\mathbf{x}, \theta) \right]
\]  

(2.33)

where \( K(\mathbf{x}) \) is the centered Gaussian Process with correlation matrix

\[
C(\mathbf{x}_1, \mathbf{x}_2) = \bar{\alpha}^2 \exp\left[ -\frac{\|\mathbf{x}_1 - \mathbf{x}_2\|^2}{2l^2} \right].
\]

The variance \( \bar{\alpha}^2 \) and the correlation length \( l \) are used to control the spatial variability of \( \tilde{E}(\mathbf{x}, \theta) \). Moreover, we define

\[
\bar{\rho} = (\mu_E + 3\sigma_E)/(\mu_E - 3\sigma_E) = (1 + 3\sqrt{e^{\bar{\alpha}^2} - 1})/(1 - 3\sqrt{e^{\bar{\alpha}^2} - 1}),
\]
where $\mu_E$ and $\sigma_E$ are the mean and variance of $\tilde{E}(x, \theta)$ at a given fixed value of $x$. This expression for $\bar{\rho}$ provides a sense of how the parameter $\bar{\alpha}$ impacts the variability of the Young’s modulus field. Note that for $\bar{\alpha} = 0$ one recovers almost surely the homogeneous field $\tilde{E}(x, \theta) = \bar{E}$.

For the analysis we generate perturbation fields for $l = 0.001, 0.1, 0.5, 1, 10$ (these values are relative to the dimension of the domain) and $\bar{\alpha} = 0.01, 0.05, 0.1$ (i.e. $\bar{\rho} \approx 1.062, 1.352, 1.860$). For each combination of these values we generated 30 realizations of the Young’s modulus field.

For each realization of the field, we selected the best model according to the evidence and plotted the value of $\rho_{MAP}$ in figure 2.10. First we notice that, as expected, the larger the value of $\bar{\alpha}$ the more our estimate for $\rho$ deviates from the actual value. Moreover, we see that for large value of the correlation length the value of $\rho_{MAP}$ remains close to one; this is because the field is highly correlated and we do not expect to see much variability. We also observe a similar behavior when $l$ is very small; in this case the there is more fluctuation in the field but the fluctuations are small relative to the size of the suspected inclusion, and as a result they are “averaged-out”. Finally, we see the largest deviations in the estimate when the correlation is about the size of the domain.

We can better understand the behavior observed in the previous figure by looking at some of the realizations of the field shown in figure 2.11. We see that for $l = 10$ the field is nearly constant, and as a result the inferred value for $\rho$ is close to one as previously seen. For $l = 0.5$ we observe some high contrast regions which are comparable in size to the inclusion’s assumed geometries, which explains the higher values of $\rho_{MAP}$. Finally, for $l = 0.01$ we also see high contrast in the material,
but the fluctuations are very small compared to the size of the assumed inclusions. Fluctuations like the ones depicted in figures 2.11b and 2.11e could in fact lead to a false-positive diagnostic, since they mimic the presence of an inclusion. However, if fluctuations like these are expected, the decision threshold for the value of $\rho$ could be adjusted accordingly. Specific analysis about calculating the probabilities of miss-detection and false-positives, and decisions about the selection of an appropriate threshold for $\rho$ are left for a future study and will be reported elsewhere.

![Figure 2.10](image_url)

Figure 2.10: MAP estimate of the ratio $\rho$ for the case of a non-homogeneous Young’s modulus for different combinations of $l$ and $\bar{\alpha}$. The MAP corresponds to the model with the highest Evidence. Each blue marker represent a different realization of the field. The purple line corresponds to a value of $\rho = 1$ that would be expected in the presence of a homogeneous field.

2.6 Conclusions

In this Chapter, we presented a method to quickly assess the presence of inclusion in soft tissue at a low computational cost. Specifically, our focus was to create a dictionary of surrogate models in order to determine the posterior probability distribution of the Young’s modulus based on noisy observations of the displacement...
field. In order to reduce the cost of the inverse problem, we used a Polynomial Chaos expansion to construct a surrogate model that provides a faithful approximation of the forward problem. A Galerkin methodology was applied for this purpose, and the accuracy of the predictions was established using systematic refinement of the expansion order.

To handle the lack of complete information about the geometry of the physical model, a family of PC surrogates was built, each with a different assumed geometry. A model selection approach based on the evidence provided by the data was used to discriminate among the different models. The availability of the surrogate was exploited in the computation of the posterior’s Hessian, which was used to estimate

\[ \bar{\alpha} = 0.10, \ l = 10 \]
\[ \bar{\alpha} = 0.10, \ l = 0.5 \]
\[ \bar{\alpha} = 0.10, \ l = 0.01 \]
\[ \bar{\alpha} = 0.05, \ l = 10 \]
\[ \bar{\alpha} = 0.05, \ l = 0.5 \]
\[ \bar{\alpha} = 0.05, \ l = 0.01 \]
the value of the evidence. Furthermore, the Hessian was used to compute the covariance of the proposal distribution in the M-H algorithm, improving its convergence.

The proposed method was shown to be advantageous since it divides the computational cost into more costly preprocessing stage where the surrogate models are computed for several geometries, and a lower cost model selection and inference stage where the decision regarding the presence of the inclusion is made using observations. Thus, the approach makes feasible the use of a diagnostic device with a low computational cost to identify the presence of an inclusion in soft tissue.

Implementation of the methodology was illustrated in a simplified setting, consisting of a two-dimensional rectangular solid containing an inclusion. Attention was focused on demonstrating the possibility of inferring the presence of the inclusion from observations of the displacement field. In the wide range of conditions considered, the computations indicate that the approach is suitable for a wide range of contrast ratios, and that the predictions are robust to measurement noise, geometrical model errors, as well as the presence of spatial inhomogeneities.

The present developments motivate several avenues for further refinement. These include non-stationary problems, the possibility of simultaneously inferring both elastic problems and geometric features, as well as optimal selection of experimental observables. These topics will be the focus of future studies, and will be reported elsewhere.
3

Parallel Domain Decomposition Method for the Computation of the Karhunen-Loève Expansion


3.1 Introduction

Stochastic Partial Differential Equations (PDEs), and elliptic ones in particular, are increasingly being used to account for situations involving uncertain or incomplete knowledge of the simulated system, and to perform, for example, inference tasks and sensitivity analyses. Computational approaches for the solution of SPDEs conceptually involve three essential steps: the modeling of the input uncertainty, the solution of the governing equations, and ultimately post-processing the output to characterize the uncertainty. This Chapter and the following focus on the two first
steps respectively. We discuss at present a domain decomposition strategy to approximate random fields using local reduced bases and local coordinates. These developments are motivated by the computational approach proposed in Chapter 4, where the structure of local representations is exploited to accelerate the Monte Carlo sampling of the solution.

As mentioned previously, modeling and approximating random processes is an important task in many applications and in particular in uncertainty quantification problems. Focusing on random processes in spatial domains, it is often convenient to perform a discretization of the process in both stochastic and spatial dimensions for computational purposes. One way to achieve this is to rely on the representation of the process in terms of random coordinates in a spatial basis, and proceed with the discretization of the spatial basis functions and random coordinates. An overview of several stochastic-discretization methods for random fields is provided in [61, 88, 87]. Among these methods, a common one in the area of computational stochastic mechanics is the Karhunen-Loève (KL) expansion [54, 44, 42]. A key advantage of the KL expansion is that it is optimal, with respect to the number of random variables involved in the representation, in the mean squared error sense. This is particularly attractive when using stochastic spectral methods, such as Polynomial Chaos expansions [28, 51], in view of analyzing the influence of the process on a model solution; that is, performing uncertainty quantification. Indeed, these functional representation methods exploit heavily the smoothness of the model solution with respect to the stochastic coordinates appearing in the KL decomposition. A challenge, however, is that finding the KL expansion of a stochastic process involves decomposing its covariance function. This leads to solving a Fredholm integral equation of the
second kind, for a kernel consisting of the two-point covariance function. Decomposing such kernels is computationally demanding, because large spatial discretization grids are routinely considered when solving partial differential equations. A detailed description of Fredholm integral equations of the second kind is provided in [4], and a comparison of different methods that solve the KL expansion is found in [12].

For some particular covariance kernels, an analytical solution of the Fredholm integral equation is available (see for instance [28]). However, this is not typically the case and numerical procedures are required for its spectral decomposition [32]. Different approaches have been devised in order to make more tractable the computation of the KL expansion. For instance, in [71] a wavelet-Galerkin approach is discussed, which provides localized support leading to sparse matrix equations that can be solved at a reduced cost. Another approach is presented in [81], which relies on fast multipole methods to speed up the computations. A more recent attempt at making the problem more tractable is provided in [12], but still finding the KL expansion remains a challenging problem, specially in situations where a large number of terms are needed or the physical space is multidimensional.

This Chapter proposes an efficient parallel method for the computation of KL expansions with a potentially large number of terms (broad spectrum). The method is based on a domain decomposition technique and we refer to it as the Domain Decomposition KL (DD-KL) method. The proposed strategy involves partitioning the computational domain into smaller non-overlapping subdomains, over which local KL decomposition problems are solved to generate local bases. Our approach elucidate the exact correlation structure between sets of local coordinates associated to different subdomains and is composed of the following three main stages: (i)
solving a *local KL expansion* problem over each subdomain; (ii) using the dominant eigenfunctions from the local expansions to assemble a *reduced eigenvalue problem*; and (iii), solving the reduced eigenvalue problem to obtain the desired (global) KL expansion. Our representation is exploited in Chapter 4 to accelerate the solution of elliptic PDEs using Monte-Carlo procedure, specifically applying the local representations to expand the condensed stochastic problem of the domain decomposition formulation as local Polynomial Chaos (PC) expansions.

In addition to the immediate computational advantages of (i) solving small independent local problems and (ii) solving a low dimensional reduced global one, the method also allows us to efficiently distribute and parallelize most of the computations. Finally, as mentioned above, the method yields a representation in terms of independent stochastic coordinates that is convenient to perform uncertainty quantification tasks and sensitivity analyses, to characterize for instance the impact of the random process on a model solution. Indeed, the expansion in terms of independent random coordinates enables both sampling based approaches (*e.g.*, Monte Carlo) and functional expansions (*e.g.*, Polynomial Chaos and low rank approximations [28, 50, 66, 90]). Furthermore, the global random coordinates can be related to small sets of local coordinates, over the subdomains, to reduce the computational complexity of the uncertainty quantification problem based on the KL approximation of the process.

The outline of the Chapter is as follows. In section 3.2, we recall the KL expansion and detail the proposed domain decomposition method. Also, in this section, a truncation strategy for the DD-KL method is established in order to control the error in the resulting approximation. In section 3.3, numerical results are provided
illustrating the effectiveness of the DD-KL method and the error control. Next, section 3.4 is dedicated to illustrating both the serial behavior of the approach and its parallel scalability and efficiency. Finally, in section 3.5, some concluding remarks are provided.

3.2 A Domain Decomposition Method for KL Expansions

In this section we introduce the proposed domain decomposition method to approximate the KL expansion of a stochastic process. We start by introducing several notations used throughout the Chapter. Consider a probability space \( \mathcal{P} = (\Theta, \Sigma_\Theta, \mu_\Theta) \), where \( \Theta \) is the set of events, \( \Sigma_\Theta \) a sigma-algebra over \( \Theta \) and \( \mu_\Theta \) a probability measure. We denote by \( \mathbb{E}[\cdot] \) the expectation operator,

\[
\mathbb{E}[U] = \int_{\Theta} U(\theta) \, d\mu_\Theta(\theta),
\]

and \( L_2(\Theta) \) the corresponding space of second-order random variables, i.e., such that \( \mathbb{E}[U^2] < +\infty \). At the deterministic level, we consider \( \Omega \) a bounded subset of \( \mathbb{R}^n \) (with \( n = 1, 2 \) or 3) and define \( L_2(\Omega) \) the space of square integrable functionals \( f : \mathbf{x} \in \Omega \mapsto f(\mathbf{x}) \in \mathbb{R} \). We shall denote by \( \|\cdot\|_\Omega \) the norm in \( L_2(\Omega) \) induced by the scalar product \( \langle \cdot, \cdot \rangle_\Omega \):

\[
\forall f \in L_2(\Omega), \quad \|f\|_\Omega^2 = \langle f, f \rangle_\Omega = \int_\Omega |f(\mathbf{x})|^2 \, d\mathbf{x}.
\]

Finally, we denote by \( L_2(\Omega, \Theta) \) the space of real-valued second-order processes \( U : \Omega \times \Theta \to \mathbb{R} \) such that \( U(\cdot, \theta) \in L_2(\Omega) \), \( U(\mathbf{x}, \cdot) \in L_2(\Theta) \) and

\[
\mathbb{E} \left[ \|U(\cdot, \theta)\|_\Omega^2 \right] < +\infty \iff U \in L_2(\Omega, \Theta).
\]
3.2.1 The KL expansion

Let $U \in L_2(\Omega, \Theta)$ be a centered random process with known covariance $C : \Omega \times \Omega \to \mathbb{R}$:

$$C(\mathbf{x}, \mathbf{x}') \equiv \mathbb{E} [U(\mathbf{x}, \cdot)U(\mathbf{x}', \cdot)]. \quad (3.4)$$

The truncated KL approximation, $U_N$, of a second-order stochastic process $U$, consists in a $N$-term expansion where each term is composed of the product of a deterministic function of $L_2(\Omega)$ with a random variable of $L_2(\Theta)$. The KL approximation $U_N$ is defined as to minimize the representation error $U - U_N$ in the $L_2(\Omega, \Theta)$ sense. As covariance functions are symmetric and non-negative, it can be shown that the sought KL expansion is given by

$$U_N(\mathbf{x}, \theta) \equiv \sum_{\alpha=1}^{N} \sqrt{\lambda_\alpha} \eta_\alpha(\theta) \Phi_\alpha(\mathbf{x}), \quad (3.5)$$

where the $\lambda_\alpha$‘s and the $\Phi_\alpha$‘s are the (dominant) eigenvalues and associated (normalized) eigenfunctions of the covariance, i.e., the solution to the integral equation

$$\int_{\Omega} C(\mathbf{x}, \mathbf{x}') \Phi(\mathbf{x}') d\mathbf{x}' = \lambda \Phi(\mathbf{x}), \quad (\Phi, \Phi)_\Omega = 1. \quad (3.6)$$

The $\eta_\alpha(\theta)$ are the stochastic coordinates of $U$, having the property of being orthonormal:

$$\mathbb{E} [\eta_\alpha \eta_\beta] = \begin{cases} 1, & \alpha = \beta, \\ 0, & \text{otherwise}. \end{cases} \quad (3.7)$$

Owing to the structure of the covariance function, $C$, the eigenvalues are non-negative and can be ordered in decreasing magnitude, leading to a natural energy criterion...
for the truncation of the expansion. Classically, $N$ is fixed as to satisfy some error tolerance in the $L_2(\Omega, \Theta)$ norm. Specifically, one sets $N$ so the following inequality is satisfied for some prescribed error tolerance $0 < \delta < 1$

$$\mathbb{E} \left[ \| U - U_N \|^2_{\Omega} \right] = \mathbb{E} \left[ \| U \|^2_{\Omega} \right] - \sum_{\alpha=1}^{N} \lambda_{\alpha} \leq \mathbb{E} \left[ \| U \|^2_{\Omega} \right] \delta^2. \quad (3.8)$$

Note that the process norm can be evaluated from the covariance function, $\mathbb{E} \left[ \| U \|^2_{\Omega} \right] = \int_{\Omega} C(x, x) dx$, or through the whole KL spectrum: $\mathbb{E} \left[ \| U \|^2_{\Omega} \right] = \sum_{\alpha} \lambda_{\alpha}$.

*Galerkin approximation*  In most cases, an analytical solution of (3.6) is not available, and we have to rely on a numerical method to approximate the eigenvalues and eigenfunctions. To this end, we choose a finite dimensional space $V$, consisting of the linear span of a basis $\{v_1(x), v_2(x), \ldots, v_Q(x)\}$, with basis functions $v_i \in L_2(\Omega)$. Let $\Phi^h(x) = \sum_{k=1}^{Q} c_k v_k(x) \in V$ approximate $\Phi(x)$ with residual

$$r(x) \equiv \lambda \Phi^h(x) - \int_{\Omega} C(x, x') \Phi^h(x') dx' = \sum_{k=1}^{Q} c_k \left( \lambda v_k(x) - \int_{\Omega} C(x, x') v_k(x') dx' \right). \quad (3.9)$$

In the Galerkin method, the vector of coefficients $c = (c_1, \cdots, c_Q)^T$ is chosen by forcing $r(x)$ to be orthogonal to all functions in $V$, i.e.:

$$\langle r, u \rangle_{\Omega} = 0, \ \forall u \in V. \quad (3.10)$$

Substituting (3.9) into (3.10) and exploiting the structure of $V$, the problem can be recast as a generalized eigenvalue problem:

$$[K]c = \lambda[M]c, \quad (3.11)$$
where \([K]\) and \([M]\) are non-negative symmetric matrices of \(\mathbb{R}^{Q \times Q}\) with entries

\[
[K]_{ij} = \int_{\Omega} \int_{\Omega} C(\mathbf{x}, \mathbf{x}') v_i(\mathbf{x}) v_j(\mathbf{x}) d\mathbf{x} d\mathbf{x}', \quad [M]_{ij} = \langle v_i, v_j \rangle_{\Omega}.
\] (3.12)

The dimension \(Q\) of the discrete generalized eigenvalue problem (3.11) depends on the dimension of the approximation space \(\mathcal{V}\). If a Finite Element (FE) method is used to discretize the problem, the dimension of the basis is equal to the number of degrees of freedom \(Q\) of the FE space, which is determined by the number and type of elements used. Depending on the covariance structure, a very fine discretization (i.e., very large \(Q\)) might be required to accurately represent the eigenfunctions and, as a result, solving the eigenvalue problem could become computationally intensive. This prevents the use of direct methods for most practical problems with large \(Q\). As one is generally not interested in the full KL decomposition of \(U\) but in the \(N\)-dimensional dominant subspace, iterative methods (e.g., sub-space iterations, Arnoldi, Lanczos; see for instance [30]) can be considered as an alternative for the resolution of (3.11). However, although for typical finite element discretizations the mass matrix \([M]\) is sparse, most covariance functions induce a full matrix \([K]\), with possibly prohibitive memory requirements for its storage. Matrix-free iterative methods and parallel implementation can be used to overcome memory limitations and accelerate computations, but the full character of the operator \([K]\) induces large computational cost and important communication burdens that can severely impact the efficiency of these approaches.

In the following subsection, we introduce an alternative strategy suitable to the parallel computation of the KL decomposition. It uses a domain decomposition approach to conveniently distribute the computational load among several processors.
and recast (3.11) in a *reduced eigenvalue problem* having a size dictated by the targeted error tolerance rather than by the dimension of the underlying discretization space, e.g. $Q$.

### 3.2.2 Domain Decomposition approach

As illustrated in Figure 3.1, our approach starts by partitioning the domain $\Omega$ into $D$ non-overlapping subdomains:

$$\bar{\Omega} = \bigcup_{d=1}^{D} \Omega_d, \quad \Omega_i \cap \Omega_{j \neq i} = \emptyset.$$  \hspace{1cm} (3.13)

![Figure 3.1: Partitioning of a square domain $\Omega$ into $D = 8$ non-overlapping subdomains.](image)

Next, for each subdomain $\Omega_d$, $d = 1, \cdots, D$, we introduce the *local eigenmodes* $\tilde{\phi}_\beta^{(d)} : \Omega_d \to \mathbb{R}$, defined as the solutions of:

$$\int_{\Omega_d} C(\mathbf{x}, \mathbf{x}') \tilde{\phi}_\beta^{(d)}(\mathbf{x}') d\mathbf{x}' = \lambda_\beta^{(d)} \tilde{\phi}_\beta^{(d)}(\mathbf{x}), \quad \|\tilde{\phi}_\beta^{(d)}\|_{\Omega_d} = 1.$$ \hspace{1cm} (3.14)

In (3.14), we have denoted by $\|\cdot\|_{\Omega_d}$ the natural restriction of norm in $L_2(\Omega)$ to the subdomain $\Omega_d$. It is seen that the $\tilde{\phi}_\beta^{(d)}$ are the eigenfunctions of the correlation
$C(x, x')$ restricted to the $d$-th subdomain. These local eigenfunctions are extended to the global domain $\Omega$ by defining

$$\forall x \in \bar{\Omega}, \phi^{(d)}_\beta(x) = \begin{cases} \tilde{\phi}^{(d)}_\beta(x), & x \in \Omega_d, \\ 0, & x \notin \Omega_d. \end{cases}$$

(3.15)

We observe that because the $\phi^{(d)}_\beta$ are orthonormal in $\Omega_d$, in light of (3.15) we have

$$\langle \phi^{(d)}_\beta, \phi^{(d')}_{\beta'} \rangle_\Omega = \begin{cases} 1, & \text{if } d = d' \text{ and } \beta = \beta', \\ 0, & \text{otherwise}. \end{cases}$$

(3.16)

For each subdomain $\Omega_d$ we retain the $m_d > 0$ dominant eigenpairs according to the criterion discussed later in section 3.2.3. The $D$ sets of dominant eigenfunctions are collected to form an orthonormal reduced basis $\mathcal{B}$ of $L^2(\Omega)$:

$$\mathcal{B} = \bigcup_{d=1}^{D} \mathcal{B}_d, \quad \mathcal{B}_d = \left\{ \phi^{(d)}_\beta, \beta = 1, \ldots, m_d \right\}.$$  

(3.17)

We denote by $\mathcal{V}_{\mathcal{B}}$ the linear span of $\mathcal{B}$. We then seek for an approximation $\hat{\Phi} \in \mathcal{V}_{\mathcal{B}}$ of the global modes solution of (3.6), that is

$$\Phi(x) \approx \hat{\Phi}(x) = \sum_{d=1}^{D} \sum_{\beta=1}^{m_d} a^{(d)}_\beta \phi^{(d)}_\beta(x).$$

(3.18)

We set $a^{(d)} = (a^{(d)}_1, \ldots, a^{(d)}_{m_d})^T$, the vector of the local coordinates of $\hat{\Phi}$ for $x \in \Omega_d$. Applying the Galerkin method,

$$\forall \phi^{(d)}_\beta \in \mathcal{B} : \left\langle \int_{\Omega} C(x, x') \hat{\Phi}(x') dx', \phi^{(d)}_\beta \right\rangle_\Omega = \Lambda \left\langle \hat{\Phi}, \phi^{(d)}_\beta \right\rangle_\Omega,$$

(3.19)
the approximate eigenfunctions are seen to solve the following discrete eigenvalue problem,

\[
\begin{bmatrix}
[\hat{K}_{11}] & [\hat{K}_{12}] & \cdots & [\hat{K}_{1D}] \\
[\hat{K}_{21}] & [\hat{K}_{22}] & \cdots & [\hat{K}_{2D}] \\
\vdots & \vdots & \ddots & \vdots \\
[\hat{K}_{D1}] & [\hat{K}_{D2}] & \cdots & [\hat{K}_{DD}]
\end{bmatrix}
\begin{bmatrix}
a^{(1)} \\
a^{(2)} \\
\vdots \\
a^{(D)}
\end{bmatrix}
= \Lambda
\begin{bmatrix}
a^{(1)} \\
a^{(2)} \\
\vdots \\
a^{(D)}
\end{bmatrix},
\tag{3.20}
\]

where the block matrices \([\hat{K}_{i,j}] \in \mathbb{R}^{m_i \times m_j}\) have for respective entries,

\[
[\hat{K}_{i,j}]_{\alpha,\beta} = \int_{\Omega_i} \int_{\Omega_j} C(x, x') \phi^{(i)}_{\alpha}(x) \phi^{(j)}_{\beta}(x') \, dx \, dx', \quad 1 \leq \alpha \leq m_i, \ 1 \leq \beta \leq m_j. \tag{3.21}
\]

We refer to (3.20) as the **condensed eigenvalue problem**. The dimension of this problem is

\[
n_t = \sum_{d=1}^{D} m_d = \text{card } \mathcal{B}. \tag{3.22}
\]

It is easily shown that the matrix \([\hat{K}] \in \mathbb{R}^{n_t \times n_t}\) is symmetric and positive definite if the covariance function is such that for all \(u \in L_2(\Omega),\)

\[
\|u\|_\Omega > 0 \Rightarrow \left\langle u, \int_{\Omega} C(\cdot, x) u(x) \, dx \right\rangle_\Omega > 0. \tag{3.23}
\]

In the following, we shall assume that this assumption is verified. Consequently, the \(n_t\) eigenvalues \(\Lambda_\alpha\) of \([\hat{K}]\) can be ordered with decreasing magnitude as

\[
\Lambda_1 \geq \Lambda_2 \geq \cdots \geq \Lambda_{n_t} \geq 0. \tag{3.24}
\]

Then, for reasons explained in section 3.2.3, we can select the smallest \(\hat{N}, 1 \leq \hat{N} \leq n_t,\) such that for a prescribed relative error tolerance \(0 \leq \delta \leq 1\) we have

\[
\sum_{\alpha=\hat{N}+1}^{n_t} \Lambda_\alpha \leq \frac{\delta^2}{2} \sum_{\alpha=1}^{n_t} \Lambda_\alpha. \tag{3.25}
\]
The truncated approximation of $U$ is then given by

$$U(x, \theta) \approx \hat{U}_N(x, \theta) = \sum_{\alpha=1}^{N} \sqrt{\Lambda_{\alpha}} \hat{\eta}_{\alpha}^{}(\theta) \hat{\Phi}_{\alpha}(x),$$

(3.26)

where

$$\hat{\Phi}_{\alpha}(x) = \sum_{d=1}^{D} \sum_{\beta=1}^{m_d} a_{\alpha,\beta}^{(d)} \phi_{\beta}^{(d)}(x).$$

(3.27)

is the eigenfunction corresponding to $\Lambda_{\alpha}$.

As illustrated in the examples section, $n_\ell$ is essentially fixed by the requested accuracy and not by the size of the discretization space. In fact, in our simulations, typical values for $n_\ell$ were small enough to permit the use of direct solvers for the solution of the reduced problem. However, any other type of solver, for instance iterative ones, can be considered for the solution of (3.20). In the following, we refer to the approximation (3.26) as the DD-KL expansion. Algorithm 1 provides a schematic of the main steps involved in the DD-KL approach. The algorithm highlights the two main steps that can be carried out in parallel, namely the construction of the local bases $B_d$ and the assembly of the matrix $[\hat{K}]$ of the condensed operator. The actual parallel implementation of the method is further discussed and tested in section 3.4.

**Remark** A numerical method is needed to solve the local problems and yield the local eigenpairs $(\lambda, \phi)$; in the present work we use Finite Element methods to discretize the Galerkin weak form of (3.6). Even though we mentioned in section 3.2.1 that using this approach could become computationally intractable for large problems, using it to solve the local problems is now a viable option, owing to the much smaller size of
Algorithm 1: Schematic steps of the DD-KL approach.

1. Partition the domain Ω into D subdomains;
   // Parallel loop – Computing local modes
2. **foreach** subdomain Ω\(_d\) **do**
   3. Discretize the local integral equation given by (3.14) to get \([K^{(d)}]\) and \([M^{(d)}]\);
   4. Solve the local generalized eigenvalue problem
      \[ [K^{(d)}]\{\tilde{\phi}^{(d)}\} = \lambda^{(d)}[M^{(d)}]\{\tilde{\phi}^{(d)}\} \]
3. **end foreach**
   // Parallel loop – Computing entries of reduced problem
4. **foreach** subdomain Ω\(_i\) **do**
5. **foreach** subdomain Ω\(_j\) **do**
6. **foreach** 1 ≤ α ≤ \(m_i\) and 1 ≤ β ≤ \(m_j\) **do**
7. Compute \([\hat{K}_{i,j}]_{\alpha,\beta} = \int_{\Omega_i} \int_{\Omega_j} C(x, x') \phi^{(i)}_\alpha(x) \phi^{(j)}_\beta(x') \, dx \, dx' \]
8. **end foreach**
9. **end foreach**
10. Assemble and solve the reduced eigenvalue problem; /* see equation (3.20) */
11. Get approximated global eigenfunctions; /* see equation (3.18) */

the local problems. Specifically, their number of degrees of freedom is approximately divided by D as compared to the global problem. Moreover, the local modes over distinct subdomains can be computed in parallel exploiting the independence of the local problems; this allows to effectively distribute the computational load among several processors.

**Remark** The cost of solving the reduced eigenvalue problem is independent of the size of the discretization space used for the resolution of the local problem. Instead, it depends on the number of local modes retained in each subdomain, \(m_d\), which add up to \(n_t\); we expect \(n_t = \mathcal{O}(N)\), which is in practice much less than \(Q\). The
computational cost and complexity analysis of the method are discussed in section 3.4 on the basis of computational examples.

**Remark** Even if the local modes \( \phi^{(d)}_\beta(x) \)'s are all approximated using the same FE method\(^1\) over each subdomain \( \Omega_d \), the condensed problem (3.20) does not ensure that the final approximation belongs to the FE space that would have been built over the whole domain \( \Omega \). For instance, a typical situation corresponds to the case of continuous FE approximations over each subdomain \( \Omega_d \) to compute the \( \phi^{(d)}_\beta \)'s, so the \( \hat{\Phi}_\alpha \) will generally be only piecewise continuous over the union of the \( \Omega_d \) as indicated by (3.18), with jumps across the interfaces between subdomains. Such jumps are generally not a problem, but they can eventually be removed if needed at a post-processing stage, \textit{e.g.} by one of the averaging procedures routinely used in Discontinuous Galerkin methods.

**Remark** Substituting (3.18) into (3.26) we get the following alternative representation of the DD-KL expansion directly in terms of the local eigenmodes

\[
\hat{U}_N(x, \theta) = \sum_{d=1}^{D} \sum_{\beta=1}^{m_d} \sqrt{\lambda^{(d)}_\beta} \xi^{(d)}_\beta(\theta) \phi^{(d)}_\beta(x), \quad \xi^{(d)}_\beta(\theta) = \sum_{\alpha=1}^{\hat{N}} \sqrt{\frac{\Lambda_{\alpha}}{\lambda^{(d)}_\beta}} a^{(d)}_{\alpha,\beta} \hat{\eta}_\alpha(\theta), \quad (3.28)
\]

where the \( \{\xi^{(d)}_\beta(\theta), \beta = 1, \ldots, m_d\} \) are called the local random variables. This expression highlights that for \( x \in \Omega_d \) the process \( U(x, \theta) \) can be approximated using an expansion that depends only on the local coordinates \( \xi^{(d)}_{1 \leq \beta \leq m_d} \). In other words, even though the stochastic dimension of the truncated expansion is \( n_t \), at the subdomain

\(^1\) In fact, distinct subdomains could be treated using different discretization methods.
level we can characterize the process using a reduced stochastic dimension \( m_d \). In addition, it can be shown that the local coordinates of a subdomain are uncorrelated but that coordinates of distinct subdomains are generally correlated. This representation of the process in terms of local random variables is exploited in Chapter 4 to accelerate the solution of stochastic elliptic PDEs using a Monte-Carlo approach.

3.2.3 Truncation strategy

In this section we address the selection of the parameters governing the proposed method. An obvious question concerns the selection of the number \( D \) of subdomains and the actual partition of \( \Omega \). In the example section 3.3.3, we numerically illustrate the robustness of the method with respect to the partitioning of \( \Omega \) in \( D \) subdomains and investigate the effects of varying \( D \). Here, we focus on the selection of the number of local modes \( m_d \) and the appropriate truncation of the final DD-KL expansion, for a fixed partition of \( \Omega \).

The method introduces two different sources of error in the approximation of \( U \) by \( \hat{U}_N \). First, an error is introduced when representing the eigenfunctions of \( U \) in the finite dimensional space \( \mathcal{V}_\mathcal{B} \) built on the local bases, see (3.18). Hereafter, we shall denote by \( U_\mathcal{B} \) the projection of \( U \) on \( \mathcal{V}_\mathcal{B} \):

\[
U_\mathcal{B}(\mathbf{x}, \theta) = \sum_{d=1}^{D} \sum_{\alpha=1}^{m_d} \sqrt{\lambda_\alpha^{(d)}} \eta_\alpha^{(d)}(\theta) \phi_\alpha^{(d)}(\mathbf{x}).
\] (3.29)

Second, the projected process \( U_\mathcal{B} \) is further reduced, through the resolution of the reduced problem, to yield the final approximation \( \hat{U}_N \).

Because \( U - U_\mathcal{B} \) is orthogonal to \( U_\mathcal{B} - \hat{U}_N \), the squared norm of the error \( U - \hat{U}_N \)
can actually be broken down into two independent parts as follows:

\[
\mathbb{E} \left[ \| U - \hat{U}_N \|_\Omega^2 \right] = \mathbb{E} \left[ \| U - U_{\beta} \|_\Omega^2 \right] + \mathbb{E} \left[ \| U_{\beta} - \hat{U}_N \|_\Omega^2 \right].
\]  

(3.30)

See appendix B for the derivation. The first term is obtained by adding up the local contributions over the subdomains, which, by construction of the local modes, are given by

\[
\epsilon_d^2 = \mathbb{E} \left[ \| U - U_{\beta} \|_{\Omega_d}^2 \right] = \mathbb{E} \left[ \| U \|_{\Omega_d}^2 \right] - \sum_{\alpha=1}^{m_d} \lambda_{\alpha}^{(d)}, \quad \forall d = 1, \ldots, D.
\]  

(3.31)

Then, gathering the local contributions, we end up with

\[
\epsilon_{\beta}^2 = \mathbb{E} \left[ \| U - U_{\beta} \|_\Omega^2 \right] = \sum_{d=1}^{D} \epsilon_d^2 = \mathbb{E} \left[ \| U \|_\Omega^2 \right] - \sum_{d=1}^{D} \sum_{\alpha=1}^{m_d} \lambda_{\alpha}^{(d)}.
\]  

(3.32)

Finally, similarly to the classical KL truncation error, the second error contribution is

\[
\epsilon_{\beta \hat{N}}^2 = \mathbb{E} \left[ \| U_{\beta} - \hat{U}_N \|_\Omega^2 \right] = \mathbb{E} \left[ \| U_{\beta} \|_\Omega^2 \right] - \sum_{\alpha=1}^{\hat{N}} \Lambda_{\alpha}.
\]  

(3.33)

Since \( \mathbb{E} \left[ \| U_{\beta} \|_\Omega^2 \right] \leq \mathbb{E} \left[ \| U \|_\Omega^2 \right] \), the overall error can be estimated from:

\[
\mathbb{E} \left[ \| U - \hat{U}_N \|_\Omega^2 \right] = \epsilon_{\beta}^2 + \epsilon_{\beta \hat{N}}^2 \leq 2 \mathbb{E} \left[ \| U \|_\Omega^2 \right] - \sum_{d=1}^{D} \sum_{\alpha=1}^{m_d} \lambda_{\alpha}^{(d)} - \sum_{\alpha=1}^{\hat{N}} \Lambda_{\alpha}.
\]  

(3.34)

This expression shows that to reduce the error, one needs to jointly increase the size of the local basis over all the subdomains and increase \( \hat{N} \). Clearly, this suggests the existence of an optimal set of values for \( m_d \) and \( \hat{N} \). In this work, in order to achieve
an overall relative error $0 \leq \delta \leq 1$, we simply enforce the following error levels. First, regarding the local errors, we require simply that the $m_d$'s are selected to ensure for each $d$

$$
\epsilon_d^2 = \mathbb{E} \left[ \| U \|_{\Omega_d}^2 \right] - \sum_{\alpha=1}^{m_d} \lambda_{\alpha}^{(d)} \leq \mathbb{E} \left[ \| U \|_{\Omega_d}^2 \right] \frac{\delta^2}{2}, \quad (3.35)
$$

such that $\epsilon_{gd}^2 \leq \mathbb{E} \left[ \| U \|_{\Omega_d}^2 \right] \delta^2 / 2$. Then, $\hat{N}$ is selected so that (3.25) holds, which ensures that

$$
\mathbb{E} \left[ \left\| U - \hat{U}_\hat{N} \right\|_{\Omega}^2 \right] \leq \delta^2 \mathbb{E} \left[ \| U \|_{\Omega}^2 \right]. \quad (3.36)
$$

**Remark** We observe that the selection of the $m_d$'s in (3.35) balances the local relative (squared) error over the subdomains; this could be further improved by collecting the local eigenvalues for all the subdomains and selecting the $n_t$ dominant ones that ensure $\epsilon_{gd}^2 \leq \delta^2 / 2$. However, numerical experiments have shown that the local selection based on (3.35) is quite satisfactory and in addition it maintains a strict independence of the local bases construction.

### 3.3 Numerical Examples

In this section we describe the test problem used to validate the domain decomposition approach (Section 3.3.1), present several numerical results that demonstrate the convergence and the error control (Section 3.3.2) of the method. Finally, we provide in Section 3.3.3 a brief analysis of the impact of the domain partitioning on the behavior of the method.
3.3.1 Test problem

The method is applied to the decomposition of a second-order stochastic process \( U(x, \theta) \) defined over the two-dimensional unit square \( \Omega = [0, 1]^2 \). For the covariance of the process, we assume the classical stationary squared exponential structure, with characteristic correlation length \( L \) and unit variance:

\[
C(x, x') = \exp \left( -\frac{\|x - x'|^2}{L^2} \right).
\]  

We stress that the proposed method is not limited to this specific covariance structure and domain shape. For the spatial discretization of the KL modes, we use piecewise constant Finite Element discretization unless otherwise indicated. This low order FE approximation space is chosen because of the application to stochastic elliptic PDEs we have in mind; in these applications, the stochastic coefficient field modeled by KL expansion is often considered as constant over the finite elements [22]. In any case, we stress that our method perfectly accommodate for higher order FE methods. The spatial mesh consists of a conforming triangulation of \( \Omega \) into a set \( \Sigma \) of \( N_e \) finite elements. Eventually, the set of elements is partitioned into \( D \) subsets forming connected non-overlapping subdomains \( \Omega_d \), for \( d = 1, \ldots, D \). For the partitioner, we rely on a \( k \)-means geometrical clustering algorithm [56, 53, 58].

The correlation length \( L \) has a marked effect on the spectral decay of the KL expansion. Figure 3.2 shows the magnitude of the ordered eigenvalues of \( C(x, x') \). From the Figure, we can appreciate the slower decay rate of \( \lambda_k \) for smaller values of \( L \). In fact, for the smallest correlation lengths tested, the leading eigenvalues are seen to be essentially equal with an asymptotic decay that is increasingly delayed as \( L \) decreases. The slower the decay rate, the larger the number of terms required in
the KL expansion to achieve a desired accuracy. For instance, setting $\delta^2 = 10^{-3}$ in equation (3.8) we get $N = 3$, $N = 8$, $N = 259$, and $N = 22,356$ for $L = 10$, $L = 1$, $L = 0.1$, and $L = 0.01$, respectively.

![Figure 3.2: Spectra of the global decomposition for the squared exponential covariance in (3.37) with different values of the correlation length $L$ as indicated.](image)

The DD-KL approach exploits the fact that the convergence behavior of the KL expansion, for fixed covariance structure, is governed by the magnitude of $L$ relative to the characteristic length of the domain. Indeed, decreasing (resp. increasing) the characteristic extent of the domain has a similar effect as increasing (resp. decreasing) the correlation length. In the DD-KL approach, increasing the number of subdomains allows to reduce the extent of the subdomains, leading to an apparent larger $L$ and a faster spectral decay for the local expansions. This is illustrated in Figure 3.3, where the spectra of local expansions are shown for different values of the number of subdomains. Note that the Figure reports the spectra for all the $D$ subdomains, so there are $D$ spectra plotted when $\Omega$ is partitioned into $D$ subdomains. Here, we
observe that for a given number $D$ of subdomains, the local subdomains all have roughly the same extent so the local expansions have a similar decay. Moreover, it is seen that, as expected, the local expansions have spectra which decay faster as $D$ increases.

The variability of the spectra among the D sudomains can be appreciated from Figure 3.3b. This variability is principally due to the partitioning procedure that generates non identical subdomains, with slightly variable apparent $L$ as a result.

![Figure 3.3](image)

**Figure 3.3**: Spectra of local decompositions for the squared Gaussian covariance (3.37), with $L = 0.1$, and different $D$ as indicated. The plot in (a) uses a linear scale for the $x$-axis, whereas the plot in (b) uses a logarithmic scale.

3.3.2 *Convergence analysis*

We start by demonstrating that the proposed method converges towards the direct decomposition estimates of the dominant or truncated decomposition of $U$. To this end, we select arbitrarily truncation levels $N > 0$ and check that the DD-KL solution
\( \hat{U}_N \) converges to the truncated direct solution \( U_N \) (i.e., the solution computed without the domain decomposition), as the error tolerance is lowered. In fact, the comparison and convergence analyses are conducted using the spatially discretized solutions, over the same finite element mesh. We compute \( U_N \) solving (3.6) discretized over the finite element mesh, as described in Section 3.2.1. The same finite element mesh with \( N_e = 40,802 \) elements and piecewise constant approximation is used for both decompositions. In this case, the differences between \( \hat{U}_N \) and \( U_N \) result from the DD-KL reduction error only. To measure these differences, we define the following error metrics. First, we consider error in the spectra, computing the normalized \( \ell_1 \)-distance between the \( N \) dominant eigenvalues of the DD-KL solution \( (\Lambda_k) \) and of the direct computation \( (\lambda_k) \):

\[
\epsilon_{\text{spec}} = \frac{\sum_{k=1}^{N} |\lambda_k - \Lambda_k|}{\sum_{k=1}^{N} |\lambda_k|}. \tag{3.38}
\]

In addition to the error in spectra, \( \epsilon_{\text{spec}} \), we quantify the distance between the \( N \)-dimensional subspaces of the direct and DD-KL eigen bases. Different error measures can be thought to characterize the distance between subspaces; here we simply rely on the expected squared \( L_2 \)-norm of the projection error in the linear span of the eigenvectors \( \hat{\Phi}_\alpha(x) \) of the DD-KL approximation. For a generic second order process \( V \), the (relative) subspace error measure \( \epsilon_{\text{sub}}^2(V) \) is defined as follows:

\[
\epsilon_{\text{sub}}^2(V) = \frac{\mathbb{E} \left[ \left| V(x, \theta) - \sum_{\alpha=1}^{N} \left( V(x, \theta), \hat{\Phi}_\alpha(x) \right)_\Omega \hat{\Phi}_\alpha(x) \right|_\Omega^2 \right]}{\mathbb{E} \left[ \| V(x, \theta) \|_\Omega^2 \right]}. \tag{3.39}
\]

Note that \( \epsilon_{\text{sub}}^2(U_N) \) is therefore the spatially discretized version of the normalized projection error \( \epsilon_\beta \) in (3.32) for the truncated process \( U_N \). In order to compute the
expectation in the previous error definition, the process $U$ is assumed to be Gaussian in what follows.

For $N$ and $D$ fixed, we expect the difference between $U_N$ and $\hat{U}_N$ to decrease as richer and richer local bases are considered. Recalling that the dimension of the local basis for subdomain $\Omega_d$ is $m_d$, we expect $\hat{U}_N \to U_N$, in the sense of the metrics defined above, when $m_d$ increases for all the subdomains. As explained before, the local basis dimension $m_d$ yields a local expansion error $\epsilon_d$ given by equation (3.31). In the following, we choose a positive tolerance parameter $\delta^*$ to control the error in the local expansions and we select the smallest $m_d \in \mathbb{N}_+$ satisfying

$$
\epsilon_d^2 \leq \mathbb{E} \left[ \|U\|_{\Omega_d}^2 \right] \times \delta^*, \quad \forall d = 1, \ldots, D.
$$

(3.40)

Figures 3.4 and 3.5 report the errors $\epsilon_{\text{spec}}$ and $\epsilon_{\text{sub}}^2$, respectively, as a function of the tolerance parameter $\delta^*$. The two figures show that one can improve the accuracy of the KL-DD expansion and approximate better and better $U_N$ by lowering the tolerance $\delta^*$, that is, by increasing the size of the local bases and accordingly enriching the reduced space. We also note that as larger $N$ are considered, $\delta^*$ has to decrease consistently, because increasing $N$ requires $n_t = \sum_d m_d \geq N$: a minimal tolerance $\delta^*$ is therefore required to induce a reduced basis $\mathcal{B}$ large enough $(\text{card } \mathcal{B} > N)$ when selecting the $m_d$ through (3.40). This explains the different ranges for $\delta^*$ considered in the plot, depending on $N$. Another interesting observation from Figures 3.4 and 3.5 is that the two errors decay proportionally with the local tolerance $\delta^*$. This trend demonstrates that the local accuracy fixes the global precision on $U_N$, for a fixed $N$ and $\delta^*$ small enough. Conversely, for a fixed and small enough $\delta^*$, the relative error $\epsilon_{\text{sub}}^2$ increases as we demand to approximate more and more modes (increasing $N$) but always remains less than $\delta^*$. In these two figures, a dashed line
Figure 3.4: Relative spectrum error $\epsilon_{\text{spec}}$, between $U_N$ and $\hat{U}_N$, as a function of the local tolerance $\delta^*$ (see (3.40)) for different subspace dimensions $N$ as indicated. Computations use $D = 80$ and $L = 0.1$. The dashed line has unit slope.

Figure 3.5: Subspace error $\epsilon_{\text{sub}}^2(U_N)$ in (3.39) as a function of the local tolerance $\delta^*$ (see (3.40)) for different subspace dimensions $N$ as indicated. Computations use $D = 80$ and $L = 0.1$. The dashed line has unit slope.

with unit slope is shown to serve as a visual reference corresponding to $\epsilon_{\text{spec}} = \delta^*$ and $\epsilon_{\text{sub}}^2 = \delta^*$, respectively. As observed, the actual error is always below the dashed line, which illustrates that we can effectively bound the error in the approximation.
to be less than any desired value $\delta^*$. 

Now that we have established that the truncated DD-KL solution $\hat{U}_N$ converges to the truncated process $U_N$ when the tolerance on the local error is lowered, we proceed to investigate the error between $\hat{U}_N$ and the non-truncated process $U$. Strictly speaking, we actually look at the differences between the spatially discretized versions of $U$ and $\hat{U}_N$. Also, to compute efficiently the subspace error $\epsilon_{\text{sub}}^2(U)$ in (3.39), we need either to directly sample realizations of $U$ or to know its (untruncated) KL expansion. Here, however, we use as a proxy for $U$ a truncated KL expansion of $U$, obtained with the direct method, with a very large number of modes, typically several times larger than the requested $N$. In Figure 3.6, the evolution of $\epsilon_{\text{sub}}^2(U)$ is shown as a function of the local tolerance $\delta^*$, for different values of $N$. These curves must be compared with the errors with respect to $U_N$ plotted in Figure 3.5. Contrary to the error with respect to $U_N$, the error with respect to $U$ is seen to level off and not to continue to decay when the tolerance $\delta^*$ is lowered. This behavior reflects the decomposition of the error in (3.30): when the projection error $\epsilon_B$ is made smaller and smaller, lowering $\delta^*$, the truncation error $\epsilon_{\hat{R}N}$ becomes dominant such that $U - \hat{U}_N$ can not be reduced further but by increasing $N$. It confirms that increasing the dimension of the local bases beyond a certain threshold does not provide any appreciable advantages. On the contrary, it requires determining more local modes and solving a larger reduced problem with higher numerical cost as a result. The plots also verify the appropriateness of the strategy described in Section 3.2.3, since we can observe that for a given $N$ the relative subspace error stagnates after $\delta^* \approx \epsilon_{\text{sub}}^2(U)$.

So far, we have kept the number of subdomains constant in analyzing the error
Figure 3.6: Projection error $\epsilon^2_{\text{sub}}(U)$ with respect to the untruncated process, as a function of tolerance $\delta^*$ defining the local truncation through (3.40). Computations are conducted for $L = 0.1$ using $D = 80$ subdomains and different values of $N$ as indicated. The dashed black line is a reference of unit slope.

in the DD-KL solution. We now show that the proposed error-control approach is not compromised when we increase the number of subdomains $D$ used to partition $\Omega$. In the following numerical test, we fix a target relative error $\delta$ and select the local and global truncation level using equations (3.25) and (3.35), to balance the projection ($\epsilon^2_{\text{proj}} \leq \frac{\delta^2}{2}$) and truncation ($\epsilon^2_{\text{trunc}} \leq \frac{\delta^2}{2}$) errors. The DD-KL solution is then computed for different numbers $D$ of subdomains and we report in Figure 3.7 the resulting relative projection error $\epsilon^2_{\text{sub}}(U)$. The curves demonstrate that a relative error less than $\delta$ is consistently achieved irrespective of the number of subdomains considered.

The numerical experiments on the convergence of the DD-KL method were repeated using continuous piecewise quadratic finite element discretizations ($P_2$). We obtained results that are similar to the piecewise constant case and, therefore, they
are not reported here. Note, however, that the discretized DD-KL and direct solutions lives in different finite dimensional space for the quadratic approximation, as the direct solution is continuous over the whole domain $\Omega$, while the DD-KL solution is only continuous over the subdomains $\Omega_d$, with possibly discontinuities across the boundaries between subdomains. Our numerical tests indicates that the DD-KL approach remains convergent for finite element discretization with order greater than 0, and is not compromised as $D$ increases. Of course, changing the order of the FE method affects the error in the computed eigenmodes of the local problems, such that a coarser mesh with a lower number of degrees of freedom $Q$ could be considered for the $P_2$ FE discretization, compared to the $P_1$ or $P_0$ discretizations, while maintaining a comparable accuracy in the final approximations. The present work is not concerned with the question of determining the optimal FE order and mesh size to achieve a prescribe accuracy at a minimal computational cost. This question
is left as a subject to future studies.

3.3.3 Effect of domain partitioning

We have just seen that changing the number of subdomains does not affect the accuracy of the final DD-KL expansion, provided that the truncations of the local and final expansions are properly set. However, the number of subdomains is obviously expected to affect the computational cost, as the DD-KL method was motivated by the complexity reduction in the first place. The analysis of the computational efficiency of DD-KL method is delayed to the next section, where a parallel implementation is presented. In this section, we focus on the behavior of the local and reduced bases when $D$ varies and depending on the type of partitioning of $\Omega$ considered.

We first assume that $\Omega$ is partitioned into $D$ subdomains having similar characteristic size. Following the discussion above, the change in apparent $L$ is proportional to $D^{1/n}$, for $n$ spatial dimensions. In addition to the reduction of the dimensionality of the discretized local problems, we also expect the size of the local bases to decrease for larger $D$. However, the spectra shown in Figure 3.3a indicate that the reduction in the average (over the $D$ subdomains) number of local modes, $\bar{m}$, becomes marginally insignificant as $D$ increases. In any cases, we have the lower bound $m_d \geq 1$, indicating that the behavior of the method for very large $D$ (compared to $N$) can be problematic. To clarify this point, we report in Table 3.1 the evolution of the reduced problem size ($n_t$), average local basis dimension ($\bar{m}$) and standard deviation of $m_d$ (denoted $\sigma_{m_d}$), when increasing the numbers of subdomains. The results reported correspond to the previous example, with a target accuracy $\delta^2 = 2 \times 10^{-3}$. 

66
We observe that as $D$ increases the dimension $n_t$ of the reduced problem increases. In fact, it is expected that asymptotically we would have $n_t = \mathcal{O}(Q)$, where $Q$ is the number of degrees of freedom. This claim is supported by the evolutions of the averaged local basis dimension $\bar{m}$ which is seen to initially drop quickly and then continues to drop at a lower rate. For the range of $D$ investigated the smallest value we reached was $\bar{m} = 3$, but if $D$ continues to increase we will eventually converge to $\bar{m} = 1$. Similarly, the standard deviation of the local basis dimension $m_d$ is reported to decay monotonically to zero. Since, for a fixed target accuracy $\delta$ we have roughly fixed number of term $N$ in the expansion, the evolution of $n_t$ with $D$ indicates the existence of an optimal number of subdomains, balancing the complexity reduction in the local problems with the progressive increase in the size $n_t$ of the reduced problem. Determining this optimum is not obvious as it depends on implementation (e.g. how many local problems can be solved in parallel) and on the numerical method used to solve the reduced problem. Regardless of these considerations, we remark that, over the range of values for $D$ shown in Table 3.1, the KL decomposition problem is recast in a reduced one that has a dimension ranging from $n_t = 431$ to $3,840$, indicating a significant complexity reduction compared to the direct approach. For example, using intermediate value $D = 160$, the DD-KL approach involves the solution of 160 independent eigen problems (possibly in parallel) with size roughly $40,802/160 = 255$ and the resolution of the reduced problem with size $n_t = 983$; these have to be compared with the direct approach in which one has to solve a single eigen problem with dimension $40,802$.

It should be clear at this point that the efficiency of the DD-KL method depends on the behavior of the local problems when the domain is partitioned into smaller
Table 3.1: Progression of \( n_t \) for different values of \( D \) with \( \delta^2 = 2 \times 10^{-3} \) and \( L = 0.1 \).

<table>
<thead>
<tr>
<th>D</th>
<th>( n_t )</th>
<th>( \bar{m} \pm \sigma )</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>431</td>
<td>21.55 ± 1.43</td>
</tr>
<tr>
<td>40</td>
<td>542</td>
<td>13.55 ± 0.59</td>
</tr>
<tr>
<td>80</td>
<td>741</td>
<td>9.26 ± 0.56</td>
</tr>
<tr>
<td>160</td>
<td>983</td>
<td>6.14 ± 0.35</td>
</tr>
<tr>
<td>320</td>
<td>1682</td>
<td>5.26 ± 0.44</td>
</tr>
<tr>
<td>640</td>
<td>2,306</td>
<td>3.60 ± 0.53</td>
</tr>
<tr>
<td>1,280</td>
<td>3,840</td>
<td>3.00 ± 0.00</td>
</tr>
</tbody>
</table>

ones. It is then important to investigate the impact of the partitioning method on this behavior. There are several approaches available to decompose \( \Omega \) into \( D \) non-overlapping subdomains \( \Omega_d \). We briefly mention two popular families: graph partitioning and \( k \)-means clustering. The graph partitioning methods convert the mesh into a dual graph, in which the vertices are the mesh elements, and edges link neighboring elements. The graph is then partitioned as to balance the number of vertices in each partition, while minimizing the number of edges that straddle different partitions. The \( k \)-means clustering [56, 34, 35, 53] consists in partitioning data points into \( k \) clusters, where each data point is assigned to the cluster whose mean (or centroid) is the nearest. In the context of domain decomposition, the data points to be partitioned correspond to the mesh element centroids or nodes. In this section, we rely on the \( k \)-means clustering method and consider different distance functions to control the geometrical properties of the subdomains \( \Omega_d \). Specifically, we consider an anisotropic Euclidean distance,

\[
\Delta_d(x, x') = \sqrt{(x - x')^2 + \rho(y - y')^2},
\]

(3.41)

to control with \( \rho > 0 \) the aspect-ratio of the \( \Omega_d \); and with \( R > 1 \) control the dispersion...
of the subdomain sizes by using coefficients $r_d$, drawn uniformly in $[1, R]$, to scale the distance to the $d$-th centroid. Figure 3.8 shows several partitions of $\Omega$ into $D = 20$ subdomains using different values of $\rho$ and $R$. It is seen that as $\rho > 1$ increases (from left to right) the subdomains are more and more stretched horizontally. Further, for $R = 1$ (top row) the subdomains have all similar size (surface), in contrast to the case with $R = 10$ (bottom row) where a significant dispersion of the subdomains size is seen.

![Subdomain partitions](image)

(a) $\rho = 1, R = 1$  
(b) $\rho = 5, R = 1$  
(c) $\rho = 20, R = 1$

(d) $\rho = 1, R = 10$  
(e) $\rho = 5, R = 10$  
(f) $\rho = 20, R = 10$

**Figure 3.8**: Partitioning of $\Omega = (0, 1)^2$ into $D = 20$ subdomains with the $k$-means clustering method, using different parameter values of $\rho$ and $R$ as indicated.
Before investigating the effects of the partition on the DD-KL method, we first check in Figure 3.9 that changing the geometry of the subdomains does not affect the accuracy of the method and that the selection of the local and global truncations remains appropriate. The plots confirm that, for all partitions tested, the method achieves the desired accuracy set with $\delta^2 = 2 \times 10^{-3}$.

![Figure 3.9](image)

**Figure 3.9**: $\epsilon^2_{\text{sub}}(U)$ versus $\rho$ for a target accuracy $\delta^2 = 2 \times 10^{-3}$. The domain is partitioned into $D = 80$ subdomains using the $k$-means clustering with parameters $\rho$ and $R$ as indicated.

Plotted in Figure 3.10 are the evolutions of the reduced problem size, $n_t$, as a function of the anisotropy parameter $\rho$, for different values of $R$ and $D$. Consistently with the results reported in Table 3.1, $n_t$ increases with $D$ when $R$ and $\rho$ are held fixed. In addition, an effect related to the aspect-ratio of the subdomains is evidenced in Figure 3.10. Specifically, the size of the reduced problem is adversely affected for stretched subdomains. This effect was anticipated from the trends and behaviors reported above: the most effective reduction of the local bases is expected for subdomains that have well balanced size in all directions. Incidentally, this be-
havior means that, in the case of anisotropic covariance functions, one would have interest in designing a partition of $\Omega$ that fits with the principal directions of $C$. We remark, however, that the effect of stretched subdomains is not so severe, with a reported increase of less than 50% in the reduced problem size when going from $1:1$ to $1:20$ aspect-ratios. Finally, we observe from Figure 3.10 that increasing the dispersion of the subdomains size (increasing $R$) has a negligible effect on $n_t$. This can be explained by compensation effects between larger $m_d$ for large subdomains and lower $m_d$ for smaller ones. Having $n_t$ roughly independent of $R$ does not necessarily translate into a constant computational cost. In fact, having subdomains with very different sizes could induce severe load balancing issues for the parallel resolution of the local problems. On the positive side, being able to tune the geometric size of the subdomains, without impacting much $n_t$, means that one can eventually adapt the partition in the case of a non-uniform spatial discretization.

![Figure 3.10: $n_t$ versus $\rho$ for a target accuracy $\delta^2 = 2 \times 10^{-3}$. Curves are generated for different values of $D$ and $R$ as indicated.](image-url)
3.4 Performance Analysis

This section is dedicated to assessing the serial behavior of our approach, as well as its parallel scalability and efficiency.

3.4.1 Serial behavior

We first investigate the influence of the number of subdomains $D$ on the computational time of the method in a serial implementation. For the reasons discussed later in Section 3.4.2, we only report the behavior of the local solves (see Eq. (3.14)) and the assembly of the reduced problem, i.e., the filling of the block matrices (see Eq. (3.20)). Figure 3.11 depicts the dependence of the computational times of these stages on the number of subdomains. We also compare the case of a discretization with either piecewise constant ($P_0$) and piecewise quadratic ($P_2$) finite elements in Figures 3.11a and 3.11b, respectively. The computational times are reported in arbitrary units.

The first observation that can be drawn from these figures is that the two computational times decrease as the number of subdomains increases, with a faster decay for the local solves than the assembly of reduced problem, and the same behavior is reported for the two finite element discretizations. Due to the faster decay of the local solves CPU time, the reduced problem assembly time becomes dominant as $D$ increases, and the combined time becomes quickly dominated by the matrix fill.

It is easy to understand that solving the local problems becomes cheaper as the number of subdomains increases: as $D$ increases, the subdomains become smaller, and thus the local problems involving fewer unknowns are cheaper to solve. In the tests presented, a direct solver was used for the local problem, so the computational
time reduction is very significant. Another important advantage of the proposed approach stems from the independence of the local solves. While solving the global generalized eigenvalue problem (3.11) would require the evaluation and possibly the storage of the full stiffness matrix $[K]$ in (3.12), the local problems only involve stiffness matrices $[K^{(d)}]$ that correspond to pairs of points $(x, x')$ belonging to a same subdomain $\Omega_d$. In other words, the set of local matrices $[K^{(d)}]$ simply correspond to the diagonal, subdomain-based blocks of the global matrix $[K]$. This can result in significant CPU time and memory savings as compared to solving the global problem (3.11).

The reason for the reduction of the computational time for the reduced problem assembly is less obvious, but can be explained as follows. A given block $[\hat{K}_{i,j}]$ has $m_i \times m_j$ entries, each one requiring an integration over $\Omega_i \times \Omega_j$, with complexity proportional to $N_{e_i} \times N_{e_j}$ ($N_{ed}$ being the number of degrees of freedom of the approximations in $\Omega_d$). If the subdomains have roughly the same numbers of local modes ($m_i \approx m_j \approx \bar{m}$) and unknowns ($N_{e_i} \approx N_{e_j} \approx N_0 / D$), and given that there are $D(D+1)/2$ such blocks to be computed (accounting for the symmetric structure), the complexity in assembling the reduced matrix $[\hat{K}]$ can be estimated to be proportional to $N_e^2 \bar{m}^2 \times (D + 1)/D$. In Section 3.3, we have seen that the average number of local modes $\bar{m} = n_t / D$ tends to decrease with $D$ (see Table 3.1), although $n_t$ increases. Consequently, the complexity of the reduced problem assembly reduces with $D$ though it is seen to progressively level-off for very large number of subdomains. Eventually, as $m_d$ can not be less than 1, increasing further $D$ would have a negative impact on the overall efficiency, because though the assembling stage may not be penalized, the resolution cost of the reduced problem would continuously increase.
Finally, from Figures 3.11a and 3.11b it is found that the $P_2$ discretization is more expensive, in terms of computational times, regardless of the number of subdomains. This is due to a higher number of unknowns for the $P_2$ discretization compared to the $P_0$ case. But a fair comparison between the two discretizations should not only account for the computational times but also for the spatial discretization error on the computed modes.

3.4.2 Parallel implementation

Our domain decomposition approach naturally lends itself to parallel computation. In addition to the reduced complexity of solving smaller problems, as reported above, further gain can be expected through parallelization. From the presentation of the method given in Section 3.2 and summarized in Algorithm 1, one can identify the three main stages of the approach. First, local generalized eigenvalue problems are solved at the subdomain level (see Eq. (3.14)); second, the stiffness matrix of the reduced (regular) eigenvalue problem is constructed (see Eq. (3.20)); and last, this
reduced problem is solved. We shall focus here on the first two stages, which represent the core of our approach. Regarding the last stage, let us simply mention that libraries exist to efficiently deal with the parallel solution of reduced problem, for example, \textit{PARPACK} (Parallel ARPACK) [60], \textit{SLEPc} (based on \textit{PETSc}) [36, 77], or the \textit{Anasazi} package of \textit{Trilinos} [6].

In what follows, we rely on a Message Passing Interface (MPI) approach to parallelize the solution of the local problems and the assembly of the reduced problem. We shall denote by $N_{\text{MPI}}$ the number of MPI processes; we then split the set of subdomains into $N_{\text{MPI}}$ subsets. Each process $p$ holds the data corresponding to the $D_p$ subdomains in the $p$-th subset. In a static \textit{a priori} load-balancing approach, we would like the number of subdomains $D_p$ handled by process $p$ to be evenly distributed among the processes, that is $D_p = D/N_{\text{MPI}}$ for all $p$. In practice, to tackle the case where $N_{\text{MPI}}$ does not divide $D$, we distribute the subdomains among the processes through

$$
\forall p = 1, \ldots, N_{\text{MPI}}, \quad D_p = \lfloor D/N_{\text{MPI}} \rfloor + \begin{cases} 
1 & \text{if } p \leq (D \mod N_{\text{MPI}}), \\
0 & \text{otherwise}, 
\end{cases} \quad (3.42)
$$

where $\lfloor \cdot \rfloor$ is the floor function. Such a distribution among processes is illustrated in Figure 3.12a, in a case where $N_{\text{MPI}}$ does not divide $D$.

\textit{Parallel resolution of the local problems}

In the first stage, the local problems are solved; they consist in the Fredholm equations defined by Eq. (3.14), which after discretization lead to the generalized eigen-
value problems (see Algorithm 1):

\[ [K^{(d)}]\{\tilde{\phi}_\beta^{(d)}\} = \lambda^{(d)}_\beta [M^{(d)}]\{\tilde{\phi}_\beta^{(d)}\}. \] (3.43)

It is clear that these problems are independent for each subdomain, and that only local data is needed to construct the stiffness and mass matrices \([K^{(d)}]\) and \([M^{(d)}]\). The parallel implementation of the local problems is thus trivial, as no communication is required between the corresponding MPI processes. Similarly, the local eigenvalues \(\lambda^{(d)}_\beta\) and local eigenvectors \(\tilde{\phi}_\beta^{(d)}\), for \(\beta = 1, \ldots, m_d\), can be conveniently stored locally in a distributed memory architecture. Because there are \(D_p\) subdomains held by process \(p\), and one single local eigenvalue problem to be solved per subdomain, process \(p\) is thus in charge of solving \(D_p\) problems. Provided that the number of degrees of freedom are roughly constant from one subdomain to another, we could expect the workload to be approximately balanced among the MPI processes and thus lead to good parallel performance.

**Construction of the reduced eigenvalue problem**

The second stage differs from the previous one in many ways. Perhaps the most relevant difference is the need to access non-local data. Indeed, as can be observed from Eq. (3.20), the block-matrices \([\tilde{K}_{i,j}]\) involve data that belong to subdomains \(\Omega_i\) and \(\Omega_j\). For diagonal blocks \((i = j)\), or for blocks corresponding to subdomains that are handled by the same process \(p\), no MPI communication is required, as all the needed data is held by the same process. On the contrary, for other blocks, data have to be exchanged between different MPI processes, namely the local modes \(\tilde{\phi}_{\beta}^{(d)}\), as well as local mesh information for discretizing the integrals. This is handled by resorting to an all-to-all communication, specifically to the MPI function \texttt{MPI\_Allgatherv}.  

76
Another important difference lies in the fact that the total number of blocks, \( N_K = D^2 \), grows quadratically with the number of subdomains \( D \). The classical way of distributing the workload would be for each process \( p \) to handle the construction of the \( D_p \) block-rows corresponding to its subdomains. In other words, process \( p \) would compute \( [\hat{K}_{i,j}] \) for all \( j = 1, \ldots, D \) and for all \( i \in I_p \), where \( I_p \) denotes the set of subdomain indices handled by process \( p \). Here, we further exploit the characteristic block-symmetric structure of the reduced problem matrix inherited from the symmetry of the covariance functions \( C \). Specifically, we compute only the \( N^\triangle K = D(D+1)/2 \) upper (or lower) triangular blocks of the matrix. The computation of these \( N^\triangle K \) blocks is distributed among the \( N_{\text{MPI}} \) processes so that process \( p \) is in charge of \( N^\triangle K_p \) blocks. The distribution of the blocks on the different processes is again handled using Eq. (3.42), substituting \( D \) and \( D_p \) with \( N^\triangle K \) and \( N^\triangle K_p \), respectively. The distribution among processes is illustrated in Fig. 3.12b, in a case where \( N_{\text{MPI}} \) does not divide \( N^\triangle K \).

The cost of computing each block matrix \( [\hat{K}_{i,j}] \) depends both on the number of degrees of freedom in subdomains \( \Omega_i \) and \( \Omega_j \) and on the number of local modes \( m_i \) and \( m_j \) retained for these subdomains. Provided that these are balanced, the workload of constructing the reduced problem should also be balanced among the processes and thus lead to good parallel performance. More general situations could be considered, in particular in the case of non uniform mesh and non-isotropic covariance structure, by introducing more advanced load-balancing strategies. This will be addressed in follow-on work.

77
Figure 3.12: Example of distribution among MPI processes of the subdomains (Fig. 3.12a) and block computation for the reduced problem assembly (Fig. 3.12b). Case of $D = 4$ subdomains and $N_{\text{MPI}} = 3$ MPI processes. The processes $p = 1, 2, 3$ are colored in blue, orange and green respectively.

3.4.3 Parallel efficiency

We now turn to investigate the efficiency of the parallel implementations of the local problem solves and the assembly of the reduced problem matrix. Specifically, we measure the scalability with the number of processes of performing these two tasks in parallel, including the MPI communication times involved. The scalability is characterized using two quantities, the parallel speedup $S$ and the parallel efficiency $E$. The speedup and efficiency are reported as functions of the number $N_{\text{MPI}}$ of MPI processes considered. The two measures are defined by

$$S(N_{\text{MPI}}) \equiv T(1)/T(N_{\text{MPI}}), \quad E(N_{\text{MPI}}) \equiv 100 \times S(N_{\text{MPI}})/N_{\text{MPI}}$$

where $T(N_{\text{MPI}})$ is the measured CPU times for the execution of the tasks using $N_{\text{MPI}}$ processes. As the smallest number of processes tested is $N_{\text{MPI}} = 16$, the definitions
of speedup and efficiency are actually based on the approximation $T(1) \approx 16T(16)$, assuming a perfect speedup for 16 processes. The tests were carried out on a parallel Blue Gene machine, fixing a constant number of MPI processes per computational node. For simplicity, we only report here the case with a single MPI process per node, such that the parallel runs use as many nodes as processes.

For a fixed finite element mesh with $N_e = 40,802$, we tested three different partitions of the domain, considering $D \in \{256; 512; 768\}$. A finer mesh with $N_e = 81,753$ elements, is also tested with $D = 256$ subdomains. The problem size being kept constant as the number of processes is increased, the reported speedup and efficiency correspond to a strong scaling experiment.

The results are reported in Fig. 3.13. Globally, the speedup and efficiency evolve satisfactorily as $N_{MPI}$ is increased, especially considering that no fine tuning of the parallel implementation has been performed. For the largest number of subdomains tested, $D = 512$ and $D = 768$, the parallel efficiency decreases slowly with $N_{MPI}$, down to approximately 85% for 256 processes. This is due to the fact that the local problems are relatively small, owing to the large number of subdomains, making the MPI communication time more significant compared to the actual computation time. This trend becomes more pronounced as $N_{MPI}$ increases.

For $D = 256$, the actual computational time is large enough compared to the communication time, even for $N_{MPI} = 256$ processes. Consequently, the parallel efficiency remains close to ideal when $N_{MPI}$ increases, with a reported efficiency of about 97% with 256 processes. This effect is confirmed considering a finer finite mesh with $N_e = 81,753$ elements for the same $D$ (256). In this case, a higher efficiency of nearly 99% is measured for $N_{MPI} = 256$. 

79
Figure 3.13: Speedup (left) and efficiency (right) versus the number of MPI processes. Plotted are curves for different values of $D$ and $N_e$ as indicated. The ideal scaling law is shown on the left plot using a solid black line. The correlation length is $L = 0.1$ and the energy criterion for the local modes is $\delta^2/2 = 10^{-3}$.

These tests demonstrate that the DD-KL approach is efficient and scalable. Not only does it benefit from cost reduction owing to the domain decomposition that breaks the problem in a set of small independent subproblems, but it also lends itself to natural parallel processing, involving limited communication, and thus enabling the efficient computation of the KL decomposition for large problems.

3.5 Conclusions

In this Chapter, we presented an efficient method to solve large-scale KL decomposition problems, based on a divide-and-conquer decomposition approach. Specifically, the computational domain is partitioned into smaller non-overlapping subdomains, over which local KL decomposition problems are solved to generate local bases. A criterion has been proposed, and numerically demonstrated, for the selection of the local bases’ dimensions in order to ensure a prescribed accuracy. The global KL de-
composition problem is subsequently reformulated by means of Galerkin projection in the subspace of the local modes. This procedure leads us to solve a reduced global problem, whose size is not related to the dimension of the underlying discretization space, but depends on the requested accuracy and the number of subdomains. The low dimensionality of the reduced problem enables efficient solution methods, including direct ones. But the approach is in fact flexible and can accommodate any type of eigenvalue solvers (direct, iterative, . . . ), both for the computing the local KL bases and solving of the reduced problem. In addition, although we focused on the problem of performing KL decomposition, it should be noted that the approach could be used in many different contexts and extended to determine the dominant subspace associated to more general integral operators.

The method was illustrated on the approximation of stochastic processes defined over the two-dimensional unit square. A squared exponential covariance structure was assumed, with short correlation lengths yielding slowly decaying spectra. Our numerical experiments demonstrated that the approach provides a fine control of the approximation error, and is very robust with respect to the number of subdomains. It was also observed that the cost of solving the local problems reduces with the number $D$ of subdomains, while the size and cost of solving the reduced problem increases with $D$. This suggests the existence of an optimal value for $D$, balancing the decreasing complexity of the local KL problems and the increasing size of the reduced problem. In any case, the range of values of $D$ over which the approach remains effective is large such that selecting the optimal $D$ is not critical. The sensitivity of the method to the shape of the subdomains was also explored. It was found that, in the case of isotropic covariance, it is more effective to use partitions
with subdomains having similar size in all directions, rather than having geometrically highly stretched subdomains. In any case, the influence of the subdomains geometry is not too pronounced, such that the method would be able to effectively accommodate complicated situations (in the covariance structure, geometry of the domain, adapted finite element mesh, . . . ) without having to construct a dedicated partitioner. This is particularly important in view of reusing standard libraries for partitioning the domain.

Beside the immediate computational advantages of breaking the scale size problem into smaller ones, the proposed approach naturally lends itself to parallel implementation. In fact, the parallelization of the local decompositions over individual subdomains is trivial, as local problems are fully independent from one subdomain to another, and the error criterion to select the local basis is also completely local. The parallel assembly of the reduced problem involves, on the contrary, more significant communication between processes, with a less obvious parallelization as a result. The parallel implementation, relying on the MPI framework, has been tested showing an excellent scaling up to 256 processes, provided that the initial global problem is large enough.

The proposed approach allows us to effectively generate approximated samples of a Gaussian process, using (3.26). Thus, as discussed in the introduction, the method is therefore an alternative to other existing methods to generate realizations of Gaussian stochastic processes with prescribed covariance structure. In fact, the solution of the reduced problem exhibits the correlation structure between local variables, such that one can also restrict the sampling to subsets of subdomains. Such a sampler is advantageous, for instance, to solve stochastic partial differential equa-
tions problems, in particular stochastic elliptic problems as considered in Chapter 4.
Acceleration of Monte Carlo Methods for Stochastic Elliptic PDEs using Domain Decomposition and PC Approximations of Local Dirichlet Maps


4.1 Introduction

Stochastic Partial Differential Equations (SPDEs) are of great importance in a wide range of applications. Computational approaches for the solution of SPDEs conceptually involve three essential steps: the modeling of the input uncertainty, the solution of the governing equations, and ultimately post-processing the output to characterize the uncertainty. This Chapter and the previous focus on the first two
steps. In Chapter 3, we discussed a domain decomposition strategy to approximate random fields using local reduced bases and local coordinates. Now, in this Chapter, the structure of local representations is exploited to accelerate the Monte Carlo sampling of the solution.

Two common approaches to solving SPDE are the Stochastic Galerkin method [52, 28, 78, 21, 100] and Monte Carlo (MC) sampling methods [14, 68, 9, 18]. One particular class of Stochastic Galerkin methods use polynomial chaos (PC) expansions. PC expansions have been studied extensively [52, 28, 99, 95, 96] and perform very well in a number of applications. Unfortunately, using PC expansions suffers from the curse of dimensionality and can quickly become computationally intractable when the stochastic dimension is large (the complexity can grow exponentially as a function of the stochastic dimension). Different methods have been devised to address the complexity issue, but when the stochastic dimension is very large some of the challenges can still remain. On the other hand, the cost of direct MC sampling methods is independent of the stochastic dimension, which is a desirable feature when dealing with high dimensional problems. However, it is known that MC methods have a slow convergence rate, with the root mean squared error inversely proportional to the square root of the number of samples. So, if the computational cost of obtaining an individual sample is high, these methods can be quite costly.

The stochastic dimension of the problem is closely related to the stochastic discretization method used to solve the problem. One common approach to discretize the stochastic space is the Karhunen-Loève (KL) expansion [54, 44, 42]. The number of terms in the KL expansion is what sets the stochastic dimension, and it turns out that for a given accuracy level, the number of terms in the expansion is actually
proportional to the size of the physical domain. Thus, the smaller the domain, the fewer terms are necessary in the expansion to achieve a desired level of accuracy. In [16], Chen et al rely on Domain Decomposition techniques to exploit this fact. By partitioning the global domain into smaller subdomains, a set of local problems is obtained, each with a significantly reduced stochastic dimension. In [73] Gosh and Pranesh present a closely related approach based on the Spectral Stochastic Finite Element Method.

In both of the approaches mentioned above, PC approximations of the local problem solution at the subdomain level are constructed. The cost of obtaining these local PC approximations is reduced by using the local random variables over each subdomain (i.e. the lower stochastic dimension at the subdomain level reduces the cost). An important point, that its acknowledged in these two papers, is that the local random variables in one subdomain have a dependence structure on the local random variables in other subdomains. Nonetheless, in these two works the local random variables are treated as independent across subdomains (corrections are made with the introduction of additional global random variables). In contrast, in our proposed approach we consider the actual dependence structure of the local variables and use them in the construction of local boundary-to-boundary maps that help us accelerate the solution of an Stochastic Elliptic PDEs via MC sampling. In Chapter 3, we analyzed in detail the local KL expansions approach and the dependences of the local random variables. The present Chapter concerns the solution of the Stochastic Elliptic PDEs by means of a MC sampling method that is accelerated by constructing PC expansions of the boundary-to-boundary maps (and not of the local problem solutions).
More precisely, our proposed approach is divided into two main stages: 1) a preprocessing stage in which PC expansions of a condensed problem are computed and 2) a Monte Carlo sampling stage where samples of the solution are computed. First, the physical domain is discretized using the finite element method; then the global domain \( \Omega \) is divided into \( D \) non-overlapping subdomains. This results in a condensed problem for the nodal values at the subdomains interfaces. Given this discretization, the preprocessing stage starts by breaking the condensed problem into individual contributions from each subdomain, and computing local KL expansions over each subdomain (as described in [19]). Then, using the local KL expansion, and taking advantage of the reduced stochastic dimension of the local problems, PC expansion of the local contributions to the condensed problem are constructed. This concludes the first stage of the approach. The second stage consists in generating samples (this requires taking into consideration the dependence structure of the local random variables), evaluating the PC expansion of the reduced problem for said samples, and solving the reduce problem to obtain samples of the solution.

In summary, our approach is a parallel solver that takes advantage of the PC method at the local level to reduce the cost of using the MC sampling method at a global level. There are two main contributions of the current work, first we take into account the dependence structure of the local random variables across subdomains. Furthermore, these local random variables are jointly sampled with the convenient approach described in [19], which allows us to accurately characterize the random process on which the SPDE depends on (which in general it is not possible when the local random variables are assumed independent across subdomains). The second contribution is that we use the local expansions to construct local PC expansions of
the condense problem (as opposed to constructing local PC expansion of the solutions at the subdomain level), and from these local expansions we build a PC expansion of the global condensed problem, which significantly reduces the sampling cost in the MC sampling method. We remark that by building the global condensed system in this manner we preserve the proper dependence structure in the final solution sought.

The outline of this Chapter is as follows. In section 4.2, we first recall how the domain decomposition method is applied to both a deterministic and a stochastic PDE and also discuss the Monte Carlo sampling method. In section 4.3, we discuss the limitations of constructing a PC expansion of the Elliptic solution and describe how instead we proceed with the construction of the PC expansion of the Condensed Problem. We also address the sampling of the condensed problem and go over some of the implementation details. Next, in section 4.4 the method is validated with some numerical results. In section 4.5, we analyze for the test case the performance of the method terms of complexity and parallel efficiency. Finally, in section 4.6, some concluding remarks are provided.

4.2 Elliptic Problem

4.2.1 Deterministic case

We consider the following elliptic problem in a bounded domain \( \Omega \subset \mathbb{R}^m \), with boundary \( \partial \Omega \):

\[
\begin{cases}
\nabla \cdot (\kappa(x) \nabla u) = -f(x), & x \in \Omega \\
\mathcal{B}(x, u) = 0, & x \in \partial \Omega,
\end{cases}
\]

(4.1)

where \( \mathcal{B} \) is the (linear) boundary condition operator and \( 0 < \kappa_{\text{min}} < \kappa(x) < \kappa_{\text{max}} < +\infty \) is the diffusion coefficient. For simplicity, we shall restrict ourselves to the case
of homogeneous Dirichlet and Neumann boundary conditions, that is

\[ u(x \in \partial\Omega_D) = 0, \quad \partial_n u(x \in \partial\Omega_N) = 0, \]

(4.2)

where \( \partial_n \) is the derivative in the normal direction, and \( \Omega_D \) and \( \Omega_N \) are the Dirichlet and Neumann parts of the boundary, such that \( \partial\Omega_N \cup \partial\Omega_D = \partial\Omega, \partial\Omega_N \cap \partial\Omega_D = \emptyset \).

To solve (4.1) we consider standard finite element (FE) methods based on a conforming triangulation of \( \Omega \) into a set, \( \mathcal{T} \), of \( N_e \) non-overlapping elements, \( \Sigma_e \). The FE approximation is based on a nodal basis representation. Let \( n \) be a node of the mesh, with position \( x_n \), we denote by \( \mathcal{N} \) the set of nodes that do not belong to the Dirichlet boundary \( \partial\Omega_D \), and \( N_n = |\mathcal{N}| \) the number of nodes in \( \mathcal{N} \). The approximation of \( u \) is sought as

\[ u(x) \approx \sum_{n \in \mathcal{N}} \Phi_n(x) u_n, \]

(4.3)

where the functions \( \Phi_n \) are nodal basis functions satisfying:

\[ \forall n, n' \in \mathcal{N}, \Phi_n(x_{n'}) = \begin{cases} 1, & n = n' \\ 0, & n' \neq n, \end{cases} \]

(4.4)

and \( \Phi_n(x \in \partial\Omega_D) = 0 \). It is further assumed that the support of \( \Phi_n \) is limited to the elements that have \( n \) as one of their nodes. The weak form of problem (4.1) is:

Find \( u \in V^\text{FE} \) such that

\[ \int_{\Omega} \kappa(x) \nabla u(x) \cdot \nabla v(x) dx = \int_{\Omega} f(x)v(x) dx \quad \forall v \in V^\text{FE}, \]

(4.5)

where \( V^\text{FE} \) is the linear span of nodal functions \( \{ \Phi_n, n \in \mathcal{N} \} \). The variational problem can be recast as a linear system of equations for the vector, \( \mathbf{u} \), of unknown nodal
values,

\[ [A]u = b, \]  

where \( u \) and \( b \in \mathbb{R}^{N_n} \). The system matrix \([A] \in \mathbb{R}^{N_n \times N_n}\) is symmetric positive definite, with entries

\[ [A]_{nn'} = \int_{\Omega} \kappa(x) \nabla \Phi_n(x) \cdot \nabla \Phi_{n'}(x) dx. \]  

The components of the system right-hand-side are given by

\[ b_n = \int_{\Omega} f(x) \Phi_n(x) dx. \]  

**Domain Decomposition method**

Owing to the compact support of the nodal basis functions, the matrix \([A]\) is sparse and efficient iterative methods (e.g. Preconditioned Conjugate Gradient) can be employed to solve (4.6). However the system size \( N_n \) may be large, inducing a significant resolution cost and motivating the introduction of domain decomposition methods [47, 74, 85, 93].

**Domain partitioning.** To this end, we first partition \( \Omega \) into a set of \( D \) non-overlapping subdomains \( \Omega^{(d)} \) consisting of subsets \( T^{(d)} \) of neighboring elements; we have

\[ \overline{\Omega^{(d)}} = \bigcup_{e \in T^{(d)}} \Sigma_e, \quad \bigcup_{d=1}^{D} T^{(d)} = T, \quad T^{(d)} \cap T^{(d') \neq d} = \emptyset. \]  

where \( \overline{\Omega^{(d)}} \) is the closure of \( \Omega^{(d)} \). We denote \( N_e^{(d)} = |T^{(d)}| \) the number of elements in \( \Omega^{(d)} \) and \( N^{(d)} \) the subset of nodes in \( N \) belonging to \( \Omega^{(d)} \):

\[ N^{(d)} = \left\{ n \in N; x_n \in \overline{\Omega^{(d)}} \right\}. \]  

[90]
The sets $N^{(d)}$ can be further split into disjoint subsets of interior nodes belonging to $\Omega^{(d)}$ only, and boundary nodes lying at the interface of more than one subdomain:

$$N^{(d)}_{in} = \left\{ n \in N^{(d)}; n \notin N^{(d') \neq d} \right\}, \quad N^{(d)}_{\Gamma} = N^{(d)} \setminus N^{(d)}_{in}. \quad (4.11)$$

Clearly, the sets $N^{(d)}_{in}$ are disjoint, while $N^{(d)}_{\Gamma} \cap N_{\Gamma}^{(d')}$ is not empty for two neighboring subdomains such that $\partial \Omega^{(d)} \cap \partial \Omega^{(d')} \neq \emptyset$. We then define the full set of inner and boundary nodes of the partitioned domain through

$$N_{in} = \bigcup_{d=1}^{D} N^{(d)}_{in}, \quad N_{\Gamma} = \bigcup_{d=1}^{D} N^{(d)}_{\Gamma}, \quad (4.12)$$

and set $N_{in} = |N_{in}|$, $N_{\Gamma} = |N_{\Gamma}|$. We can now rewrite the FE approximation of $u$ in (4.3) as

$$u(x) \approx \sum_{n \in N^{(d)}_{in}} u_{n} \Phi_{n}(x) + \sum_{n \in N_{\Gamma}} u_{n} \Phi_{n}(x). \quad (4.13)$$

**Iterative Domain Decomposition solver.** Upon reordering of the nodes, the linear system in (4.6) can be recast in the following block matrix form,

$$\begin{bmatrix} [A_{\Gamma,\Gamma}] & [A_{\Gamma, in}] \\ [A_{in,\Gamma}] & [A_{in, in}] \end{bmatrix} \begin{bmatrix} u_{\Gamma} \\ u_{in} \end{bmatrix} = \begin{bmatrix} b_{\Gamma} \\ b_{in} \end{bmatrix}, \quad (4.14)$$

with the previous expressions for the matrix and right-hand-sides entries. This system can be further manipulated to eliminate the internal unknowns in $u_{in}$ to come up with the condensed problem for the nodal values at the subdomains’ interfaces,

$$\hat{A}u_{\Gamma} = \hat{b}, \quad (4.15)$$
where
\[ \widehat{A} = [A_{\Gamma,r}] - [A_{\Gamma,in}]^{-1}[A_{in,r}], \quad \widehat{b} = b_{\Gamma} - [A_{\Gamma,in}]^{-1}b_{in}. \] (4.16)

Considering an iterative method to solve (4.15), the main computationally heavy task amounts to performing matrix-vector products between \( \widehat{A} \) and successive iterates vectors of \( \mathbb{R}^{N_{r}} \). A closer inspection reveals that multiplying a vector by \( \widehat{A} \) involves solving for \( v \) by inverting a system of the form \( [A_{in,in}]v = w \). This step is actually the heaviest one in the iterative solution, as it requires the solution of a linear system whose dimension, \( N_{n} - N_{r} \), is generally close to the dimension of the non-condensed problem, that is \( N_{n} \). However, it is crucial to remark that \( [A_{in,in}] \) has diagonal block structure when the nodes in \( N_{in} \) are ordered by subdomains; in this case, we have

\[
\left[
\begin{array}{ccc}
[A_{in,in}^{(1)}] & [0] & \cdots & [0] \\
[0] & \ddots & \ddots & \vdots \\
\vdots & \ddots & \ddots & [0] \\
[0] & \cdots & [0] & [A_{in,in}^{(D)}]
\end{array}
\right]
\begin{pmatrix}
v^{(1)} \\
v^{(2)} \\
\vdots \\
v^{(D)}
\end{pmatrix}
= \begin{pmatrix}
w^{(1)} \\
w^{(2)} \\
\vdots \\
w^{(D)}
\end{pmatrix}
\Rightarrow v^{(d)} = [A_{in,in}^{(d)}]^{-1}w^{(d)}. 
\] (4.17)

It shows that computing \( v = [A_{in,in}]^{-1}w \), given \( w \in \mathbb{R}^{N_{in}} \), amounts to solving \( D \) subsystems or local problems \emph{independently} over each subdomain. Not only does this call for the inversion of systems with much smaller sizes, typically \( N_{in}^{(d)} \approx N_{n}/D \), but these computations can be carried out in parallel for different subdomains. The same remark also applies to the determination of the right-hand-side \( \widehat{b} \) of the reduced problem (4.15). The possibility of applying efficiently the condensed operator \( \widehat{A} \) on a vector \( u \in \mathbb{R}^{N_{r}} \), through local solves over subdomains, motivates the use of matrix-free type iterative methods where \( \widehat{A} \) is never formally assembled. In such
an approach, one eventually only computes the sparse matrices $[A_{r,r}]$ and $[A_{i,n}]$ and the local problem matrices $[A_{i,n}]_{d}$. The latter, owing to their low dimension, can even be factorized to speed-up subsequent products with $[A]$.

Finally, when the reduced problem solution $u_{r}$ is obtained, one can compute the solution over selected subdomains solving local problems with corresponding Dirichlet boundary conditions in $u_{r}$ (see below).

**Subdomains expansion of the condensed operator**

The discussion above highlighted the role of the local problems in the structure of the condensed problem. In fact, the system in (4.15) can be formally recast to highlight independent contributions from the subdomains, namely according to:

$$
[A]u_{r} = \sum_{d=1}^{D} [A]_{d}^{(d)} u_{r}^{(d)}, \quad \hat{b} = b_{r} + \sum_{d=1}^{D} b_{(d)}. \tag{4.18}
$$

Focusing first on the right-hand-side expansion, we identify

$$
\hat{b}^{(d)} = -[A_{r,i}^{(d)}][A_{i,n}^{(d)}]^{-1}b_{n}^{(d)}. \tag{4.19}
$$

Note that for simplicity, the expressions above are formal and involved an abuse of notations. In particular, we allow varying the size of matrices and vectors, by removing unnecessary entries or nodal components, or by padding with zeros when necessary. For instance, depending on the context the matrix $[A_{i,n}^{(d)}]^{-1}$ can be understood as a $N_{n}^{(d)} \times N_{n}^{(d)}$ matrix or as its padded to zero to form the $N_{n} \times N_{n}$ version with only a nonzero $(d)$-th diagonal block. With this abuse of notation, the
inverse of $[A_{in,in}]$ in (4.17) can be formally written as

$$[A_{in,in}]^{-1} = \sum_{d=1}^{D} [A_{in,in}^{(d)}]^{-1}.$$  

Similarly, the expansion of $[\hat{A}]$ in (4.18) means that the expansion term $[\hat{A}]^{(d)}u_r^{(d)}$ accounts for the effect of the $(d)$-th subdomain only. To derive an expression for these matrices, we fix a subdomain $d$, select $n \in \mathcal{N}_r^{(d)}$ and consider the solution of

$$[A_{in,in}^{(d)}] u_{in,n}^{(d)} = -[A_{in,r}^{(d)}] e_n^{(d)},$$  

(4.20)

where $e_n^{(d)}$ is the canonical vector with all zero component except the n-th one equal to 1. The solution $u_{in,n}^{(d)}$ are the (internal) nodal values of the finite element approximation of the elliptic problem over $\Omega^{(d)}$ for homogeneous boundary conditions all over $\partial \Omega^{(d)}$, except at node $n \in \mathcal{N}_r^{(d)}$ where the nodal value is set to one. From this family of elementary solutions we define the vector

$$I_n^{(d)} = [A_{r,in}^{(d)}] u_{in,n}^{(d)} + [A_{r,r}^{(d)}] e_n^{(d)},$$  

(4.21)

where

$$[A_{r,r}^{(d)}]_{n,n'} = \begin{cases} \int_{\Omega^{(d)}} \kappa \nabla \Phi_n \cdot \nabla \Phi_{n'} dx, & n, n' \in \mathcal{N}_r^{(d)} \\ 0, & \text{otherwise.} \end{cases}$$  

(4.22)

We observe that the computation of the vector $I_n^{(d)}$ involves only quantities and operators localized on the considered subdomain. In particular, we note that the definition of the matrix in (4.22) involves an integral restricted to $\Omega^{(d)}$, such that
Finally, exploiting the linearity of elliptic equation and superposition principle, we obtain:

\[ \widehat{[A]} u_{\Gamma} = \sum_{d=1}^{D} \widehat{[A]}(d) u_{\Gamma}(d) \],

\[ \widehat{[A]} u_{\Gamma} = \sum_{n \in \mathcal{N}_{\Gamma}(d)} I_{n}^{(d)} \left( u_{\Gamma}(d) \right)_{n}, \quad (4.23) \]

showing that the columns of the matrices \( \widehat{[A]}(d) \) are made of the vectors \( I_{n}^{(d)} \).

Constructing the condensed operator expansion in (4.18) involves the solution over each subdomain of a set of local elliptic problems (4.20), in fact the same elliptic problem with \( \mathcal{N}_{\Gamma}(d) \) right-hand-sides. Although it can be performed efficiently in parallel, the explicit construction of the condensed operator is generally not considered in the practical implementation of domain decomposition approaches for elliptic problems, because of its computational complexity which is generally larger than that of the direct matrix-free iterative method described in the previous section. However, the case of stochastic elliptic problems is different as many stochastic samples may have to be computed, so that having an explicit representation of the (now stochastic) condensed operator may be interesting. We expand on this idea in the following sections.

4.2.2 Stochastic elliptic problem

Formulation of the stochastic problem

We now extend the deterministic problem in (4.1) to the stochastic case. The case of stochastic forcing \( f \) induces no particular difficulty and can be treated in the framework to be introduced below. For simplicity with restrict the presentation to the case of a random diffusion field \( \kappa \). Let \( (\Theta, \Sigma, \mu) \) be a probability triplet; the
problem now becomes

$$\nabla \cdot (\kappa(x, \theta) \nabla u(x, \theta)) = -f(x), \quad x \in \Omega, \theta \in \Theta,$$

(4.24)

with additional (almost sure) homogeneous Neumann and Dirichlet boundary conditions for $x \in \partial \Omega$. For the well-posedness of the problem, we assume that the random field $\kappa(x, \theta)$ is almost surely bounded below and above for almost every $x$. Then the stochastic solution $u$ has finite second order moments,

$$\mathbb{E} \left[ u(x, \cdot)^2 \right] = \int_{\Theta} u(x, \theta)^2 d\mu(\theta) < +\infty,$$

(4.25)

where $\mathbb{E} [\cdot]$ denotes the expectation operator.

As in the deterministic case, we proceed with the spatial discretization over a deterministic finite element space $V^{FE}$, expressing the discrete solution from its random nodal values over the mesh,

$$u(x, \theta) = \sum_{n \in N} u_n(\theta)\Phi_n(x) \in V^{FE}.$$

(4.26)

Above, we denoted the solution space $V^{FE}$ which results from the tensorization of the spatial FE space with the space of second order random variables: $V^{FE} = V^{FE} \otimes L_2(\Theta, \mu)$. The (semi) weak form is obtained multiplying (4.24) by $v \in V^{FE}$, and integrating (by parts) first over $\Omega$; this results in

$$\int_{\Omega} \kappa(x, \theta) \nabla u(x, \theta) \cdot \nabla v(x) dx = \int_{\Omega} f(x)v(x) dx, \quad \forall v \in V^{FE}.$$

Note that the equality stands in the almost sure sense. Given the approximation form in (4.26), the variational formulation can be recast in a linear system of equations.
involving the vector of random nodal values \( \mathbf{u}(\theta) \), the stochastic analogous of (4.6),

\[
[A](\theta)\mathbf{u}(\theta) = \mathbf{b}, \quad [A]_{n,n'}(\theta) = \int_{\Omega} \kappa(x, \theta) \nabla \Phi_n(x) \cdot \nabla \Phi_{n'}(x) dx. \tag{4.27}
\]

Direct Monte Carlo sampling

A common approach to solve the discrete stochastic problem is to resort to Monte Carlo (MC) sampling methods. In a MC approach, samples \( \kappa(x, \theta_i) \) of the random field are generated, leading to samples of the stochastic matrix \([A](\theta_i)\) and corresponding realizations \( \mathbf{u}(x, \theta_i) \in V^{FE} \) of the stochastic solution. Note that different samples can be computed in parallel. Different moments and statistics of the solution can be computed, in particular the solution mean and the two-points correlations can be estimated from

\[
\mathbb{E}[\mathbf{u}(x, \cdot)] = \lim_{M \to \infty} \frac{1}{M} \sum_{i=1}^{M} \mathbf{u}(x, \theta_i), \quad \mathbb{E}[\mathbf{u}(x, \cdot), \mathbf{u}(x', \cdot)] = \lim_{M \to \infty} \frac{1}{M} \sum_{i=1}^{M} \mathbf{u}(x, \theta_i)\mathbf{u}(x', \theta_i).
\]

The computational complexity of the method is thus proportional to the number of samples \( M \) one uses in the MC estimation, and there is an obvious interest in reducing the computational cost of generating individual samples. Applying efficient deterministic strategies is therefore critical, and for this purpose MC is well suited to reuse the domain decomposition method detailed in the previous section. To do so, we can first derive formally the stochastic form of the condensed problem,

\[
[\widehat{A}](\theta) \mathbf{u}_r(\theta) = \widehat{\mathbf{b}}(\theta), \tag{4.28}
\]
where,

$$\bar{A}(\theta) = [A_{\Gamma,r}] - [A_{\Gamma,in}]^{-1} [A_{in,r}] \theta, \quad \hat{b}(\theta) = b_{\Gamma} - [A_{\Gamma,in}]^{-1} b_{in},$$

(4.29)

and subsequently proceed with the MC sampling of the condensed problem to yield samples of subdomain boundary nodal values $u_{\Gamma}(\theta_i)$ and solution $u(x, \theta_i)$. If one uses a matrix-free iterative scheme without explicit construction of $\bar{A}(\theta_i)$, the heaviest part of the computation is dedicated to the assembly of the local problem operators $[A_{in,in}]^{(d)}(\theta_i)$ and possibly their factorizations.

At this point we remark that, contrary to the deterministic case, the stochastic condensed problem is going to be queried multiple times, as large values of $M$ are generally needed to obtain well converged MC estimators. This is quite a different situation from the deterministic case where the actual assembly of $\bar{A}$ appears computationally too expensive if it is to be queried only once. This observation suggests that there could be an interest in actually assembling the stochastic condensed problem, to sample from, as the overhead of the assembly would be factorized (amortized) over the subsequent $M$ samples. If such a strategy is feasible, one would jointly sample directly the matrix $\bar{A}(\theta)$ and right-hand-side $\hat{b}(\theta)$, to get samples of the boundary solution $u_{\Gamma}(\theta)$ by means of a matrix-based iterative method. As a result, one would only have to solve a unique local problem per subdomain for each sample, and only for the subdomain where the solution is sought.

To be effective, the approach just sketched would have to fulfill two conditions. First, the stochastic condensed problem matrix and right-hand-side must be represented in a format amenable to sampling. Second, the assembly overhead must
remain reasonable for the method to be practical. Below, we rely on stochastic spectral expansions to approximate the problem in a suitable format; then we exploit the underlying structure of the condensed problem, namely its expression as a sum of local stochastic operators, to come up with representation having manageable complexity.

4.3 Stochastic Spectral Expansion of the Condensed Problem

4.3.1 PC expansion of the elliptic solution

Stochastic spectral expansions have been proposed as an alternative to Monte-Carlo methods. The key observation supporting the spectral approach is the smooth dependences of the elliptic equation solution with respect to the diffusivity coefficients. This fact motivates the expansion of the solution $u(x, \theta)$ as a series of the form

$$u(x, \theta) = \sum_{\alpha} u_\alpha(x) \Psi_\alpha(\theta),$$

where the $\Psi_\alpha$ are random functionals. Typically, one starts by approximating the diffusion field $\kappa$ as a functional of a finite set of $N_\kappa \geq 1$ independent random variables $\eta(\theta)$ with known density:

$$\kappa(x, \theta) \approx \hat{\kappa}(x, \eta(\theta)). \quad (4.30)$$

Such parametrization of $\kappa$ can be obtained for instance by computing Karhunen-Loève expansions as in the following sections. As a result, the solution is a functional of $\eta(\theta)$, and the truncated spectral expansion becomes

$$u(x, \theta) \approx \tilde{u}(x, \eta(\theta)) = \sum_{\alpha \in A} u_\alpha(x) \Psi_\alpha(\eta(\theta)). \quad (4.31)$$
Classically, one considers expansions using orthonormal functionals $\Psi_\alpha$, in particular polynomials in $\eta$. In this case, the expansion in (4.31) is called the Polynomial Chaos expansion of $u$. The multi-index $\alpha = (\alpha_1, \ldots, \alpha_{N_\kappa}) \in \mathbb{N}^{N_\kappa}$ indicates the maximal polynomial degree $\alpha_k$ in each component $\eta_k$, and we shall denote $|\alpha| = \sum_{k=1}^{N_\kappa} \alpha_k$ the total degree of $\Psi_\alpha$. The functionals are orthonormal in the sense that

$$\mathbb{E}[\Psi_\alpha \Psi_\beta] = \int_{\Theta} \Psi_\alpha(\eta(\theta))\Psi_\beta(\eta(\theta))d\mu(\theta) = \begin{cases} 1, & \alpha = \beta, \\ 0, & \text{otherwise}. \end{cases}$$

Finally, the summation in (4.31) is restricted to $\alpha$ belonging to the multi-index set $\mathcal{A} \subset \{\alpha \in \mathbb{N}^{N_\kappa}\}$. Different strategies can be used to define this set; without loss of generality and unless specified otherwise, we shall control $\mathcal{A}$ by the maximal total polynomial degree $N_o$ of the expansion, setting

$$\mathcal{A} = \{\alpha \in \mathbb{N}^{N_\kappa}, |\alpha| \leq N_o\}.$$

Under mild assumptions on $\kappa$, the solution $u$ has exponentially converging expansions with respect to the number of random variables in $\eta$ and with the polynomial degree $N_o$ of the truncated form of the expansion. Regarding the computation of the expansion coefficients $u_\alpha$, different approaches have been proposed and improved over the last 25 years. These include the Galerkin methods, consisting in deriving a set of coupled problems for the $u_\alpha$ from the original stochastic elliptic problem, and the non-intrusive methods, where the $u_\alpha$ are estimated from a set of resolutions of the deterministic elliptic problem corresponding to realizations of $\eta$.

The main limitation in the applicability of the spectral expansions to the solution of stochastic elliptic problems comes with the number of terms in the series that can
be prohibitively large in some situations. This has motivated adaptive strategies, in particular low rank approximations. However, the case of diffusion fields $\kappa$ with large variances and short correlation lengths remains challenging because it requires, first, a large number $N_\kappa$ of random variables for their parametrization in (4.30) and, second, a high degree $N_o$ for the polynomial expansions. The issue can be seen from the expression of the number of terms in an expansion (with total degree truncation) involving $N_\kappa$ random variables and degree $N_o$:

$$P = |\mathcal{A}| = \frac{(N_\kappa + N_o)!}{N_\kappa!N_o!}. \quad (4.32)$$

Although more advanced truncation strategies have been proposed, in particular adapting the expansion order in the different variables of $\eta$, the relation (4.32) shows that cases of large dimensional problems ($\eta$) remain critical even for low orders, and that it is highly desirable to keep the dimension of $\eta$ as small as possible.

### 4.3.2 Spectral expansion of the condensed problem

**Local parametrization**

It is well known that the dimensionality of $\eta$ relates to the intrinsic stochastic dimensionality of $\kappa$ which, roughly speaking, corresponds to the minimal number of random variables in its parametrization. It is also known from the properties of second-order orthogonal decompositions à la Karhunen-Loève, that the stochastic dimensionality of a field over a fixed domain increases as its correlation length decreases. The stochastic dimensionality of a stationary process is actually governed by the ratio of correlation length and domain extension, expressing the fact that a lower number of random variables can be used to parametrize the process over a subdomain. This
feature is exploited in [19] where we proposed a reduced basis method to perform Karhunen-Loève decompositions (factorization of correlation functions) within a domain decomposition framework. Specifically, the stochastic parametrization of $\kappa$ is written as

$$\hat{\kappa}(x, \theta) = \sum_{d=1}^{D} 1_{\Omega^{(d)}}(x) \hat{\kappa}^{(d)}(x, \eta^{(d)}), \quad 1_{\Omega^{(d)}}(x) = \begin{cases} 1 & x \in \Omega^{(d)} \\ 0 & \text{otherwise} \end{cases}$$ (4.33)

In (4.33), $1_{\Omega^{(d)}}$ is the indicator function of a subdomain and $\hat{\kappa}^{(d)}$ is a local approximation of $\kappa$ over $\Omega^{(d)}$ which uses local random variables $\eta^{(d)}$ whose number $N^{(d)}_\kappa$ will be shown to be much less than for the global parametrization of $\kappa$ over the whole domain $\Omega$, see [19].

One cannot express the elliptic equation solution $u$ in a format similar to (4.33), using the same local random vectors $\eta^{(d)}$ as for the parametrization of $\kappa$. Indeed, the stochastic solution $u$ over a subdomain $\Omega^{(d)}$ depends on the whole set of local random variables $\{\eta^{(d)}, d = 1, \ldots, D\}$, because of the elliptic nature of the problem. In other words, it is not possible to expand $u$ for $x \in \Omega^{(d)}$ in terms of the local random variables $\eta^{(d)}$ only. This prevents the direct construction of a local expansion for $u(x \in \Omega^{(d)}, \theta)$ using a low dimensional polynomial basis constructed on the reduced set of $N^{(d)}_\kappa$ local random variables in $\eta^{(d)}$. Alternatively, the construction of a global expansion of $u(x \in \Omega, \theta)$ using the whole set of local variables would require a prohibitively large PC basis as it would involve $N_\kappa = \sum_{d=1}^{D} N^{(d)}_\kappa$ random dimensions. Note that the $\eta^{(d)}$ will be generally not independent so it could be possible to reduce the global number of random variables, but the approach would eventually remain at least as costly as for a direct parametrization of $\kappa$ as in (4.30). We thus consider a
different approach in the following, avoiding to seek a PC expansion of the solution.

**Local PC expansion of the condensed problem**

Although computational complexity reduction using direct local expansions of the solution cannot be achieved, we propose to take advantage of the low dimensionality of the local parametrization of $\kappa$ to accelerate the Monte-Carlo sampling of the stochastic solution discussed in the previous section. The key idea supporting the proposed approach comes from the following observation. Contrary to the solution over a subdomain, the contribution to the condensed problem of the subdomain can be approximated solely in terms of its local random variables $\eta^{(d)}$. Specifically, we can write

$$[\widehat{A}](\theta) = \sum_{d=1}^{D} [\widehat{A}]^{(d)}(\theta) \approx \sum_{d=1}^{D} [\widehat{A}]^{(d)}(\eta^{(d)}(\theta)), \quad (4.34)$$

with similar expressions of the right-hand-side $\widehat{b}(\theta)$. It suffices to remind that, in the deterministic case, the local condensed operator $[\widehat{A}]^{(d)}$ and right-hand-side $\widehat{b}^{(d)}$ can be determined solving local elliptic problems over $\Omega^{(d)}$ with selected boundary conditions. Our objective is therefore to construct local PC approximation as

$$[\widehat{A}]^{(d)}(\eta^{(d)}(\theta)) \approx [\widehat{A}]^{(d)}(\eta^{(d)}(\theta)) \equiv \sum_{\alpha \in \mathcal{A}^{(d)}} [\widehat{A}]_{\alpha}^{(d)} \Psi_{\alpha}(\eta^{(d)}). \quad (4.35)$$

To this end, we rely on the decomposition of $\kappa$ in (4.33) and we first consider the stochastic problems which are the counterpart of (4.20), namely for $n \in \mathcal{N}^{(d)}$ we solve

$$[A]^{(d)}_{\text{in, in}}(\eta^{(d)})u^{(d)}_{\text{in, n}}(\eta^{(d)}) = -[A]^{(d)}_{\text{in, f}}(\eta^{(d)})e^{(d)}_{n}. \quad (4.36)$$
The stochastic matrices $A_{\text{in,in}}^{(d)}(\eta^{(d)})$ and $A_{\text{in,r}}^{(d)}(\eta^{(d)})$ appearing in these problems now have entries of the form

$$\int_{\Omega^{(d)}} \hat{\kappa}^{(d)}(x, \eta^{(d)}) \nabla \Phi_n(x) \cdot \nabla \Phi_{n'}(x) dx.$$ 

Further, the solutions of the elementary problems (4.36) can be approximated on a local PC basis through

$$\bar{u}_{\text{in,n}}^{(d)}(\eta^{(d)}) \approx u_{\text{in,n}}^{(d)}(\eta^{(d)}) = \sum_{\alpha \in A^{(d)}} \left( u_{\text{in,n}}^{(d)} \right)_{\alpha} \Psi_{\alpha}(\eta^{(d)}).$$

The local basis defined by the local multi-index set $A^{(d)}$ may be based on different truncation strategies. In this work, we shall restrict ourselves to the simplest case of total order truncation using a fixed polynomial order $N_o \geq 1$ for all the subdomains; the local basis cardinality $P^{(d)}$ is then function of the number $N^{(d)}$ and given by (4.32). We stress that, as we expect $N^{(d)} \ll N$, $P^{(d)}$ is much reduced because of its exponential dependence on the number of random variables ($N^{(d)}$).

For the computation of the expansion coefficients $\left( u_{\text{in,n}}^{(d)} \right)_{\alpha}$ we shall rely on the Galerkin approximation of (4.36). Specifically, we solve

$$\sum_{\alpha \in A^{(d)}} \mathbb{E} \left[ \left[ A_{\text{in,in}}^{(d)} \right]_{\alpha} \psi_{\beta} \right] \left( u_{\text{in,n}}^{(d)} \right)_{\alpha} = -\mathbb{E} \left[ \left[ A_{\text{in,r}}^{(d)} \right] \psi_{\beta} \right] \epsilon_{\text{n,\beta}}^{(d)}, \quad \forall \beta \in A^{(d)}. \quad (4.37)$$

Note that the size of this linear problem is $N^{(d)}_{\text{in}} \times P^{(d)}$, stressing the importance of achieving low-dimensional local parameterization. We also remark that only the right-hand side of this system is changing for different $n \in N^{(d)}_{\text{in}}$. This can be exploited to efficiently compute the set of local solutions, for instance by pre-factorizing the

104
linear system or employing an iterative solver designed to handle multiple right-hand sides. Further, these sequences of problems are independent from a subdomain to another, and so they can be carried out in parallel. Finally, from the PC expansion of \( u_{in,n}^{(d)}(\eta^{(d)}) \) we derive the PC expansion of the columns \( I_n^{(d)}(\eta^{(d)}) \) for the subdomain contribution to the stochastic condensed operator (see (4.23)),

\[
I_n^{(d)}(\eta^{(d)}) \approx \tilde{I}_n^{(d)}(\eta^{(d)}) = \sum_{\alpha \in A(d)} (I_n^{(d)})_{\alpha} \Psi_{\alpha}(\eta^{(d)}),
\]

using the Galerkin interpretation of the matrix-vector product:

\[
(I_n^{(d)})_{\alpha} \doteq \sum_{\beta \in A(d)} \mathbb{E} \left[ \left[ A_{\Gamma,in}^{(d)} \right] \Psi_{\alpha} \Psi_{\beta} \right] (u_{in,n}^{(d)})_{\beta} + \mathbb{E} \left[ \left[ A_{\Gamma,\Gamma}^{(d)} \right] \Psi_{\alpha} \right] e_n^{(d)}. \tag{4.38}
\]

A similar procedure is employed to derive the PC approximations of the stochastic subdomain contributions to the condensed problem right-hand side, namely

\[
\hat{b}^{(d)}(\eta^{d}) \approx \tilde{b}^{(d)}(\eta^{d}) = \sum_{\alpha \in A(d)} \hat{b}_{\alpha}^{(d)} \Psi_{\alpha}(\eta^{(d)}). \tag{4.39}
\]

### 4.3.3 Sampling the stochastic condensed problem

At this point, we have described a strategy to compute a composite PC expansion of the condensed problem. These approximations can be used to generate approximate samples of the solution à la Monte Carlo. This task amounts to sampling jointly the local random variables \( \eta^{(d)} \) of the subdomains as illustrated in the following example section. We shall denote \( \eta^{(d)}_i = (\eta^{(d)}_i(\theta_i) \) a sample of the local random variables; the corresponding sample of the condensed problem solution \( u_{\Gamma}(\theta_i) \) is defined through

\[
[\widehat{A}](\theta_i)u_{\Gamma}(\theta_i) = \tilde{b}(\theta_i), \tag{4.40}
\]

105
The key advantage of the proposed approach is the substitution of the exact condensed operator with its composite PC approximation. As a result, applying $\mathbf{A}(\theta_i)$ to a given vector in an iterative solution method for (4.40) is much less costly than having to solve local problems in the classical method. Indeed, forming the reduced problem essentially amounts to evaluating polynomial expansions for the subdomains contribution, which can be made in parallel. Obviously, this comes at the cost of having first to compute the PC approximation $\mathbf{A}(\theta_i)$ in the preprocessing stage; however this overhead is factorized over the number of samples subsequently generated.

Note that when the sample $\mathbf{u}_\Gamma(\theta_i)$ solving (4.40) is obtained, the local problems can be independently solved for (and only for) the subdomains where the solution is sought. Specifically, once $\mathbf{u}_\Gamma(\theta_i)$ is computed, one can solve (independently)

$$\mathbf{A}^{(d)}_{\text{in,in}}(\eta_i^{(d)}, \theta_i) \mathbf{u}_{\text{in}}^{(d)}(\theta_i) = \mathbf{b}_{\text{in}}^{(d)} - \mathbf{A}^{(d)}_{\text{in,r}}(\eta_i^{(d)}, \theta_i) \mathbf{u}_{\Gamma}^{(d)}(\theta_i),$$

(4.42)

to get the finite element approximation of $u(x, \theta_i)$ for $x \in \Omega^{(d)}$.

4.3.4 Monte Carlo algorithm and implementation

The proposed method thus involves two distinct steps as summarized in Algorithm 2: a preprocessing stage where the PC approximations of the condensed problem are constructed and the Monte Carlo sampling of the approximate solution.
Given a partition of $\Omega$ into $D$ subdomains and associated local random variables for the parametrization of $\kappa$, the preprocessing stage is dedicated to the construction of the local approximations for the condensed problem. The treatments of different subdomains are fully independent and can be trivially carried out in parallel (loop starting at line 2). For each subdomain, the main computational effort is the solution of a local stochastic elliptic problem (with multiple right-hand-sides), whose size is made reasonable by considering sufficiently many subdomains so $N_{in}^{(d)}$ and $N_{\kappa}^{(d)}$ are sufficiently small. The memory requirement to store the local PC expansions $\mathbb{A}^{(d)}(\eta^{(d)})$ and $\mathbb{b}^{(d)}(\eta^{(d)})$ is proportional to $N_{r}^{(d)} \times N_{\Gamma}^{(d)} \times P^{(d)}$ and $N_{r}^{(d)} \times P^{(d)}$ respectively.

In the sampling stage, starting at line 11, one generates joint samples $\eta^{(d)}(\theta_i)$ and evaluates the subdomain contributions to the sample condensed problems. This involves polynomial evaluations which can be carried out in parallel over distinct subdomains (loop starting at line 13). The resulting sampled problem (4.40) can be solved for instance by means of an iterative method, without having to resort to any local problem solve. When the sample $u_{\Gamma}(\theta_i)$ is computed, see line 16, one can eventually recompute classically the solution over subdomains of interest (loop starting at line 17). Again these final solves over different subdomains can be carried out in parallel.

The solver for the approximate condensed problem (4.40) can eventually be implemented in parallel, and another advantage of the proposed approach is the possibility of relying on a preconditioned iterative method. For instance, we show in the example section how to take advantage of the manageable dimension and explicit
representation of the condensed operator to determine an effective preconditioner for the sampled problems. This preconditioner is a carefully selected realization of $\hat{[A]}$, whose LU decomposition is computed at the preprocessing stage and subsequently employed in the sampling stage to further accelerate the convergence of the iterative solves.

**Algorithm 2:** Proposed method.

**Data:** Partitioning of the domain, local parametrization of $\kappa$, polynomial order $N_o$

**Result:** Produce $M$ samples of the stochastic solution

1. **Preprocessing stage:** approximation of the condensed problem
   
   for subdomain with index $d = 1, \ldots, D$ do
   
   2. Set local PC basis
   
   3. Compute PC expansion $\tilde{b}(\eta^{(d)})$
   
   4. for boundary node $n \in N_{\Gamma}^{(d)}$ do
   
   5. Solve local stochastic problem (4.37)
   
   6. Set PC expansion of n-th column of $\hat{[A]}^{(d)}(\eta^{(d)})$ using (4.38)
   
   7. end for
   
   8. end for

9. **Monte-Carlo Sampling Stage:** Generate approximate samples of solution

   for sample index $i = 1, \ldots, M$ do

   10. Generate a random sample of $\eta_i = (\eta_i^{(1)} \ldots \eta_i^{(D)})$

      11. for subdomain with index $d = 1, \ldots, D$ do

      12. Compute $\hat{[A]}^{(d)}(\eta_i^{(d)})$ and $\tilde{b}^{(d)}(\eta_i^{(d)})$ using (4.41)

      13. end for

   14. Solve sampled condensed problem (4.40) for $u_{\Gamma}(\theta_i)$

      15. for subdomain with index $d = 1, \ldots, D$ do

      16. Solve local problem (4.42) for the inner unknowns $u^{(d)}_{in}$

      17. end for

   18. end for
4.4 Example of Stochastic Elliptic Problem

In the following sections we illustrate the application of the proposed methods to an elliptic equation with log-normal coefficient field. The problem settings are detailed in Section 4.4.1. Next, we provide various convergence studies in Section 4.4.2 to investigate the behavior of the method with respect to its principal numerical parameters, namely the number of subdomains, $D$, and the PC order, $N_o$, of the PC expansions of operators $\tilde{A}^{(d)}$ and right-hand-side $\tilde{b}^{(d)}$. In Section 4.4.3 we focus on the case of random coefficient $\kappa$ with high variability to highlight the main mechanism driving the error in the method in extreme problems. Finally, the efficiency and parallel implementation of the method are discussed in Section 4.5.

4.4.1 Test problem

We consider the elliptic problem (4.24) over a two-dimensional domain consisting of the unit square, $\Omega = (0,1)^2$. We set $f(x) = 1$ and adopt homogeneous boundary conditions as follows:

$$u(x) = 0 \text{ for } x \in \partial \Omega_D, \text{ and } \nabla u \cdot \hat{n} = 0 \text{ for } x \in \partial \Omega_N,$$

(4.43)

where $\partial \Omega_D$ corresponds to the West, South, and East sides of the domain; $\partial \Omega_N$ corresponds to the North side of the domain; and $\hat{n}$ is the unit normal to the boundary $\partial \Omega_N$.

For the random field $\kappa$, we assume that $\kappa - \kappa_{\text{min}}$ is a stationary log-normal process, such that

$$G(x, \theta) \equiv \log (\kappa(x, \theta) - \kappa_{\text{min}}) \sim N(\mu_G(x), C(x, x')).$$

Here, we have denoted $N(\mu_G, C)$ the Gaussian process with mean $\mu_G$ and covariance
function $C$, whereas $\kappa_{\text{min}}$ is a small positive constant ensuring the well-posedness of the problem (in $L_2$-sense). We shall classically a covariance function having a square exponential decay,

$$C(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2L^2}\right),$$

where $L$ is the correlation length and $\sigma^2$ the variance. In the following, we use $L = 0.1$, unless otherwise indicated.

For the local parametrization of the process, we consider the local Karhunen-Loeve expansion of $G$ over each of the subdomain. Denoting $G^{(d)}$ the restriction of $G$ over $\Omega^{(d)}$, we have

$$G^{(d)}(\mathbf{x}, \theta) = \mu_G(\mathbf{x}) + \sum_{k=1}^{\infty} \sqrt{\lambda_k^{(d)}} g_k^{(d)}(\mathbf{x}) \eta_k^{(d)}(\theta),$$

where the $\lambda_k^{(d)}$ and $g_k^{(d)}$ are the (dominant) eigenvalues and normalized eigenfunctions of the covariance satisfying

$$\int_{\Omega^{(d)}} C(\mathbf{x}, \mathbf{x}') g_k^{(d)}(\mathbf{x}') d\mathbf{x}' = \lambda_k^{(d)} g_k^{(d)}(\mathbf{x}).$$

It is a standard result that the random variables in the KL expansion above are independent standard Gaussian random variables, that is $\eta_k^{(d)} \sim N(0, 1)$. Obviously, the KL expansion must be truncated; we shall truncate (4.45) to the $N_k^{(d)}$ first dominant (largest eigenvalues); accordingly, we have:

$$G^{(d)}(\mathbf{x}, \theta) \approx \hat{G}^{(d)}(\mathbf{x}, \eta^{(d)}(\theta)) = \mu_G(\mathbf{x}) + \sum_{k=1}^{N_k^{(d)}} \sqrt{\lambda_k^{(d)}} g_k^{(d)}(\mathbf{x}) \eta_k^{(d)}(\theta)$$

110
and where $N^\kappa(d)$ is selected from the following criteria (see [19]):

$$
\sum_{k=1}^{N^\kappa(d)} \lambda_k(d) \geq (1 - \epsilon_G)\sigma^2 \frac{|\Omega(d)|}{|\Omega|}.
$$

(4.48)

Here, $\epsilon_G < 1$ is a small positive constant measuring the approximation error in the $L_2$-sense and $|\Omega|$ (resp. $|\Omega(d)|$) is the volume of the domain $\Omega$ (resp. $\Omega(d)$). Extending to zero the eigenfunctions outside of their respective supports $\Omega^d$, we end up with

$$
\kappa \approx \hat{\kappa} = \kappa_{\min} + \sum_{d=1}^{D} \hat{\eta}^d(x)\hat{\kappa}^d(x, \eta^d), \quad \hat{\kappa}^d(x, \eta^d) = \exp \left[ \hat{G}^d(x, \eta^d) \right],
$$

(4.49)

which has a structure similar to (4.33).

For the Monte Carlo sampling of the problem, we will have to sample jointly the $\eta^d$. Since we know that the random variable are Gaussian and centered, we must provide the correlation structure between the random variables $\eta_k^d$. In [19], we have shown that the correlation structure is given by

$$
\sqrt{\lambda_l^{(d)} \lambda_l^{(d')}} \mathbb{E} \left[ \eta_l^{(d)} \eta_l^{(d')} \right] = \int \int \limits_{\Omega^{(d)} \times \Omega^{(d')}} g_l^{(d)}(x)g_l^{(d')}(x')C(x, x')dx'dx'.
$$

In particular, one observes that by construction the $\eta_l^{(d)}$ of a subdomain are uncorrelated as expected. Sampling the whole set of $\eta = \{\eta^d, d = 1, \ldots, D\}$ can be achieved by standard techniques, e.g., decomposing the covariance matrix.

As an illustration of the parametrization of the random field $\kappa$, we provide in the top row of Figure 4.1 three realizations for $\mu_G = 0$ and an increasing value of $\sigma^2$ of $G$ from left to right. In these examples, the number of subdomains is set to $D = 480$.
and the boundaries of the subdomains are outlined in the plot. With $L = 0.1$ and $\epsilon_G = 0.01$, a total of $N_\kappa = 178$ would be necessary in a global construction. Instead, one only needs $N^{(d)}_\kappa = 3$ random variables per subdomain for the same level of approximation error, $\epsilon_G$. The plots show the effect of varying $\sigma^2$ with its direct impact on the range of variability for $\kappa$, which is roughly 10 times larger for $\sigma^2 = 0.5$ compared to the case with $\sigma^2 = 0.05$. The plots also illustrate the spatial structure of the fields with multiple local minima and maxima, due to the small correlation length, and the exponentiation effects that emphasize the maxima and stiffen the gradients.

4.4.2 Validation of the method

Unless specified otherwise, the computation of this section uses $L = 0.1$ and a finite element mesh having $N_e = 16,441$ triangular quadratic elements and $N_n = 32,747$ unknowns.

Solution samples

We first verify that the proposed method with PC approximation of the condensed problem approximates the MC samples obtained with the original approach described in Section 4.2.2. To this extent, we refer to our approach as the DD-PC method and denote $\hat{u}(\eta_i)$ a corresponding finite element solution sample, while $u(\eta_i)$ is a finite element solution sample for the direct-sampling method. For fairness, when comparing two solution samples of $\hat{u}$ and $u$ we use the same approximation of the random field $\hat{\kappa}(\eta_i)$, so their difference $u - \hat{u}$ is solely due to the PC approximation error of the condensed problem.
First, we look at three different realizations of $\kappa(\eta_i)$ corresponding to different variances for the underlying Gaussian process $G$. The realizations of $\kappa$ are shown in the top row of Figure 4.1. The second row of Figure 4.1 shows the difference between $u(\eta_i)$ and the corresponding mean, $E[u]$, which enables us to highlight the complexity and length-scales in the solution samples. (The means, $E[u]$, are depicted in the top row of Figure 4.2). Finally, the third row of Figure 4.1 depicts the differences between the realizations computed with the DD-PC and the direct method. Here, the DD-PC solutions are computed using PC approximations with order $N_o = 2$ for all the 3 variances of $G$.

Focusing on the case with lowest variance, $\sigma^2 = 0.05$ (left column), the realization $u(\theta_i)$ is seen to be rather smooth, with differences less than $6 \times 10^{-4}$ between $u$ and $\hat{u}$. As $\sigma^2$ is increased to 0.20 (center column) the realization has now steeper gradients whereas the error level is now as high as $2 \times 10^{-3}$, roughly 1% of the maximum of $E[u]$. For the largest variance $\sigma^2 = 0.50$, the solution presents even steeper gradients and the peak error is as high as 10% of the maximum of $E[u]$. These observations are expected, because with increasing $\sigma^2$ a higher PC order $N_o$ would be needed to achieve a certain relative accuracy in the local condensed problem. This is verified in the following.

**Convergence with PC order**

We now analyze the behavior of the DD-PC method, starting from the Monte Carlo error in the estimation of the mean of the finite element solution, namely $E[\hat{u}] - E[u]$. These errors are reported in Figure 4.2; shown are the mean fields ($E[u]$, top row) and error fields for two expansion orders ($N_o = 2$ in the middle row and $N_o = 6$...
<table>
<thead>
<tr>
<th>$\sigma^2 = 0.05$</th>
<th>$\sigma^2 = 0.20$</th>
<th>$\sigma^2 = 0.50$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\kappa(x, \eta_i)$</td>
<td>$\kappa(x, \eta_i)$</td>
<td>$\kappa(x, \eta_i)$</td>
</tr>
<tr>
<td>$u(x, \eta_i) - \mathbb{E}[u(x, \cdot)]$</td>
<td>$u(x, \eta_i) - \mathbb{E}[u(x, \cdot)]$</td>
<td>$u(x, \eta_i) - \mathbb{E}[u(x, \cdot)]$</td>
</tr>
<tr>
<td>$\hat{u}(x, \eta_i) - u(x, \eta_i)$</td>
<td>$\hat{u}(x, \eta_i) - u(x, \eta_i)$</td>
<td>$\hat{u}(x, \eta_i) - u(x, \eta_i)$</td>
</tr>
</tbody>
</table>

**Figure 4.1:** Realizations of $\kappa$ (top row), deviation to the mean, $u - \mathbb{E}[u]$, of the direct solution (middle row) and differences between DD-PC and direct solutions (bottom row). The columns correspond to different realizations of $\kappa$ drawn at random using Gaussian fields with increasing variance: $\sigma^2 = 0.05, 0.2$ and $0.5$ from left to right. The DD-PC solutions use $N_o = 2$, and $D = 480$ (the subdomains partition is shown in all the figures).

in the bottom row) for the DD-PP method and the three variances $\sigma^2$ as before. Again, a total of $D = 480$ subdomains is used. Focusing first on the lower order case, $N_o = 2$, we observe that the error increases with $\sigma^2$, with higher values in $\Omega$ where $\mathbb{E}[u]$ is larger. This indicates that the DD-PC method is biased. Note also
that in the case with $\sigma^2 = 0.05$, when the error on the mean is the lowest, the field $\mathbb{E} [\hat{u}] - \mathbb{E} [u]$ appears noisy. This is due to the finite number of samples used in the Monte Carlo estimate of the expectations ($M = 500,000$) which induces a sampling error that is significant compared to the true ($M = \infty$) value of error on the mean solution. Increasing the PC order to $N_o = 6$ is seen to reduce by several orders of magnitude the error in mean of the DD-PC method. In fact, with $N_o = 6$ the error is so low that even for the largest $\sigma^2$ the MC sampling error remains significant and visible, whereas for the smallest $\sigma^2$ it is completely dominant.

To better understand the impact of $N_o$ on the bias in the DD-PC method, we define the normalized $L_2$ error on the mean, $\epsilon_{\text{mean}}$, according to:

$$
\epsilon_{\text{mean}}^2 = \frac{\| \mathbb{E} [\hat{u}] - \mathbb{E} [u] \|_{L_2(\Omega)}^2}{\| \mathbb{E} [u] \|_{L_2(\Omega)}^2}, \quad \| u \|_{L_2(\Omega)}^2 = \int_\Omega |u(x)|^2 dx.
$$

(4.50)

In practice the mean solutions $\mathbb{E} [\hat{u}]$ and $\mathbb{E} [u]$ are estimated by their empirical averages using $M$ Monte Carlo samples. We report in Figure 4.3 the evolution with $M$ of the estimate of $\epsilon_{\text{mean}}$ for the different values of $N_o$ and $\sigma^2$. We observe that for small values of $M$ the error norm $\epsilon_{\text{mean}}$ is overestimated because of the sampling error. The sampling error decreases as $M$ increases, and for $M$ large enough we see that $\epsilon_{\text{mean}}$ converges to a non-zero value, reflecting the bias in the DD-PC method. Moreover, as we saw before, the bias depends on both $\sigma$ and $N_o$. Specifically, higher values of $\sigma$ result in higher errors on the mean, and higher values of $N_o$ increase the accuracy of the PC expansion and reduce the bias. An important remark is that for high polynomial orders the sampling error will be dominant unless a large number of samples is used in estimating any desired statistic. Thus, there is not point in using
a large polynomial order for a small sample size.

The convergence with $N_o$ of the DD-PC method is not restricted to the mean solution but can be expected for other quantities of interest derived from $u$, albeit possibly with different rates. For instance, we report in Figure 4.4 the convergence of the error in the standard deviation of $u$, namely $\text{Std} [\hat{u}(\mathbf{x})] - \text{Std} [u(\mathbf{x})]$ for $N_o = 2$ and $N_o = 6$, and the 3 values of the variance $\sigma^2$. The plots show a similar trend as
for the mean solution, although the spatial structure of the standard deviation error appears to depend more heavily on \( \text{No} \).

### 4.4.3 \( L_2 \)-error norm

We now consider the more generic error measure as the full (or stochastic) \( L_2 \)-norm of the difference \( \hat{u}(x, \theta) - u(x, \theta) \) and define the relative stochastic error norm as

\[
\epsilon_u^2 = \frac{\mathbb{E}\left[ \| \hat{u} - u \|_{L_2(\Omega)}^2 \right]}{\mathbb{E}\left[ \| u \|_{L_2(\Omega)}^2 \right]}.
\]  

(4.51)

Figure 4.5 reports \( \epsilon_u \) as a function of the PC order \( \text{No} \). Shown are plots for different values of \( \sigma^2 \) and curves for different \( D \). The relative error on the mean, \( \epsilon_{\text{mean}} \), is also shown for comparison. For \( \sigma^2 = 0.05 \) (left plot) we notice that the behavior of both errors is very similar, decaying monotonically with \( \text{No} \), with the relative stochastic error higher than the relative error on the mean. Further, the number of subdomains \( D \) is seen to have negligible effect on the two errors. These
observations are in sharp contrast with the high variability case, $\sigma^2 = 0.5$, shown in the right plot of Figure 4.5, where the error decay with $N_o$ is no longer monotonic over the reported range. In fact, the convergence curves highlight an even-odd effect with a smaller error for even order $N_o = 2n$ than for the next odd order $N_o = 2n + 1$. In addition, the relative stochastic error reaches dramatically large levels for odd
values of $N_o$ and has much more severe and non trivial dependences on $D$. We should remark that the case with $\sigma^2 = 0.5$ leads to a very high variability in $\kappa$ and can be considered as an extreme case. In the following, we proceed to analyze the stochastic error in this large variability case.

![Figure 4.5](image_url)

**Figure 4.5**: Relative stochastic error $\epsilon_u$ and relative error on the mean $\epsilon_m$ as functions of the PC order $N_o$, for different values of $D$ as indicated and $\sigma^2 = 0.05$ (left plot) and 0.5 (right plot).

To better understand the error mechanism, we first reduce the computational cost of this analysis, namely by increasing the correlation length of $G$ to $L = 1$ but keeping $\sigma^2 = 0.5$. The increased $L$ allows to consider a coarser finite element mesh (with $N_o = 1,630$ elements and $N_n = 3,204$ unknowns), owing to the increased length scales in the solution $u$. However, this change does not affect the odd-even order effects just discussed, as shown by the convergence curves reported in Figure 4.6 which are similar to the previous case (Figure 4.5, left plot). Note that due to the coarser nature of the mesh, we also considered different values for the number of subdomains; $D = 120, 240, \text{ and } 480$, instead of $D = 240, 480, \text{ and } 960$.

For the purpose of the analysis, we compute 100,000 samples of the solutions $\hat{u}(\eta_i)$

119
and $u(\eta_i)$, using the two approaches, and retrieve the corresponding samples of error
norm, $\|\hat{u} - u\|_{L^2(\Omega)}$, norm of the DD-PC samples solution, $\|\hat{u}(\eta_i)\|_{L^2(\Omega)}$, Frobenius
norm of the error on the condensed problem operator, $\|\hat{\mathbf{A}}(\eta_i) - \mathbf{A}(\eta_i)\|_F$, and
finally condition number of its PC approximation, cond $\|\mathbf{A}(\eta_i)\|_F$. These samples are
used to estimate the statistics of these quantities, which are summarized in Figure 4.7
using histograms in log-log scale, contrasting the cases of $N_o = 2, 3$ and 9 for the PC
approximation of the condensed problem.

First, the statistics of the error norms $\|\hat{u} - u\|$, depicted in Figure 4.7a, are seen to
be more spread for $N_o = 3$ than for $N_o = 2$ with a much longer tail towards the high
error side: extreme samples for $N_o = 3$ are standing more than 3 orders of magnitude
away from the extreme samples for $N_o = 2$. The presence of very large error samples
induces the average-error behavior shown in Figure 4.6, even though the mode of the
histogram for $N_o = 3$ is at a lower error level compared to the mode for $N_o = 2$. On
the contrary, the error distribution for even order $N_o = 2$ does not exhibit a long tail
towards higher error values. In addition, increasing the PC order to $N_o = 9$ results
in a distribution of the error that remains quite broad (in the logscale) with broad right-tail but shifted to the low error values compared to $N_o = 3$. Overall the error samples for $N_o = 9$ remain lower than for $N_o = 2$. One can conclude that the large stochastic error for odd orders is caused by a fraction of samples having abnormally very high error compared to their median error, but with a probability that decreases with increasing order.

Comparing the statistics of the error on the condensed operator $\| [\hat{A}] - [A] \|_F$ at different orders, shown in Figure 4.7b, we observe a monotonic shift of the histograms when $N_o$ increases with similar tails for both odd and even orders. This is expected as one globally improves the PC approximation with increasing $N_o$. This distribution of the operator error must be contrasted with the statistics of the condensed operator condition number $\text{cond} [\hat{A}]$ reported in Figure 4.7c: the histogram for $N_o = 3$ is seen to exceed by several orders of magnitude the highest values for $N_o = 2$ and $N_o = 9$. In fact, the histogram for $N_o = 3$ reveals samples with poorly conditioned systems. Since the error on the operator itself behaves well, one can suspect the PC approximation of $[\hat{A}]$ to induce error on the lowest part of the spectrum, that is the smallest eigenvalues and eigenfunctions of $[\hat{A}]$ (recall that $[\hat{A}]$ is symmetric positive definite).

To evidence the role of the condition number and error on the lowest eigen values of $[\hat{A}]$ on the error, we present in Figure 4.8 samples of the condition number of $[\hat{A}](\eta_i)$ as a function of the corresponding samples of the error $\| u - \hat{u} \|_{L_2(\Omega)}$ for different PC orders $N_o$. The sample points have also been colored by the sign of the smallest eigenvalue of $[\hat{A}](\eta_i)$: in blue for a positive value and in green for a
Figure 4.7: Log-Histograms of the error norm \( \| u - \tilde{u} \|_{L^2(\Omega)} \) (left), of the approximation error on condensed operator \( \| [A] - \tilde{A} \|_F \) (center), and of the condition number of the approximate system \( \text{cond} \left( \tilde{[A]} \right) \) (right) for PC orders \( N_o = 2, 3, 9 \). Case of \( G \) with \( \sigma^2 = 0.5 \) and \( L = 1 \).

Negative value. Focusing first in the case \( N_o = 2 \) reported in Figure 4.8a, we observe that the error tends to be correlated with the condition number of the system. In particular, the minimal error increases when \( \text{cond} \tilde{[A]} \) increases. The case of \( N_o = 3 \) in Figure 4.8b appears to have an even more pronounced correlation of the error with the condition number with additional events associated to large condition number and high error level. The color clearly highlights the fact that the highest errors and condition number events are associated with a loss of positivity in \( \tilde{[A]} \). In fact, the error distribution is somehow bimodal, with one or the other mode depending highly \( \tilde{[A]} \) having negative eigenvalues. On the contrary, we report no sample with negative eigenvalues in our experiments for \( N_o = 2 \) (and also for \( N_o = 4, 6, \) and 8; not shown for brevity). Further, increasing the order to \( N_o = 5 \) and \( N_o = 9 \) in Figure 4.8c and 4.8d we observe the reduction of the probability of loss of positivity.
events (which is not at all observed in the whole sample set with \( N_o = 9 \)), and correspondingly a reduction of the resulting solution error. As a closing remark, detection of the loss of positivity in the samples of \( \hat{[A]} \) would be a good indicator of an insufficient PC order in the approximation. In our experiments, we found that checking for the positivity of the diagonal elements of \( \hat{[A]}(\eta_i) \), a necessary condition for the positivity of the sample, was sufficient for this purpose.

4.5 Performance Analysis

In Section 4.5.1 we provide a brief analysis of the computational complexity and memory requirements of the DD-PC method. A few alternative parallel implementations are discussed in Section 4.5.2, and subsequently compared in Section 4.5.3.

4.5.1 Complexity analysis

As highlighted in Algorithm 2, the proposed method has two distinct stages: a preprocessing stage during which the PC approximation of the condensed problem is computed, and a sampling stage where approximate samples of the solution are computed.

For the first stage, one has to solve on each subdomain a stochastic problem for a set of \( N^{(d)} \) distinct boundary conditions; this discretized stochastic problem has \( N^{(d)} \) unknowns expanded on a \( P^{(d)} \) dimensional PC basis. Eventually, the storage of the PC approximation for the subdomain contributions \( \hat{[A]}(d) \) and \( \hat{b}^{(d)} \) has a memory requirement of \( N^{(d)} \times \left( N^{(d)} + 1 \right) \times P^{(d)} \). Clearly, \( N^{(d)} \), \( N^{(d)} \) and \( P^{(d)} \) are the parameters driving of the computational complexity of the preprocessing stage on a subdomain, and we illustrate their evolutions when one considers an increasing number D of
Figure 4.8: Samples of the error in the solution $\|u - \hat{u}\|_{L^2(\Omega)}$ as a function of the condition number $\text{cond}(A)$. The samples are colored according to the sign of the smallest eigenvalue of $[A]$. Different PC orders as indicated.

Subdomains to partition a fixed mesh ($N_e = 163,272$) on the previous problem with $L = 0.1$, $\sigma^2 = 0.2$ and $\epsilon_G = 0.01$. Note that the underlying unstructured mesh is essentially isotropic with uniform refinement. The Metis software [45] is employed.
here to partition the domain; several examples are shown in Figure 4.9.

\[ (a) \ D = 15. \]
\[ (b) \ D = 60. \]
\[ (c) \ D = 960. \]

**Figure 4.9:** Partitions of the computational mesh into different numbers of subdomains D as indicated.

The results are reported in Table 4.1. The second column shows the evolution with D of the condensed problem dimension $N_\Gamma$. The third and fourth columns report the corresponding values of $N_\Gamma^{(d)}$ and $N_{in}^{(d)}$ (rounded averages over the set of subdomains, with ± RMS values). It is seen that while $N_{in}^{(d)} \sim 1/D$, the decay of $N_\Gamma^{(d)}$ is slower denoting the number of interfaces increasing with D (see Figure 4.9). Similarly, the number of local random variables $N_\kappa^{(d)}$ decreases at a sub-linear rate with respect to 1/D and would tend asymptotically to 1 for $D \to N_e$ (see the discussion in [19]). The decay behavior of $N_\kappa^{(d)}$ induces an extremely fast decay rate of the local polynomial basis dimensions $P^{(d)}$ with D as reported in the last two rows of Table 4.1, corresponding to PC degrees $N_o = 2$ and 6 respectively. For instance, when $N_o = 6$ the local PC basis dimension is 10,000 times smaller for $D = 480$ than for $D = 8$. However, when D becomes too large, $N_\kappa^{(d)}$ levels off and so does the dimension of the local PC bases.
Table 4.1: Evolutions with the number of subdomains $D$ of the dimension of the condensed problem ($N_r$), (averaged) numbers of local unknowns ($N^{(d)}_r$ and $N^{(d)}_{in}$), local random variables ($N^{(d)}_\kappa$) and local PC basis dimension $P^{(d)}$ for $N_o = 2$ and 6.

<table>
<thead>
<tr>
<th>$D$</th>
<th>$N_r$</th>
<th>$N^{(d)}_r$</th>
<th>$N^{(d)}_{in}$</th>
<th>$N^{(d)}_\kappa$</th>
<th>$P^{(d)}$</th>
<th>$P^{(d)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>2,233</td>
<td>752 ± 93</td>
<td>40,477 ± 91</td>
<td>28.0 ± 0.0</td>
<td>(4.35 ± 0.00) × 10^2</td>
<td>(1.34 ± 0.00) × 10^6</td>
</tr>
<tr>
<td>15</td>
<td>3,337</td>
<td>549 ± 66</td>
<td>21,514 ± 62</td>
<td>17.0 ± 0.0</td>
<td>(1.71 ± 0.00) × 10^2</td>
<td>(1.01 ± 0.00) × 10^6</td>
</tr>
<tr>
<td>30</td>
<td>5,258</td>
<td>404 ± 48</td>
<td>10,693 ± 41</td>
<td>10.7 ± 0.4</td>
<td>(7.48 ± 0.53) × 10^4</td>
<td>(1.12 ± 0.19) × 10^4</td>
</tr>
<tr>
<td>60</td>
<td>7,582</td>
<td>280 ± 26</td>
<td>5,308 ± 23</td>
<td>7.0 ± 0.2</td>
<td>(3.57 ± 0.14) × 10^1</td>
<td>(1.70 ± 0.14) × 10^3</td>
</tr>
<tr>
<td>120</td>
<td>11,205</td>
<td>201 ± 17</td>
<td>2,624 ± 14</td>
<td>5.0 ± 0.0</td>
<td>(2.10 ± 0.00) × 10^4</td>
<td>(4.62 ± 0.00) × 10^2</td>
</tr>
<tr>
<td>240</td>
<td>15,921</td>
<td>141 ± 11</td>
<td>1,292 ± 9</td>
<td>3.2 ± 0.4</td>
<td>(1.11 ± 0.21) × 10^1</td>
<td>(1.11 ± 0.52) × 10^2</td>
</tr>
<tr>
<td>480</td>
<td>22,726</td>
<td>100 ± 8</td>
<td>632 ± 6</td>
<td>3.0 ± 0.0</td>
<td>(1.00 ± 0.00) × 10^1</td>
<td>(8.40 ± 0.00) × 10^1</td>
</tr>
<tr>
<td>960</td>
<td>32,618</td>
<td>72 ± 6</td>
<td>306 ± 4</td>
<td>2.8 ± 0.4</td>
<td>(9.23 ± 1.58) × 10^0</td>
<td>(7.32 ± 2.21) × 10^1</td>
</tr>
<tr>
<td>1920</td>
<td>46,047</td>
<td>51 ± 5</td>
<td>146 ± 3</td>
<td>2.0 ± 0.0</td>
<td>(6.00 ± 0.09) × 10^0</td>
<td>(2.80 ± 0.13) × 10^1</td>
</tr>
</tbody>
</table>

The results in Table 4.1 enable us to quantify the reduction in the local stochastic problem complexity and memory requirements to store $\tilde{A}$ and $\tilde{b}$. This is illustrated in Figure 4.10 which shows the evolution of the local complexity measured by the (averaged) value of $(N_{in}^{(d)})^2 \times P^{(d)}$ is reported for $N_o = 2$ and 6 (left plot). We do not report here the consolidated computational complexity, or the sum of local complexities, as the solves at the preprocessing stage are fully independent over the subdomains and can be carried out in parallel. Instead, we remark that in the case of $N_o = 6$, small values of $D$ yield too many local variables $N^{(d)}_\kappa$ with large local PC bases and prohibitive complexities: increasing $D$ makes the local solves tractable. Similarly, increasing $D$ reduces the memory requirements for storing each local contribution to the condensed operator, as depicted in the right plot of Figure 4.10. The plot shows both the local memory requirement, measured by (averaged) $(N^{(d)}_r)^2 \times P^{(d)}$, and the global memory requirement defined as the sum of the local ones. It is seen that
the local requirements have essentially the same evolution with \( D \) as the complexity. However, the reduction in the global requirement tends to level off as \( D \) becomes large, as it could be expected from the behavior of \( N^{(d)}_{\Gamma} \) and \( N^{(d)}_{\kappa} \) shown in Table 4.1.

![Figure 4.10: Local complexity (left plot) and local and global memory requirements (right plot), as a function of the number of subdomains \( D \) and for two PC degree \( N_o = 2 \) and \( N_o = 6 \). Note that both plots use a log-log scale.](image)

These findings support the use of the largest possible number of subdomains to reduce the computational complexity and memory requirements of the preprocessing stage. However, we might not want to make \( D \) as large as possible because \( N_{\Gamma} \), the size of the condensed problem, increases as \( D \) increases (see the second column of Table 4.1). The cost of solving the reduced problem at the sampling stage, therefore, increases as the number of subdomains increases. Thus, as it is typically the case for methods involving domain decomposition, the best value for \( D \) will depend on the specific problem at hand and the available computational resources.
4.5.2 Implementation details

In this section, we discuss choices for the design and implementation of sampling stage algorithms. As described in Section 4.3.4 and shown in Algorithm 2, the computation of a sample with index $i$, in the loop starting at line 11 (Algorithm 2), involves four main steps. For a given realization, i.e., for one particular index $i$ in these steps can be summarized as follows. First, generate a joint random sample of the local random variables (line 12); second, evaluate the subdomain contributions to the condensed problem (4.40) (line 13), which amounts to evaluating polynomials; third, solve the sample domain decomposition problem (4.40) (line 16); and finally, if desired, recompute the solution inside selected subdomains (line 17). The parallelization of the first and second steps is trivial, as well as the solution of the local problems in the fourth stage when the boundary data are known; see (4.42). For the latter step, our PC approach can even bypass the final local solves if the local PC approximations $\tilde{u}^{(d)}_{in,n}$ of the elementary solutions $u^{(d)}_{in,n}$ can be stored (see Section 4.3.2).

In contrast, different strategies can be envisioned to solve the sampled condensed problem, as further discussed below.

Strategies for solving the condensed problem

Our PC-based sampling approach aims at accelerating direct MC sampling. As discussed earlier in Sections 4.2.1 and 4.2.2, such direct MC methods are usually based on matrix-free iterative solvers where the realizations of the condensed operator and corresponding right-hand side in (4.28) are never explicitly assembled. Instead, using a CG algorithm, the application of the condensed operator to successive conjugate
vectors is implicitly performed in a matrix-free manner by computing residuals from local PDE solutions at the subdomain level. This approach will be referred to as \texttt{Dir-loc-CG} and will serve as a reference. It will be compared with different PC-based strategies, also relying on the CG method to solve the condensed problems, and using the same stopping criterion in order to ensure the fairness of the comparisons.

In our PC-based approach, we investigate two main strategies for solving of the condensed problem (4.40). The first strategy mimics the reference \texttt{Dir-loc-CG} above, in that it never assembles the full condensed problem (4.40). Inside the CG iterations, the subdomain contribution to the residual of the successive conjugate vectors is computed locally, by matrix multiplication with the sample value of $\mathbf{A}^{(d)}$ instead of solving a local PDE problem. This approach will be referred to as \texttt{PC-loc-(P)CG}, where the optional \texttt{P} indicates whether or not a preconditioner is involved in the CG method (see Section 17 below). The second strategy, on the contrary, is based on assembling for each sample the corresponding full condensed operator and right-hand side. The condensed problem (4.40) is still solved using CG, leading to the approach referred to as \texttt{PC-glo-(P)CG} in the following. Note that \texttt{PC-loc-(P)CG} and \texttt{PC-glo-(P)CG} are equivalent, as they solve the same problem, but are expected to have different parallel efficiencies as they will have different communication patterns as discussed in the following.

\textit{Parallelism}

For \texttt{PC-loc-(P)CG}, the realizations are processed sequentially, as in the reference method. For each sample, the solution of the condensed problem is performed in parallel, in a fashion following closely the \texttt{Dir-loc-CG} strategy. Specifically, each
MPI process is in charge of computing the local contributions to the residual of the set of subdomains handled by the process. We will refer to this strategy as parallelism across subdomains because the workload is distributed among the MPI processes according to the spatial domain decomposition. An overview of this parallel implementation is given in the schematic Algorithm 3.

**Algorithm 3**: Schematic algorithm illustrating the parallelism across subdomains for strategy PC-loc-(P)CG.

1. for sample index $i = 1, \ldots, M$ do // [SEQUENTIAL LOOP]
   2. Generate a random sample of $\eta_i = (\eta_i^{(1)} \ldots \eta_i^{(D)})$
   3. for subdomain with index $d = 1, \ldots, D$ do // [PARALLEL LOOP]
      4. Compute $[A]^{(d)}(\eta_i^{(d)})$ and $b^{(d)}(\eta_i^{(d)})$ using (4.41)
   5. end for
   // PARALLEL solve (except preconditioning)
   6. Solve sampled condensed problem (4.40) for $u_{r}(\theta_i)$ using (local) CG iterations
   7. for subdomain with index $d = 1, \ldots, D$ do // [PARALLEL LOOP]
      8. Solve local problem (4.42) for the inner unknowns $\mu_{in}^{(d)}$
   9. end for
10. end for

Regarding PC-glo-(P)CG, a parallelism across samples is more appropriate because the global condensed problem is explicitly assembled. In this strategy, the full condensed operator and right-hand side are assembled in batches of samples, each batch being processed in parallel. For the sake of simplicity, and without loss of generality, we assume that a batch has as many samples as the number of MPI processes, $N_{MPI}$. In a given batch, the first and second steps are performed sequentially for the $N_{MPI}$ samples, parallelizing the tasks across the subdomains for each sample element of the batch. To each of the $N_{MPI}$ samples of the batch corresponds a sample of the
condensed problem, which is globally assembled, through collective communications, on its dedicated MPI process. Once all the samples of the batch have been processed this way, each MPI process owns one particular sample of the condensed problem, and can then proceed with its solution. This amounts to a parallelism across samples, in the sense that the current batch of $N_{\text{MPI}}$ samples has been distributed among the $N_{\text{MPI}}$ MPI processes and are solved independently. An overview of this parallel implementation is given in the schematic Algorithm 4. Optionally, as for the other strategies, the full solutions (step 4) may be retrieved by final local solves using the solutions of the condensed problem, returning to a parallelization across subdomains.

**Preconditioning**

One advantage of having an expression of the condensed operator is the possibility to propose a preconditioner for the CG solver. Classical domain decomposition methods can be preconditioned, in particular using two-levels strategies[]. Here, we rely on an alternative preconditioner based on the condensed operator’s expectation, $\mathbb{E}[\overline{A}]$, defined as:

\[
\overline{A} \doteq \mathbb{E}[\overline{A}] \approx \sum_{d=1}^{D} \mathbb{E}[\overline{A}^{(d)}] = \sum_{d=1}^{D} \overline{A}^{(d)}_0,
\]

(4.52)

where $0 \in \mathbb{N}^{N_{\kappa}(d)}$ is the multi-index of the constant polynomial. Hereafter, $\overline{A}$ will be referred to as the mean condensed operator. It is expected that $\overline{A}^{-1}\overline{A}$ remains close to the identity for all samples so the mean operator can be used as a preconditioner to the full residual iterate appearing in the CG algorithms. In practice, the LU decomposition of $\overline{A}$ is once precomputed prior to the sampling stage, and subsequently used to precondition the CG iterations when solving the condensed
Algorithm 4: Schematic algorithm showing the mixed subdomains and samples parallel processing for the strategy PC-glo-(P)CG.

1. $i \leftarrow 0$
2. while $i < M$ do // [SEQUENTIAL LOOP]
   3. // Start a new batch
   4. for process index $p = 1, \ldots, N_{\text{MPI}}$ do // [SEQUENTIAL LOOP]
   5. $i \leftarrow i + 1$
   6. Generate a random sample of $\eta_i = (\eta_{i}^{(1)} \ldots \eta_{i}^{(D)})$
   7. for subdomain with index $d = 1, \ldots, D$ do // [PARALLEL LOOP]
      // Each process handles $D_p \approx D/N_{\text{MPI}}$ subdomains
      8. Compute $\widetilde{A}^{(d)}(\eta_{i}^{(d)})$ and $\widetilde{b}^{(d)}(\eta_{i}^{(d)})$ using (4.41)
   9. end for
10. Assemble and store the global $\widetilde{A}(\theta_i)$ and $\widetilde{b}(\theta_i)$ on process $p$
11. end for
12. // Each process now owns one global realization of the condensed problem
13. for process index $p = 1, \ldots, N_{\text{MPI}}$ do // [PARALLEL LOOP]
14. // Each process handles 1 realization
15. Solve sampled condensed problem (4.40) using (global) CG
16. end for
17. // Optional
18. for subdomain with index $d = 1, \ldots, D$ do // [PARALLEL LOOP]
19. Solve local problem (4.42) for the inner unknowns $\mathbf{u}_{in}^{(d)}$
20. end for
21. end while

problem for different samples. Note that for the (PC-loc-PCG) strategy, where the full operator is not assembled, further gain may be obtained by parallelizing the application of the preconditioner, although this direction is not further investigated here.
4.5.3 Computational behavior

The analysis of the computational behavior of the method is broken down into two parts. First, we investigate the scalability of the preprocessing stage. Second, we discuss the computational behavior of the sampling stage, for the different solving strategies described above, and compare it with the behavior of the classical matrix-free MC sampling approach. Unless specified otherwise, the computations of this section use $\sigma^2 = 0.2$, $L = 0.1$, $D = 512$ and $N_o = 2$.

Preprocessing stage

We characterize the scalability with the number $N_{MPI}$ of MPI processes of the preprocessing stage by the parallel efficiency $E$, expressed as a percentage:

$$E(N_{MPI}) = 100 \frac{T_{\text{ref}}}{N_{MPI} T(N_{MPI})},$$

where $T_{\text{ref}}$ and $T(N_{MPI})$ are the measured CPU times of the tasks execution for a reference case and the execution using $N_{MPI}$ processes. For the reference, we take the smallest number of processes tested, $N_{MPI} = 16$, and use $T_{\text{ref}} = 16T(16)$, assuming a perfect parallel efficiency from 1 to 16 processes.

Figure 4.11 shows the parallel efficiency of the preprocessing stage for 3 meshes of increasing size. In Fig. 4.11a, we observe that the parallel efficiency slightly decreases with $N_{MPI}$, but remains above 80% on 512 processes for all three meshes. It shows that the preprocessing stage is scaling decently, even using a naive, static, a priori load balancing strategy. The moderate loss of efficiency can be explained by processes waiting for each other to get to a certain point, caused by load imbalance. Although the preprocessing stage involves no communication (either point-to-point...
or collective) between processes, the preprocessing stage ends when all processes have terminated leading to a worst case idle-time scenario. In addition, each process \( p \) handles a certain number of subdomains whose indices \( d \) are collected in \( \mathcal{I}_p \). Note that in the present tests we used numbers of processes such that \( D = 512 \) is always a multiple of \( N\text{_{MPI}} \), and that the processes handle exactly the same number of subdomains, namely \( D/N\text{_{MPI}} \). However, the subdomains support FE meshes having different sizes, as well as possibly different numbers of local random variables \( N_{\kappa}^{(d)} \), see Table 4.1. As a consequence, the number and size of the local Galerkin problems that a process \( p \) has to solve may change slightly from one process to another. As \( N\text{_{MPI}} \) increases, fewer subdomains are handled by a process, down to the case of 512 processes each handling one single subdomain, tending to increase the load imbalance between processes with a degradation of the parallel efficiency. It is clear than more advanced partitioning and load balancing techniques can be employed to improve the scaling properties of this stage. As a side note, we point out that using carefully designed regular structured meshes should theoretically lead to quasi-ideal scaling. Finally, we observe in Fig. 4.11b that the parallel efficiency is not affected by the PC degree \( N_o \).

**Sampling stage**

We now investigate the computational behavior of the sampling stage. In particular, we compare the different strategies discussed in Section 4.5.2.

Figure 4.12 reports the CPU times needed to generate a single sample of the condensed problem solution, as a function of \( N\text{_{MPI}} \). These measurements only include the first three steps of the sampling procedure, leaving aside the final calculation of
the full solution over the subdomains. Moreover, for PC-glo-PCG and a parallelization over samples, we consider a batch of size $M = D = 512$ and report the average computational time (divided by $M$) for a fair comparison. In addition, the CPU times are scaled so that the reported time using PC-glo-PCG on 16 processes equals 1.

In Fig. 4.12a, corresponding to a spatial mesh with $N_e = 163,272$ quadratic finite elements, we observe that the strategy PC-loc-CG outperforms the reference approach Dir-loc-CG with an acceleration factor of about 3.5 on 16 processes. As the number of processes increases, the two approaches lose parallel efficiency and seem to converge to the same CPU time. This trend can be explained by the collective data communication which needs to be performed at each CG iteration. This communication time does not decrease as $N_{\text{MPI}}$ increases, while on the contrary, the workload of the processes for solving local problems (in Dir-loc-CG) or performing local matrix-vector products (in PC-loc-CG) decreases, due to a good parallel scaling.
of these computations. Eventually, the communication cost becomes comparable to the computational cost of the rest of the CG algorithm (e.g. dot products), which scale poorly, and consequently the overall sampling cost converges to this flat cost. Concerning PC-\texttt{loc-PCG}, the effect of the mean preconditioner can be appreciated comparing its computational time with the PC-\texttt{loc-CG} strategy: for 16 processes, the CPU time is reduced by another factor of about 3.5. This reduction is due to the improved convergence of the iterative solver, allowing to save many CG iterations. However, the overall cost of PC-\texttt{loc-PCG} is quickly dominated by the application of the preconditioner, which is not performed in parallel in the present implementation (see Section 17), with a very poor parallel scaling of PC-\texttt{loc-PCG} as a result. Eventually, the savings of the preconditioner are lost and the overall CPU time converges to that of the non-preconditioned version.
Finally, the strategy $\text{PC-glo-PCG}$, based on the full assembly of the global condensed system (4.40) and using the mean operator as a preconditioner, has a parallel efficiency behavior similar to that of $\text{PC-loc-CG}$, but with a computational cost up to 16 times less. $\text{PC-glo-PCG}$ outperforms the reference strategy $\text{Dir-loc-CG}$ by a factor of about 48 for $N_{\text{MPI}} = 16$ and remains asymptotically 20 times faster despite its efficiency drop. Again, the drop in efficiency for $\text{PC-glo-PCG}$ is caused by the collective communication needed to assemble the global condensed system from its local contributions. For the present example, this communication step involves the exchange of about 4.5 million double precision values between all the $N_{\text{MPI}}$ processes. In addition to being much more efficient than the other strategies, this last approach lends itself to a task-based parallel framework, where data locality would be preserved and collective communication would be avoided. Although outside the scope of this Chapter, it is important to point out that adopting such a parallel processing paradigm could potentially improve significantly the parallel scaling of this approach. In any case, having a different treatment of the PC evaluation (parallelized across subdomains) and of the condensed system solve (parallelized across samples) clearly allows for more flexibility.

To conclude this analysis, let us mention that the reported trends in the parallel efficiency for the different methods does not significantly depend on the FE discretization of the problem. This can be seen comparing the similar evolution of the CPU times in Fig. 4.12a and Fig. 4.12b, the latter corresponding to an FE mesh with twice as many elements as before $N_e = 327,334$ elements (leading to computational times roughly twice as long). We also note that for this refined mesh and 16 processes, $\text{PC-glo-PCG}$ is roughly 64 times faster than for the reference $\text{Dir-loc-CG}$.  

137
4.6 Conclusions

We have presented an acceleration strategy for a Monte Carlo sampling-based Stochastic Elliptic PDE solver. The method employs a domain decomposition technique to partition the computational domain into smaller non-overlapping subdomains. In a first stage, an approximation of the local boundary-to-residual map is constructed, independently over each subdomain. This approximation uses a PC expansion to represent the dependencies of the map on the stochastic coefficient of the elliptic equation. The cost of computing this local PC approximation is reduced owing to the possibly low dimensional representation of the stochastic coefficient over the considered subdomain, compared to its global representation. These local PC expansions can be combined together to obtain an approximation of the (global) condensed problem relating the stochastic solution at the interface of the subdomains. The local PC-based representations of the condensed problem can be sampled with a low computational cost, amounting to simple polynomial evaluations. This feature is exploited in a second stage to generate, at a reduced computational cost, realizations of the stochastic solution via MC sampling.

We validated the accuracy of the proposed approach on a numerical example that also served to analyze convergence with the polynomial degree of the PC expansion. An important finding is that, as desired, the domain decomposition allows for significant computational time saving while having a negligible effect on the approximation error which is essentially driven by the polynomial degree of the expansion. We also analyzed the performance of different parallel implementations of the approach. Specifically, we showed that the cost of the preprocessing stage can be conveniently
distributed our multiple processors with a close to ideal parallel efficiency (higher than 80% in our experiments). Given that we used a naive, static, a priori load balancing strategy, the scaling properties of the first stage could even be improved by employing more advanced partitioning and load balancing techniques. Concerning the sampling stage, all the parallel strategies involving the PC approximation of the condensed problem perform better than the reference approach. A noticeable degradation in the parallel efficiency is however reported when the number of MPI processes is increased. Despite this efficiency drop, the best sampling strategy is found to remain at least 20 times faster than the reference for the largest number of processes tested (512) when it is up to 60 times faster when only 16 MPI processes are used. The collective communications involved in the assembly of the condensed problem, from the local contributions, are responsible for the efficiency drop. A possible way to mitigate this issue would be to rely on a task-based parallel framework, where data locality would be preserved and collective communication would be avoided.

In addition to improving parallel efficiency, future works should focus on improved partitioning strategies and the determination of the optimal number of subdomains yielding the lowest computational cost. The later aspect is non-obvious, but involves several trade-offs between different steps of the method, and clearly depends on the computational architecture and the resources available. Another potential route to develop the proposed approach is exploring the potential interest of considering a hierarchy of FE meshes. This hierarchy could be used to accelerate the resolution of the condensed problem (as in two-level domain decomposition methods), on the one hand, and to optimize the computational complexity of the MC method (as in
multilevel MC methods), on the other hand.
Conclusion

In this dissertation we presented two acceleration techniques for Uncertainty Quantification (UQ). The first approach was a model reduction strategy that was used in the context of inferring the presence of an inclusion embedded in a soft matrix, mimicking tumors in soft tissues. The presence of the inclusion was determined based on changes in the Youngs modulus of the material. Obtaining the value of the Youngs modulus requires the solution of an inverse problem. The Youngs modulus was obtained from noisy displacement data through the solution of an inverse problem. For any given set of displacement data, solving the inverse problem can be very computationally demanding. Thus, to address this issue we built a surrogate model that reduced the computational cost of solving the inverse problem. However, building a surrogate model required knowledge of the geometry of the inclusion, which most likely is not available. To handle the lack of complete information about the geometry of the physical model, a family of PC surrogates was built, each with
a different assumed geometry. A model selection approach based on the evidence provided by the data was used to discriminate among the different models. The proposed method was shown to be advantageous since it divides the computational cost into more costly preprocessing stage where the surrogate models are computed for several geometries, and a lower cost model selection and inference stage where the decision regarding the presence of the inclusion is made using observations.

The other acceleration technique that was presented relied on Domain Decomposition methods and exploited the fact that a reduced number of random variables can be used to characterize the uncertainty at the subdomain level. The work related to this technique was divided into two parts. In the first part we focused on the computation of the KL expansion by means of a divide-and-conquer strategy where the computational domain was partitioned into smaller non-overlapping subdomains, over which independent local KL decompositions were performed to generate local bases which were subsequently used to discretize the global modes over the entire domain. The relationship between the local KL expansion and the global KL expansion was fully analyzed, and it was observed that local expansion can be used to effectively characterize the uncertainty at the global level. The second part of the work exploited the availability of local coordinates to reduce the cost of solving a Stochastic Elliptic Equation (SEE) using Monte Carlo sampling methods. Using the local random variables, it is possible to construct at a low cost a PC expansion of the linear system that needs to be solved in order to compute a sample of the solution. In summary, the approach consists of two main stages: 1) a preprocessing stage in which PC expansions of a condensed problem are computed and 2) a Monte Carlo sampling stage where samples of the solution are computed in order
to solve the SEE. Both the computational savings offered by the approach and its parallel efficiency were demonstrated quite satisfactorily with a numerical example. Nonetheless, further developments will be required to improve the parallel efficiency of the sampling stage.
Appendix A

MCMC Posterior Sampling

Given observations $Y_i$, the joint posterior distribution of $\xi$ and $\epsilon$ can be sampled by Markov Chain Monte Carlo (MCMC) algorithms [92, 11] in the corresponding $(\xi, \epsilon)$ domain. In this work, we use an adaptive Metropolis-Hastings (M-H) algorithm with a multivariate centered Gaussian proposal distribution. The approach consists in using a proposal distribution with a $(\xi, \epsilon)$-dependent covariance matrix based on the decomposition of the Hessian of the log-posterior distribution.

From the expressions of the priors and likelihood function, the log-posterior distribution can be expressed as

$$LP(\xi, \epsilon|\{Y_i\}) = -\sum_{i=1}^{m_o} \frac{(\hat{u}(x_i, \xi) - Y_i)^2}{2\epsilon^2} - (m_o + 1) \log(\epsilon) + \log(p_\xi(\xi)) + C. \quad (A.1)$$

where $C$ groups all the constant terms in the expression. Under mild conditions on $p_\xi$ (recall that the component $\xi_i$ of $\xi$ are independent), the log-posterior can be
differentiated twice, and we denote $[H_{LP}] \in \mathbb{R}^{(N+1)\times(N+1)}$ the Hessian matrix of the log-posterior distribution. Observing that $[H_{LP}]$ is symmetric, it can be decomposed as $[H_{LP}] = [V][\Lambda][V]^T$, where $[\Lambda]$ is a diagonal matrix whose elements are the real eigenvalues of $[H_{LP}]$, and $[V]$ is the matrix whose columns are the eigenvectors of $[H_{LP}]$ such that $[V][V]^T = [I]$. We then define the covariance matrix $\Sigma^2$ of the centered Gaussian proposal distribution according to:

$$\Sigma^2 = \sigma_\Sigma^2 \times [V][\Lambda][V]^T,$$

where the non-negative diagonal matrix $\mathcal{D}$ is any convenient transformation of $[\Lambda]$ and $\sigma_\Sigma^2 > 0$ is a scaling factor. A suitable choice for $\mathcal{D}$ is

$$\mathcal{D}_{ii} ([\Lambda]) = \begin{cases} \frac{\min_k |\lambda_{kk}|}{|\lambda_{ii}|} & \Lambda_{ii} \neq 0 \\ 1 & \Lambda_{ii} = 0 \end{cases} \quad (A.3)$$

This transformation allows us to take a larger step along the direction where the log-posterior is flat (the posterior varies slowly) and smaller steps along the directions of high curvature.

Note that in general the Hessian matrix $[H_{LP}]$ will be a function of the current state $(\xi, \epsilon)$ of the chain. Thus, as the chain evolves from one state to another the Hessian matrix will change accordingly. However, in many cases the changes might not be significant, and it is possible to further reduce the computational cost by reducing the frequency at which the covariance matrix of the proposal distribution is updated, without significant performance degradation. Furthermore, in some instances we only need to compute the Hessian once to determine an adequate proposal distribution.
An advantage of using PC representations is that it is straightforward to compute derivatives [3, 86]. In particular one can derive expressions for the successive derivatives of \( \hat{u}(x, \xi) \) with respect to components \( \xi_i \) as needed for the computation of the Hessian. For instance, considering the partial derivative with respect to \( \xi_i \) of \( \hat{u}(x, \xi) \), we can write it as a PC expansion with the same basis used for \( \hat{u}(x, \xi) \):

\[
\frac{\partial}{\partial \xi_i} \hat{u}(x, \xi) = \frac{\partial}{\partial \xi_i} \sum_{\alpha=0}^{P} \hat{u}_\alpha(x) \Psi_{\alpha}(\xi) = \sum_{\alpha=0}^{P} (\partial_i \hat{u})_{\alpha}(x) \Psi_{\alpha}(\xi), \tag{A.4}
\]

where the PC coefficients \((\partial_i \hat{u})_{\alpha}\) of the derivative are related to the PC coefficients of \( \hat{u} \) through

\[
\begin{pmatrix}
(\partial_i \hat{u})_0 \\
(\partial_i \hat{u})_1 \\
\vdots \\
(\partial_i \hat{u})_P
\end{pmatrix} = [L_i] \begin{pmatrix}
\hat{u}_0 \\
\hat{u}_1 \\
\vdots \\
\hat{u}_P
\end{pmatrix} = \begin{pmatrix}
\langle \frac{\partial}{\partial \xi_i} \Psi_0, \Psi_0 \rangle \\
\langle \frac{\partial}{\partial \xi_i} \Psi_0, \Psi_1 \rangle \\
\vdots \\
\langle \frac{\partial}{\partial \xi_i} \Psi_0, \Psi_P \rangle \\
\langle \frac{\partial}{\partial \xi_i} \Psi_1, \Psi_0 \rangle \\
\langle \frac{\partial}{\partial \xi_i} \Psi_1, \Psi_1 \rangle \\
\vdots \\
\langle \frac{\partial}{\partial \xi_i} \Psi_1, \Psi_P \rangle \\
\vdots \\
\langle \frac{\partial}{\partial \xi_i} \Psi_P, \Psi_0 \rangle \\
\langle \frac{\partial}{\partial \xi_i} \Psi_P, \Psi_1 \rangle \\
\vdots \\
\langle \frac{\partial}{\partial \xi_i} \Psi_P, \Psi_P \rangle
\end{pmatrix} \begin{pmatrix}
\hat{u}_0 \\
\hat{u}_1 \\
\vdots \\
\hat{u}_P
\end{pmatrix}. \tag{A.5}
\]

The derivative operators \([L_i], i = 1, \cdots, N\) can be precomputed exactly and stored, so that when computing the Hessian in the MCMC algorithm, finding \( \frac{\partial}{\partial \xi_i} \hat{u} \) reduces to a single matrix multiplication of \([L_i]\) with the coefficients vector \( \hat{u} \). Higher-order derivatives amount to additional matrix multiplications. Consequently, the computational complexity of the adaptive proposal covariance essentially reduces to the spectral decomposition of the Hessian, which remains low provided that \( N \) is not too large. Finally, the scale factor \( \sigma_{\xi}^2 \) is classically tuned as to obtain a prescribed rejection rate for the chain.
Appendix B

Derivation of the Error Expression

We have an initial process $U(x, \theta) \in L_2(\Omega, \Theta)$, its projection $U_\mathcal{B}(x, \theta) \in \mathcal{V}_\mathcal{B} \times L_2(\Theta)$ and finally $\hat{U}(x, \theta) \in \mathcal{V}_\mathcal{B} \times L_2(\Theta)$ the truncated KL expansion of $U_\mathcal{B}$ using $\hat{N} \leq n_t$ modes. Our concern is to derive an expression for the error $U - \hat{U}$ in the $L_2(\Omega, \Theta)$-norm. By definition this is the square root of

\[
\mathbb{E} \left[ \left\| U - \hat{U} \right\|^2_\Omega \right] = \mathbb{E} \left[ \left\langle U - \hat{U}, U - \hat{U} \right\rangle_\Omega \right].
\]

We have

\[
\left\langle U - \hat{U}, U - \hat{U} \right\rangle_\Omega = \left\langle (U - U_\mathcal{B}) + (U_\mathcal{B} - \hat{U}), (U - U_\mathcal{B}) + (U_\mathcal{B} - \hat{U}) \right\rangle_\Omega
\]

\[
= \left\langle (U - U_\mathcal{B}), (U - U_\mathcal{B}) + (U_\mathcal{B} - \hat{U}) \right\rangle_\Omega + \left\langle (U_\mathcal{B} - \hat{U}), (U - U_\mathcal{B}) + (U_\mathcal{B} - \hat{U}) \right\rangle_\Omega
\]

\[
= \left\| U - U_\mathcal{B} \right\|^2_\Omega + \left\| U_\mathcal{B} - \hat{U} \right\|^2_\Omega + 2 \left\langle (U - U_\mathcal{B}), (U_\mathcal{B} - \hat{U}) \right\rangle_\Omega.
\]

(B.1)
However the last term is zero as \((U_{\mathcal{B}} - \hat{U}) \in \mathcal{V}_{\mathcal{B}} \times L_2(\Theta)\) while \((U - U_{\mathcal{B}})\) is orthogonal to \(\mathcal{V}_{\mathcal{B}} \times L_2(\Theta)\). Then it follows that

\[
\mathbb{E}\left[\|U - \hat{U}\|_\Omega^2\right] = \mathbb{E}\left[\|U - U_{\mathcal{B}}\|_\Omega^2 + \|U_{\mathcal{B}} - \hat{U}\|_\Omega^2\right] = \mathbb{E}\left[\|U - U_{\mathcal{B}}\|_\Omega^2\right] + \mathbb{E}\left[\|U_{\mathcal{B}} - \hat{U}\|_\Omega^2\right].
\]
Bibliography


Biography

Andres Anibal Contreras was born in Santo Domingo, Dominican Republic in 1986. He earned Bachelor of Science degrees in Electrical Engineering and Mathematics in May 2009 and a Master of Science degree in May 2010, all from Utah State University. In August 2012 he started his Ph.D. studies at Duke University, in the areas of Computational Mechanics and Uncertainty Quantification, in the department of Civil and Environmental Engineering.