Università degli Studi di Trieste

Dipartimento di Matematica e Geoscienze

Corso di Laurea Magistrale in Matematica

WEIGHTED REDUCED ORDER METHODS FOR PARAMETRIZED PDEs IN UNCERTAINTY QUANTIFICATION PROBLEMS

Advisor:
Prof. Gianluigi Rozza
Co-advisor:
Dr. Francesco Ballarin

Master Thesis of:
Luca Venturi

Academic year 2015-2016

Contents

Introduction

When modeling a physical systems, uncertainties inevitably arise from various sources, e.g., computational geometries, physical parameters, external forces, and initial or boundary conditions, and may significantly impact the computational results. Physical systems are usually modeled by parametrized partial differential equations. When uncertainties are incorporated into these equations, we are facing stochastic problems or uncertainty quantification. Various computational methods have been developed to solve such equations, including Monte-Carlo, stochastic collocation and reduced basis.
The most commonly used Monte-Carlo methods [21,45], typically converge slowly and require an high computational effort. The stochastic collocation methods [35, 48], instead, employs multivariate polynomial interpolations for the integral in the variational formulation of the stochastic system with respect to probability space. Due to the heavy computation of a deterministic system at each collocation point in high-dimensional spaces, sparse grids with suitable quadrature rules have been analyzed and applied to reduce the computational load.
Reduced basis (RB) methods [26] have been applied to stochastic problems in order to reduce the computational effort of Monte-Carlo methods. The RB method has been proposed primarly to solve parametric systems, in works [37,42], and has been lately applied to stochastic problems, in works [5, 6]. In the latter context, it regards the random variable as parameters and selects the most representative points in the parameter space by greedy sampling, based on a posteriori error estimation. The underlying idea for the deterministic and stochastic reduced basis method is to separate the whole procedure into an offline stage and an online stage. During the former, the large computational ingredients are computed and stored once and for all, including sampling parameters, assembling matrices and vectors, solving and collecting snapshots of solutions, etc. In the online stage, only the parameter-related elements are left to be computed and a small Galerkin projection problem needs to be solved.
Nevertheless, in [5, 6] the RB method was used only for stochastic problems with uniformly distributed random inputs. In order to deal with more general stochastic problems with other distributed random inputs, the works [8–11], have proposed and analyzed a new version of RB method, named 'weighted' reduced basis method. The basic idea is to suitably assign a larger weight to samples that are more important or have a higher probability of occuring than the others, according to the probability distribution function.
Reduced basis method are part of a larger family, known as reduced order meth-

ods [16].
Another well known reduced order technique is the so called Proper Orthogonal Decomposition (POD). The basic idea behind the POD method is to minimize the error in a L^2 norm over the parameter space. To our knowledge, a weighted POD method has never been used for the solution of stochastic problems.
In this thesis it has been proposed and numerically analyzed a new version of POD method for stochastic problems, which has been named 'weighted' POD. The basic idea behind the weighted POD method is to minimize the distance between the solution in a L^2_P norm over the parameter space, where P indicates the probability distribution of the parameter.
The thesis is organized as follows.
In the first chapter we present reduced order methods for the solution of parametrized PDEs, in the deterministic case. We introduce the general framework and we analyze, respectively, how POD and RB algorithms work.
In the second chapter we present weighted reduced order methods for PDEs with stochastic parameters. The general framework of stochastic partial differential equations (PDEs) and the hypotesis necessary to use weighted reduced order methods are set. Thus, we apply them for the resolutions of two linear stochastic elliptic equations, with different random dimensions. We then discuss the results obtained numerically. For all the implementation, it has been used the `RBniCS` [2] library, which we properly modified to allow the use of weighted reduced order algorithms.
In the third chapter, we discuss a possible application of Smolyak quadrature [23, 36] to weighted POD algorithms. Stochastic collocation methods and in particular Smolyak algorithms are presented. We then explained how they could be useful in weighted POD method. We extended the `RBniCS` library with functions which allow to compute Smolyak quadrature approximations and to use them for the weighted POD method. Numerical results and remarks are also reported.
Finally, in the fourth chapter, we analyze the numerical solutions of a 2d linear elasticity equation, with some stochastic parameters, using weighted reduced order methods.
Some concluding remarks are reported in the final chapter.

Introduzione

Nel modellizzare un sistema fisico, diverse fonti di incertezza si presentano inevitabilmente sotto molti aspetti, quali ad esempio, parametri fisici, forze esterne, o condizioni iniziali o al contorno, e possono significativamente influenzare i risultati finali. I sistemi fisici sono spesso descritti tramite equazioni alle derivate parziali dipendenti da parametri. Quando fonti di incertezza sono incorporate nei parametri, siamo di fronte a problemi stocastici o di quantificazione dell'incertezza.
Diversi metodi computazionali sono stati sviluppati per risolvere tali equazioni, quali i metodi Monte-Carlo e di collocazione stocastica. Tra questi, quelli più comunemente usati sono i metodi Monte-Carlo [21, 45]. Tuttavia tali metodi convergono lentamente e richiedono un grande sforzo computazionale. I metodi di collocazione stocastica [35, 48], d'altro lato, utilizzano un interpolazione polinomiale multivariata nello spazio di probabilità per gli integrali nella formulazione variazionale del problema stocastico. Dato l'alto costo computazionale, per spazi di dimensione elevata, di risolvere sistemi deterministici in ogni punto di collocazione, metodi sparse grid sono stati utilizzati per ridurre il carico computazionale.
I metodi alle basi ridotte [26] sono stati applicati a problemi stocastici per ridurre il costo computazionale dei metodi Monte-Carlo. I metodi alle basi ridotte sono stati proposti prima di tutto per risolvere problemi parametrici, nei lavori [37, 42], e successivamente applicati a problemi stocastici, nei lavori [5, 6]. In quest'ultimo contesto, le variabili aleatorie sono viste come parametri e vengono selezionati i punti più rappresentativi nello spazio dei parametri con un sampling di tipo greedy, basato su una stima a posteriori dell'errore. L'idea alla base dei metodi alle basi ridotte è di separare l'intera procedura in una fase offline e una online. Durante la prima, gli ingredienti di alto costo computazionale (quali, ad esempio, i parametri di sampling, le matrici e i vettori di assemblaggio, gli snapshot delle soluzioni, etc.) sono calcolati e salvati una volta per tutte. Nella fase online, rimangono da calcolare solamente gli elementi legati ai parametri e il problema è risolto cercando una proiezione di Galerkin piccola.
Nei lavori [5,6] il metodo alle basi ridotte era usato solamente per problemi stocastici con input casuali di distribuzione uniforme. Per trattare problemi stocastici più generali, ovvero con diverse distribuzioni degli input, i lavori [8–11] hanno proposto e analizzato una nuova versione di metodi RB, chiamati metodi alle basi ridotte 'pesati'. L'idea alla base è di assegnare pesi maggiori a samples che sono più importanti o hanno una maggior probabilità di apparire di altri, secondo la distribuzione di probabilità.

I metodi alle basi ridotte fanno parte di una famiglia più grande, conosciuta come metodi di ordine ridotto [16].
Un altra tecnica di ordine ridotto è la cosiddetta Proper Orthogonal Decomposition (POD). L'idea alla base del metodo POD è di minimizzare l'errore in norma L^2 sullo spazio dei parametri.
In questa tesi viene proposta una versione 'pesata' del metodo POD. Per quanto ne sappiamo, un tale metodo non è mai stato utilizzato finora per la risoluzione di problemi stocastici. L'idea alla base del metodo POD pesato è di minimizzare l'errore in norma L^2_P sullo spazio dei parametri, dove P indica la distribuzione di probabilità sui parametri.
La tesi è organizzata come segue.
Nel primo capitolo vengono presentati i metodi di ordine ridotto per la risoluzione di PDEs parametrizzate, nel caso deterministico. Viene introdotto il contesto generale e sono analizzati, rispettivamente, gli algoritmi dei metodi RB e POD.
Nel secondo capitolo sono presentati i metodi di ordine ridotto pesati per la soluzione di PDEs dipendenti da parametri stocastici. Sono riportati il contesto generale delle equazioni alle derivate parziali stocastiche (SPDEs) e le ipotesi necessarie per applicarvi i metodi di ordine ridotto. Quindi, tali metodi sono applicati alla risoluzione di due equazioni stocastiche lineari ellittiche, con diverse dimensioni del parametro stocastico. Infine sono discussi i risultati ottenuti numericamente. Per tutte le implementazioni, è stata usata la libreria `RBniCS` [2], modificata in modo da permettere l'uso di metodi pesati.
Nel terzo capitolo, viene discussa una possibile applicazione della quadratura di Smolyak [23, 36] ai metodi POD pesati. Innanzitutto sono presentati i metodi di collocazione stocastica e in particolare gli algoritmi di Smolyak. Viene quindi spiegato il loro possibile utilizzo nei metodi POD pesati. La libreria `RBniCS` è stata estesa con funzioni per il calcolo le formule di quadratura di Smolyak e per il loro utilizzo nei metodi POD pesati. Vengono quindi riportati i risultati numerici ottenuti e le relative osservazioni.
Infine, nel quarto capitolo, vengono analizzate le soluzioni numeriche di un problema di elasticità lineare bidimensionale, dipendente da parametri stocastici, ottenute utilizzando metodi di ordine ridotto pesati.
Nel capitolo finale sono riportate delle osservazioni conclusive.

Prerequisites and notation

We go through some of the necessary terminology used in the course of this thesis. Let V be an arbitrary normed space. The set of continuous linear mappings from V to $\mathbb{R}$ is called the topological dual space of V and it is denoted by V'.
Throughout this thesis we employ so-called multi-index notation. If $\alpha \in \mathbb{N}^d$, then we refer to its $j^{\mathtt{th}}$ coordinate universally as α_j. If $\beta \in \mathbb{N}^d$ we write $\alpha \geq \beta$ if $\alpha_j \geq \beta_j$ for all $j = 1, \ldots, d$. We define additionally the shorthand $\mathbf{1} = (1, \ldots, 1)$ and $\mathbf{0} = (0, \ldots, 0)$. We introduce the following convention for the mixed derivative operator:

$$\frac{\partial^\alpha}{\partial x^\alpha} = \frac{\partial^{|\alpha|_1}}{\partial x_1^{\alpha_1} \cdots x_d^{\alpha_d}}, \qquad x = (x_1, \ldots, x_d) \in \mathbb{R}^d.$$

Given a open subset Ω in $\mathbb{R}^d$, we denote with $\mathcal{D}(\Omega)$ the space of all infinitely differentiable functions $v : \Omega \to \mathbb{R}$ with compact support and with $\mathcal{D}'(\Omega)$ the space of distributions on Ω.
For each integer $m \geq 0$, the Sobolev space $W^{m,p}(\Omega)$ consists of those functions $v \in L^p(\Omega)$, for which all partial derivatives $\partial^\alpha v$ (in the distribution sense), with $|\alpha|_1 \leq m$, belong to the space $L^p(\Omega)$. The space $W^{m,p}(\Omega)$ is a Banach space equipped with the norm

$$\|v\|_{m,p,\Omega} = \left(\sum_{|\alpha|_1 \leq m} |v|^p_{m,p,\Omega} \right)^{1/p}, \quad \text{where} \quad |v|_{m,p,\Omega} = \left(\sum_{|\alpha|_1 = m} \int_\Omega |\partial^\alpha v|^p \, dx \right)^{1/p}$$

are seminorms. In particular $H^m(\Omega) \doteq W^{m,2}(\Omega)$ is an Hilbert space with scalar product

$$(u, v)_{m,\Omega} = \sum_{|\alpha|_1 \leq m} \int_\Omega \partial^\alpha u \, \partial^\alpha v \, dx$$

We denote with $\|\cdot\|_{m,\Omega}$ the norm of $H^m(\Omega)$ and with $|\cdot|_{m,\Omega} = |\cdot|_{m,2,\Omega}$ the seminorms. We define the Sobolev space

$$H^m_0(\Omega) = \overline{\mathcal{D}(\Omega)},$$

the closure being understood in the sense of the norm $\|\cdot\|_{m,\Omega}$. When the set Ω is bounded, there exists a constant $C(\Omega)$ such that

$$|v|_{0,\Omega} \leq C(\Omega)|v|_{1,\Omega}, \qquad \forall\, v \in H^1_0(\Omega) \tag{0.1}$$

In the following we will consider open sets Ω with Lipschitz continouous boundary $D = \partial\Omega$, i.e., such that there exist constants $\alpha, \beta > 0$, and a finite number of local coordinate systems and local maps a_r, $1 \le r \le R$, which are Lipschitz continuous on their respective domains of definitions $\{\hat{x}^r \in \mathbb{R}^{d-1} \ : \ \|\hat{x}^r\| \le \alpha\}$, such that:

$$
\begin{aligned}
&D = \bigcup_{r=1}^{R} \{(x_1^r, \hat{x}^r) \ : \ x_1^r = a_r(\hat{x}^r), \ \|\hat{x}^r\|_\infty < \alpha\} \\
&\{(x_1^r, \hat{x}^r) \ : \ a_r(\hat{x}^r) < x_1^r < a_r(\hat{x}^r) + \beta, \ \|\hat{x}^r\|_\infty < \alpha\} \subseteq \Omega, \quad 1 \le r \le R, \\
&\{(x_1^r, \hat{x}^r) \ : \ a_r(\hat{x}^r) - \beta < x_1^r < a_r(\hat{x}^r), \ \|\hat{x}^r\|_\infty < \alpha\} \subseteq (\overline{\Omega})^c, \quad 1 \le r \le R,
\end{aligned}
$$

where $\hat{x}^r = (x_2^r, \dots, x_d^r)$. If the maps a_r are C^k, we say that D is of class C^k. If D is Lipschitz continuous, a superficial measure, which we shall denote $d\gamma$, can be defined along the boundary, so that it makes sense to consider the spaces $L^2(D)$. Then it can be proved that there exists a constant $C(\Omega)$ such that

$$
\|v\|_{0,D} \le C(\Omega)\|v\|_{1,\Omega}, \qquad \forall v \in \mathcal{C}^\infty(\overline{\Omega}).
$$

Since in this case $\overline{\mathcal{C}^\infty(\overline{\Omega})} = H^1(\Omega)$, there exists a continuous linear mapping $\mathrm{tr} : v \in H^1(\Omega) \mapsto \mathrm{tr}\, v \in L^2(D)$, which is called the trace operator. However when no confusion arise, we shall simply write $\mathrm{tr}\, v = v$. The following characterization holds:

$$
H_0^1(\Omega) = \{v \in H^1(\Omega) \ : \ v = 0 \quad \text{on } D\}.
$$

Moreover, since D is Lipschitz, the unit outer normal $\mathbf{n} = (n_1, \dots, n_d)$ exists $d\gamma$ almost everywhere (a.e.). Therefore it can be shown that given two functions $u, v \in H^1(\Omega)$, the following fundamental Green's formula holds for any $i = 1, \dots, d$:

$$
\int_\Omega u\, \partial_i v \, dx = -\int_\Omega \partial_i u\, v \, dx + \int_D u\, v\, n_i \, d\gamma. \tag{0.2}
$$

In the following we shall also consider vectorial problem and so we will consider the spaces

$$
\begin{aligned}
\mathbf{L}^p(\Omega) &\doteq (L^p(\Omega))^d, \quad \mathbf{W}^{m,p}(\Omega) \doteq (W^{m,p}(\Omega))^d, \\
\mathbf{H}^m(\Omega) &\doteq (H^m(\Omega))^d, \quad \mathbf{H}_0^m(\Omega) \doteq (H_0^m(\Omega))^d.
\end{aligned}
$$

With abuse of notation we will denote the norm and the semi-norms on this spaces as, respectively,

$$
\|\mathbf{v}\|_{m,p,\Omega} \doteq \left(\sum_{i=1}^{d} \|v_i\|_{m,p,\Omega}^2\right)^{1/2} \qquad \text{and} \qquad |\mathbf{v}|_{m,p,\Omega} \doteq \left(\sum_{i=1}^{d} |v_i|_{m,p,\Omega}^2\right)^{1/2},
$$

for $\mathbf{v} = (v_1, \dots, v_d) \in \mathbf{W}^{m,p}(\Omega)$. For every detail about the above definitions and property we refer to [1, 7, 43].
Finally, in this thesis we will make use of $\mathbb{P}^1$ finite element ($\mathbb{P}^1$-F.E.) approximation. For further details about it we refer to [18, 38, 41].

Chapter 1

Reduced order methods for parametrized PDEs

In this chapter we introduce reduced order methods (ROMs) for parametrized partial differential equations (PPDEs), in which a number of input parameters are used to characterize a particular problem and possible variations in its geometric configuration, physical properties, boundary conditions or source terms. Reduced order models are used when the solution is sought for a large number of different parameter values or when we are interested in a real time input-output computation. In such situations, we need the ability to accurately and efficiently evaluate an output of interest when the input parameters are being varied. However, the complexity and computational cost associated with solving the full partial differential equation for each new parameter value rules out a direct approach. We must therefore seek a different approach that allows us to evaluate the desired output at minimal cost, yet without sacrificing the predictive accuracy of the complex model. Reduced order method are based on a two stage procedure, comprising an offline and an online stage. During the potentially very costly offline stage, one empirically explores the solution manifold to construct a reduced order space that approximates any member of the solution manifold to within a prescribed accuracy. The order space is built as the space spanned by some linear combinations of solutions. The online stage consists of a Galerkin projection onto the order spaces. During this stage, one can explore the parameter space at a substantially reduced cost. The chapter is split in three sections. In the first one we set up the general framework for reduced order methods. In the two next sections we describe two different techniques to build reduced order spaces, i.e., the proper orthogonal decomposition and the greedy basis generation.

1.1 Parametrized PDEs

In this section we describe the framework and the main features of reduced order methods. In particular we will focus on linear coercive elliptic PPDEs. In the first two sections we give the abstract formulation of such problems. Then, in the next three sections, we briefly discuss discretization techniques for the computation of

high-fidelity solutions, how reduced order methods work and a main assumption that must be done to ensure efficiency of the ROMs. This section is based on the book [26] and we remand the reader to it for further details.

1.1.1 Parametric weak formulation

Let $\Omega \subseteq \mathbb{R}^d$ $(d = 1, 2, 3)$ be a domain[1] with Lipschitz boundary $\Gamma = \partial\Omega$. We will consider both scalar-valued and vector-valued field variables $v : \Omega \to \mathbb{R}^{d_v}$, where $d_v = 1$ for scalar-valued problems and $d_v = d$ for vector-valued problems. Let $\mathbb{V}_i$, $i = 1, \dots, d_v$ be functional spaces over Ω. In particular we assume $\mathbb{V}_i$ to be Hilbert spaces. In general we will consider $H_0^k(\Omega) \subseteq \mathbb{V}_i \subseteq H^k(\Omega)$, for a fixed integer $k \geq 1$. We then consider the space $\mathbb{V} = \mathbb{V}_1 \times \cdots \times \mathbb{V}_{d_v}$, equipped with an inner product $(v, w)_{\mathbb{V}}$ and the induced norm $\|v\|_{\mathbb{V}} = \sqrt{(v, v)_{\mathbb{V}}}$, for all $v, w \in \mathbb{V}$, such that this norm is equivalent to the $(H^k(\Omega))^{d_v}$ one. Thus, $\mathbb{V}$ is an Hilbert space. We finally introduce the parameter space $\mathbb{P} \subseteq \mathbb{R}^P$, where P denotes the number of parameters. We will so consider parametric field variables $u : \mathbb{P} \to \mathbb{V}$, $u = u(\mu)$, for $\mu \in \mathbb{P}$.
Our general problem will be settled in a parametrized weak formulation. Let $a : \mathbb{V} \times \mathbb{V} \times \mathbb{P} \to \mathbb{R}$ be a parametrized bilinear (with respect to the first two variables) form and $f, l : \mathbb{V} \times \mathbb{P} \to \mathbb{R}$ be two parametrized linear forms (with respect to the first variable). The abstract formulation of our problem reads: find $u : \mathbb{P} \to \mathbb{V}$ such that

$$a(u(\mu), v; \mu) = f(v; \mu), \qquad \forall\, v \in \mathbb{V}, \mu \in \mathbb{P}, \tag{1.1}$$

and evaluate

$$s(\mu) = l(u(\mu); \mu), \qquad \forall\, \mu \in \mathbb{P}. \tag{1.2}$$

In order to have a well-posed problem for every parameter $\mu \in \mathbb{P}$, we make additional assumptions on the parametrized forms $a(\cdot, \cdot; \mu)$ and $f(\cdot; \mu)$, for every $\mu \in \mathbb{P}$:

(i) $a(\cdot, \cdot; \mu)$ is coercive and continuous with respect to the norm $\|\cdot\|_{\mathbb{V}}$, i.e., there exist two constants $\alpha(\mu) \geq \hat{\alpha} > 0$ and $\gamma(\mu) \leq \hat{\gamma} < +\infty$ such that

$$a(v, v; \mu) \geq \alpha(\mu)\|v\|_{\mathbb{V}}^2 \qquad \text{and} \qquad a(v, w; \mu) \leq \gamma(\mu)\|v\|_{\mathbb{V}}\|w\|_{\mathbb{V}}, \tag{1.3}$$

for every $v, w \in \mathbb{V}$;

(ii) $f(\cdot; \mu)$ is continuous with respect to the norm $\|\cdot\|_{\mathbb{V}}$, i.e., there exists a constant $\lambda(\mu) \leq \lambda < +\infty$ such that

$$f(v; \mu) \leq \lambda(\mu)\|v\|_{\mathbb{V}}, \qquad \forall\, v \in \mathbb{V}. \tag{1.4}$$

These properties ensure well-posedness of problem (1.1), thanks to the Lax-Milgram theorem (cf. [7], p. 140). We also assume that the problem is *compliant*. This means that we assume that the form $a(\cdot, \cdot; \mu)$ is symmetric and that

[1] By domain we mean an open bounded connected subset.

$l(\cdot;\mu) = f(\cdot;\mu)$, for every $\mu \in \mathbb{P}$. We also introduce energy inner product and energy norm as

$$(v, w)_\mu = a(v, w; \mu), \qquad \forall\, v, w \in \mathbb{V}, \tag{1.5}$$

$$\|v\|_\mu = \sqrt{a(v, v; \mu)}, \qquad \forall\, v \in \mathbb{V}, \tag{1.6}$$

respectively, for every $\mu \in \mathbb{P}$. In many cases the $\mathbb{V}$-norm $\|\cdot\|_\mathbb{V}$ coincides with energy norm for a fixed parameter $\bar{\mu}$:

$$(v, w)_{\bar{\mu}} = (v, w)_\mathbb{V} \qquad \text{and} \qquad \|v\|_{\bar{\mu}} = \|v\|_\mathbb{V}$$

for every $v, w \in \mathbb{V}$.

Example 1.1.1 (Linear elliptic equation)**.** *Let $\Omega \subseteq \mathbb{R}^d$ ($d = 1, 2, 3$) be a domain with Lipschitz boundary $\Gamma = \partial\Omega$. We consider the following equation: find $u \in C^2(\overline{\Omega})$ such that*

$$-\nabla \cdot (b(x)\nabla u(x)) = g(x) \qquad \text{for } x \in \Omega, \tag{1.7}$$

with the boundary Dirichlet condition

$$u(x) = 0 \qquad \text{for } x \in \partial\Omega,$$

where $g \in L^2(\Omega)$, $b \in L^\infty(\Omega)$ and $b(x) > 0$ for every $x \in \Omega$. In general existence of a solution to (1.7) is not guaranteed; we have to make further assumptions on the regularity of b, g and Ω. However, equation (1.7) can be formulated in a weak form. Let u be a solution of (1.7). Then, for every $v \in H_0^1(\Omega)$, it holds that

$$-\int_\Omega (\nabla \cdot (b(x)\nabla u(x)))\, v(x)\, dx = \int_\Omega g(x)v(x)\, dx. \tag{1.8}$$

Thanks to Green theorem, the right hand side of (1.8) can be reformulated as

$$\int_\Omega b(x)\, \nabla u(x) \cdot \nabla v(x)\, dx \; - \int_\Omega \nabla \cdot (b(x)v(x)\nabla u(x))\; dx,$$

where, since $v \in H_0^1(\Omega)$, we have that

$$\int_\Omega \nabla \cdot (b(x)v(x)\nabla u(x))\; dx = \int_{\partial\Omega} b(x)v(x)\nabla u(x) \cdot n(x)\, dS = 0.$$

Thus, (1.8) is equivalent to

$$\int_\Omega b(x)\, \nabla u(x) \cdot \nabla v(x)\, dx = \int_\Omega g(x)v(x)\, dx. \tag{1.9}$$

Let us denote with $a : H_0^1(\Omega) \times H_0^1(\Omega) \to \mathbb{R}$ and $f : H_0^1(\Omega) \to \mathbb{R}$ the, respectively, bilinear and linear form such that, for every $u, v \in H_0^1(\Omega)$,

$$a(u, v) = \int_\Omega b(x)\, \nabla u(x) \cdot \nabla v(x)\, dx, \tag{1.10}$$

$$f(v) = \int_\Omega g(x)v(x)\, dx.$$

It is easy to show that a is a coercive and continuous bilinear form and that $f \in H_0^{-1}(\Omega)$. Thus we define the weak formulation of problem (1.7) as: find $u \in H_0^1(\Omega)$ such that

$$a(u, v) = f(v) \qquad \forall\, v \in H_0^1(\Omega). \tag{1.11}$$

Thanks to Lax-Milgram theorem there exists an unique solution to the weak problem (1.11).

1.1.2 Geometrical parametrization

An interesting case that could be treated in this framework is the one where also the domain Ω depends on the parameter, i.e., $\Omega = \Omega_\mu$. Suppose our problem can be formulated as follows: for each $\mu \in \mathbb{P}$, find $u_0(\mu) \in \mathbb{V}_\mu$ such that

$$a_0(u_0(\mu), v; \mu) = f_0(v; \mu), \qquad \forall\, v \in \mathbb{V}_\mu, \tag{1.12}$$

where $\mathbb{V}_\mu$ is a functional space on Ω_μ (we suppose $H_0^1(\Omega_\mu) \subseteq \mathbb{V}_\mu \subseteq H^1(\Omega_\mu)$) and $a_0(\cdot, \cdot; \mu)$ and $f_0(\cdot; \mu)$ are a bilinear and a linear form on $\mathbb{V}_\mu$ satisfying (1.3) and (1.4), respectively. Let us choose a particular value of the parameter $\bar{\mu} \in \mathbb{P}$ and define the *reference domain* as $\Omega = \Omega_{\bar{\mu}}$. The reference domain is related to domains Ω_μ through a parametric transformation $T(\cdot; \mu)$ such that $T(\Omega; \mu) = \Omega_\mu$. We now focus only on a particular class of transformations and problems. First, we introduce a domain decomposition of Ω_μ such that

$$\overline{\Omega_\mu} = \bigcup_{k=1}^{L} \overline{\Omega_\mu^k}$$

where Ω_μ^k, for $k = 1, \ldots, L$ are mutually non overlapping open subsets of Ω_μ. We now suppose that the transformations can be defined between subdomains, i.e., there are $T^k(\cdot; \mu) : \Omega_\mu \to \Omega$ such that:

$$T^k(\Omega^k; \mu) = \Omega_\mu^k,$$
$$T(\cdot; \mu)|_{\Omega^k} = T^k(\cdot; \mu),$$

for $k = 1, \ldots, L$. Moreover we assume that, for each $\mu \in \mathbb{P}$, the map $T(\cdot; \mu)$ is continuous and the maps $T(\cdot; \mu)^k$ are individually bijective and affine. Therefore, each local map T^k can be described by

$$T_i^k(x; \mu) = C_i^k(\mu) + \sum_{j=1}^{d} G_{ij}^k(\mu) x_j, \qquad \text{for } x \in \Omega^k,\ i = 1, \ldots, d,$$

where $C^k : \mathbb{P} \to \mathbb{R}^d$, $G^k : \mathbb{P} \to \mathrm{GL}_d(\mathbb{R})$ are smooth maps. We also denote with $J^k(\mu)$ the determinant of the matrix $G^k(\mu)$. Under these assumptions in many cases the problem (1.12) can be easily formulated in the form (1.1), like in the following Example 1.1.2.

Example 1.1.2 (Advection-diffusion-reaction operators)**.** *An important class of parametrized domain problems that can be effectively treated within an affine framework is the one of advection-diffusion-reaction operators:*

$$Lv = \nabla \cdot (D(\mu)\nabla v) + A(\mu) \cdot \nabla v + \lambda(\mu)v,$$

being $D(\mu)$ the $d \times d$ diffusivity tensor, $A(\mu)$ the advection field in $\mathbb{R}^d$ and $\lambda(\mu)$ the reaction coefficient. Let us consider the case $d = 2$. The bilinear form associated is, for each $v, w \in \mathbb{V}_\mu$,

$$a_0(v, w; \mu) = \sum_{k=1}^{L} \int_{\Omega_\mu^k} \left(\frac{\partial v}{\partial x}, \ \frac{\partial v}{\partial y}, \ v \right) K_{0,k}(\mu) \left(\frac{\partial w}{\partial x}, \ \frac{\partial w}{\partial y}, \ w \right)^T dxdy, \qquad (1.13)$$

where $K_{0,k} : \mathbb{P} \to \mathbb{R}^{3\times 3}$, $k = 1, \ldots, L$, is a smooth mapping such that, for every $\mu \in \mathbb{P}$, is defined as

$$K_{0,k}(\mu) = \begin{pmatrix} D(\mu) & A(\mu) \\ \underline{0}^T & \lambda(\mu) \end{pmatrix}.$$

Denoting with $\mathbb{V}$ the space $\mathbb{V}_{\bar{\mu}}$, given a value $\mu \in \mathbb{P}$, for each $\widehat{v} \in \mathbb{V}_\mu$ we can define $v \in \mathbb{V}$ as $v = \widehat{v} \circ T(\cdot; \mu)$. We can now track back to $\Omega = \Omega_{\bar{\mu}}$ the integrals in (1.13) obtaining:

$$a(v, w; \mu) \doteq \sum_{k=1}^{L} \int_{\Omega^k} \left(\frac{\partial v}{\partial x}, \ \frac{\partial v}{\partial y}, \ v \right) K_k(\mu) \left(\frac{\partial w}{\partial x}, \ \frac{\partial w}{\partial y}, \ w \right)^T dxdy,$$

for $v, w \in \mathbb{V}$. We denoted by $K_k(\mu)$, $k = 1, \ldots, L$, the transformed operator:

$$K_k(\mu) = J^k(\mu)\, \widehat{G}^k(\mu)\, K_{0,k}(\mu)\, \widehat{G}^k(\mu)^T,$$

where

$$\widehat{G}^k(\mu) = \begin{pmatrix} (G^k(\mu))^{-1} & \underline{0} \\ \underline{0}^T & 1 \end{pmatrix}.$$

Similarly we can require that the linear form $f_0(\cdot; \mu) : \mathbb{V}_\mu \to \mathbb{R}$ in (1.12) is, for all $v \in \mathbb{V}_\mu$:

$$f_0(v; \mu) = \sum_{k=1}^{L} \int_{\Omega_\mu^k} F_{0,k}(\mu) v \, dxdy.$$

Here $F_{0,k}$, $k = 1, \ldots, L$, is a function $\mathbb{P} \to \mathbb{R}$. Acting exactly as before, we can obtain a linear form defined on the reference space $\mathbb{V}$. This form turns out to be, for $v \in \mathbb{V}$:

$$f(v; \mu) \doteq \sum_{k=1}^{L} \int_{\Omega^k} F_k(\mu) v \, dxdy,$$

where, for $k = 1, \ldots, L$, the parametric coefficient $F_k(\mu)$ is

$$F_k(\mu) = J^k(\mu) F_{0,k}(\mu).$$

1.1.3 Discretization techniques

In the following part of the thesis we will assume that given a problem (1.1), the solution is sought in a conforming approximation space, i.e., a finite-dimensional subspace $\mathbb{V}_\delta \subseteq \mathbb{V}$. This will be usually constructed as a finite elements subspace. Let $\{\psi_i\}_{i=1}^{N_\delta}$ be a basis of $\mathbb{V}_\delta$, where $N_\delta = \dim(\mathbb{V}_\delta) < +\infty$. For every $\mu \in \mathbb{P}$, we consider the discrete problem: find $u_{N_\delta}(\mu) \in \mathbb{V}_\delta$ such that

$$a(u_{N_\delta}(\mu), v; \mu) = f(v; \mu), \qquad \forall\, v \in \mathbb{V}_\delta, \tag{1.14}$$

and evaluate

$$s_{N_\delta}(\mu) = l(u_{N_\delta}(\mu); \mu).$$

We will refer problem (1.14) as the *truth problem* and u_{N_δ} as the *truth solution*. We will regard at this as an high fidelity approximation of the solution to problem (1.1), in the sense we require the error $\|u(\mu) - u_{N_\delta}(\mu)\|_{\mathbb{V}}$ to be sufficiently small. A desired accuracy can be achieved if we allow the dimension of the approximation space, N_δ, to be sufficiently high. However, this implies that the computation of truth solution is potentially very expensive.
Thanks to the conformity of approximation space, the truth solution satisfies the Galerkin orthogonality property:

$$a(u(\mu) - u_{N_\delta}(\mu), v; \mu) = 0, \qquad \forall\, v \in \mathbb{V}_\delta. \tag{1.15}$$

Therefore the continuity and coercivity of the bilinear form allow us to recover Cea's Lemma. Indeed, it holds that

$$\begin{aligned}\alpha(\mu)\|u(\mu) - u_{N_\delta}(\mu)\|_{\mathbb{V}}^2 &\leq a(u(\mu) - u_{N_\delta}(\mu), u(\mu) - u_{N_\delta}(\mu); \mu) = \\ &= a(u(\mu) - u_{N_\delta}(\mu), u(\mu) - v; \mu) \leq \gamma(\mu)\|u(\mu) - u_{N_\delta}(\mu)\|_{\mathbb{V}}\|u(\mu) - v\|_{\mathbb{V}},\end{aligned}$$

for every $v \in \mathbb{V}_\delta$, which gives

$$\|u(\mu) - u_{N_\delta}(\mu)\|_{\mathbb{V}_\delta} \leq \frac{\gamma(\mu)}{\alpha(\mu)} \inf_{v \in \mathbb{V}_\delta} \|u(\mu) - v\|_{\mathbb{V}_\delta}.$$

Therefore, for every $\mu \in \mathbb{P}$, the approximation error is strictly related with the distance of the *solution manifold* $\mathcal{M} \doteq \{u(\mu) : \mu \in \mathbb{P}\}$ from the truth approximation space $\mathbb{V}_\delta$.
So, the problem (1.1) has been transformed into a linear algebra one. Indeed, let us denote with A_δ^μ, $M_\delta \in \mathbb{R}^{N_\delta \times N_\delta}$ and $f_\delta^\mu \in \mathbb{R}^{N_\delta}$ the matrices associated, respectively, with the energy inner product, the inner product and the right hand side, defined as

$$(M_\delta)_{ij} = (\psi_i, \psi_j)_{\mathbb{V}}, \qquad (A_\delta^\mu)_{ij} = a(\psi_i, \psi_j; \mu), \qquad (f_\delta^\mu)_i = f(\psi_i; \mu),$$

for all $i, j = 1, \ldots, N_\delta$. Matrix A_δ^μ is usually referred as the *stiffness* matrix. Then, the truth problems reads: for every $\mu \in \mathbb{P}$, find $u_\delta^\mu \in \mathbb{R}^{N_\delta}$ such that

$$A_\delta^\mu u_\delta^\mu = f_\delta^\mu.$$

Thus the truth solution is $u_{N_\delta}(\mu) = \sum_{i=1}^{N_\delta} (u_\delta^\mu)_i \psi_i$ and output can eventually be evaluated as (in the compliant case)

$$s_{N_\delta}(\mu) = (u_\delta^\mu)^T f_\delta^\mu.$$

Depending on the solver of choice to invert the linear system and the properties of the stiffness matrix, the computational cost of output evaluation for a given μ, is $O(N_\delta^\alpha)$, with $\alpha \geq 1$.

1.1.4 Reduced order methods

As we would like to require an high accuracy in the truth approximation of solution of problem (1.1), this leads to an high computation cost, depending on N_δ. Moreover, if we are interested in solutions for many (let's say K) different parameter values, the computational cost is K times that of the truth problem. Therefore we should find a way to reduce the computational effort. In this context reduced order methods come very useful. Let us introduce the discrete version of the solution manifold, the *truth solution manifold* $\mathcal{M}_\delta = \{u_{N_\delta}(\mu) : \mu \in \mathbb{P}\}$. A main assumption in reduced order methods is that $\mathcal{M}_\delta$ is of low dimension, i.e., that the span of a low number of appropriately chosen basis functions represents $\mathcal{M}_\delta$ with a small error. Therefore, the idea is to seek for a small amount, let's say $N \ll N_\delta$, of basis functions $\xi_1, \dots, \xi_N \in \mathbb{V}_\delta$, and then to solve the problem (1.1) in the space spanned by those. So, the solving procedure is split in two steps. During the potentially very costly *Offline* stage, a N-dimensional reduced order space $\mathbb{V}_{\mathtt{rb}} = \text{span}\{\xi_1, \dots, \xi_N\} \subseteq \mathbb{V}_\delta$ is built. Then, in the *Online* stage, the reduced order approximation is sought as: for any given $\mu \in \mathbb{P}$, find $u_N \in \mathbb{V}_{\mathtt{rb}}$ such that

$$a(u_N(\mu), v; \mu) = f(v; \mu), \qquad \forall\, v \in \mathbb{V}_{\mathtt{rb}}, \tag{1.16}$$

and evaluate

$$s_N(\mu) = l(u_N(\mu); \mu).$$

As in the previous case, this is just a linear algebra problem. Now one just have to solve a $N \times N$ linear system, which makes the online stage potentially very cheap, taking $N \ll N_\delta$. If we denote with $B \in \mathbb{R}^{N_\delta \times N}$ and $u_N^\mu \in \mathbb{R}^N$ respectively the matrix such that $u_N(\mu) = \sum_{i=1}^N (u_N^\mu)_i \xi_i$ and $\xi_i = \sum_{j=1}^{N_\delta} B_{ji} \varphi_j$, for $j = 1, \dots, N$, (note that $u_\delta^\mu = B u_N^\mu$) then problem (1.16) is equivalent to solving the linear system

$$A_N^\mu u_N^\mu = f_N^\mu, \tag{1.17}$$

where $A_N^\mu = B^T A_\delta^\mu B$ and $f_N^\mu = B^T f_\delta^\mu$. The output of interest can thus (in the compliant case) be evaluated as $s_N(\mu) = (u_N^\mu)^T f_N^\mu$. So, for each given $\mu \in \mathbb{P}$, the computational cost of the output evaluation in the online stage is $O(N^\alpha)$, for some $\alpha \geq 1$. The error resulting from the approximation can be split in two parts:

$$\|u(\mu) - u_N(\mu)\|_{\mathbb{V}} \leq \|u(\mu) - u_{N_\delta}(\mu)\|_{\mathbb{V}} + \|u_{N_\delta}(\mu) - u_N(\mu)\|_{\mathbb{V}}. \tag{1.18}$$

The first part only depends on how good is the discretization technique used. In the following we will not focus on such discretization techniques, but we will

assume that the truth solution is a good approximation of the real solution. For further details on discretization techniques we refer to [38,41]. Instead, the second part of (1.18) depends on the distance of $\mathcal{M}_\delta$ from $\mathbb{V}_{\mathtt{rb}}$. Indeed Cea's Lemma still holds:

$$\|u_{N_\delta}(\mu) - u_N(\mu)\|_{\mathbb{V}} \leq \frac{\gamma(\mu)}{\alpha(\mu)} \inf_{v \in \mathbb{V}_{\mathtt{rb}}} \|u_{N_\delta}(\mu) - v\|_{\mathbb{V}}.$$

If we introduce the Kolmogorov N-width as

$$E(\mathcal{M}_\delta, \mathbb{V}_{\mathtt{rb}}) = \sup_{u_{N_\delta} \in \mathcal{M}_\delta} \inf_{v \in \mathbb{V}_{\mathtt{rb}}} \|u_\delta - v\|_{\mathbb{V}}, \tag{1.19}$$

then $d_N(\mathcal{M}_\delta) = \inf\{E(\mathcal{M}_\delta, \mathbb{V}) : \mathbb{V} \subseteq \mathbb{V}_\delta, \dim(\mathbb{V}) = N\}$ is an index of how good $\mathcal{M}_\delta$ can be approximated by a N-dimensional subspace. Another quantity that can be considered instead of $E(\mathcal{M}_\delta, \mathbb{V}_{\mathtt{rb}})$ is

$$\sqrt{\int_{\mu \in \mathbb{P}} \inf_{v \in \mathbb{V}_{\mathtt{rb}}} \|u_{N_\delta}(\mu) - v\|_{\mathbb{V}}^2 \, d\mu}. \tag{1.20}$$

The former corresponds to an L^∞ norm over the parameter space, while the latter corresponds to an L^2 one. Depending on which of these quantities one wants to minimize we get two different methods for building reduced order spaces. These methods are presented in sections 1.2 and 1.3.

1.1.5 Affine decomposition

To ensure efficiency of the online stage we have to make a further assumption. Indeed, to solve problem (1.17) for a new parameter $\mu \in \mathbb{P}$, the matrix A_N^μ needs to be assembled. To do this, one generally would need to first assemble the matrix A_δ^μ and then construct $A_N^\mu = B^T A_N^\mu B$. This computation depends on N_δ, and would severely limit the potential for rapid online evaluation of new reduced order solution. The same observation holds for term f_N^μ (and for the reduced order vector of output, in case of non-compliant problems). However, this restriction can be overcome if we assume that the forms $a(\cdot, \cdot; \mu)$, $f(\cdot; \mu)$ and $l(\cdot; \mu)$ fulfill the affine decomposition:

$$a(\cdot, \cdot; \mu) = \sum_{q=1}^{Q_a} \theta_q^a(\mu) a_q(\cdot, \cdot), \tag{1.21}$$

$$f(\cdot; \mu) = \sum_{q=1}^{Q_f} \theta_q^f(\mu) f_q(\cdot), \tag{1.22}$$

$$l(\cdot; \mu) = \sum_{q=1}^{Q_l} \theta_q^l(\mu) l_q(\cdot), \tag{1.23}$$

where each form

$$a_q : \mathbb{V} \times \mathbb{V} \to \mathbb{R}, \qquad f_q : \mathbb{V} \to \mathbb{R}, \qquad l_q : \mathbb{V} \to \mathbb{R},$$

is independent of the parameter value μ, and the parametric coefficients

$$\theta_q^a : \mathbb{P} \to \mathbb{R}, \qquad \theta_q^f : \mathbb{P} \to \mathbb{R}, \qquad \theta_q^l : \mathbb{P} \to \mathbb{R},$$

are scalar quantities. With this assumption matrices A_N^μ, f_N^μ can be computed as

$$A_\delta^\mu = \sum_{q=1}^{Q_a} \theta_q^a(\mu) A_\delta^q, \qquad f_\delta^\mu = \sum_{q=1}^{Q_f} \theta_q^a(\mu) f_\delta^q,$$

where A_δ^q and f_δ^q can be calculated once for all in the offline stage (and analogously for the output term, in case of non-compliant problems). So the cost of the assembly of the matrix A_δ^μ is independent of N_δ; e.g., for matrix A_N^μ, the computational cost scales proportionally to $Q_a \cdot N^2$. For cases where an affine decomposition does not hold naturally for the operator or the linear forms, one can often find an approximate form that satisfies this property using a technique known as Empirical Interpolation. A description of this method can be found in [26], chapter 5. However, in this thesis, we always assume an affine decomposition already holds.

1.2 Proper Orthogonal Decomposition

The Proper Orthogonal Decomposition (POD) is a method for building reduced basis spaces. The main idea of this method is to construct a discrete version of the integral in (1.20) and to find the N-dimensional subspace which minimize this quantity.
Let us first introduce a discrete training set $\mathbb{P}_h = \{\mu_1, \dots, \mu_M\} \subseteq \mathbb{P}$ in parameter domain (it can consist of a regular lattice or a set of randomly generated point in $\mathbb{P}$), where $M = |\mathbb{P}|_h < +\infty$. We denote the discrete truth solution manifold

$$\mathcal{M}_\delta(\mathbb{P}_h) \doteq \{u_{N_\delta}(\mu) \,:\, \mu \in \mathbb{P}_h\}.$$

In particular, we denote $\varphi_1 = u_{N_\delta}(\mu_1), \dots, \varphi_M = u_{N_\delta}(\mu_M)$ the elements of $\mathcal{M}_\delta(\mathbb{P}_h)$, which we can suppose to be linearly independent, and $\mathbb{V}_M$ the linear space spanned by these. Now, for a given positive integer $N < M$, we want to find the N-dimensional subspace $\mathbb{V}_N \subseteq \mathbb{V}_M$ which minimize the quantity

$$\sqrt{\frac{1}{M} \sum_{i=1}^{M} \|\varphi_i - P_N(\varphi_i)\|_{\mathbb{V}}^2}, \tag{1.24}$$

where $P_N : \mathbb{V}_M \to \mathbb{V}_N$ is the projection operator associated with the subspace $\mathbb{V}_N$. The quantity (1.24) can be regarded as a discretized version of (1.20). Let us suppose that $x_1, \dots, x_N$ is an orthonormal basis of $\mathbb{V}_N$ (with respect to the inner product $(\cdot,\cdot)_{\mathbb{V}}$). Then, we have that

$$\begin{aligned}\sum_{i=1}^{M} \|\varphi_i - P_N(\varphi_i)\|_{\mathbb{V}}^2 &= \sum_{i=1}^{M} \|\varphi_i\|_{\mathbb{V}}^2 - \sum_{i=1}^{M}\sum_{j=1}^{N} (x_j, \varphi_i)_{\mathbb{V}}^2 \\ &= C - \sum_{i=1}^{M}\sum_{j=1}^{N} (x_j, \varphi_i)_{\mathbb{V}}^2,\end{aligned}$$

where $C > 0$ is a constant. Therefore we just have to find an orthonormal N-dimensional basis that maximize the quantity

$$\begin{aligned} H(X) &\doteq \frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{N}(x_j,\varphi_i)_{\mathbb{V}}^2 \\ &= \sum_{j=1}^{N}(x_j,\frac{1}{M}\sum_{i=1}^{M}(x_j,\varphi_i)_{\mathbb{V}}\varphi_i)_{\mathbb{V}} = \sum_{j=1}^{N}(x_j,C(x_j))_{\mathbb{V}}, \end{aligned}$$

where we denote $X=\{x_1,\dots,x_N\}$ and $C(x_j)=\frac{1}{M}\sum_{i=1}^{M}(x_j,\varphi_i)_{\mathbb{V}}\varphi_i$. The linear map $C:\mathbb{V}_M\to\mathbb{V}_M$ is symmetric (with respect to the inner product induced on $\mathbb{V}_M$), so we can find an orthonormal basis of eigenvectors $\xi_1,\dots,\xi_M$, associated with the eigenvalues $\lambda_1\geq\cdots\geq\lambda_M>0$ (note that the eigenvalues are all positive since $(C(\varphi_j),\varphi_j)_{\mathbb{V}}=\frac{1}{M}>0$). We denote $\Xi=\{\xi_1,\dots,\xi_N\}$; we want to show that $H(X)\leq H(\Xi)$ for all X N-dimensional orthonormal basis of $\mathbb{V}_M$. Indeed, let $X=\{x_1,\dots,x_N\}$ be such a basis. Since $x_j=\sum_{i=1}^{M}(\xi_i,x_j)_{\mathbb{V}}\xi_i$, we can write

$$\begin{aligned} H(X) &= \sum_{j=1}^{N}(\sum_{i=1}^{M}(\xi_i,x_j)_{\mathbb{V}}\xi_i,\sum_{k=1}^{M}\lambda_k(\xi_k,x_j)_{\mathbb{V}}\xi_k)_{\mathbb{V}} \\ &= \sum_{j=1}^{N}\sum_{k=1}^{M}\lambda_k(x_j,\xi_k)_{\mathbb{V}}^2 = \sum_{k=1}^{M}\lambda_k p_k, \end{aligned}$$

where $p_k=\sum_{j=1}^{N}(x_j,\xi_k)_{\mathbb{V}}^2$. Note that $p_k=\|P_X(\xi_k)\|_{\mathbb{V}}^2\in[0,1]$ and $\sum_{k=1}^{M}p_k=\sum_{j=1}^{N}\|x_j\|_{\mathbb{V}}^2=N$. Therefore, since $\lambda_1\geq\cdots\geq\lambda_M>0$, it holds that

$$H(X)\leq\sum_{k=1}^{N}\lambda_k=H(\Xi).$$

So, taking $\mathbb{V}_{\mathtt{rb}}=\operatorname{span}\{\xi_1,\dots,\xi_N\}$, the minimum of (1.24) is obtained and it is equal to

$$\sqrt{\frac{1}{M}\sum_{i=1}^{M}\sum_{j=N+1}^{M}(\xi_j,\varphi_i)_{\mathbb{V}}^2}=\sqrt{\sum_{j=N+1}^{M}(\xi_j,C(\xi_j))_{\mathbb{V}}}=\sqrt{\sum_{j=N+1}^{M}\lambda_j}.$$

Therefore, the above described procedure provides a method to get a subspace $\mathbb{V}_{\mathtt{rb}}\subseteq\mathbb{V}_\delta$ of prescribed dimension N which is optimal in the sense it minimizes a discrete version of (1.20). Computationally, this can be done finding the N eigenvectors $v_1,\dots,v_N$ associated with the maximum eigenvalues $\lambda_1\geq\cdots\geq\lambda_N$ of the symmetric matrix $C\in\mathbb{R}^{M\times M}$ defined as

$$C_{ij}=\frac{1}{M}(\varphi_i,\varphi_j)_{\mathbb{V}}\qquad\text{for } i,j=1,\dots,M. \tag{1.25}$$

Thus, one defines $\xi_n=\sum_{i=1}^{M}(v_n)_i\varphi_i$, for $n=1,\dots,N$, and takes $\mathbb{V}_{\mathtt{rb}}$ as the space spanned by them.

1.3 Greedy basis generation

The greedy algorithm is an iterative procedure for building reduced order spaces. Reduced order spaces built with a greedy algorithm are usually referred to as reduced *basis* spaces. At each iteration one new basis function is added and the overall precision of the basis set is improved. This procedure aims to minimize the L^∞ norm of the error over the parameter space. It requires one truth solution to be computed at each iteration and the N-dimensional reduced basis space will be the one spanned by the first N truth solutions computed. The main assumption is the availability, for every $\mu \in \mathbb{P}$, of an upper bound $\eta(\mu)$ which provides an estimate of the error induced by replacing $\mathbb{V}_{\mathtt{rb}}$ with $\mathbb{V}_\delta$ in the variational formulation. We will describe how to obtain such an estimator in the next subsection and simply assume here that one is available, satisfying

$$\|u_{N_\delta}(\mu) - u_N(\mu)\|_\mu \leq \eta(\mu), \qquad \forall\, \mu \in \mathbb{P}. \tag{1.26}$$

Alternatively, a different norm, e.g., the intrinsic norm $\|u_{N_\delta}(\mu) - u_N(\mu)\|_{\mathbb{V}}$, or even the measure of the output $|s_{N_\delta}(\mu) - s_N(\mu)|$ error can be chosen. The iterative procedure works like that: provided we have a reduced basis space $\mathbb{V}_{\mathtt{rb}}$ of dimension N, which is given by $\mathbb{V}_{\mathtt{rb}} = \text{span}\{u_{N_\delta}(\mu_1), \dots, u_{N_\delta}(\mu_N)\}$ for some $\mu_1, \dots, \mu_N$ previously selected, we then compute

$$\mu_{N+1} = \arg\max_{\mu \in \mathbb{P}} \eta(\mu), \tag{1.27}$$

and compute $u_{N_\delta}(\mu_{N+1})$ to enrich the reduced basis space as $\mathbb{V}_{\mathtt{rb}} = \text{span}\{u_{N_\delta}(\mu_1), \dots, u_{N_\delta}(\mu_{N+1})\}$. This algorithm is repeated until a prescribed accuracy is gained or until a prescribed dimension N is reached. Actually, in practice, the maximum of (1.27) is sought only on a discretized parameter space $\mathbb{P}_h \subseteq \mathbb{P}$. However, since for every point in $\mathbb{P}_h$ we just have to compute the error estimator, and not a truth solution, the cost per point is small and $\mathbb{P}_h$ can be chosen considerably larger than the one used in the POD algorithm, provided the error can be evaluated efficiently.

1.3.1 A posteriori error estimators

In the following we describe how to obtain a good a posteriori error estimate $\eta(\mu)$. With 'good', we mean it should satisfy certain properties. First of all it must satisfy (1.26), that is, it should be an upper bound. Then it should be *sharp*: an overly conservative error bound can cause inefficient approximation spaces, that is with a dimension N unnecessarily high. Moreover it has to be computational *efficient*: its computation must be very fast in order to speed up the offline stage and to allow its use in the online stage. The computational cost should be independent of N_δ.
Let us denote

$$\alpha_\delta(\mu) = \inf_{v \in \mathbb{V}_\delta} \frac{a(v, v; \mu)}{\|v\|^2_{\mathbb{V}}}, \qquad \lambda_\delta(\mu) = \sup_{v, w \in \mathbb{V}_\delta} \frac{a(v, w; \mu)}{\|v\|_{\mathbb{V}} \|w\|_{\mathbb{V}}}.$$

Note that $\alpha(\mu) \leq \alpha_\delta(\mu)$ and $\lambda(\mu) \geq \lambda_\delta(\mu)$, for every $\mu \in \mathbb{P}$. Moreover, let $e(\mu) \doteq u_{N_\delta}(\mu) - u_N(\mu)$ be the error induced by the reduced basis approximation. We first note that

$$a(e(\mu), v; \mu) = r(v; \mu), \qquad \forall\, v \in \mathbb{V}_\delta, \tag{1.28}$$

where $r(\cdot; \mu) \in \mathbb{V}'_\delta$ defined as

$$r(v; \mu) = f(v; \mu) - a(u_N(\mu), v; \mu),$$

for $v \in \mathbb{V}_\delta$. In particular, thanks to Riesz' theorem (cf. [7], p. 135), there exists $\hat{r}_\delta(\mu) \in \mathbb{V}_\delta$ such that

$$(\hat{r}_\delta(\mu), v)_{\mathbb{V}} = r(v; \mu), \qquad \forall\, v \in \mathbb{V}_\delta.$$

Moreover it holds

$$\|\hat{r}_\delta(\mu)\|_{\mathbb{V}} = \|r(\cdot; \mu)\|_{\mathbb{V}'_\delta} = \sup_{v \in \mathbb{V}_\delta \setminus \{0\}} \frac{r(v; \mu)}{\|v\|_{\mathbb{V}}}.$$

Therefore, (1.28) can be re-written as

$$a(e(\mu), v; \mu) = (\hat{r}_\delta(\mu), v)_{\mathbb{V}}, \qquad \forall\, v \in \mathbb{V}_\delta.$$

The following lemma provides a relation between output error and approximation error in energy norm for compliant problems.

Lemma 1.3.1. *If the problem (1.1) is compliant, then it holds that*

$$s_{N_\delta}(\mu) - s_N(\mu) = \|u_{N_\delta}(\mu) - u_N(\mu)\|^2_\mu, \qquad \forall\, \mu \in \mathbb{P}.$$

Moreover, it holds that $s_{N_\delta}(\mu) \geq s_N(\mu)$, for every $\mu \in \mathbb{P}$.

Proof. Now we have that

$$a(u_{N_\delta}(\mu) - u_N(\mu), v; \mu) = 0 \qquad \forall\, v \in \mathbb{V}_{\mathtt{rb}}.$$

Therefore it follows

$$s_{N_\delta}(\mu) \;-\; s_N(\mu) \;\;=\;\; f(e(\mu); \mu) \;\;=\;\; a(e(\mu), u_{N_\delta}(\mu); \mu) \;\;=\;\; \|e(\mu)\|^2_\mu \;\;\geq\;\; 0.$$

□

From now on, we suppose it is available a lower bound for coercivity constant $a_\delta(\mu)$, namely

$$\alpha_{\mathtt{LB}}(\mu) \leq \alpha_\delta(\mu), \qquad \forall\, \mu \in \mathbb{P}, \tag{1.29}$$

which is independent of $\mathbb{V}_\delta$ and efficiently computable. The approximation procedure of such a lower bound is explained in the next subsection. We can now

define the a posteriori error estimators for, respectively, energy norm, output and relative output as

$$\eta_{\mathtt{en}}(\mu) = \frac{\|\hat{r}_\delta(\mu)\|_{\mathbb{V}}}{\alpha_{\mathtt{LB}}^{1/2}(\mu)}, \tag{1.30}$$

$$\eta_{\mathtt{s}}(\mu) = \frac{\|\hat{r}_\delta(\mu)\|_{\mathbb{V}}^2}{\alpha_{\mathtt{LB}}(\mu)} = (\eta_{\mathtt{en}}(\mu))^2, \tag{1.31}$$

$$\eta_{\mathtt{s,rel}}(\mu) = \frac{\|\hat{r}_\delta(\mu)\|_{\mathbb{V}}^2}{\alpha_{\mathtt{LB}}(\mu) s_N(\mu)} = \frac{\eta_{\mathtt{s}}(\mu)}{s_N(\mu)}. \tag{1.32}$$

The following proposition justifies the names of these estimators.

Proposition 1.3.2. *The following estimates hold:*

$$\|u_{N_\delta}(\mu) - u_N(\mu)\|_\mu \leq \eta_{\mathtt{en}}(\mu), \tag{i}$$

$$s_{N_\delta}(\mu) - s_N(\mu) \leq \eta_s(\mu), \tag{ii}$$

$$\frac{s_{N_\delta}(\mu) - s_N(\mu)}{s_{N_\delta}(\mu)} \leq \eta_{\mathtt{s,rel}}(\mu). \tag{iii}$$

Proof. We prove (i). Since it holds that

$$\|e(\mu)\|_\mu^2 = a(e(\mu), e(\mu); \mu) \leq \|\hat{r}_\delta(\mu)\|_{\mathbb{V}} \|e(\mu)\|_{\mathbb{V}}$$

and

$$\alpha_{\mathtt{LB}}(\mu)\|e(\mu)\|_{\mathbb{V}}^2 \leq a(e(\mu), e(\mu); \mu) = \|e(\mu)\|_\mu^2$$

we get

$$\|e(\mu)\|_{\mathbb{V}} \leq \frac{\|\hat{r}_\delta(\mu)\|_\mu}{\alpha_{\mathtt{LB}}(\mu)} \qquad \text{and} \qquad \|e(\mu)\|_\mu \leq \eta_{\mathtt{en}}(\mu). \tag{1.33}$$

The equations (ii) and (iii) can now be derived from Lemma 1.3.1. □

We now introduce the effectivity indeces associated to the estimators, which can be regarded as a measure of how sharp the estimators are:

$$\mathrm{eff}_{\mathtt{en}}(\mu) = \frac{\eta_{\mathtt{en}}(\mu)}{\|u_{N_\delta}(\mu) - u_N(\mu)\|_\mu},$$

$$\mathrm{eff}_{\mathtt{s}}(\mu) = \frac{\eta_{\mathtt{s}}(\mu)}{s_{N_\delta}(\mu) - s_N(\mu)},$$

$$\mathrm{eff}_{\mathtt{s,rel}}(\mu) = \frac{\eta_{\mathtt{s,rel}}(\mu) s_{N_\delta}(\mu)}{s_{N_\delta}\mu - s_N(\mu)}.$$

While all of these effectivity indexes are bigger than 1, one would desire these to be as close to 1 as possible. The following proposition provides upper estimates for the effectivities.

Proposition 1.3.3. *The following estimates hold:*

$$\mathrm{eff}_{\mathtt{en}}(\mu) \leq \sqrt{\frac{\lambda_\delta(\mu)}{\alpha_{\mathtt{LB}}(\mu)}}, \tag{i}$$

$$\mathrm{eff}_{\mathtt{s}}(\mu) \leq \frac{\lambda_\delta(\mu)}{\alpha_{\mathtt{LB}}(\mu)}, \tag{ii}$$

$$\mathrm{eff}_{\mathtt{s,rel}}(\mu) \leq (1+\eta_{\mathtt{s,rel}}(\mu))\frac{\lambda_\delta(\mu)}{\alpha_{\mathtt{LB}}(\mu)}. \tag{iii}$$

Proof. Now we have that

$$\|\hat{r}_\delta(\mu)\|_{\mathbb{V}}^2 = (\hat{r}_\delta(\mu),\hat{r}_\delta(\mu))_{\mathbb{V}} = a(e(\mu),\hat{r}_\delta(\mu);\mu) \leq \|e(\mu)\|_\mu \|\hat{r}_\delta(\mu)\|_\mu,$$

$$\|\hat{r}_\delta(\mu)\|_\mu^2 = a(\hat{r}_\delta(\mu),\hat{r}_\delta(\mu);\mu) \leq \lambda_\delta(\mu)\|\hat{r}_\delta(\mu)\|_{\mathbb{V}}^2 \leq \lambda_\delta(\mu)\|e(\mu)\|_\mu \|\hat{r}_\delta(\mu)\|_\mu$$

which gives

$$\|\hat{r}_\delta(\mu)\|_\mu \leq \lambda_\delta(\mu)\|e(\mu)\|_\mu.$$

Therefore it holds

$$(\eta_{\mathtt{en}}(\mu))^2 = \frac{\|\hat{r}_\delta(\mu)\|_{\mathbb{V}}^2}{\alpha_{\mathtt{LB}}(\mu)} \leq \frac{\lambda_\delta(\mu)}{\alpha_{\mathtt{LB}}(\mu)}\|e(\mu)\|_\mu^2$$

from which one derives estimates (i) and (ii). Finally, from the effictivities definition, we have that

$$\mathrm{eff}_{\mathtt{s,rel}}(\mu) = \frac{s_{N_\delta}(\mu)}{s_N(\mu)}\mathrm{eff}_{\mathtt{s}}(\mu)$$

and, since

$$\frac{s_{N_\delta}(\mu)}{s_N(\mu)} = 1 + \frac{s_{N_\delta}(\mu)-s_N(\mu)}{s_N(\mu)} \leq 1 + \frac{\eta_{\mathtt{s}}(\mu)}{s_N(\mu)} = 1+\eta_{\mathtt{s,rel}}(\mu),$$

we get (iii). □

Analogous estimators can be obtained for the error in $\mathbb{V}$-norm. Let us define

$$\eta_{\mathbb{V}}(\mu) = \frac{\|\widehat{r_\delta}(\mu)\|_{\mathbb{V}}}{\alpha_{\mathtt{LB}}(\mu)},$$

$$\eta_{\mathbb{V},\mathtt{rel}}(\mu) = \frac{2\|\widehat{r_\delta}(\mu)\|_{\mathbb{V}}}{\alpha_{\mathtt{LB}}(\mu)\|u_N(\mu)\|_{\mathbb{V}}} = \frac{2\eta_{\mathbb{V}}(\mu)}{\|u_N(\mu)\|_{\mathbb{V}}},$$

respectively the a posteriori estimators for intrinsic norm and relative intrinsic norm. An analogous to Proposition 1.3.2 holds (see [26], p. 51, for a proof).

Proposition 1.3.4. *The following estimates hold:*

$$\|u_{N_\delta}(\mu) - u_N(\mu)\|_{\mathbb{V}} \leq \eta_{\mathbb{V}}(\mu) \tag{i}$$

and, if $\eta_{\mathbb{V},\mathtt{rel}}(\mu) \leq 1$,

$$\frac{\|u_{N_\delta}(\mu) - u_N(\mu)\|_{\mathbb{V}}}{\|u_{N_\delta}(\mu)\|_{\mathbb{V}}} \leq \eta_{\mathbb{V},\mathtt{rel}}(\mu). \tag{ii}$$

The respective effectivities are defined as in the previous case:

$$\begin{aligned}\text{eff}_{\mathbb{V}}(\mu) &= \frac{\eta_{\mathbb{V}}(\mu)}{\|u_{N_\delta}(\mu) - u_N(\mu)\|_{\mathbb{V}}},\\ \text{eff}_{\mathbb{V},\texttt{rel}}(\mu) &= \eta_{\mathbb{V},\texttt{rel}}(\mu)\frac{\|u_{N_\delta}(\mu)\|_{\mathbb{V}}}{\|u_{N_\delta}(\mu) - u_N(\mu)\|_{\mathbb{V}}}\end{aligned}$$

Still, an analogous proposition holds (see [26], p. 52, for a proof).

Proposition 1.3.5. *The following estimates hold:*

$$\text{eff}_{\mathbb{V}}(\mu) \leq \frac{\lambda_\delta(\mu)}{\alpha_{\texttt{LB}}(\mu)} \tag{i}$$

and, if $\eta_{\mathbb{V},\texttt{rel}}(\mu) \leq 1$,

$$\text{eff}_{\mathbb{V},\texttt{rel}}(\mu) \leq 3\frac{\lambda_\delta(\mu)}{\alpha_{\texttt{LB}}(\mu)}. \tag{ii}$$

Now, to efficiently evaluate the effectivities, one must be able to efficiently compute $\|\hat{r}_\delta(\mu)\|_{\mathbb{V}}$. To do this, we take advantage of the affine decomposition of $a(\cdot,\cdot;\mu)$ and $f(\cdot;\mu)$. Therefore, we have that

$$r(v;\mu) = \sum_{q=1}^{Q_f}\theta_q^f(\mu)f_q(v) - \sum_{q=1}^{Q_a}\sum_{n=1}^{N}\theta_q^a(\mu)(u_N^\mu))_n a_q(\xi_n, v), \qquad \forall\, v \in \mathbb{V}_\delta,$$

where $\{\xi_1, \dots, \xi_N\}$ is a basis of $\mathbb{V}_{\texttt{rb}}$ and $u_N(\mu) = \sum_{n=1}^N (u_N^\mu)_n \xi_n$. We now introduce the coefficients vector

$$r(\mu) = (\theta_1^f(\mu), \dots, \theta_{Q_f}^f(\mu), -(u_N^\mu)^T\theta_1^a(\mu), \dots, -(u_N^\mu)^T\theta_{Q_a}^a(\mu)) \in \mathbb{R}^{Q_r},$$

where $Q_r = Q_f + NQ_a$. Analogously we define

$$\begin{aligned}F &= (f_1, \dots, f_{Q_f}) \in (\mathbb{V}_\delta')^{Q_f},\\ A_q &= (a_q(\xi_1,\cdot), \dots, a_q(\xi_N,\cdot)) \in (\mathbb{V}_\delta')^N, \qquad \text{for } q = 1, \dots, Q_a,\\ R &= (F, A_1, \dots, A_{Q_a}) \in (\mathbb{V}_\delta')^{Q_r}.\end{aligned}$$

Thus, we obtain that

$$(\hat{r}_\delta(\mu), v)_{\mathbb{V}} = r(v;\mu) = \sum_{q=1}^{Q_r} r_q(\mu)R_q(v), \qquad \forall\, v \in \mathbb{V}.$$

Since, for $q = 1, \dots, Q_r$, $R_q(\cdot) = (\hat{r}_\delta^q, \cdot)_{\mathbb{V}}$ for a $\hat{r}_\delta^q \in \mathbb{V}_\delta$, we can write

$$\hat{r}_\delta^q(\mu) = \sum_{q=1}^{Q_r} r_q(\mu)\hat{r}_\delta^q.$$

Therefore, we finally get that

$$\|\hat{r}_\delta(\mu)\|^2_{\mathbb{V}} = \sum_{q,p=1}^{Q_r} r_q(\mu) r_p(\mu) (\hat{r}^q_\delta, \hat{r}^p_\delta)_{\mathbb{V}}.$$

Note that the inner products $(\hat{r}^q_\delta, \hat{r}^p_\delta)_{\mathbb{V}}$, for $p, q = 1, \dots, Q_r$, can be computed once and for all in the offline stage, so that $\|\hat{r}_\delta(\mu)\|^2_{\mathbb{V}}$ can be efficiently computed in the online stage, with a computational cost independent of N_δ.
This procedure is implemented as follows. Remember that we have the affine decomposition of matrix A^μ_δ,f^μ_δ:

$$A^\mu_\delta = \sum_{q=1}^{Q_a} \theta^a_q(\mu) A^q_\delta, \qquad f^\mu_\delta = \sum_{q=1}^{Q_f} \theta^a_q(\mu) f^q_\delta.$$

We therefore define $R \in \mathbb{R}^{N_\delta \times Q_r}$ as $R_{i,q} = R_q(\varphi_i)$, for $i = 1, \dots, N_\delta$, $q = 1, \dots, Q_r$. Note that we have

$$R = (f^1_\delta, \dots, f^1_\delta, B^T A^1_\delta, \dots, B^T A^1_\delta).$$

Thus, if we define $G \in \mathbb{R}^{Q_r \times Q_r}$ as $G = R^T M_\delta R$, we finally have

$$\|\hat{r}_\delta(\mu)\|_{\mathbb{V}} = \sqrt{r(\mu)^T \, G \, r(\mu)}.$$

We can compute G once for all during the offline stage.

1.3.2 Coercivity constant

In this section we explore two possible methods to calculate efficiently a lower bound for the coercivity constant (1.29). The first one is called (Multi-parameter) Min-θ-approach and it applies only to parametrically coercive problems. Otherwise the second one, called Successive Constraint Method (SCM), can be generalized to non-symmetric problems, although this is not discussed in this thesis (we refer [13–15, 25, 28, 29] for further details). Both these methods make use of the affine decompositions (1.21),(1.22) and (1.23). Let us first note that the coercivity constant $\alpha_\delta(\mu)$ can be defined as the smallest eigenvalue of a generalized eigenvalue problem: find $(\lambda, w) \in \mathbb{R}^+ \times \mathbb{V}_\delta$ such that

$$a(v, w; \mu) = \lambda (v, w)_{\mathbb{V}}, \qquad \forall \, v \in \mathbb{V}_\delta. \tag{1.34}$$

Min-θ-approach

Parametrically coercive problems are characterized by the following assumptions:

(i) $\theta^a_q(\mu) > 0$, $\forall \, \mu \in \mathbb{P}$, $q = 1, \dots, Q_a$,

(ii) $a_q(\cdot, \cdot)$ is semi-positive definite for all $q = 1, \dots, Q_a$.

Under these assumptions and further assuming that the coercivity constant $\alpha_\delta(\bar{\mu})$ has been computed for a single parameter value $\bar{\mu} \in \mathbb{P}$, we observe that

$$\begin{aligned}
\alpha_\delta(\mu) &= \inf_{v \in \mathbb{V}_\delta} \sum_{q=1}^{Q_a} \theta_q^a(\mu) \frac{a_q(v,v)}{\|v\|_{\mathbb{V}}^2} = \inf_{v \in \mathbb{V}_\delta} \sum_{q=1}^{Q_a} \frac{\theta_q^a(\mu)}{\theta_q^a(\bar{\mu})} \theta_q^a(\bar{\mu}) \frac{a_q(v,v)}{\|v\|_{\mathbb{V}}^2} \\
&\geq \left(\min_{q=1,\ldots,Q_a} \frac{\theta_q^a(\mu)}{\theta_q^a(\bar{\mu})} \right) \inf_{v \in \mathbb{V}} \sum_{q=1}^{Q_a} \theta_q^a(\bar{\mu}) \frac{a_q(v,v)}{\|v\|_{\mathbb{V}}^2} \\
&= \alpha_\delta(\bar{\mu}) \min_{q=1,\ldots,Q_a} \frac{\theta_q^a(\mu)}{\theta_q^a(\bar{\mu})} \doteq \alpha_{\mathtt{LB}}(\mu)
\end{aligned}$$

While this approach provides a positive lower bound $\alpha_{\mathtt{LB}}(\mu)$ for $\alpha_\delta(\mu)$, it is generally not a sharp bound, possibly resulting in error bounds that are overly conservative. However, it can be refined as follows. Let us consider a set of M parameter eigenvalues $\mu_1, \ldots, \mu_M$ for which we compute the coercivity constant $\alpha_\delta(\mu_m)$ from the lowest eigenvalues of (1.34). Now we have that, for each $m = 1, \ldots, M$,

$$\alpha_\delta(\mu_m) = \min_{q=1,\ldots,Q_a} \frac{\theta_q^a(\mu)}{\theta_q^a(\mu_m)}$$

is a guaranteed lower bound for $\alpha_\delta(\mu)$. It follows that

$$\alpha_{\mathtt{LB}}(\mu) = \max_{m=1,\ldots,M} \left(\alpha_\delta(\mu_m) \min_{q=1,\ldots,Q_a} \frac{\theta_q^a(\mu)}{\theta_q^a(\mu_m)} \right)$$

is the sharpest of the lower bounds of $\alpha_\delta(\mu)$ among all candidates. While this is a more accurate approach, it also requires more eigenvalue problems to be solved.

Successive Constraint Method

As for the multi-parameter Min-θ-approach, the SCM is an offline/online procedure where generalized eigenvalue problems of size N_δ need to be solved during the offline phase; the online part is then reduced to provide a lower bound $\alpha_{\mathtt{LB}}(\mu)$ of the coercivity constant $\alpha_\delta(\mu)$ for each new parameter value $\mu \in \mathbb{P}$ with an operation count that is independent of N_δ. The idea of the SCM is to express the problem (1.34) as a minimization problem of the functional $S : \mathbb{P} \times \mathbb{R}^{Q_a} \to \mathbb{R}$ such that

$$S(\mu, y) = \sum_{q=1}^{Q_a} \theta_q^a(\mu) y_q$$

over the set of admissible solutions

$$\Lambda = \left\{ y \in \mathbb{R}^{Q_a} \, : \, \exists \, v \in \mathbb{V}_\delta \text{ s.t. } y_q = \frac{a_q(v,v)}{\|v\|_{\mathbb{V}}^2}, q = 1, \ldots, Q_a \right\}.$$

Then, we can equivalently write

$$\alpha_\delta(\mu) = \min_{y \in \Lambda} S(\mu, y)$$

and a lower and upper bound can be found by enlarging or restricting the admissible set of solution vectors y. This is done by introducing $\Lambda_{\mathtt{UB}} \subseteq \Lambda \subseteq \Lambda_{\mathtt{LB}}$ and defining

$$\alpha_{\mathtt{LB}}(\mu) = \min_{y \in \Lambda_{\mathtt{LB}}} S(\mu, y), \qquad \text{and} \qquad \alpha_{\mathtt{UB}}(\mu) = \min_{y \in \Lambda_{\mathtt{UB}}} S(\mu, y).$$

The remaining question is how to efficiently design the spaces $\Lambda_{\mathtt{UB}}$ and $\Lambda_{\mathtt{LB}}$ to ensure that any target accuracy for the error quantity $1 - \alpha_{\mathtt{LB}}(\mu)/\alpha_{\mathtt{UB}}(\mu)$ can be achieved. Remember that, for the greedy algorithm, we introduced the discretized parameter space $\mathbb{P}_h$. The offline part of the SCM is based on a greedy approach where the n-th iteration of the offline procedure is initiated by assuming that:

(i) we know the coercivity constants $\alpha_\delta(\mu_j)$, $j = 1, \ldots, n$, for some parameter values $\mathbb{C}_n = \{\mu_1, \ldots, \mu_n\} \subseteq \mathbb{P}_h$;

(ii) for each $\mu \in \mathbb{P}_h$, we have some lower bound $\alpha_{\mathtt{LB}}^{n-1}(\mu) \geq 0$ of $\alpha_\delta(\mu)$ from the previous iteration.

For $n = 1$, we set $\alpha_{\mathtt{LB}}^0(\mu) = 0$ for all $\mu \in \mathbb{P}_h$. The constants $\alpha_\delta(\mu_j)$ are computed as solutions of the eigenvalue problem (1.34); we denote with w_δ^j the corresponding eigenfunctions. These eigenfunctions provide the corresponding vectors $y^j \in \mathbb{R}^{Q_a}$ by

$$(y^j)_q = \frac{a_q(w_\delta^j, w_\delta^j)}{\|w_\delta^j\|_{\mathbb{V}}^2}, \qquad \text{for } q = 1, \ldots, Q_a.$$

We can now set

$$\Lambda_{\mathtt{UB}}^n = \left\{ y^j : j = 1, \ldots, n \right\}.$$

So, the evaluation of $\alpha_{\mathtt{UB}}(\mu)$ consists of solving a discrete minimization problem, whose cost is clearly independent of N_δ once the vectors y^j have been built. For $\Lambda_{\mathtt{LB}}$ we define first a rectangular box $\Theta = \prod_{q=1}^{Q_a} [\sigma_q^-, \sigma_q^+] \subseteq \mathbb{R}^{Q_a}$ containing Λ by setting

$$\sigma_q^- = \inf_{v \in \mathbb{V}_\delta} \frac{a_q(v, v)}{\|v\|_{\mathbb{V}}^2} \qquad \text{and} \qquad \sigma_q^+ = \sup_{v \in \mathbb{V}_\delta} \frac{a_q(v, v)}{\|v\|_{\mathbb{V}}^2}.$$

This corresponds to computing the smallest and the largest eigenvalues of an eigenvalue problem for each $a_q(\cdot, \cdot)$ and can be computed once at the beginning of the SCM algorithm. To ensure that the set $\Lambda_{\mathtt{LB}}$ is as small as possible while containing Λ, we impose some additional restrictions, which result in sharper lower bounds. These constraints depend on the value of the actual parameter μ, which implies $\Lambda_{\mathtt{LB}}$ to depend on the parameter too, i.e., $\Lambda_{\mathtt{LB}} = \Lambda_{\mathtt{LB}}(\mu)$. We introduce the function that provides close parameter values

$$P_M(\mu; E) = \begin{cases} M \text{ closest point to } \mu \text{ in } E & \text{if } |E| > M, \\ E & \text{if } |E| \leq M, \end{cases}$$

for either $E = \mathbb{C}_n$ or $E = \mathbb{P}_h$. For some M_1, M_2, we define

$$\begin{aligned} \Lambda_{\mathtt{LB}}^n(\mu) = \big\{ y \in \Theta \; : & S(\mu_1, y) \geq \alpha_\delta(\mu_1), \forall\, \mu_1 \in \mathbb{P}_{M_1}(\mu; \mathbb{C}_n), \\ & S(\mu_1, y) \geq \alpha_{\mathtt{LB}}^{n-1}(\mu_2), \forall\, \mu_2 \in \mathbb{P}_{M_2}(\mu; \mathbb{P}_h \setminus \mathbb{C}_n) \big\}. \end{aligned}$$

It can be shown that $\Lambda^n_{\mathtt{UB}} \subseteq \Lambda \subseteq \Lambda^n_{\mathtt{LB}}(\mu)$ (see [29] for the proof). Consequently, $\Lambda^n_{\mathtt{UB}}$, Λ and $\Lambda^n_{\mathtt{LB}}(\mu)$ are nested as

$$\Lambda^1_{\mathtt{UB}} \subseteq \Lambda^2_{\mathtt{UB}} \subseteq \cdots \subseteq \Lambda^n_{\mathtt{UB}} \subseteq \cdots \subseteq \Lambda \subseteq \cdots \subseteq \Lambda^n_{\mathtt{LB}}(\mu) \subseteq \cdots \subseteq \Lambda^2_{\mathtt{LB}}(\mu) \subseteq \Lambda^1_{\mathtt{LB}}(\mu).$$

Note that finding $\alpha^n_{\mathtt{LB}}(\mu) = \min_{y \in \Lambda^n_{\mathtt{LB}}(\mu)} S(\mu, y)$ corresponds to a linear programming problem of Q_a design variables and $2Q_a + M_1 + M_2$ conditions; the complexity of the linear programming problem is thus independent of N_δ. Now we can define a greedy selection to enrich the space $\mathbb{C}_n$ and build $\mathbb{C}_n + 1$ at all stages of n. Given the n-th step, we select

$$\mu_{n+1} = \arg\max_{\mu \in \mathbb{P}_h} \left(1 - \frac{\alpha^n_{\mathtt{LB}}(\mu)}{\alpha^n_{\mathtt{UB}}(\mu)}\right)$$

and we set $\mathbb{C}_{n+1} = \mathbb{C}_n \cup \{\mu_n + 1\}$. This procedure is repeated until a fixed tolerance is reached. Once that this process has been completed in the offline phase, one can so compute $\alpha_{\mathtt{LB}}(\mu)$, for each $\mu \in \mathbb{P}$, in the online stage just solving a linear programming problem, whose cost is independent of N_δ. Note that, since we now consider any $\mu \in \mathbb{P}$ not necessarily contained in $\mathbb{P}_h$, we must add the additional constraint that $S(\mu, y) \geq 0$.

A comparison between Min-θ-approach and SCM

We report here a theorem that provides an inequality between the results of the two methods presented (for a proof see [26], pp. 63-65).

Proposition 1.3.6. *For parametrically coercive problems, consider the multiparameter Min-θ-approach based upon the computation of $\alpha_\delta(\mu_1), \ldots, \alpha_\delta(\mu_M)$ and the SCM assuming that $\mu_m \in \mathbb{C}_n$ for all $1 \leq m \leq M$. Then, the multiparameter Min-θ-approach lower bound is at most as sharp as the lower bound provided by the SCM based upon $\mathbb{C}_n$ and any $M_2 > 0$, i.e.,*

$$\alpha^{SCM}_{\mathtt{LB}}(\mu) \geq \alpha^{\theta}_{\mathtt{LB}}(\mu), \qquad \forall\, \mu \in \mathbb{P}.$$

Chapter 2

Weighted reduced order methods for SPDEs

In this chapter we introduce weighted reduced order algorithms for the solution of Stochastic Partial Differential Equations (SPDEs). In the first section, the general framework for weighted reduced order algorithms is settled; we consider a particular class of linear stochastic partial differential equations. Then, in the second section, we briefly present the classical Monte-Carlo procedure for computing an approximate solution and discuss why reduced order approximation could be important in such a context. We therefore present some possible ways to modify the previous presented greedy and POD algorithms in order to keep into account the probability distribution of the parameters; the modified algorithms will be referred as *weighted*. Finally, in the third section, we implement the weighted algorithms for the solution of two simple linear elliptic PDEs and we make some numerical tests.

2.1 Parametrized PDEs with random inputs

Let $(\mathcal{S}, \mathcal{A}, P)$ be a complete probability space, where $\mathcal{S}$ is the event space, $\mathcal{A} \subseteq 2^{\mathcal{S}}$ the σ-algebra, and P the probability measure. Consider a d-dimensional domain $\Omega \subseteq \mathbb{R}^d$ $(d = 1, 2, 3)$ with Lipschitz boundary $\partial\Omega$, and we study the following problem: find a stochastic function, $u : \mathcal{S} \times \overline{\Omega} \to \mathbb{R}$, such that for P-almost every (a.e.) $\omega \in \mathcal{S}$, the following equation holds:

$$\mathcal{L}(\omega, x; u) = g(\omega, x), \qquad x \in \Omega, \tag{2.1}$$

subject to the boundary condition

$$\mathcal{B}(\omega, x; u) = h(\omega, x), \qquad x \in \partial\Omega, \tag{2.2}$$

where $\mathcal{L}$ is a (linear/nonlinear) differential operator, $\mathcal{B}$ is a boundary operator and $g : \mathcal{S} \times \Omega \to \mathbb{R}$, $h : \mathcal{S} \times \partial\Omega \to \mathbb{R}$ are the forcing terms. In the most general settings, the operators $\mathcal{L}$ and $\mathcal{B}$, as well as the driving terms f and g, can all have random components. Finally, we assume that the boundary $\partial\Omega$ is sufficiently

regular and the driving terms g and h are properly posed, such that (2.1)-(2.2) is well posed P-a.e. $\omega \in \mathcal{S}$.
To solve (2.1)-(2.2) numerically, one needs to reduce the infinite-dimensional probability space to a finite-dimensional space. This can be accomplished by characterizing the probability space by a finite number of random variables. In this thesis we assume that such a characterization is already available. However, such a situation can be achieved via a certain type of decomposition which can approximate the target random process with desired accuracy, such as the Karhunen-Loève type expansion (see, for example, [8]). Thus, assuming that the random inputs can be characterized by N random variables, we can re-write the random inputs in abstract form, e.g.,

$$\mathcal{L}(\omega, x; u) = \mathcal{L}(Y^1(\omega), \ldots, Y^N(\omega), x; u), \qquad g(\omega, x) = f(Y^1(\omega), \ldots, Y^N(\omega), x),$$

where $\{Y^i\}_{i=1}^N$ are real random variables with zero mean value and unit variance. Hence, the solution of (2.1)-(2.2) can be described by the same set of random variables $\{Y^i\}_{i=1}^N$, i.e.,

$$u(\omega, x) = u(Y^1(\omega), \ldots, Y^N(\omega), x).$$

From a mathematical and computational point of view, it is most convenient to further assume that these random variables are mutually independent. Such an assumption is non-trivial and could introduce more errors in approximating the random input processes. However, in this thesis, we take the customary approach of assuming that the random inputs are already characterized by a set of mutually independent random variables. Moreover, we assume their probability distributions are characterized by density functions $\rho_i : \Gamma_i \to \mathbb{R}^+$ and their images $\Gamma_i = Y^i(\mathcal{S})$ are bounded intervals in $\mathbb{R}$ for $i = 1, \ldots, N$. Then

$$\rho(y) = \prod_{i=1}^{N} \rho_i(y_i), \qquad \text{for } y \in \Gamma,$$

is the joint probability density of $Y = (Y^1, \ldots, Y^N)$ with support

$$\Gamma = \prod_{i=1}^{N} \Gamma_i \subseteq \mathbb{R}^N.$$

This allows us to re-write (2.1)-(2.2) as an $(N + d)$-dimensional differential equation in the strong form

$$\mathcal{L}(y, x; u) = g(y, x), \qquad (y, x) \in \Gamma \times \Omega, \tag{2.3}$$

subject to the boundary condition

$$\mathcal{B}(y, x; u) = h(y, x), \qquad (y, x) \in \Gamma \times \partial\Omega. \tag{2.4}$$

The numerical methods we present in this thesis need the resolution of an amount of deterministic problems, i.e., for a fixed set of realizations of Y, $\{y^1, \ldots, y^M\}$. In

the following, we assume it is possible to reformulate the deterministic problem in a weak formulation, as in Section 1.1.1. Doing so, we obtain the Ω-weak/Γ-strong formulation, which reads: for every $y \in \Gamma$, find $u(y) \in \mathbb{V}_y$ such that

$$a(u(y), v; y) = f(v; y), \qquad \forall\, v \in \mathbb{V}_y, \tag{2.5}$$

where $\mathbb{V}_y$ is a suitable functional space over Ω, $a(\cdot, \cdot; y) : \mathbb{V}_y \times \mathbb{V}_y \to \mathbb{R}$ is a bilinear form and $f(\cdot; y) \in \mathbb{V}_y'$. Moreover we also assume that the space $\mathbb{V}_y$ does not actually depend on y, i.e., $\mathbb{V}_y = \mathbb{V}$. A further assumption we make is that a and f admits an affine decomposition:

$$a(\cdot, \cdot; y) = a_0(\cdot, \cdot) + \sum_{k=1}^{N_a} y_k a_k(\cdot, \cdot) \qquad \text{and} \qquad f(\cdot; y) = f_0(\cdot; y) + \sum_{k=N_a+1}^{N_a+N_f} y_k f_k(\cdot),$$

where $N = N_a + N_f$, $a_k : \mathbb{V} \times \mathbb{V} \to \mathbb{R}$ is a bilinear form, $k = 0, \dots, N_a$, and $f_k \in \mathbb{V}'$, for $k = 0, N_a + 1, \dots, N$. Moreover we assume that (1.3) and (1.4) still hold and that the bilinear form $a(\cdot, \cdot; y)$ is symmetric for every $y \in \Gamma$. More often, we are not interested in the solution $u(y)$ itself but on a functional $s(u(y); y)$ of the solution as model output, e.g., the compliant output $s(u(y); y) = f(u(y); y)$, as well its statistics, e.g., the expectations

$$\mathbb{E}[s] = \int_\Gamma s(u(y); y)\rho(y)\, dy \qquad \text{or} \qquad \mathbb{E}[u] = \int_\Gamma u(y)\rho(y)\, dy.$$

Example 2.1.1 (Linear stochastic elliptic problem)**.** *Consider the following problem: find $u : \Omega \times \Gamma \to \mathbb{R}$ such that*

$$\begin{aligned} -\nabla \cdot (b(x, y)\nabla u(x, y)) &= g(x, y) \qquad \forall\, (x, y) \in \Omega \times \Gamma, \\ u(x, y) &= 0 \qquad \forall\, (x, y) \in \partial\Omega \times \Gamma, \end{aligned} \tag{2.6}$$

where the divergence $\nabla\cdot$ and the gradient ∇ are taken with respect to x. For the random forcing term g and the diffusivity b, we consider the following assumptions:

(i) the random diffusivity b is of the form

$$b(x, y) = b_0(x) + \sum_{k=1}^{N_a} y_k b_k(x),$$

with $b_k \in L^\infty(\Omega)$, for $k = 0, \dots, N_a$;

(ii) the random forcing term f is of the form

$$g(x, y) = g_0(x) + \sum_{k=N_a+1}^{N} y_k g_k(x),$$

with $g_k \in L^2(\Omega)$, for $k = 0, N_a + 1, \dots, N$.

The Ω-weak/Γ-strong formulation of this problem reads: for every $y \in \Gamma$ find $u(y) \in H_0^1(\Omega)$ such that

$$a(u(y), v; y) = f(v; y) \qquad \forall v \in H_0^1(\Omega), \tag{2.7}$$

where the bilinear and linear forms are given by

$$a(u, v; y) = \int_\Omega b_0(x)(\nabla u \cdot \nabla v)\, dx + \sum_{k=1}^{N_a} y_k \int_\Omega b_k(x)(\nabla u \cdot \nabla v)\, dx,$$

and

$$f(v; y) = \int_\Omega g_0(x) v\, dx + \sum_{k=N_a+1}^{N} y_k \int_\Omega g_k(x) v\, dx.$$

2.2 Weighted reduced order methods

A typical way to solve numerically stochastic differential equations is to use a Monte-Carlo simulation. The procedure of applying a Monte-Carlo method to problem (2.3)-(2.4) takes the following steps:

1. for a prescribed number of realizations M, generate independent and identically distributed (i.i.d.) random variables $\{Y_j^i\}_{i=1}^N = \{Y^i(w_j)\}_{i=1}^N$, for $j = 1, \ldots, M$, with density distribution ρ;

2. for each $j = 1, \ldots, M$, solve a deterministic problem (2.3)-(2.4) with $y = y^j = (Y_j^1, \ldots, Y_j^N)$ and obtain solution $u_j = u(y^j)$;

3. Postprocess the results to evaluate the solution statistics, e.g.,

$$\mathbb{E}[u] \simeq \langle u \rangle = \frac{1}{M} \sum_{j=1}^{M} u_j \qquad \text{or} \qquad \mathbb{E}[s] \simeq \langle s(u) \rangle = \frac{1}{M} \sum_{j=1}^{M} s(u_j; y_j)$$

   for some suitable output function s.

Although the convergence rate it is formally independent of the dimension of the random space, the convergence rate of a Monte Carlo method is relatively slow (typically $1/\sqrt{M}$). Thus, one requires to solve a large amount of deterministic problems to obtain a desired accuracy, which implies a very high computational cost. In this framework reduced order methods turn out to be very useful in order to reduce the computational cost, at cost of a (possibly) small additional error. In the following sections we present a way to modify the previously presented reduced order algorithm in order to take into account the stochastic distribution of the parameters. We will refer to these two modified methods as weighted greedy algorithm and weighted POD. From now on we consider problems formulated as in (2.5) with all the previously made assumptions.

2.2.1 Weighted greedy algorithm

In this section we present an adaption to stochastic problems of the greedy algorithm presented in the previous chapter. This technique has been originally developed by P. Chen et al. (cf. [8–11]). The basic idea of this weighted method is to assign different weights in the construction of reduced basis space at different values of parameter $y \in \Gamma$ according to a prescribed weight function $w(y)$, where $w : \Gamma \to \mathbb{R}^+$ is a weight function. The weighted greedy algorithm consists of the same elements presented in chapter Section 1.3. Thus, here we use the previous introduced notation and we only highlight the new weighted scheme.
For sake of simplicity, we omit the indexes δ,N_δ and we act like our original problem (1.1) coincides with the truth problem (1.14). The greedy scheme is based on the availability of an error estimator $\eta(y)$ such that

$$\|u(y) - u_N(y)\|_y \leq \eta(y)$$

(note that this formula is just (1.30; now the parameters are the possible realizations of a random variable) where we replaced μ with y). Thus, the idea is to modify $\eta(y)$, multiplying it by a weight $w(y)$, chosen accordingly to the distribution of $y \in \Gamma$. Let us introduce, for $y \in \Gamma$, the spaces $\mathbb{V}_y = \mathbb{V}$ with the norm

$$\|v\|_{\mathbb{V}_y} = w(y)\|v\|_{\mathbb{V}}, \qquad \forall\, v \in \mathbb{V}_y.$$

Thus, if we define $\widehat{\eta}(y) = \eta(y)w(y)$, we have that

$$\alpha\|u(y) - u_N(y)\|_{\mathbb{V}_y} \leq w(y)\|u(y) - u_N(y)\|_y \leq \widehat{\eta}(y),$$

for all $y \in \Gamma$. So, except of a positive constant α, a greedy routine with error estimator $\widehat{\eta}$ aims to minimize the distance of the solution manifold from the reduced basis space in a weighted L^∞-norm on the parameter space. Now, depending on what one wants to compute, different choices can be made for the weight function w. For example, if we are interested in a statistics of the solution, e.g., $\mathbb{E}[u]$, we can choose $w(y) = \rho(y)$. Thus, for the error committed computing the expected value using the reduced basis, the following estimates holds:

$$\big\|\mathbb{E}[u] - \mathbb{E}[u_N]\big\|_{\mathbb{V}} \leq \int_\Gamma \|u(y) - u_N(y)\|_{\mathbb{V}}\rho(y)\,dy \leq \frac{1}{\alpha}|\Gamma| \sup_{y\in\Gamma} \widehat{\eta}(y).$$

If we are interested in the expectance of a linear output s, $\mathbb{E}[s]$, using the same weight function, we get the error estimate:

$$\big|\mathbb{E}[s(u)] - \mathbb{E}[s(u_N)]\big| \leq \int_\Gamma \|s\|_{\mathbb{V}'}\|u(y) - u_N(y)\|_{\mathbb{V}}\rho(y)\,dy \leq \frac{1}{\alpha}|\Gamma|\|s\|_{\mathbb{V}'} \sup_{y\in\Gamma} \widehat{\eta}(y).$$

Instead, taking $w(y) = \sqrt{\rho(y)}$ we obtain that

$$\int_\Gamma \|u(y) - u_N(y)\|^2_{\mathbb{V}}\rho(y)\,dy = \int_\Gamma \|u(y) - u_N(y)\|^2_{\mathbb{V}_y}\,dy \leq \frac{|\Gamma|}{\alpha} \sup_{y\in\Gamma} \widehat{\eta}(y)^2.$$

so that we obtain the following estimate for the quadratic error:

$$\mathbb{E}\Big[\|u(Y) - u_N(Y)\|^2_{\mathbb{V}}\Big]^{1/2} \leq \sqrt{\frac{|\Gamma|}{\alpha}} \sup_{y\in\Gamma} \widehat{\eta}(y). \tag{2.8}$$

2.2.2 Weighted POD

In this section we present a possible way to modify also the POD algorithm to keep into account the probability distribution of the parameters. This is an original contribution of this work. We recall that, chosen a discretization $\Xi = \{y_1, \dots, y_M\}$ of the parameter space Γ, the POD algorithm provides the N-dimensional linear subspace of $\mathbb{V}$ which minimizes the quantity

$$\frac{1}{M}\sum_{i=1}^{M}\|u(y_i) - u_N(y_i)\|_{\mathbb{V}}^2.$$

The first approach, to keep into account the probability distribution of y, is to consider, as before, the $\mathbb{V}_y$-norm instead of the $\mathbb{V}$-norm. Thus, we look for the N-dimensional subspace of $\mathbb{V}$ which minimize the quantity

$$\frac{1}{M}\sum_{i=1}^{M} w(y_i)\|u(y_i) - u_N(y_i)\|_{\mathbb{V}}^2, \tag{2.9}$$

for a given weight function $w : \Xi \to \mathbb{R}^+$. For example, we can choose $w(y) = \rho(y)$. Following the same procedure of section 1.2 one can show that this is equivalent to find the N greatest eigenvalues $\lambda_1 \geq \cdots \geq \lambda_N$, and corresponding eigenvectors $\xi_1, \dots, \xi_N$, of the linear application $C : \mathbb{V} \to \mathbb{V}$ defined as

$$C(v) = \frac{1}{M}\sum_{i=1}^{M} w(y_i)(v, u(y_i))_{\mathbb{V}} u(y_i)$$

The sought best approximation (in sense of minimal quantity (2.9)) N-dimensional subspace is hence given by $\mathbb{V}_N = \text{span}\{\xi_1, \dots, \xi_N\}$. Computationally, this can be achieved computing the N maximum N eigenvalues $\lambda_1 \geq \cdots \geq \lambda_N$, and respective eigenvectors $v_1, \dots, v_N$, of the preconditioned matrix

$$\widehat{C} = P \cdot C,$$

where C is the matrix defined in (1.25) and

$$P = \begin{pmatrix} w_1 & & \\ & \ddots & \\ & & w_M \end{pmatrix}. \tag{2.10}$$

Thus, the basis function are given by $\xi_n = \sum_{i=1}^{M}(v_n)_i \varphi_i$, for $n = 1, \dots, N$. We note that the matrix $\widehat{C}$ is not symmetric in the usual sense, but respect to the scalar product induced by the matrix C, i.e., it holds the relation $\widehat{C}^T C = C\widehat{C}$. Thus, spectral theorem still holds and there exists an orthonormal basis of eigenvectors, i.e., $\widehat{C}$ is diagonalizable with an orthogonal change of basis matrix. The discretized parameter space Ξ can be selected with a sampling technique, e.g., using an equispaced tensor product grid on Γ or taking M realizations of a uniform distribution on Γ. Note that if we build Ξ as the set of M realizations of a

random variable on Γ, with distribution ρ, and we put $w \equiv 1$, the quantity (2.9) we minimize is just a Monte Carlo approximation of the integral

$$\int_\Gamma \|u(y) - u_N(y)\|_{\mathbb{V}}^2 \rho(y)\, dy = \mathbb{E}[\|u - u_N\|_{\mathbb{V}}^2]. \tag{2.11}$$

Therefore, in this case, the difference between the weighted and the standard POD algorithm is (at least theoretically) we replaced quantity (1.20) with (2.11), which, since we are considering the stochastic problem (2.5), could be reasonable. Following this reasoning, the idea is to select Ξ and w as the nodes and the weights of a quadrature algorithm that approximate the integral (2.11). We consider a quadrature operator $\mathcal{U}$, defined as

$$\mathcal{U}(f) = \sum_{i=1}^{M} \omega_i f(x_i)$$

for every integrable function $f : \Gamma \to \mathbb{R}$, where $x_1, \ldots, x_M \in \Gamma$ are the nodes of the algorithm and $\omega_1, \ldots, \omega_M$ the respective weights. Then we can approximate (2.11) as

$$\int_\Gamma \|u(y) - u_N(y)\|_{\mathbb{V}}^2 \rho(y)\, dy \simeq \sum_{i=1}^{M} w_i \|u(x_i) - u_N(x_i)\|_{\mathbb{V}}^2, \tag{2.12}$$

where $w_i = \omega_i \rho(y_i)$ or, if $\mathcal{U}$ is a quadrature rule for integration with respect to weight ρ, $w_i = \omega_i$. Thus we can look for the N-dimensional linear subspace which realizes the minimum of the right hand side of (2.12). This can be found by following the above described procedure with preconditioning matrix (2.10) where $w_i = \omega_i \rho(y_i)$ or, if $\mathcal{U}$ is a quadrature rule for integration with respect to weight ρ, $w_i = \omega_i$. Therefore, varying the quadrature rule $\mathcal{U}$ used, one obtain diverse ways of preconditioning the matrix C. Unfortunately, the main problem with POD algorithm is that there is not an a posteriori error estimator. This does not allow us to say a priori which weighted variant is the best. Moreover, since it requires to compute the eigenvalues of a $M \times M$ matrix, the dimension M of Ξ should better not be too high. In Chapter 3 we will describe a method that allows to build multivariate quadrature formulas with a reduced number of nodes with respect to tensor product quadrature rules.

2.3 Numerical tests

In this section we test the algorithms described above on a specific problem. We consider the linear stochastic elliptic equation (2.7) on $\Omega = [0, 1]^2$.

2.3.1 The `RBniCS` library

To solve the following problems we used the `RBniCS` library (cf. [2]). `RBniCS` is an implementation in `FEniCS` of several reduced order modeling techniques for coercive problems. We modified the present algorithms in order to treat stochastic problems and apply weighted reduced order methods. All the scripts were written in `Python`.

2.3.2 First study case: 4 diffusivity zones equation

Let $\Omega = \cup_{i=1}^4 \Omega_i$ be a decomposition of $\Omega = [0,1]^2$ such that

$$\Omega_1 = [0,1/2] \times [0,1/2], \quad \Omega_2 = [1/2,1] \times [0,1/2],$$
$$\Omega_3 = [0,1/2] \times [1/2,1], \quad \Omega_4 = [1/2,1] \times [1/2,1].$$

We consider equation (1.11) with $g \equiv 1$ and b of the form

$$b(x;y) = \sum_{i=1}^{4} y_i \mathbb{1}_{\Omega_i}(x), \qquad \text{for } x \in \Omega,$$

for some fixed $y = (y_1, \dots, y_4) \in (0,\infty)^4$. In practice we are considering a diffusion

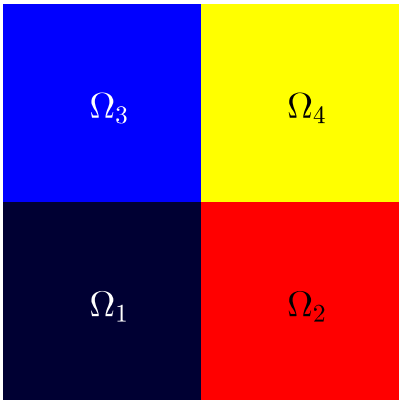

Figure 2.1: Geometrical set-up of problem (2.13).

problem on Ω, where the diffusion coefficient is constant on the elements of a partition of Ω in four zones. In particular we want to consider the following stochastic problem. Let $\Gamma = \prod_{i=1}^4 \Gamma_i = \prod_{i=1}^4 [a_i, b_i] \subseteq (0,+\infty)^4$ and $Y = (Y_1, \dots, Y_4)$ be a random vector taking values in Γ. Moreover we suppose that $Y_1, \dots, Y_4$ are independent and that, for $i = 1, \dots, 4$,

$$\frac{Y_i - a_i}{b_i - a_i} \sim \text{Beta}(\alpha_i, \beta_i),$$

for some positive distribution parameters α_i, β_i. We consider the problem: find $u : \Gamma \to H_0^1(\Omega)$ such that, a.s.,

$$a(u(Y), v; Y) = f(v) \qquad \forall\, v \in H_0^1(\Omega), \tag{2.13}$$

where $a(\cdot,\cdot;y)$ denotes the bilinear form (1.10) with $b = b(\cdot;y)$. We implemented weighted greedy algorithm and weighted POD for the evaluation of the expectation of the solution, and we compared them. In particular we calculated the error

$$\mathbb{E}\left[\|u_{N_\delta}(Y) - u_N(Y)\|^2_{H_0^1(\Omega)}\right] = \int_\Gamma \|u_{N_\delta}(y) - u_N(y)\|^2_{H_0^1(\Omega)} \rho(y)\, dy \tag{2.14}$$

using a Monte Carlo approximations, i.e.,

$$\mathbb{E}\left[\|u_{N_\delta}(Y) - u_N(Y)\|^2_{H^1_0(\Omega)}\right] \simeq \frac{1}{M}\sum_{m=1}^{M}\|u_{N_\delta}(Y^m) - u_N(Y^m)\|^2_{H^1_0(\Omega)}, \qquad (2.15)$$

where $Y^1, \dots, Y^M$ are M realizations of Y. In particular we tested the case with $\Gamma_i = [1, 3]$ and $(\alpha_i, \beta_i) = (10, 10)$, for all $i = 1, \dots, 4$. As discretized space $\mathbb{V}_\delta$ we used the classical $\mathbb{P}^1$-FE approximation space.

Greedy algorithm

We implemented the standard and the weighted greedy algorithm for construction of reduced basis space, taking $\omega = \sqrt{\rho}$ in the weighted case (this choice being motivated by (2.8)). For the train set selection, we sampled it using various techniques as uniform sampling, equispaced grid and sampling of the distribution. We also tried to use other samples techniques as Clenshaw-Curtis or Gauss-Legendre tensor product grids (see appendix). We took $|\Xi| = 1000$ and we chose the first parameter μ_1 as $\mu_1^i = 2$, for $i = 1, \dots, 4$ (2 is the mode of the distribution of Y). We found that the best accuracy is achieved using the weighted algorithm with a sampling of the distribution of Y. In Figure 2.2, we reported the graph of

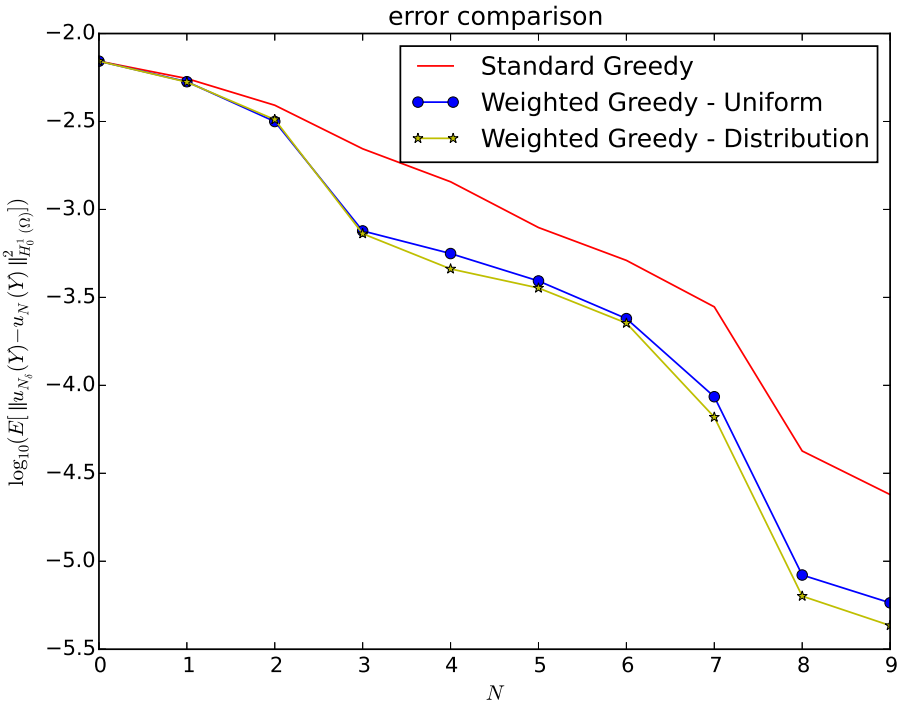

Figure 2.2: Comparison of error (2.14) using a standard greedy algorithm and a weighted greedy algorithm with a uniform sampling or a sampling of Y.

the error (2.14) (in a logarithmic scale) as a function of the reduced basis space

dimension N, using a standard greedy algorithm (with an uniform sampling) and weighted greedy algorithms with a uniform sampling or a sampling of Y. As we can see, using the weighted version instead of the standard one we earn almost one order of accuracy, for $N = 9$. The weighted greedy with sampling of the distribution seems to perform a little better than the weighted greedy with uniform sampling. This should be due to the fact to a more 'representative' choice of the samples. However the difference is almost null.

POD method

We implemented several versions of weighted POD algorithm and we compared their performances. As explained above, weighted versions of POD algorithm are based on quadrature formulas. As before, let us denote with $\Xi = \{y_1, \dots, y_M\}$ the sampling point (which correspond to the nodes of the chosen quadrature formula), $w = (w_1, \dots, w_M)$ the correspondent weights used in POD algorithm, $\rho_1, \dots, \rho_M$ the values of the density ρ in the points of Γ_h and $\omega = (\omega_1, \dots, \omega_M)$ the weights of the chosen quadrature formula. A description of the quadrature formula used can be found in Appendix A.1. We chose to use the versions that minimize the following approximations of (2.11):

(i) Uniform Monte-Carlo approximation:

$$\frac{1}{M}\sum_{i=1}^{M}\rho_i\|u_{N_\delta}(Y^i) - u_N(Y^i)\|_{\mathbb{V}}^2, \tag{2.16}$$

where $Y^1, \dots, Y^M$ are M realizations of a random variable with uniform distribution on Γ and ρ_i are the values of the density ρ in these points. Therefore in this case $\Xi = \{Y^1, \dots, Y^M\}$ and $w_i = \rho_i$, for all $i = 1, \dots, M$;

(ii) Monte-Carlo approximation:

$$\frac{1}{M}\sum_{i=1}^{M}\|u_{N_\delta}(Y^i) - u_N(Y^i)\|_{\mathbb{V}}^2, \tag{2.17}$$

where $Y^1, \dots, Y^M$ are M realizations of Y. In this case $\Xi = \{Y^1, \dots, Y^M\}$ and $w_i = 1$, for all $i = 1, \dots, M$;

(iii) Tensor product Clenshaw-Curtis rule approximation:

$$\frac{1}{M}\sum_{i=1}^{M}\rho_i\omega_i\|u_{N_\delta}(y^i) - u_N(y^i)\|_{\mathbb{V}}^2, \tag{2.18}$$

where $y^1, \dots, y^M$ are the nodes of the tensor product Clenshaw-Curtis quadrature rule and $\omega_1, \dots, \omega_M$ are the respective quadrature weights. In this case $\Xi = \{y^1, \dots, y^M\}$ and $w_i = \rho_i\omega_i$, for all $i = 1, \dots, M$. We do the same using also the tensor product Gauss-Legendre quadrature rule (which gives a different choice of weights and nodes);

(iv) Tensor product Gauss-Jacobi rule approximation:

$$\frac{1}{M}\sum_{i=1}^{M}\omega_i\|u_{N_\delta}(y^i)-u_N(y^i)\|_{\mathbb{V}}^2, \tag{2.19}$$

where $y^1,\dots,y^M$ are the nodes of the tensor product Gauss-Jacobi quadrature rule and $\omega_1,\dots,\omega_M$ are the respective quadrature weights. We choose parameters of the Gauss-Jacobi formula to be (α_i,β_i) in each dimension, accordingly to the distribution of Y. In this case $\Xi=\{y^1,\dots,y^M\}$ and $w_i=\omega_i$, for all $i=1,\dots,M$.

	w_i	$\Xi^1\|$	$\|\Xi^1\setminus\partial\Gamma\|$	$\|\Xi^2\|$	$\|\Xi^2\setminus\partial\Gamma\|$
Standard	1	100	100	500	500
Uniform Monte-Carlo	ρ_i	100	100	500	500
Monte-Carlo	1	100	100	500	500
Clenshaw-Curtis	$\omega_i\rho_i$	1296	256	2401	625
Gauss-Legendre	$\omega_i\rho_i$	256	256	625	625
Gauss-Jacobi$(10,10)$	ω_i	256	256	625	625

Table 2.1: Description of the weights used in the weighted POD algorithms and of the number of points of Ξin the two different trials.

From now on we will refer to POD algorithms coming from previous formulas as Uniform Monte-Carlo, Monte-Carlo, Clenshaw-Curtis (Gauss-Legendre) and Gauss-Jacobi, respectively for formulas (2.16), (2.17), (2.18) and (2.19). We report here two different trials made for different values of M. In the first trial we selected $\Xi=\Xi^1$ to be the smallest possible such that $|\Xi^1\setminus\partial\Gamma|\geq 100$ for the various algorithms (when we are using tensor product quadrature rule, we can not impose the cardinality of Ξ a priori). Instead, in the second trial we selected $\Xi=\Xi^2$ to be the smallest possible such that $|\Xi^2\setminus\partial\Gamma|\geq 500$ for the various algorithms We also note that when we use the Clenshaw-Curtis approximation, the majority of the points in Γ_h lies on $\partial\Gamma$: these point are completely negligible, since $\rho|_{\partial\Gamma}\equiv 0$. So in this case, we need to take a bigger M to get a good accuracy. The number of sampling points used for every algorithm in the two trials is reported in Table 2.1. In Figure 2.3 we confronted a standard POD algorithm (not weighted, with sampling uniformly distributed) with uniform Monte-Carlo and Monte-Carlo POD algorithms. As we can see (on the left), weighted algorithms perform better (up to an order of accuracy) than the standard one and the best accuracy is obtained with the Monte-Carlo POD algorithm. For bigger values of M, the accuracy improves both for standard and Uniform Monte-Carlo POD algorithms, while remains almost the same for the Monte-Carlo POD. In particular, in this case, the two Monte-Carlo POD algorithms almost show the same accuracy. For further trials (with bigger M) the results show hardly any change. Therefore, in this case the conclusion is that the better algorithm to be

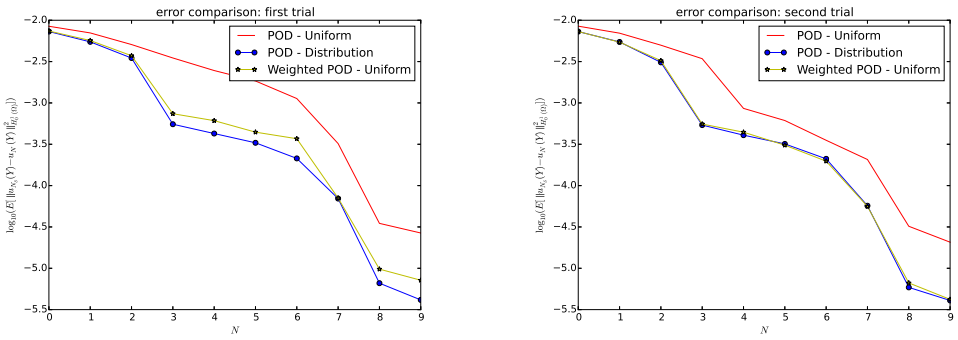

Figure 2.3: Comparison for the error (2.14) obtained for two different values of M, using standard, uniform Monte-Carlo and Monte-Carlo POD.

used seems to be Monte-Carlo POD.

In Figure 2.4 we reported instead the performance for Clenshaw-Curtis, Gauss-

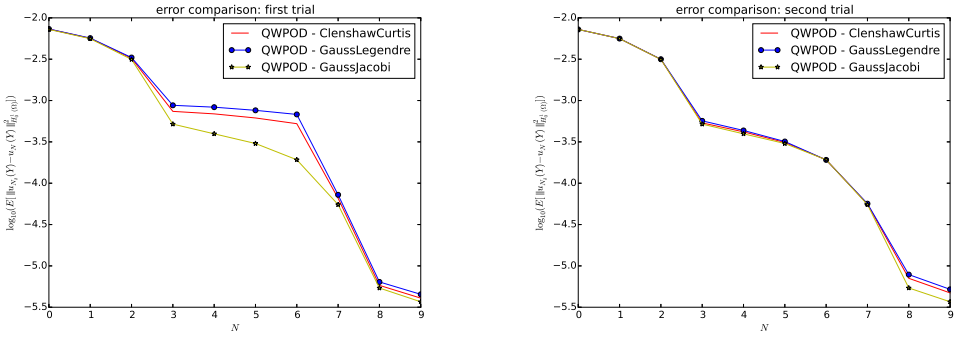

Figure 2.4: Comparison for the error (2.14) obtained for two different values of M, using Clenshaw-Curtis, Gauss-Legendre and Gauss-Jacobi (of same parameters of distribution of Y) POD.

Legendre and Gauss-Jacobi (with the same parameter of distribution of Y) POD algorithms, in the two cases (different values of M). We note (from Table 2.1) that for Clenshaw-Curtis POD we have to take $M = 1296$ and $M = 2401$ to ensure the same number of points in $\mathring{\Gamma}$ as in other POD algorithms. As we can see Clenshaw-Curtis and Gauss-Legendre POD algorithms bring almost the same accuracy, while Gauss-Jacobi POD seems to work a little bit better (which makes sense, since Gauss-Jacobi quadrature rule is designed exactly for integrals with Beta weight). However, in the second trial, Gauss-Jacobi POD shows hardly any change while the other two improve their accuracy almost up to the one provided by Gauss-Jacobi POD. For further trials (with bigger M) the results show hardly any change. Thus, in this case the algorithm with best performance is the Gauss-Jacobi one. Finally, in Figure 2.5 we compared the standard POD algorithm with the two previous best weighted POD algorithm: Monte-Carlo and Gauss-Jacobi. In particular we used $M = 100, 100, 256$, respectively. As we can see the latter two show almost the same order of accuracy, which is improved, with respect the

standard one, up to an order of accuracy.

Greedy vs POD: a comparison

In Figure 2.6 we compare standard and weighted greedy algorithm performances with the ones of standard and Gauss-Jacobi (which was the best weighted one) POD algorithm. We used $M = 1000$ for the greedy algorithms and $M = 100,256$ for the POD algorithms. As noted before, the weighted algorithms perform better, up to an order of accuracy. The weighted POD works a little bit better than the weighted greedy, which makes sense, since the POD is designed to minimize (in some sense) the quantity (2.14); however, the difference is practically null.

2.3.3 Second study case: 9 diffusivity zones equation

We consider the same problem as before, but subdividing Ω in 9 sub-domains. So we consider $\Omega = \cup_{i=1}^{9}\Omega_i$ such that

$$\begin{array}{ccc} \Omega_1 = [0,1/3]\times[0,1/3], & \Omega_2 = [1/3,2/3]\times[0,1/3], & \Omega_3 = [2/3,1]\times[0,1/3], \\ \Omega_4 = [0,1/3]\times[1/3,2/3], & \Omega_5 = [1/3,2/3]\times[1/3,2/3], & \Omega_6 = [2/3,1]\times[1/3,2/3], \\ \Omega_7 = [0,1/3]\times[2/3,1], & \Omega_8 = [1/3,2/3]\times[2/3,1], & \Omega_9 = [2/3,1]\times[2/3,1], \end{array}$$

and equation (1.11) with $g \equiv 1$ and b of the form

$$b(x;y) = \sum_{i=1}^{9} y_i \mathbb{1}_{\Omega_i}(x), \qquad \text{for } x \in \Omega,$$

for some fixed $y = (y_1,\dots,y_9) \in (0,\infty)^9$. As before, we consider the stochastic problem: find $u : \Gamma \to H_0^1(\Omega)$ such that, a.e.,

$$a(u(Y), v; Y) = f(v) \qquad \forall\, v \in H_0^1(\Omega), \tag{2.20}$$

where $a(\cdot,\cdot;y)$ denotes the bilinear form (1.10) with $b = b(\cdot;y)$ and $Y = (Y_1,\dots,Y_9)$ is a random vector taking values in $\Gamma = \prod_{i=1}^{9}\Gamma_i = \prod_{i=1}^{9}[a_i,b_i]$. Analogously, we suppose that $\Gamma_i = [1,3]$ and Y_i are i.i.d. random numbers such that

$$\frac{Y_i - 1}{2} \sim \mathrm{Beta}(\alpha_i,\beta_i), \tag{2.21}$$

for $i = 1,\dots,9$. Again we implemented diverse versions of weighted greedy and POD algorithms for the construction of reduced order basis spaces, taking $\mathbb{V}_\delta$ as the classical $\mathbb{P}^1$-FE approximation space. We hence compared them, confronting the values of error (2.15), as a function of the reduced order basis space dimension N.

As one can observe, the only difference with the problem of section (2.3.2) is practically just the higher number of parameters. We made two different tests for different parameter values: the first one with $\alpha_i = \beta_i = 10$, for $i = 1,\dots,9$, the second one with $\alpha_i = \beta_i = 75$, for $i = 1,\dots,9$.

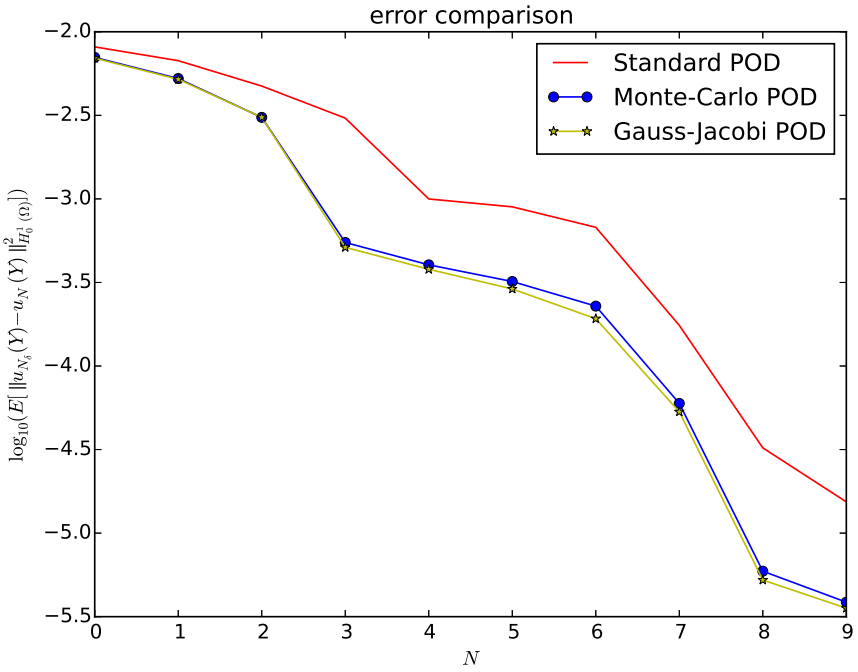

Figure 2.5: Comparison of error (2.14) obtained using standard, Monte-Carlo and Gauss-Jacobi POD algorithms with, respectively, $M = 100, 100, 256$.

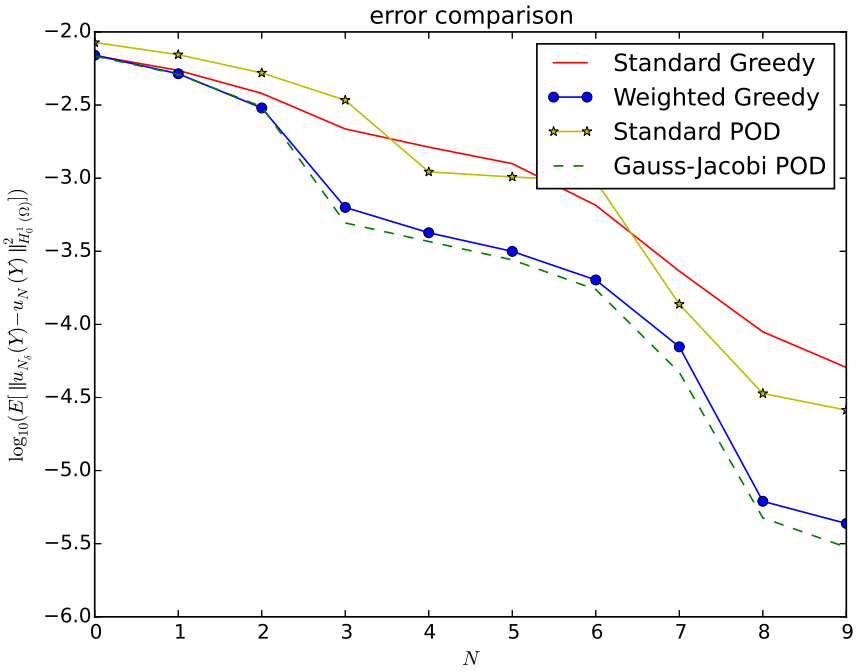

Figure 2.6: Comparison of error (2.14) obtained using standard Greedy and weighted greedy and standard and Gauss-Jacobi POD algorithms with, respectively, $M = 1000, 1000, 100, 256$.

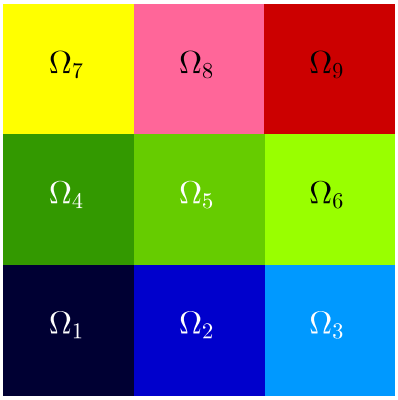

Figure 2.7: Geometrical set-up of problem (2.20).

Greedy algorithm

We implemented the standard and the weighted greedy algorithm for construction of reduced basis space, taking $\omega = \sqrt{\rho}$ in the weighted case as before. Again, for the train set selection, we sampled it using various techniques. We took $|\Xi| = 2000$ and we chose the first parameter μ_1 as $\mu_1^i = 2$, for $i = 1, \dots, 9$ (2 is the mode of the distribution of Y). The best accuracy is still achieved using the weighted algorithm with a sampling of the distribution of Y. In Figure 2.8, we reported the graph of the error (2.14) (in a logarithmic scale) as a function of the reduced basis space dimension N, using a standard greedy algorithm (with an uniform sampling) and weighted greedy algorithms with a uniform sampling or a sampling of Y for $\alpha_i = \beta_i = 10$ (left) and $\alpha_i = \beta_i = 75$ (right). In the first case we observe that the weighted algorithm with sampling of Y performs better than the standard one. However, the weighted algorithm with uniform sampling does not show a much better accuracy, providing an accuracy hardly different from that of the standard POD. This implies that with an higher dimension of the parameter space the sampling technique assumes much more importance. This is strongly underlined in the case $\alpha_i = \beta_i = 75$. Here, the distribution of Y is much more concentrated and the difference between the standard greedy and the weighted one with sampling of Y is more evident. Moreover, the performance of the weighted algorithm with uniform sampling gets much worst than the standard one: this is clearly due the fact that, having a bad sampling, the weighted greedy algorithm forces us to take points $\mu_1, \dots, \mu_N$ for which the respective solutions are almost linearly dependent. This is highlighted by the fact that for $N \geq 8$ the reduced matrix generated by the greedy algorithm becomes singular.

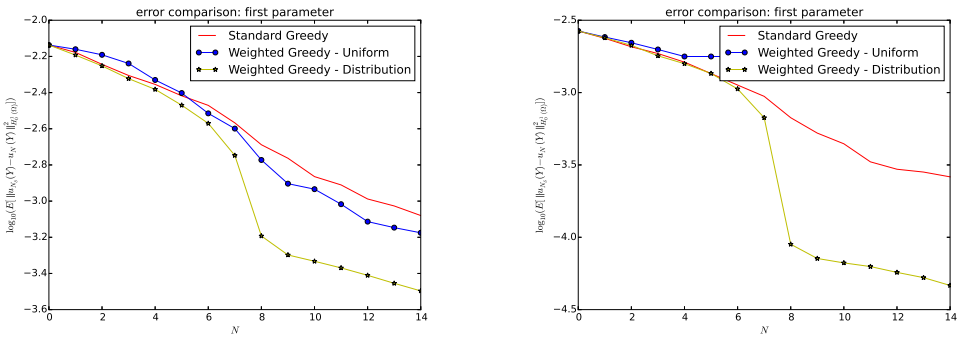

Figure 2.8: Comparison for the error (2.14) obtained for $\alpha_i = \beta_i = 10$ (left) and $\alpha_i = \beta_i = 75$ (right), using standard greedy algorithm and weighted greedy algorithms with uniform sampling or sampling of Y.

POD method

We implemented the various weighted versions of POD algorithm and compared the results obtained in the cases $\alpha_i = \beta_i = 10$ and $\alpha_i = \beta_i = 75$. In Table 2.2 we reported the weights and the number of sample points used in the various algorithms. Since Gauss-Legendre and Gauss-Jacobi POD algorithms are based

	w_i	$\lvert\Xi\rvert$
Standard	1	500
Uniform Monte-Carlo	ρ_i	2000
Monte-Carlo	1	500
Gauss-Legendre	$\omega_i\rho_i$	512
Gauss-Jacobi	ω_i	512

Table 2.2: Description of the weights used in the weighted POD algorithms and of the number of points of Ξin the two different trials.

on a tensor product rule the only possibility was to take $M = 2^9 = 512$. Indeed the next possible choice is $M = 3^9 = 19683$, which is computationally impracticable. We did not tested Clenshaw-Curtis POD, since $|\Xi \cap \mathring{\Gamma}| = 0$ for $M = 2^9$ and $|\Xi \cap \mathring{\Gamma}| = 1$ for $M = 3^9$, so that for having an enough representative set Ξ we would need $M \geq 4^9$, which is clearly impracticable. Figure 2.9 shows that the weighted algorithms perform better of the standard one, even if Monte-Carlo POD outperforms uniform Monte-Carlo POD. The situation is highlighted in the case $\alpha_i = \beta_i = 75$, where the distribution of Y is much more concentrated. Moreover, for obtaining good results for uniform Monte-Carlo POD we had to take $M = 2000$ while we run Monte-Carlo POD with $M = 500$ (for bigger M the performance does not change). The difference between the two weighted algorithms can be addressed to the fact that rule (2.17) is a better approximation of (2.14) than (2.16) and to the more representative choice of the points in Ξ. Therefore, it seems that, in the case of high dimensional parameter space, the choice of the sampling points plays

a key role in the POD algorithm, as in the greedy. In Figure 2.10 we reported

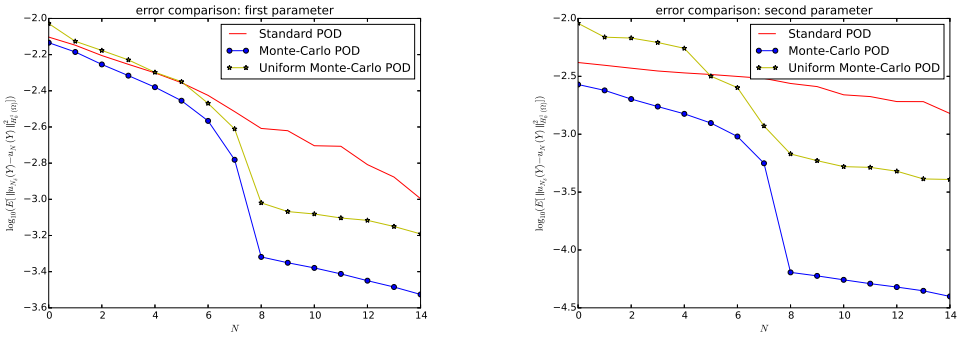

Figure 2.9: Comparison for the error (2.14) obtained for $\alpha_i = \beta_i = 10$ (left) and $\alpha_i = \beta_i = 75$ (right), using standard, uniform Monte-Carlo and Monte-Carlo POD.

the performances of Gauss-Legendre and Gauss-Jacobi POD algorithms. As in the case considered in section 2.3.2 Gauss-Jacobi POD shows performances almost equal to the Monte-Carlo POD ones. However this time Gauss-Legendre performs very poorly. This is due to a bad selection of the sampling points Ξ by the Gauss-Legendre algorithm: from $N \geq 8$, for both parameter values, the accuracy shows hardly any change for higher N. Finally, in Figure 2.11 we confronted the two

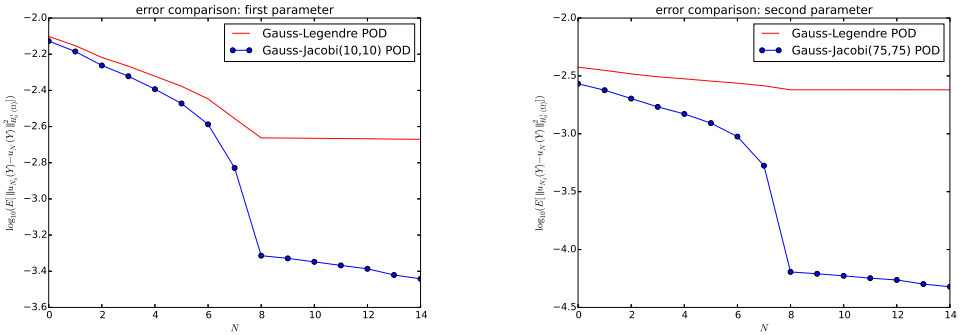

Figure 2.10: Comparison for the error (2.14) obtained for $\alpha_i = \beta_i = 10$ (left) and $\alpha_i = \beta_i = 75$ (right), using Gauss-Legendre and Gauss-Jacobi (of same parameters of distribution of Y) POD.

well working weighted POD algorithms (Monte-Carlo and Gauss-Jacobi) with the standard one. In both the cases a better accuracy is obtained with the weighted algorithms. The difference is much more visible in the case $\alpha_i = \beta_i = 75$, when the distribution is highly concentrated.

Greedy vs POD: a comparison

We report in Figure 2.12 the comparisons of the errors obtained using standard and weighted reduced order approximation. We selected, both for POD and greedy algorithms, the weighted versions with the best performance. We observe

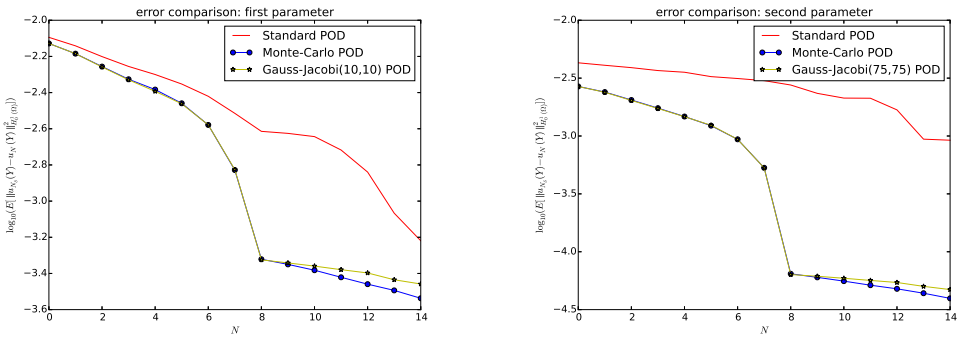

Figure 2.11: Comparison for the error (2.14) obtained for $\alpha_i = \beta_i = 10$ (left) and $\alpha_i = \beta_i = 75$ (right), using standard, Monte-Carlo and Gauss-Jacobi (of same parameters of distribution of Y) POD.

that in both cases a better accuracy is obtained with a weighted algorithm instead of a standard one. The differences between weighted and standard algorithms are much more evident when the distribution is highly concentrated.

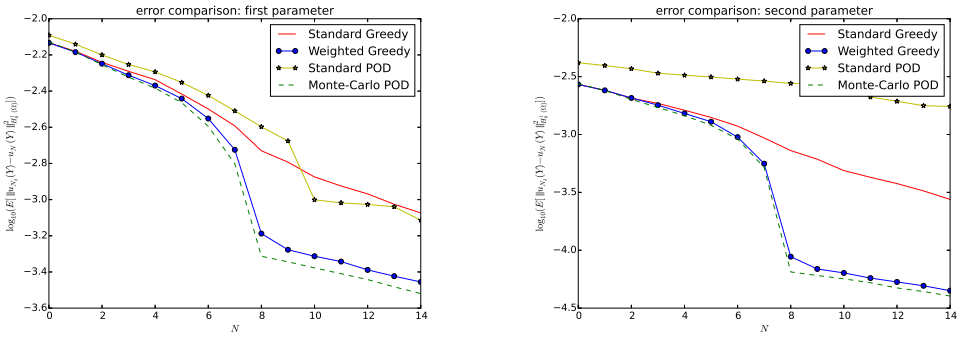

Figure 2.12: Comparison of error (2.14) obtained using standard Greedy and weighted Greedy and standard and Monte-Carlo POD algorithms for $\alpha_i = \beta_i = 10$ (left) and $\alpha_i = \beta_i = 75$ (right).

2.3.4 Remarks

We implemented the weighted reduced order methods presented in Section 2.2 and we tested them on two easy linear stochastic elliptic problems, with random space dimension $N = 4$ and $N = 9$, respectively. We observed that weighted reduced order methods performs better than the not weighted ones. However, if the dimension of the random space is high enough ($N = 9$ in our case), not all the weighted methods perform well; indeed in this case the choice of a well representative sampling Ξ is fundamental. Moreover, the importance of a good sampling is still more important when the distribution of the random parameters is highly concentrated. This fact can be easily explained by the fact that the subspace built by the reduced order methods is a subspace of $\{u(y) : y \in \Xi\}$.

Thus, if Ξ contains only points y with probability $P\{Y = y\}$ too low, adding a linear combination of solutions $\{u(y) : y \in \Xi\}$ to the reduced subspace is useless, since we are interested in a statistics of the solution. Even if every method presented before could generate a sampling set Ξ enough fine, this is computationally impracticable for high N. Therefore, we need to use a method which samples accordingly to the distribution. This can be achieved in the greedy just taking Ξ as a set of realizations of Y and in the POD using the Monte-Carlo POD. Moreover, in our example we were considering Y with a Beta distribution function, so also Gauss-Jacobi POD, which is based on Gauss integration with a Beta weight function, was working almost as well as Monte-Carlo POD. However to use such a rule we are forced to take, in our case, at least $|\Xi| = 2^9$. In the next chapter we describe Smolyak quadrature rules, which allow to define multivariate quadrature rules with a reduced number of nodes with respect to tensor product quadrature rules, and we apply them to weighted POD algorithms.

Chapter 3

Sparse grid sampling techniques

In this chapter we present a possible way to define a multivariate quadrature rule, in order to use it in weighted POD algorithms. The algorithm presented here is known as Smolyak algorithm and it allows to define a multivariate quadrature rule, based on univariate quadrature rules, which make use of a nodal set with significantly less number of nodes compared to the tensor product rule. The resulting nodal set is called *sparse grid*. Sparse grids have been firstly used, in the framework of stochastic partial differential equations, for stochastic collocation methods. In the first section we present stochastic collocation methods for the solution of a stochastic partial differential equation (2.3)-(2.4). These methods are based on polynomial interpolation in the random parameter space. Then, in the second section we present Smolyak algorithms for quadrature rules. Their possible use in stochastic collocation methods and POD algorithms is discussed. Finally, in the third section, we report some numerical tests for the use of Smolyak quadrature algorithms in weighted POD algorithms.

3.1 Stochastic collocation methods

In this section we present the stochastic collocation methods for the solution of problem (2.1)-(2.2). This section is based on works [35, 48]. We will keep the notation introduced in section 2.1. The construction of stochastic collocation methods is based on polynomial interpolation in the multidimensional random space $\Gamma \subseteq \mathbb{R}^N$, where N is the dimension of the random space. Let us denote by $\mathbb{P}^N$ the space of all N-variate polynomials with all real coefficients and by $\mathbb{P}_d^N$ the subspace of polynomials of total degree at most d. Now, given a finite number of distinct points $y_1, \dots, y_M \in \mathbb{R}^N$, some real constants $b_1, \dots, b_M$, and a subspace $V_I \subseteq \mathbb{P}^N$, the *Lagrange interpolation* problem is defined as find a polynomial $l \in V_I$ such that

$$l(y_j) = b_j, \qquad \text{for each } j = 1, \dots, M. \tag{3.1}$$

The points $y_1, \dots, y_M$ are called interpolation nodes, and V_I is the interpolation space. The Lagrange interpolation point is said to be poised in V_I, for the points $y_1, \dots, y_M \in \mathbb{R}^N$, if, for any given data $b_1, \dots, b_M \in \mathbb{R}$ there exists a function $l \in V_I$ that satisfies (3.1). Suppose now that we have a set of prescribed nodes

$\Theta_N = \{y_j\}_{j=1}^M \subseteq \Gamma$ such that Langrange interpolation (3.1) in the N-dimensional random space Γ is poised in a corresponding interpolation space V_I. Subsequently, Lagrange interpolation of a smooth function $f : \Gamma \to \mathbb{R}$, can be viewed as follows: find a polynomial $\mathcal{I}(f) \in V_I$ such that $\mathcal{I}(f)(y_j) = f(y_j)$, for each $j = 1, \ldots, M$. The polynomial approximation $\mathcal{I}(f)$ can be expressed by using the Lagrange interpolation polynomials, i.e.,

$$\mathcal{I}(f)(y) = \sum_{k=1}^{M} f(y_k) L_k(y),$$

where

$$L_i \in V_I, \quad L_i(y_j) = \delta_{ij}, \quad \text{for } 1 \leq i, j \leq M, \tag{3.2}$$

are the Lagrange polynomials. We equip the spaces $C^0(\Gamma)$ and $\mathbb{P}^N(\Gamma)$ with the L^∞-norm, and we consider the dual norm of Lagrange operator $\mathcal{I} : C^0(\Gamma) \to V_I$. It is easy to show that

$$\|\mathcal{I}\| = \sup_{f \neq 0} \frac{\|\mathcal{I}(f)\|_\infty}{\|f\|_\infty} = \Lambda \doteq \max_{y \in \Gamma} \sum_{k=1}^{M} |L_i(y)|.$$

The constant Λ is known as the Lebesgue constant. Therefore, one gets that

$$\|f - f^*\|_\infty \leq \|f - \mathcal{I}(f)\|_\infty \leq (1 + \Lambda)\|f - f^*\|_\infty,$$

where f^* is the best approximating polynomial of f. The collocation procedure consists to approximate the solution $u(y)$ to the problem (2.3)-(2.4) with

$$\hat{u}(y) = \sum_{k=1}^{M} u(y_k) L_k(y),$$

where $u(y_k)$ are the solutions to (2.3)- (2.4) for $y = y_k$, $k = 1, \ldots, M$. By using the property of Lagrange interpolation (3.2), we immediately obtain:

$$\mathcal{L}(y_k, x; u(y_k)) = g(y_k, x), \qquad x \in D, \tag{3.3}$$

with boundary condition

$$\mathcal{B}(y_k, x; u(y_k)) = h(y_k, x), \qquad x \in \partial D, \tag{3.4}$$

for $k = 1, \ldots, M$. Thus, the stochastic collocation method is equivalent to solving M deterministic problems (3.3)-(3.4), the deterministic counterpart of problem (2.3)- (2.4), at each nodal point y_k, $k = 1, \ldots, M$, in a given nodal set Θ_N. Problem (3.3)-(3.4) for each k is naturally decoupled, and existing deterministic solvers can be readily applied.
Once the numerical solutions of (3.3)-(3.4) are obtained at all collocation points, the statistics of the random solution can be evaluated, e.g.,

$$\mathbb{E}(\hat{u}) = \sum_{k=1}^{M} u(y_k) \int_\Gamma L_k(y) \rho(y)\, dy. \tag{3.5}$$

The evaluations of such expectation require explicit knowledge of the Lagrange polynomials $\{L_k\}_{k=1}^M$. For a given nodal set Θ_N, the polynomials can be determined numerically by inverting a Vandermonde-type matrix (the matrix is invertible under the assumption that the Lagrange is poised on Θ_N) (see [40], p. 376). In multivariate case, such a procedure can be cumbersome, but can be accomplished once and for all at the preprocessing stage.
An alternative is to choose the set Θ_N to be a quadrature. Given a quadrature rule

$$\int_\Gamma f(y)\,dy \simeq \sum_{k=1}^{M} f(y_k)w_k,$$

where $\{w_k\}_{k=1}^M$ and $\{y_k\}_{k=1}^M$ are, respectively, the weights and the nodes of the quadrature formula, we can choose the interpolation point set Θ_N to be the same as the quadrature point set $\{y_i\}_{i=1}^M$, then the evaluation of (3.5) is reduced to

$$\mathbb{E}(\hat{u}) = \sum_{k=1}^{M} u(y_k)w_k. \tag{3.6}$$

The computational complexity of the stochastic collocation methods is M times that of a deterministic problem, where M is the total number of collocation points. Thus, we need to choose a nodal set Θ_N with fewest possible number of points under a prescribed accuracy requirement. Generally we will have multidimensional random spaces, i.e., $\Gamma \subseteq \mathbb{R}^N$ with $N > 1$. The first idea is to choose the multi-dimensional interpolation nodes set Θ_N as the tensor product of an univariate interpolation nodes sets. Remember that the random space has the form $\Gamma = \prod_{i=1}^N \Gamma_i$, with $\Gamma_i = [a_i, b_i]$, for $i = 1, \ldots, N$. If we have a sequence of N unidimensional nodal sets

$$\Theta_1^i = \left\{x_1^i, \ldots, x_{m_i}^i\right\} \subseteq [-1, 1], \tag{3.7}$$

for $i = 1, \ldots, N$, then we can take

$$\Theta_N = \Theta_1^{i_1} \times \cdots \times \Theta_1^{i_N}.$$

Clearly, the Lagrange interpolation with the above defined nodal set needs $M = m_{i_1} \cdot m_{i_2} \cdots m_{i_N}$ nodal points. If we choose the same nodal sets (3.7) in each dimension, with number of points m, then the total number of points is $M = m^N$. This number grows quickly in high dimension $N \gg 1$, even for a poor approximation with two points ($m = 2$), $M = 2^N \gg 1$ for $N \gg 1$. The same situation holds when we consider tensor product quadrature formulas (see Appendix, Section A.2.1) for the evaluation of (3.6). In the next section we describe the Smolyak algorithm for the construction of multivariate quadrature rules. These algorithms provide *sparse* grid as nodal set, and a correspondent set of weights. This approach allows to reduce significantly the number of nodes used, compared to the number of points of a tensor product rule.

3.2 Sparse grids: Smolyak quadrature rules

In this section we present the Smolyak algorithm for the construction of multivariate quadrature formulas. Given a sequence of quadrature rules, the Smolyak quadrature rule is formulated as a sum of tensor product taken over the consecutive differences in the univariate sequence. In the following we suppose we want to define a quadrature formula $\mathcal{U} : C^0(\Gamma) \to \mathbb{R}$, with $\Gamma = \prod_{i=1}^n \Gamma_i = \prod_{i=1}^n [a_i, b_i]$. Let $(U_i^{(j)})_{i=1}^{\infty}$ be a sequence of univariate quadrature rules in the interval $I_j \subseteq \mathbb{R}$, for $j = 1, \ldots, n$. We call i the order of the operator . We introduce the *differences operators* in I_j by setting

$$\Delta_0^{(j)} = 0 \qquad \text{and} \qquad \Delta_{i+1}^{(j)} = U_{i+1}^{(j)} - U_i^{(j)} \quad \text{for } i \geq 0,$$

where we put $U_i^{(0)} = 0$. We define the Smolyak quadrature rule of order q in Γ as the operator:

$$\mathcal{Q}_q^n = \sum_{\substack{|\alpha|_1 \leq q \\ \alpha \in \mathbb{N}^n}} \bigotimes_{i=1}^n \Delta_{\alpha_i}^{(i)}. \tag{3.8}$$

Note that the tensor product $\Delta_{\alpha_1}^{(1)} \otimes \cdots \otimes \Delta_{\alpha_n}^{(n)}$ in the summation of (3.8) vanishes whenever $\alpha_i = 0$ for some index i. Thus in the sequel we always assume $\boldsymbol{\alpha} \geq \mathbf{1}$ and hence $q \geq n$. Moreover, note that, for $n = 1$, we have $\mathcal{Q}_q^1 = U_q^{(1)}$ for all $q \geq 1$. Using the differences operators defined above, we can write the tensor product operator of order q in the form

$$\begin{aligned} \bigotimes_{i=1}^n U_q^{(i)} &= \left(\sum_{\alpha_1=0}^{q} \Delta_{\alpha_1}^{(1)} \right) \otimes \cdots \otimes \left(\sum_{\alpha_1=0}^{q} \Delta_{\alpha_n}^{(n)} \right) \\ &= \sum_{\alpha_1=0}^{q} \cdots \sum_{\alpha_n=0}^{q} \bigotimes_{i=1}^{q} \Delta_{\alpha_i}^{(i)} = \sum_{\substack{|\alpha|_\infty \leq q \\ \alpha \in \mathbb{N}^n}} \bigotimes_{i=1}^n \Delta_{\alpha_i}^{(1)}. \end{aligned}$$

Therefore the rule (3.8) can be considered as a delayed sum of the ordinary tensor product operator. The following theorem provides an alternative representation of operator (3.8). A detailed proof of this theorem is given in the Appendix, Section A.4.

Theorem 3.2.1. *The operator $\mathcal{Q}_q^n$ defining the Smolyak quadrature rule can be written as*

$$\mathcal{Q}_q^n = \sum_{\substack{\max\{n, q-n+1\} \leq |\alpha|_1 \leq k \\ \alpha \in \mathbb{N}^n,\, \alpha \geq \mathbf{1}}} (-1)^{q-|\alpha|_1} \binom{d-1}{k-|\alpha|_1} \bigotimes_{i=1}^n U_{\alpha_i}^{(i)}. \tag{3.9}$$

An immediate consequence of the formula (3.9) is the way it renders the evaluation point sets of the Smolyak rule explicitly known. Let $X_i^{(j)}$ be the

point sets containing the evaluation points of the quadrature rule $U_i^{(j)}$, for each $j = 1, \ldots, n$ and $i \geq 1$. Then by (3.9) the evaluation points of $\mathcal{Q}_q^n$ form the set

$$\Theta_n^q = \bigcup_{\substack{|\alpha|_1 = q \\ \alpha \in \mathbb{N}^n,\, \alpha \geq \mathbf{1}}} X_{\alpha_1}^{(1)} \times \cdots \times X_{\alpha_n}^{(n)} \qquad \text{for all } q \geq n.$$

We call elements of the set Θ_n^q the nodes of $\mathcal{Q}_q^n$. The cardinality of the set Θ_n^q determines the cost of $\mathcal{Q}_q^n$ since it determines the minimum number of function evaluations required to compute the Smolyak rule. We can now formulate the numerical implementation of the Smolyak rule for the evaluation of integrals of the form

$$I_W^n f = \int_\Gamma W_1(x_1) \cdots W_n(x_n) f(x_1, \ldots, x_n)\, dx_1 \cdots dx_n$$

where $W(x_1, \ldots, x_n) = W_1(x_1) \cdots W_n(x_n)$ is the weight function, for $f \in C^0(\Gamma)$. Suppose that $U_i^{(j)}$ are the quadrature rules for univariate integrals with weight function W_j, i.e.,

$$\int_{\Gamma_i} f(x) W_j(x)\, dx \simeq U_i^{(j)}(f) \doteq \sum_{l=1}^{m_i^{(j)}} w_{i,l}^{(j)} f(x_{i,l}^{(j)})$$

for $f \in C^0(\Gamma_i)$, $i = 1, \ldots, n$. Then, for $f \in C^0(\Gamma)$, the Smolyak rule gives the approximation of integral

$$\begin{aligned} I_W^n f &\simeq \mathcal{Q}_q^n(f) = \\ &= \sum_{s=\max\{n, q-n+1\}}^{q} \sum_{\substack{|\alpha|_1 = s \\ \alpha \in \mathbb{N}^n,\, \alpha \geq \mathbf{1}}} \sum_{l_1=1}^{m_{\alpha_1}^{(1)}} \cdots \sum_{l_n=1}^{m_{\alpha_n}^{(n)}} C_{n,q}^s\, w_{1,l_1}^{(\alpha_1)} \cdots w_{n,l_n}^{(\alpha_n)} f(x_{1,l_1}^{(\alpha_1)}, \ldots, x_{n,l_n}^{(\alpha_n)}), \end{aligned}$$

where $C_{n,q}^s = (-1)^{q-n} \binom{n-1}{q-s}$. The above is just the Smolyak rule (3.9) written down explicitly. Observe that the cumbersome nested sums can be replaced by summation over all occurring combinations of univariate nodes and weights. Clearly, the formulation discussed above can be formulated with interpolation operators $L_i^{(j)}$ instead of quadrature operators $U_i^{(j)}$. For more details about Smolyak interpolation we refer to works [4, 47].
We report here two results on order of polynomial exactness from Smolyak quadrature rules.

Theorem 3.2.2. *Let $U_i^{(j)}$ be univariate quadrature rules that corresponds to the weight W_j and have polynomial exactness $m_i^{(j)} = m_i$ such that $m_i \leq m_{i+1}$. Then*

$$I_W^n f = \mathcal{Q}_q^n(f) \qquad \forall\, f \in \sum_{\substack{|\alpha|_1 = q \\ \alpha \in \mathbb{N}^n}} \bigotimes_{i=1}^{n} \mathbb{P}^1_{m_{\alpha_i}} \quad \textit{and} \quad q \geq n.$$

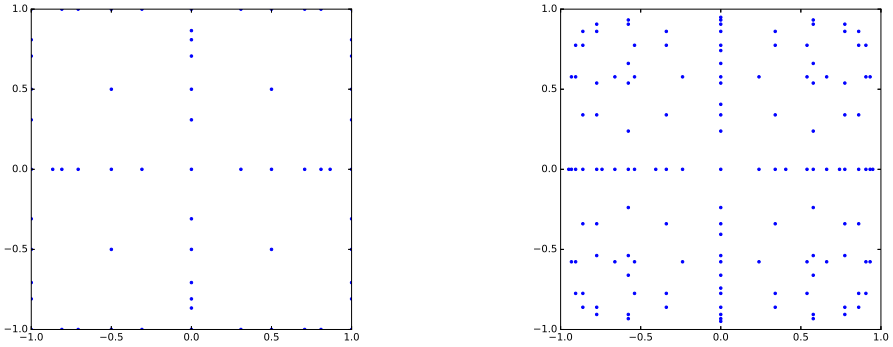

Figure 3.1: Example of sparse grids. The right one is obtained using Clenshaw-Curtis nodes gor $q = 9$, the left one with Gauss-Legendre nodes for $q = 7$.

For the proof of the theorem above and further results on order of polynomial exactness and approximation errors for Smolyak quadrature we refer to works [23, 27, 36]. Smolyak methods are frequently used in implementing stochastic collocation techniques. They indeed allow to use interpolation or quadrature rules with a reduced number of nodes with respect to tensor product rules. Our idea is to try to use them for the construction of weighted POD algorithms. In the next section we solve problem (2.20) and compare the results obtained with that of the previous chapter.

3.3 Numerical tests

In this section we solve problem (2.20) using Smolyak quadrature rules and we discuss the results. We use the notation of Section 2.3.3. We wrote `python` scripts for Smolyak algorithms and we implemented them in `RBniCS` to allow weighted reduced order methods to use them.
We used Smolyak algorithms built using Gauss-Legendre or Gauss-Jacobi rules in each dimension, i.e., we defined $\mathcal{Q}_q^n$ taking $U_i^{(j)}$ to be the Gauss-Legendre quadrature operator of order i, for each $j = 1, \dots, n$, or taking $U_i^{(j)}$ to be the Gauss-Jacobi quadrature operator of order i and parameters α_i, β_i, for each $j = 1, \dots, n$. We refer to the POD algorithm deriving from this quadrature rule as, respectively, sparse Gauss-Legendre POD and sparse Gauss-Jacobi POD. We did not used Clenshaw-Curtis rules for the following reason. If we define the Smolyak quadrature operator taking $U_i^{(j)}$ to be the Clenshaw-Curtis quadrature operator of order i, for each $j = 1, \dots, n$, the number of nodes not in $\partial\Gamma$ is too low (see Table 3.1). In Figure 3.2 we reported the performances of Gauss-Legendre POD and Sparse Gauss-Legendre POD for problem (2.20), for $\alpha_i = \beta_i = 10$. As we can see the sparse Gauss-Legendre POD method performs much worse than the standard Gauss-Legendre POD, providing a constant error for $N > 1$. This is due to an unconvenient choice of nodes. Indeed, if $Y = (Y^1, \dots, Y^9)$ is a random variable distributed as in (2.21), for $\alpha_i = \beta_i = 10$, and we denote $\Gamma^* = [1.5, 2.5]^9 \subseteq \Gamma$, then $P\{Y \in \Gamma^*\} \simeq 0.85$. However if we use a sparse Gauss-Legendre training set

	$q = 9$	$q = 10$	$q = 11$	$q = 12$	$q = 13$
$\lvert\Xi\rvert$	1	19	163	853	3157
$\lvert\Xi \setminus \partial\Gamma\rvert$	1	1	1	19	37

Table 3.1: Cardinalities of the sets Ξ of nodes of Smolyak quadrature algorithms built from Clenshaw-Curtis rules of different orders q.

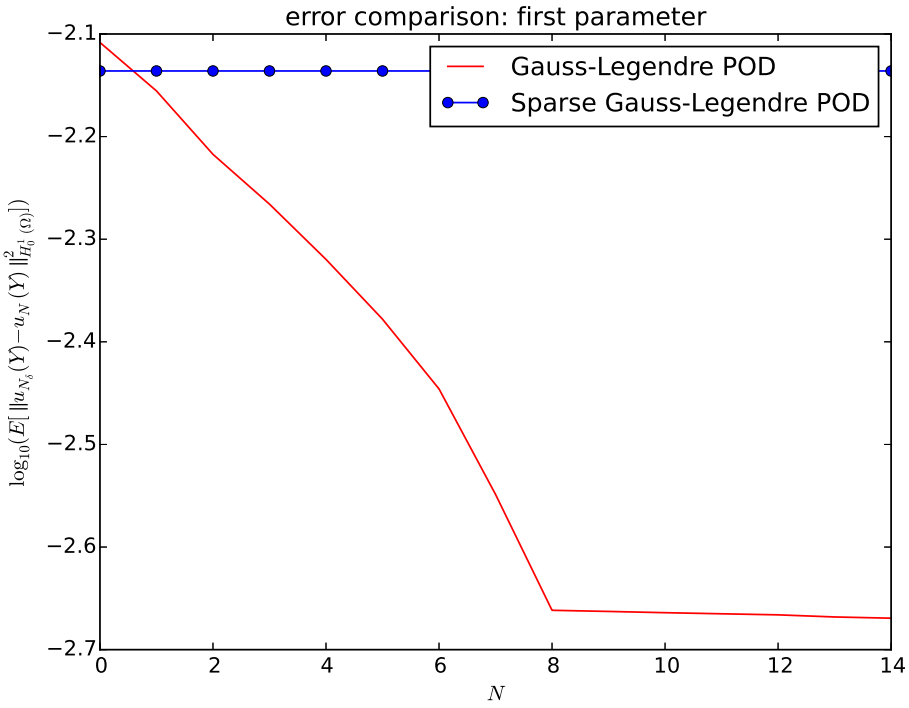

Figure 3.2: Comparison of error (2.14) using Gauss-Legendre POD and sparse Gauss-Legendre POD methods.

with cardinality $|\Xi| = 181$ (which corresponds to the Smolyak formula of order $q = 11$), then only one point of Ξ lies in Γ^*. This means that the sparse Gauss-Legendre POD (of such order q) samples nodes that are mostly irrelevant with respect to the distribution of Y. The situation does not get better for higher orders: for $q = 12$ one gets $|\Xi \cap \Gamma^*| = 1122$ while $|\Xi| = 1177$ and for $q = 13$ one gets $|\Xi \cap \Gamma^*| = 5748$ while $|\Xi| = 5965$. Bigger values of q are computationally impracticable. Because of the bad performance in the case $\alpha_i = \beta_i = 10$ we did not implemented Gauss-Jacobi for the case $\alpha_i = \beta_i = 75$, where the distribution is much more concentrated.
In Figure 3.3 we report the performance of Gauss-Jacobi POD and Sparse Gauss-Jacobi POD for problem (2.20), for $\alpha_i = \beta_i = 10$ and $\alpha_i = \beta_i = 75$. In this

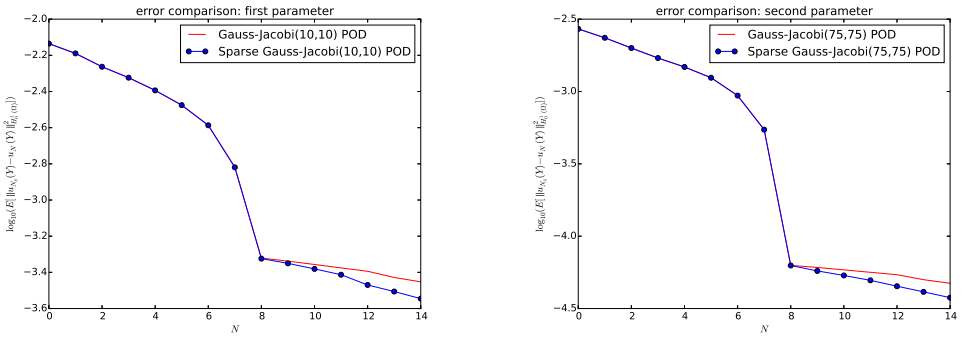

Figure 3.3: Comparison of error (2.14) obtained using sparse Gauss-Jacobi POD and standard Gauss-Jacobi POD algorithms for $\alpha_i = \beta_i = 10$ (left) and $\alpha_i = \beta_i = 75$ (right).

case the use of sparse grids actually improve the performances of POD algorithms. However, the difference between the errors obtained is not so relevant. The important difference between the two algorithms is the number of nodes used. Indeed, in the sparse algorithm we took a training set Ξ consisting of $|\Xi| = 181$ nodes, versus the 512 nodes of the tensor product grid. Finally in Figure 3.4 we

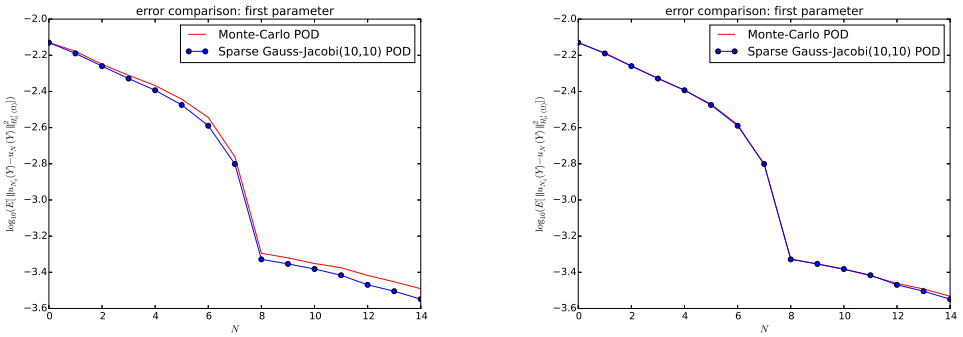

Figure 3.4: Comparison of error (2.14) obtained using sparse Gauss-Jacobi POD and Monte-Carlo POD algorithms for $\alpha_i = \beta_i = 10$. In the left the training set Ξ for Monte-Carlo algorithm is taken with $|\Xi| = 100$, and with $|\Xi| = 500$ in the right figure.

compared the performance of sparse Gauss-Jacobi POD and Monte-Carlo POD algorithms in the case $\alpha_i = \beta_i = 10$. In the left figure we used a training set of $|\Xi| = 100$ nodes for the Monte-Carlo POD and in the right figure we used a training set of $|\Xi| = 500$ nodes. In the first case the sparse Gauss-Jacobi POD performs a little better than the Monte-Carlo POD, even if the difference is not relevant, while in the second case both algorithms provide almost the same approximation error. Therefore, the sparse Gauss-Jacobi POD method provides the best approximation with the least number of nodes in the training set Ξ.

3.3.1 Remarks

We implemented weighted POD algorithms based on Smolyak quadrature formulas. We applied them to problem (2.20). The results highlighted the importance of a well representative, in the sense of the distribution of Y, training set Ξ. This induces a dual effect. If the quadrature rule was already working well and providing a 'good' train set in the weighted POD based on the tensor product quadrature, the weighted POD based on Smolyak method for the same quadrature rule provides a slightly better approximation with the use of a smaller training set. This is very important because it makes possible to reduce the computational effort in the offline stage. In our example, this is the case of the Gauss-Jacobi POD method.
Instead, if the quadrature rule is not optimal, using a Smolyak algorithm we have the opposite effect and the situation worsens. Indeed, with the Smolyak algorithm the number of points representative of the distribution of Y drastically decreases. This has implied, in our case, to have an error constant in the dimension N of the reduced order space, for $N = 1, \dots, 15$.

Chapter 4

Application to a stochastic vectorial problem

In this chapter we apply the weighted reduced order methods presented in the previous chapters to a linear elasticity problem. In the first section, we present the problem in its deterministic version, then, in the second section, we add some stochastic parameters and we apply weighted reduced order methods to solve it. Therefore, the obtained results are discussed.

4.1 The linear 2d elastic block problem

Let $\Omega \subseteq \mathbb{R}^2$ be a domain with Lipschitz boundary $D = \partial\Omega$. Also, let $D_0, D_1 \subseteq D$ be a partition of D, such that D_0 has a positive $d\gamma$-measure. We consider the following problem: find $\mathbf{u} \in \mathbf{H}^1(\Omega)$ such that

$$\begin{aligned} -\mu\Delta\mathbf{u} - (\lambda+\mu)[\nabla(\nabla\cdot\mathbf{u})] &= \mathbf{f} \qquad \text{in } \Omega, \\ \mathbf{u} &= \mathbf{0} \qquad \text{on } D_0, \\ \sigma(\mathbf{u})\mathbf{n} &= \mathbf{g} \qquad \text{on } D_1, \end{aligned} \tag{4.1}$$

where μ, λ are two positive constants, $f \in \mathbf{L}^2(\Omega)$, $\mathbf{g} \in \mathbf{L}^2(D_1)$, and

$$\sigma(\mathbf{u}) = \lambda(\nabla\cdot\mathbf{u})\mathbf{I} + \mu(\nabla\mathbf{u} + \nabla\mathbf{u}^T). \tag{4.2}$$

Equations (4.1) and (4.2) can be formulated in terms of the *linearized strain tensor*

$$\mathbf{e}(\mathbf{v}) = \frac{1}{2}\left(\nabla\mathbf{v} + \nabla\mathbf{v}^T\right)$$

as, respectively,

$$-\nabla\cdot\{\lambda\,\mathrm{tr}(\mathbf{e}(\mathbf{u}))\,\mathbf{I} + 2\mu\,\mathbf{e}(\mathbf{u})\} = \mathbf{f} \qquad \text{in } \Omega,$$

and

$$\sigma(\mathbf{v}) = \lambda\,\mathrm{tr}(\mathbf{e}(\mathbf{v}))\,\mathbf{I} + 2\mu\,\mathbf{e}(\mathbf{v}). \tag{4.3}$$

The above problem is called the *system of equations of two-dimensional, or plane, elasticity.* Assuming 'small' displacements $\mathbf{u}$ and 'small' strains $\mathbf{e}(\mathbf{u})$, this system describes the equilibrium state of a homogeneous isotropic elastic body which occupies the set $\overline{\Omega}$, $\mathbf{u}$ denoting the displacement of the points of $\overline{\Omega}$ under the influence of given forces $\mathbf{f}$ and $\mathbf{g}$. The body $\overline{\Omega}$ can not move along Γ_0, and along Γ_1 surfaces forces of density $\mathbf{g}$ are given. In addition, a volumic force, of density $\mathbf{f}$, is prescribed inside the body $\overline{\Omega}$. The tensor $\sigma(\mathbf{u})$ is called the *linearized stress tensor* and the relationship (4.3) between the linearized strain tensor and the linearized stress tensor is known in elasticity theory as *Hooke's law* for isotropic bodies. The constants λ and μ are the *Lamé coefficients* of the material of which the body is composed. For further details on equation (4.1) and its physical meaning we refer to [17].
Problem (4.1) can be reformulated in a weak version. Let us define the space

$$\mathbb{V} = \{\mathbf{v} = (v_1, v_2) \in (H^1(\Omega))^2 \ : \ v_i|_{D_0} = 0, \quad \text{for } i = 1, 2\}. \tag{4.4}$$

Then if we integrate (4.1) over Ω against a function $\mathbf{v} \in \mathbb{V}$ we get

$$\int_\Omega (\nabla \cdot \mathbf{S}(\mathbf{u})) \cdot \mathbf{v} \, dx = \int_\Omega \mathbf{f} \cdot \mathbf{v} \, dx$$

where we put $\mathbf{S}(\mathbf{u}) = -\sigma(\mathbf{u})$. Now, from the fundamental Green's formula (0.2), it follows that

$$\begin{aligned} \int_\Omega (\nabla \cdot \mathbf{S}(\mathbf{u})) \cdot \mathbf{v} \, dx &= -\int_\Omega \mathbf{S}(\mathbf{u}) : \nabla \mathbf{v} \, dx + \int_D \mathbf{S}(\mathbf{u})\mathbf{n} \cdot \mathbf{v} \, d\gamma \\ &= -\int_\Omega \mathbf{S}(\mathbf{u}) : \mathbf{e}(\mathbf{v}) \, dx - \int_{D_1} \mathbf{g} \cdot \mathbf{v} \, d\gamma. \end{aligned}$$

Therefore, we can reformulate problem, in the following weak form: find $\mathbf{u} \in \mathbb{V}$ such that, for all $\mathbf{v} \in \mathbb{V}$,

$$\int_\Omega \sigma(\mathbf{u}) : \mathbf{e}(\mathbf{v}) = \int_\Omega \mathbf{f} \cdot \mathbf{v} \, dx + \int_{D_1} \mathbf{g} \cdot \mathbf{v} \, d\gamma.$$

Clearly, we can rewrite this problem as: find $\mathbf{u} \in \mathbb{V}$ such that

$$a(\mathbf{u}, \mathbf{v}) = f(\mathbf{v}) \qquad \text{for all } \mathbf{v} \in \mathbb{V}, \tag{4.5}$$

where we defined the bilinear form

$$\begin{aligned} a(\mathbf{u}, \mathbf{v}) &= \int_\Omega \operatorname{tr}\left(\sigma(\mathbf{v})\mathbf{e}(\mathbf{v})\right) \, dx \\ &= \int_\Omega \left\{\lambda \left(\nabla \cdot \mathbf{u}\right)\left(\nabla \cdot \mathbf{v}\right) + 2\mu \, \mathbf{e}(\mathbf{u}) : \mathbf{e}(\mathbf{v})\right\} \, dx \end{aligned}$$

and the linear form

$$f(\mathbf{v}) = \int_\Omega \mathbf{f} \cdot \mathbf{v} \, dx + \int_{D_1} \mathbf{g} \cdot \mathbf{v} \, d\gamma.$$

The space $\mathbb{V}$ is a normed space with the norm $\|\cdot\|_{\mathbb{V}} = \|\cdot\|_{1,\Omega}$. Moreover, applying repeatedly the Cauchy-Schwartz inequality, and thanks to the continuity of the trace operator, it is easy to show these bilinear and linear forms are continuous. Instead, coercivity of the bilinear form a is not trivial, and we need some preliminary result to prove it. The following theorem provides a fundamental inequality, which is due to Korn.

Theorem 4.1.1 (Korn's inequality)**.** *Let Ω be a domain in $\mathbb{R}^d$. Then there exists a constant $c > 0$ such that*

$$\|\mathbf{v}\|_{1,\Omega} \leq c\left\{|\mathbf{v}|_{0,\Omega}^2 + |\mathbf{e}(\mathbf{v})|_{0,\Omega}^2\right\}^{1/2} \qquad \textit{for all } \mathbf{v} \in \mathbf{H}^1(\Omega),$$

and thus, on the space $\mathbf{H}^1(\Omega)$, the mapping

$$\mathbf{v} \mapsto \left\{|\mathbf{v}|_{0,\Omega}^2 + |\mathbf{e}(\mathbf{v})|_{0,\Omega}^2\right\}^{1/2}$$

is a norm, equivalent to the norm $\|\cdot\|_{1,\Omega}$.

For a proof of this theorem we refer to [34]. With Korn's inequality, we now prove the coercivity of the bilinear form

$$\mathbf{v} \mapsto \int_\Omega \mathbf{e}(\mathbf{v}) : \mathbf{e}(\mathbf{v})\, dx,$$

whence that of the bilinear form a. In the proof of the following theorem we make use of Theorem A.5.1 and Theorem A.5.2, which are reported in the appendix.

Theorem 4.1.2. *Let $\mathbb{V}$ be the space defined in (4.4). Then $\mathbb{V}$ is a closed subspace of $\mathbf{H}^1(\Omega)$ (i.e., $\mathbb{V}$ is an Hilbert space). Moreover, there exists a constant $c > 0$ such that*

$$\frac{1}{c}\|\mathbf{v}\|_{1,\Omega} \leq |\mathbf{e}(\mathbf{v})|_{0,\Omega} \leq c\|\mathbf{v}\|_{1,\Omega} \qquad \textit{for all } \mathbf{v} \in \mathbb{V}, \tag{4.6}$$

i.e, on the space $\mathbb{V}$, the seminorm $\mathbf{v} \mapsto |\mathbf{e}(\mathbf{v})|_{0,\Omega}$ is a norm, equivalent to the norm $\|\cdot\|_{1,\Omega}$.

Proof. We first prove that $\mathbb{V}$ is closed. Let $(\mathbf{v}_k)_k$ be a sequence of elements of $\mathbf{V}$ such that $\mathbf{v}_k \to \mathbf{v}$ in $\mathbf{H}^1(\Omega)$. Then $\mathbf{v}_k \to \mathbf{v}$ in $L^2(D)$, and so there exists a subsequence $(\mathbf{v}_{k_i})_i$ such that $\mathbf{v}(x) = \lim_{i\to\infty} \mathbf{v}_{k_i}(x)$ for $d\gamma$-a.e. $x \in \Gamma$. Hence, $\mathbf{v} = 0$ $d\gamma$-a.e. on Γ_0, and thus the space $\mathbb{V}$ is closed in $\mathbf{H}^1(\Omega)$.
As a preliminary to the proof of inequality (4.6), we now show that $\mathbf{v} \mapsto |\mathbf{e}(\mathbf{v})|_{0,\Omega} = 0$ implies $\mathbf{v} = \mathbf{0}$ if $\mathbf{v} \in \mathbb{V}$. To this end, we observe that the following identities hold:

$$\partial_{jk} v_i = \partial_j e_{ik}(\mathbf{v}) + \partial_k e_{ij}(\mathbf{v}) - \partial_i e_{jk}(\mathbf{v}) \qquad \text{in } \mathcal{D}'(\Omega),$$

for $i, j, k = 1, 2$. Hence we deduce that

$$|\mathbf{e}(\mathbf{v})|_{0,\Omega} = 0 \quad \Rightarrow \quad \mathbf{e}(\mathbf{v}) = \mathbf{0} \quad \Rightarrow \quad \partial_{jk} v_i = 0 \quad \text{in } \mathcal{D}'(\Omega),$$

for $i, j, k = 1, 2$. By Theorem A.5.1, we deduce that each function v_i is a polynomial of degree less or equal than 1 in the variables x_i, i.e., $v_i(x) = a_i + b_{i1}x_1 + b_{i2}x_2$.

Since $e_{ij}(\mathbf{v}) = 0$ further implies that $b_{ij} = -b_{ji}$, we reach the conclusion that there are constants a_1, a_2 and b such that

$$v_1(x) = a_1 + bx_2, \qquad v_2(x) = a_2 - bx_1,$$

equivalently, there exists $\mathbf{a} \in \mathbb{R}^2$ and $b \in \mathbb{R}$ such that

$$\mathbf{v}(x) = \mathbf{a} + \begin{pmatrix} 0 & b \\ -b & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \qquad \text{for all } x \in \overline{\Omega}.$$

Hence the set $\{x \in \mathbb{R}^2 \ : \ \mathbf{v}(x) = \mathbf{0}\}$ is always of zero area, unless $b = 0$, $\mathbf{a} = \mathbf{0}$, and thus

$$\text{area}(D_0) > 0 \quad \Rightarrow \quad \{\mathbf{v} \in \mathbf{H}^1(\Omega) \ : \ \mathbf{e}(\mathbf{v}) = 0, \ \mathbf{v} = 0 \ d\gamma\text{-a.e. on } D_0\} = \{\mathbf{0}\},$$

as was to be proven.
Now we can prove the inequalities (4.6). The inequality $|\mathbf{e}(\mathbf{v})|_{0,\Omega} \leq c\|\mathbf{v}\|_{1,\Omega}$ clearly holds for all $\mathbf{v} \in \mathbb{V}$ (in fact, for all $\mathbf{v} \in \mathbf{H}^1(\Omega)$), for a constant $c > 0$ big enough (it can be proved just applying repeatedly the Cauchy-Schwartz inequality). Instead, if the other inequality was false, there exists a sequence $(\mathbf{v}_k)_k$ in $\mathbb{V}$ such that

$$\|\mathbf{v}_k\|_{1,\Omega} = 1 \quad \text{for all } k \geq 1, \quad \text{and} \quad \lim_{k\to\infty} \|\mathbf{e}(\mathbf{v}_k)\|_{0,\Omega} = 0.$$

Since the sequence $(\mathbf{v}_k)_k$ is bounded in the space $\mathbf{H}^1(\Omega)$, there exists a subsequence $(\mathbf{v}_{k_i})_i$ that converges in the space $\mathbf{L}^2(\Omega)$, as a consequence of Theorem A.5.2. Since the subsequence $(\mathbf{e}(\mathbf{v}_{k_i}))_i$ also converges in the spaces $\mathbf{L}^2(\Omega)$, we conclude that the sequence $(\mathbf{v}_{k_i})_i$ is a Cauchy sequence with respect to the norm

$$\mathbf{v} \mapsto \{|\mathbf{v}|^2_{0,\Omega} + |\mathbf{e}(\mathbf{v})|^2_{0,\Omega}\}^{1/2}.$$

By Korn's inequality, this norm is equivalent to the norm $\|\cdot\|_{1,\Omega}$ on the space $\mathbf{H}^1(\Omega)$. Hence this Cauchy sequence converges to some element $\mathbf{v} \in \mathbb{V}$, since the space $\mathbb{V}$ is complete. Therefore, the limit $\mathbf{v}$ satisfies

$$\|\mathbf{e}(\mathbf{v})\|_{0,\Omega} = \lim_{i\to\infty} \|\mathbf{e}(\mathbf{v}_{k_i})\|_{0,\Omega} = 0,$$

and hence $\mathbf{v} = \mathbf{0}$. But this contradicts the equalities $\|\mathbf{v}_{k_i}\|_{1,\Omega} = 1$ for all $i \geq 1$. $\square$

From the previous result, it easily follows the coercivity of the bilinear form a. Indeed, if $\mathbf{v} \in \mathbb{V}$ we have that

$$a(\mathbf{v}, \mathbf{v}) = \int_\Omega \{\lambda(\nabla \cdot \mathbf{v})^2 + 2\mu\, \mathbf{e}(\mathbf{v}) : \mathbf{e}(\mathbf{v})\}\, dx \geq 2\mu |\mathbf{e}(\mathbf{v})|^2_{0,\Omega}$$

since $\lambda > 0$, which implies coercivity of a, since $\mu > 0$ and (4.6). We therefore conclude that problem (4.5) is well-posed.

4.2 Stochastic formulation and numerical results

We consider equation (4.5) with the following stochastic parametrization. Let $\Omega, \Omega_1, \dots, \Omega_4$ be as in Section 2.3.2. We consider problem (4.5) where we put

$$a(\mathbf{u}, \mathbf{v}; Y^1, \dots, Y^4) = \sum_{i=1}^{4} Y^i \int_{\Omega_i} \{\lambda\, (\nabla \cdot \mathbf{u})(\nabla \cdot \mathbf{u}) + 2\mu\, \mathbf{e}(\mathbf{u}) : \mathbf{e}(\mathbf{v})\}\, dx,$$

$$f(\mathbf{v}; Y^5, Y^6) = \sum_{i=1}^{2} Y^{i+4} \int_{(i-1)/2}^{i/2} v_2(1, x_2)\, dx_2,$$

for every $\mathbf{u}, \mathbf{v} \in \mathbb{V}$, where $\mathbb{V} = \{\mathbf{v} \in \mathbf{H}^1(\Omega) \ : \ \mathbf{v}|_{D_0} = 0\}$, with $D_0 = [0,1] \times \{0,1\} \subseteq \partial\Omega$. This corresponds to consider the equation of a linear elastic block,

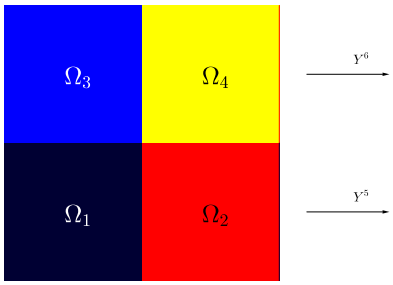

Figure 4.1: Geometrical set-up of problem (4.7).

split in four parts, with Lamé constants re-scaled by a parameter Y^i, $i = 1, \dots, 4$, on each part, and with the volumetric and surface forces, respectively,

$$\mathbf{f}(x) \equiv \mathbf{0}, \qquad \text{for } x \in \Omega,$$

$$\mathbf{g}(x; Y^4, Y^5) = (Y^5 \mathbb{1}_{[0,1/2]}(x_2) + Y^6 \mathbb{1}_{[1/2,1]}(x_2)) \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \qquad \text{for } x \in D_1 = \partial\Omega \setminus D_0,$$

acting on the block. Moreover we consider the parameters Y^i, $i = 1, \dots, 6$, to be random numbers such that

$$\frac{Y^i - 1}{2} \sim \text{Beta}(\alpha_i, \beta_i), \qquad \text{for } i = 1, \dots, 4,$$

$$\frac{Y^i - 2}{4} \sim \text{Beta}(\alpha_i, \beta_i), \qquad \text{for } i = 5, 6.$$

Thus we look for the solution of: find $\mathbf{u} = \mathbf{u}(Y) \in \mathbb{V}$ such that

$$a(\mathbf{u}, \mathbf{v}; Y^1, \dots, Y^6) = f(\mathbf{v}; Y^5, Y^6) \tag{4.7}$$

for all $\mathbf{v} \in \mathbb{V}$.

POD method

The forms a and f are such that we can apply the weighted POD methods presented in the previous sections. Therefore we implemented POD algorithms for

the solution of the above problem. We also implemented greedy algorithms, but since in this case it had not a priori a lower bound α_{LB} of the coercivity constant, we had to use SCM to compute it. These resulted in a much longer computation time with respect to the computation time needed to the POD algorithm. Moreover, the two algorithms showed practically the same behavior, so we decided to report here only the results obtained with the weighted POD methods.
Firstly we analyzed the error (2.14) obtained with standard and various POD methods for the cases . In Table 4.1 we reported the cardinalities of the training sets Ξ for the various algorithms. We implemented standard, uniform Monte-Carlo and Monte-Carlo PODs. In Figure 4.2 we reported the results. The situation resembles perfectly that of the precedent cases, i.e., both the weighted POD methods work better, up to an order of approximation, than the standard one, when the distribution of Y is not much concentrated, while, when the distribution is highly concentrated, the uniform Monte-Carlo POD loose its accuracy, because of a bad sampling, while the Monte-Carlo POD performs much better than the standard one, up to two order of approximation. Then, we implemented weighted

	Standard - Monte-Carlo	Gauss-Legendre	Sparse Gauss-Jacobi
$\lvert\Xi\rvert$	500	729	389

Table 4.1: Description of the number of points of Ξ in the weighted POD algorithms used.

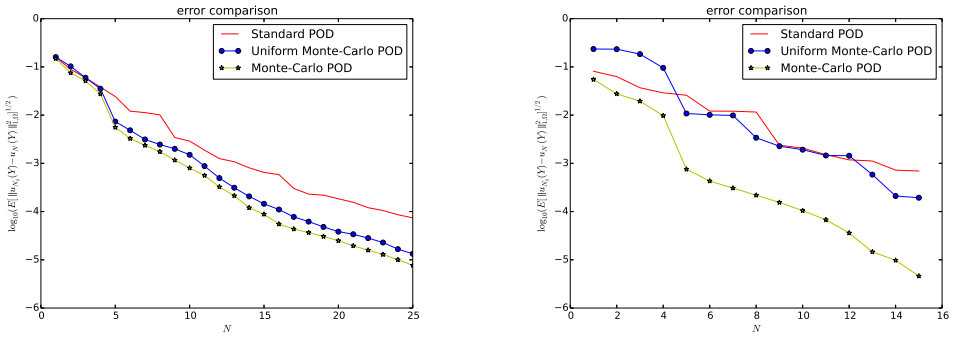

Figure 4.2: Comparison of error (2.14) using standard, Monte-Carlo and Uniform Monte-Carlo POD algorithms, for the cases $\alpha_i = \beta_i = 10$ (left) and $\alpha_i = \beta_i = 75$ (right).

POD methods based on quadrature formula. Also in this case we did not implemented Clenshaw-Curtis POD, since it was providing a training set such that $|\Xi| = 728$, but with only one point not on D. The next choice of training set has a cardinality of $|\Xi| = 15625$, which was computationally impracticable. Following the results of the previous chapter, we instead chose to use sparse algorithms for Gauss-Jacobi POD, and tensor product algorithms for Gauss-Legendre POD. The results are reported in Figure 4.3 and they still resemble the situation of the previous chapters. The Gauss-Jacobi POD performs better than standard POD, up to an order of accuracy in the case $\alpha_i = \beta_i = 10$ and up to two order of accu-

racy in the case $\alpha_i = \beta_i = 75$. Instead, the Gauss-Legendre POD, perform little better than the standard one, but worse than the Gauss-Jacobi POD. This is due to a bad training set Ξ choice, which causes the algorithm to produce singular basis, in the case $\alpha_i = \beta_i = 75$, for $N \geq 10$. We therefore reported in Figure 4.4

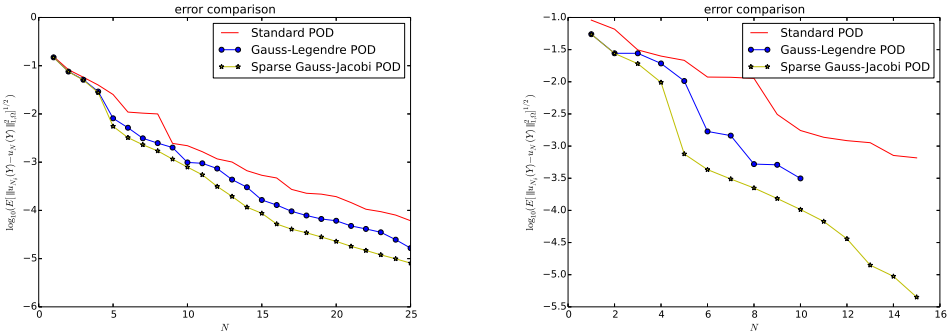

Figure 4.3: Comparison of error (2.14) using standard, Gauss-Legendre and sparse Gauss-Jacobi POD algorithms, for the cases $\alpha_i = \beta_i = 10$ (left) and $\alpha_i = \beta_i = 75$ (right).

the performances of Gauss-Jacobi and Monte-Carlo POD versus the ones of the standard POD. Finally, for sake of completeness, we reported also the graphs for the following errors:

(i) The distance between the mean of the solution computed with Monte-Carlo and the mean of the solution computed with the reduced order method:

$$\|\mathbb{E}[u_{N_\delta}(Y)] - \mathbb{E}[u_N(Y)]\|^2_{1,\Omega}, \tag{4.8}$$

(ii) The error for the computation of the mean of the compliant output:

$$|\mathbb{E}[s_{N_\delta}(Y)] - \mathbb{E}[s_N(Y)]|, \tag{4.9}$$

(iii) The relative error for the computation of the mean of the compliant output:

$$\frac{|\mathbb{E}[s_{N_\delta}(Y)] - \mathbb{E}[s_N(Y)]|}{|\mathbb{E}[s_{N_\delta}(Y)]|}. \tag{4.10}$$

For such errors we implemented only the standard POD method and the two best working weighted POD methods: Monte-Carlo and sparse Gauss-Jacobi. As we can see in figures 4.5, 4.6 and 4.7, the good performances of the weighted PODs versus those of the standard POD still hold.

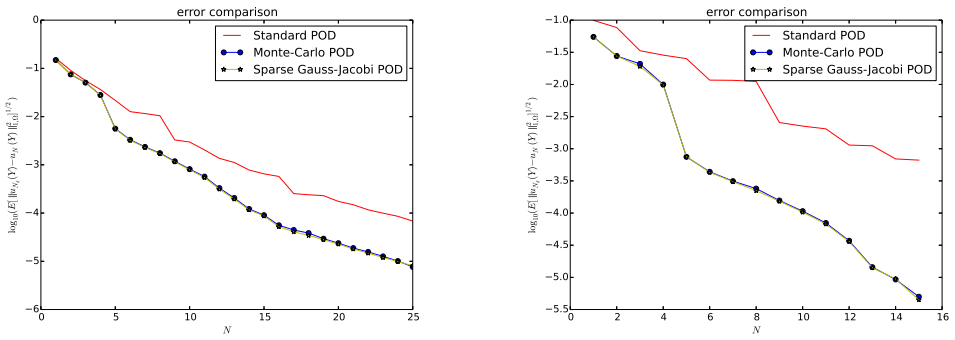

Figure 4.4: Comparison of error (2.14) using standard, Monte-Carlo and sparse Gauss-Jacobi POD algorithms, for the cases $\alpha_i = \beta_i = 10$ (left) and $\alpha_i = \beta_i = 75$ (right).

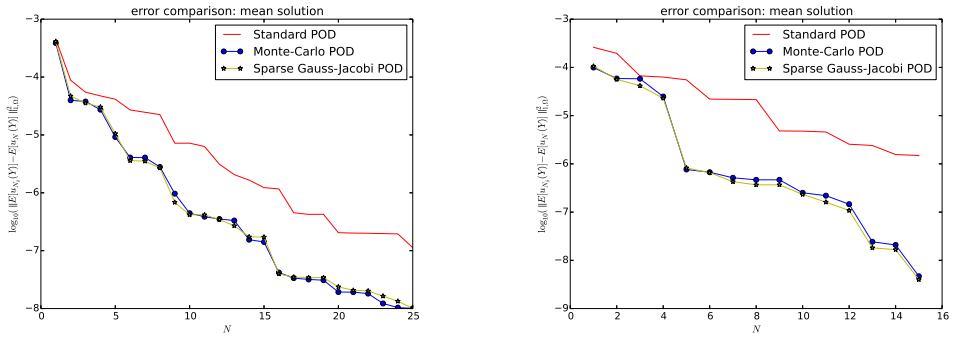

Figure 4.5: Comparison of error (4.8) using standard, Monte-Carlo and sparse Gauss-Jacobi POD algorithms, for the cases $\alpha_i = \beta_i = 10$ (left) and $\alpha_i = \beta_i = 75$ (right).

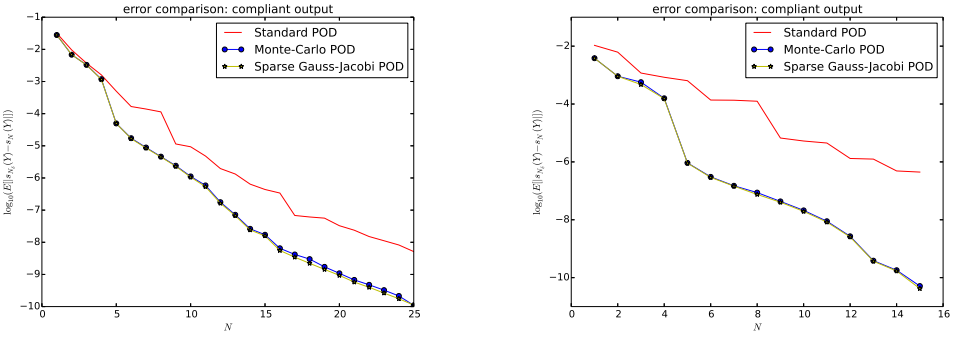

Figure 4.6: Comparison of error (4.9) using standard, Monte-Carlo and sparse Gauss-Jacobi POD algorithms, for the cases $\alpha_i = \beta_i = 10$ (left) and $\alpha_i = \beta_i = 75$ (right).

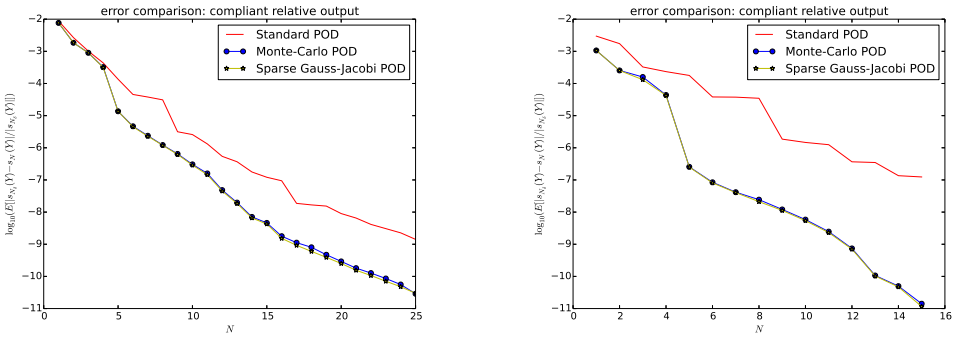

Figure 4.7: Comparison of error (4.10) using standard, Monte-Carlo and sparse Gauss-Jacobi POD algorithms, for the cases $\alpha_i = \beta_i = 10$ (left) and $\alpha_i = \beta_i = 75$ (right).

Chapter 5

Conclusions and perspectives

In this thesis we analyzed weighted reduced order methods for the solution of partial differential equations depending on stochastic parameters.
Firstly, we analyzed the already existing weighted reduced basis method for the evaluation of statistics of solutions of stochastic partial differential equations. We considered two multivariate parametrizations of a diffusion problem, with different random space dimension. The weighted RB method has proved to be numerically efficient and reliable at reducing the cost of computation for the evaluation of statistics of the solutions. Moreover the results showed the importance of choosing a well representative (in the sense of the parameters distribution) training sample set.
Secondly, our main effort has been to understand how to properly modify a different reduced order method. We proposed and analyzed a weighted POD method. The basic idea was to assign different weight to samples, according to the probability distribution function. Since the purpose of POD method is to minimize the approximation error in a L^2 norm over the parameter space, we decided to choose samples and respective weights according to a quadrature formula. We used both Monte-Carlo and tensor product quadrature rules. For low dimensional random spaces, all the various quadrature formulas we applied (Monte-Carlo, Clenshaw-Curtis and Gauss quadrature) showed the same results. Nevertheless, for higher dimensional random spaces and more concentrated probability distribution, the numerical results highlighted the importance of choosing rules which provide a well representative set of nodes. The best performance has been obtained with quadrature rules tailored for the probability distribution (Beta), i.e., Monte-Carlo and Gauss-Jacobi rules. We compared these results with those of RB method.
To reduce the computational effort in the offline stage of POD, we tested Smolyak quadrature rules. The use of Smolyak algorithms was derived from the stochastic collocation method, which makes use of sparse grid Lagrange interpolation. This method showed that is possible to actually reduce the computational effort for weighted POD based on Gauss quadrature with orthogonal polynomials with respect to density weight (Gauss-Jacobi in our case), maintaining the same accuracy.
Finally, we applied weighted POD to a linear elasticity problem, which is a vec-

torial problem. The tests show the same performance of the scalar cases. This could be the starting point to study the application of weighted reduced order methods to non-linear elastic equations.
In all our tests, we considered the case of a Beta distribution. This choice was motivated by the the fact that the Beta density is unimodal and with compact support. However, other distributions could be considered as well. For example, Gauss formulas for diverse distributions are available in literature and they can be applied in these cases.
All the computations have been carried out using the `RBniCS` library, which is based on `FEniCS`. We properly modified the library, allowing to select the desired distribution and weighting method for RB and POD methods. Moreover we wrote additional functions which allow to compute the nodes and weights sets of the tensor product, for the chosen univariate rules.
Possible future investigations could concern applications to problems with more involved stochastic dependence, as well as non-affinely parametrized problems. The latter ones could require the use of an ad hoc weighted empirical interpolation technique. Another problem, would be that of providing accurate estimation for the error. Such estimation were obtained for linear elliptic coercive problems in [9], but it would be useful to generalize them to different problems. Finally, the proposed tests and methodology could also be used as the first step to study non-linear problems.

Appendix A

A.1 Univariate quadrature rules

In this section we present some methods of numerical integration. In particular we will focus on integration of univariate functions $f : I \subseteq \mathbb{R} \to \mathbb{R}$, where $I \subseteq \mathbb{R}$ is an interval. The rules we discuss are usually referred to as *quadrature* rules and have the form

$$\int_I f(x)\, d\mu(x) \simeq \mathcal{U}(f) \doteq \sum_{k=1}^{m} w_k f(x_k),$$

for a sequence of positive weights $\{w_k\}_{k=1}^m$ and a sequence of nodes $\{x_k\}_{k=1}^m \subseteq I$. In the following, we assume that the measure μ is of the form $d\mu(x) = W(x)dx$, where W is a non-negative function such that $x \mapsto W(x)x^k$ is integrable in I for all $k \in \mathbb{N}$; we will refer to W as the weight function. We say that the quadrature rule $\mathcal{U}$ has *order of polynomial exactness* $n \in \mathbb{N}$ if $\int_I p(x)\, d\mu(x) = \mathcal{U}(p)$ for every $p \in \mathbb{P}_n^1$, where $\mathbb{P}_n^1$ denotes the space of all univariate polynomials of order at most n. The following lemma holds (for a proof, see [45], p. 166).

Lemma A.1.1. *No quadrature rule with n distinct nodes in the interior of* $\operatorname{supp}(W)$ *can have order of polynomial exactness* $2n$ *or greater.*

A classical to way to define a quadrature rule is, given a set of nodes $x_1, \ldots, x_n$, to approximate the function f, of which we want to compute the integral, with its Lagrange interpolation in the nodes $x_1, \ldots, x_n$, and then to calculate the integral of this function. That is:

$$\int_I f(x)\, d\mu(x) \simeq \mathcal{Q}(f) \doteq \int_I I(f)(x)\, d\mu(x) = \sum_{k=1}^{m} f(x_k) \int_I l_k(x)\, d\mu(x), \qquad \text{(A.1)}$$

where $I(f) = \sum_{k=1}^m f(x_k) l_k(x)$ is the Lagrange interpolation of the function f and l_i are the Lagrange polynomials

$$l_i(x) = \prod_{\substack{1 \le j \le n \\ j \neq i}} \frac{x - x_j}{x_i - x_j}.$$

Thus, equation (A.1) defines a quadrature formula with nodes x_k and weights $w_k = \int_I l_k(x)\, d\mu(x)$. Clearly, such a formula has order of polynomial exactness at

least n. In section A.1.1 we describe how to obtain a method of the form (A.1) with maximum order of polynomial exactness, while in section A.1.2 we describe a powerful method for integration with respect to Lebesgue measure.

A.1.1 Gauss quadrature rule

Gauss quadrature rule are such that both the nodes and the weights are chosen so as to maximize the order of polynomial exactness of the quadrature formula. This method is based on the choice of a system $\mathcal{Q} = \{q_n : n \in \mathbb{N}\}$ of orthogonal polynomials for μ, that is, for $n \in \mathbb{N}$, q_n is a polynomial of degree exactly n such that

$$\int_I p(x) q_n(x)\, d\mu(x) = 0 \qquad \text{for all } p \in \mathbb{P}_n^1.$$

One can show that, for each $n \in \mathbb{N}$, q_n has exactly n distinct roots in I (for a proof, see [45], p. 146). The n-points Gauss quadrature rule is defined as

$$\mathcal{Q}_n^G(f) \doteq \sum_{i=1}^{n} w_i f(x_i),$$

where $x_1, dots, x_n$ are the roots of q_n and the weights $w_1, \ldots, w_n$ are given in terms of Lagrange basis polynomial l_i for the nodes $x_1, \ldots, x_n$ by

$$w_i \doteq \int_I l_i(x)\, d\mu(x) = \int_I \prod_{\substack{1 \le j \le n \\ j \neq i}} \frac{x - x_j}{x_i - x_j}\, d\mu(x),$$

for $i = 1, \ldots, n$. With such a choice of nodes and weights, the n-points Gauss quadrature rule has maximum order of polynomial exactness $2n - 1$ (for a proof of this fact, see [45], p. 170).
In section A.3 we describe the sets of orthogonal polynomials for some weight functions W.

A.1.2 Clenshaw-Curtis quadrature rules

The Clenshaw-Curtis quadrature formula for the integration of a function $f : [-1, 1] \to \mathbb{R}$ with respect to Lebesgue measure on [-1,1] begins with a change of variables:

$$\int_{-1}^{1} f(x)\, dx = \int_0^{\pi} f(\cos\theta) \sin\theta\, d\theta.$$

Now suppose that f has a cosine series

$$f(\cos\theta) = \frac{a_0}{2} + \sum_{k=1}^{\infty} a_k \cos(k\theta)$$

where the cosine series coefficients are given by

$$a_k = \frac{2}{\pi} \int_0^{\pi} f(\cos\theta) \cos(k\theta)\, d\theta.$$

If so, then

$$\int_0^\pi f(\cos\theta)\sin\theta\,d\theta = a_0 + \sum_{k=1}^{\infty}\frac{2a_{2k}}{1-(2k)^2}.$$

The Clenshaw-Curtis formula of order n is defined as the approximation

$$\int_{-1}^{1} f(x)\,dx \simeq \mathcal{Q}_n^C(f) \doteq a_0 + \sum_{k=1}^{n}\frac{2a_{2k}}{1-(2k)^2}$$

Note that we can re-write the above formula as

$$\mathcal{Q}_n^C(f) = \mathbf{w}\cdot\mathbf{a}$$

where we put

$$\mathbf{a} = \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix}, \qquad \mathbf{w} = \begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{pmatrix}, \qquad w_j = \begin{cases} 1, & j = 0, \\ \frac{2}{1-j^2}, & j \neq 0 \text{ even}, \\ 0, & j \text{ odd}. \end{cases}$$

The coefficients a_k can be approximated as

$$a_k \simeq \frac{2}{n}\sum_{\substack{j=0\\ *}}^{n} f(\cos\theta_j)\cos(k\theta_j), \tag{A.2}$$

where $*$ in the sum indicates that the first and the last terms are to be weighted by a factor of $1/2$, and where

$$\theta_j = \frac{j\pi}{n}.$$

Introducing some convenient shorthand notations, we can write equation (A.2) in the form

$$a_k \simeq \sum_{j=0}^{n}\Lambda_{kj} f(t_j),$$

where

$$t_j = \cos\frac{\pi j}{n}, \qquad \Lambda_{kj} = \begin{cases} \frac{1}{n}\cos\left(\frac{jk\pi}{n}\right), & j = 0, \\ \frac{2}{n}\cos\left(\frac{jk\pi}{n}\right), & j = 1, 2, \ldots, n-1, \\ \frac{1}{n}\cos\left(\frac{jk\pi}{n}\right), & j = n. \end{cases}$$

Therefore we can write

$$\mathbf{a} \simeq \mathbf{\Lambda}\mathbf{f},$$

where $\mathbf{\Lambda} = (\Lambda_{kj})_{kj}$ and $\mathbf{f} = (f(t_0), \ldots, f(t_n))$. It follows that the Clenshaw-Curtis quadrature formula can be written as

$$\mathcal{Q}_n^C(f) = \sum_{k=0}^{n}\omega_k f(t_k), \tag{A.3}$$

where we denote $\omega_k = (\mathbf{w}^T \mathbf{\Lambda})_k$. Note that the nodes of the formula A.3 are the Chebyshev nodes. Moreover it is possible to prove the following explicit formula for the weights:

$$\omega_k = \begin{cases} \frac{1}{n^2-1}, & k = 0, \\ \frac{2}{n}\left[1 + \left\{\sum_{j=1}^{n/2-1} \left(\frac{2}{1-4j^2}\right) \cos\left(\frac{2kj\pi}{n}\right)\right\} + \frac{\cos k\pi}{1-n^2}\right], & k = 1, \ldots, n-1 \\ \frac{1}{n^2-1}, & k = n. \end{cases}$$

In contrast to Gaussian quadrature, which evaluates the integrand at $n+1$ points and exactly integrates polynomials up to degree $2n+1$, Clenshaw-Curtis quadrature evaluates the integrand at $n+1$ points and exactly interpolates polynomials only up to degree n. However, in practice, the fact that Clenshaw-Curtis quadrature has lower order of polynomial exactness is not of great concern, and has accuracy comparable to Gaussian quadrature for 'most' integrands (which are ipso facto not polynomials). The work [46] presented numerical evidence that the 'typical' error for both Gauss and Clenshaw-Curtis quadrature of an integrand in $\mathcal{C}^k$ is of the order $\frac{1}{(2n)^k k}$. Moreover the weights of the Clenshaw-Curtis rule can be computed in $\mathcal{O}(n \log n)$ time, while classical methods for computing weights of Gauss formula takes $\mathcal{O}(n^2)$.

A.2 Multivariate quadrature rules

In this section we describe two possible ways to calculate multi-dimensional integrals, i.e., integrals of the form

$$\int_\Gamma f(x_1, \ldots, x_d)\, d\mu(x_1, \ldots, x_d)$$

where $\Gamma \subseteq \mathbb{R}^d$. The first method it is just the natural extension of univariate rules to multi-dimensional domains. The second one is known as Monte-Carlo integration and works drawing samples from the measure against which the integrand is to be integrated.

A.2.1 Tensor product rules

Suppose that $\Gamma = \prod_{i=1}^d I_i$, for some intervals $I_i \subseteq \mathbb{R}$, and that $d\mu(x_1, \ldots, x_d) = W_1(x_1) \cdots W_d(x_d)\, dx_1 \cdots dx_d$. Then the first, obvious, strategy to try is to treat d-dimensional integration as a succession of d one-dimensional integrals and apply a chosen univariate quadrature formula d times. This method can be described in terms of tensor products of univariate quadrature operators. Let $\mathcal{U}_i : L^1_{\mu_i}(I_i) \to \mathbb{R}$ be univariate quadrature operators, where $L^1_{\mu_i}(I_i)$ is the set of the functions integrable with respect to the measure $d\mu_i(x_i) = W(x_i)dx_i$, for $i = 1, \ldots, d$. Then we can define the tensor product operator $\mathcal{U} : L^1_\mu(\Gamma) \to \mathbb{R}$ as

$$\mathcal{U} = \mathcal{U}_1 \otimes \cdots \otimes \mathcal{U}_d. \tag{A.4}$$

If the operators $\mathcal{U}_i$ are defined as

$$\mathcal{U}_i(f) = \sum_{k=1}^{m_i} w_i^k f(x_i^k)$$

then we can write (A.4) explicitly as

$$\mathcal{U}(f) = \sum_{k_1=1}^{m_1} \cdots \sum_{k_d=1}^{m_d} w_1^{k_1} \cdots w_d^{k_d} f(x_1^{k_1}, \ldots, x_d^{k_d}).$$

In general, when the univariate rules use n nodes the error for an integrand in $\mathcal{C}^r$ using a tensor product rule is $\mathcal{O}(n^{-r/d})$. The main drawback of tensor product rule is that, if we use univariate rules with n nodes, then the tensor product rule uses $N = n^d$ nodes, which very rapidly leads to an impractically large number of integrand evaluations for even moderately large values of n and d.

A.2.2 Monte-Carlo methods

Monte-Carlo methods are, in essence, an application of the Law of Large Numbers (LLN). Recall that the LLN states that if $Y^1, Y^2, \ldots$ are independently and identically distributed according to the law of a random variable Y with finite expectation $\mathbb{E}[Y]$, then the sample average

$$S_n = \frac{1}{n} \sum_{i=1}^{n} Y^i$$

converges a.s. to $\mathbb{E}[Y]$ as $n \to +\infty$. Suppose now that we want to compute

$$\mathbb{E}[f(X)] = \int_\Gamma f(x)\, d\mu(x).$$

where X is a random variable distributed according to the probability measure μ on $\Gamma \subseteq \mathbb{R}^d$. Assuming that one can generate independent and identically distributed samples $X^1, X^2, \ldots$ from the probability measure μ, the n^{th} Monte-Carlo approximation is

$$\mathbb{E}[f(X)] \simeq S_n(f) \doteq \frac{1}{n} \sum_{i=1}^{n} f(X^i)$$

To obtain an error estimate for such Monte Carlo integrals, we simply apply Chebyshev's inequality to $S_n(f)$, which has expected value $\mathbb{E}[S_n(f)] = \mathbb{E}[f(X)]$ and variance

$$\mathrm{Var}[S_n(f)] = \frac{1}{n^2} \sum_{i=1}^{n} \mathrm{Var}[f(X)] = \frac{\mathrm{Var}[f(X)]}{n},$$

to obtain that, for any $t \geq 0$,

$$\mathrm{P}\left\{|S_n(f) - \mathbb{E}[f(X)]| \geq t\right\} \leq \frac{\mathrm{Var}[f(X)]}{nt^2}.$$

That is, for any $\varepsilon \in (0,1]$, with probability at least $1-\varepsilon$ with respect to the n Monte-Carlo samples, the Monte Carlo average $S_n(f)$ lies within $(\mathrm{Var}[f(X)])^{1/2}$ of the true expected value $\mathrm{E}[f(X)]$. Thus, for a fixed integrand f, the error decays like $n^{-1/2}$ regardless of the dimension of the domain of integration and of the smoothness of f, and this is major advantage of Monte Carlo integration. However, the slowness of the $n^{-1/2}$ decay rate is a major limitation of Monte-Carlo methods.
In the case that we wish to evaluate an expected value for some integrand $f(X)$, where $X \sim \mu$, but can only easily draw samples from some other measure ν, one approach is to re-weight the samples of ν: if the density $\frac{d\mu}{d\nu}$ exists and is computationally accessible, then we can estimate $\mathbb{E}[f(X)]$ via

$$\mathbb{E}[f(X)] = \int_\Gamma f(x)\frac{d\mu}{d\nu}(x)\,d\mu(x) \simeq \frac{1}{n}\sum_{i=1}^{n} f(Y^i)\frac{d\mu}{d\nu}(Y^i),$$

where $Y^1, \dots, Y^n$ are independent and identically ν-distributed.

A.3 Some classes of orthogonal polynomials

We report here some systems of orthogonal polynomials, with their respective weights.

Legendre polynomials

The Legendre polynomials are orthogonal on the interval $(-1,1)$ with respect to the weight function $w(x)=1$. They can be defined by:

$$P_n(x) = \frac{(-1)^n}{2^n n!}\frac{\partial^n}{\partial x^n}[(1-x^2)^n], \qquad n \in \mathbb{N}.$$

They can be described by the recurrent relation

$$\begin{aligned} P_0(x) &= 1, \qquad P_1(x) = x, \\ P_{n+1}(x) &= \frac{2n+1}{n+1}xP_n(x) - \frac{n}{n+1}P_{n-1}(x), \qquad n \geq 1. \end{aligned}$$

The following orthogonality relation holds

$$\int_{-1}^{1} P_m(x)P_n(x)\,dx = \frac{2}{2n+1}\delta_{nm}, \qquad n,m \in \mathbb{N}.$$

Hermite polynomials

The Hermite polynomials are orthogonal on the interval $(-\infty,+\infty)$ with respect to the weight function $w(x) = e^{-x^2}$. They can be defined by:

$$P_n(x) = (-1)^n e^{x^2}\frac{\partial^n}{\partial x^n}e^{-x^2}, \qquad n \in \mathbb{N}.$$

They can be described by the recurrent relation

$$\begin{aligned} P_0(x) &= 1, \qquad P_1(x) = 2x, \\ P_{n+1}(x) &= 2xP_n(x) - 2nP_{n-1}(x), \qquad n \geq 1. \end{aligned}$$

The following orthogonality relation holds

$$\int_{-1}^{1} P_m(x)P_n(x)e^{-x^2}\,dx = 2^n n!\sqrt{\pi}\delta_{nm}, \qquad n, m \in \mathbb{N}.$$

Jacobi polynomials

The Jacobi polynomials are orthogonal on the interval $(-1, 1)$ with respect to the weight function $w(x) = (1-x)^\alpha(1+x)^\beta$, for some fixed $\alpha, \beta > -1$. They can be defined by:

$$P_n(x) = \frac{(-1)^n}{2^n n!}(1-x)^{-\alpha}(1+x)^{-\beta}\frac{\partial^n}{\partial x^n}[(1-x)^{n+\alpha}(1+x)^{n+\beta}], \qquad n \in \mathbb{N}.$$

The following orthogonality relation holds

$$\int_{-1}^{1} P_m(x)P_n(x)\,dx = \frac{2^{\alpha+\beta+1}\Gamma(n+\alpha+1)\Gamma(n+\beta+1)}{(2n+\alpha+\beta+1)\Gamma(n+\alpha+\beta+1)n!}\delta_{nm}, \qquad n, m \in \mathbb{N},$$

where $\Gamma : \mathbb{R}^+ \to \mathbb{R}$ is the Euler Gamma function defined as

$$\Gamma(x) = \int_0^{+\infty} t^{x-1}e^{-t}\,dt, \qquad x > 0.$$

Chebyshev polynomials

For $x \in [-1, 1]$ the Chebyshev polynomials $P_n^1(x)$ of the first kind and the Chebyshev polynomials $P_n^2(x)$ of the second can be defined by:

$$P_n^1(x) = \cos(n\theta) \quad \text{and} \quad P_n^2(x) = \frac{\sin((n+1)\theta)}{\sin\theta}, \quad x = \cos\theta, \quad n \in \mathbb{N}.$$

They can be described by the recurrent relation

$$\begin{aligned} P_0^1(x) = P_0^2(x) &= 1, \qquad P_1^1(x) = x, \qquad P_1^2(x) = 2x, \\ P_{n+1}^i(x) &= 2xP_n^i(x) - P_{n-1}^i(x), \qquad n \geq 1, \quad i = 1, 2. \end{aligned}$$

The following orthogonality relations hold

$$\int_{-1}^{1} \frac{1}{\sqrt{1-x^2}} P_m^i(x)P_n^i(x)\,dx = \frac{\pi}{2}\delta_{nm}, \qquad n, m \in \mathbb{N}, \quad i = 1, 2.$$

Laguerre polynomials

The Legendre polynomials are orthogonal on the interval $(0, +\infty)$ with respect to the weight function $w(x) = e^{-x}x^{\alpha}$, for a fixed $\alpha > 0$. They can be defined by:

$$P_n(x) = \frac{1}{n!}e^x x^{-\alpha}\frac{\partial^n}{\partial x^n}[e^{-x}x^{n+\alpha}], \qquad n \in \mathbb{N}.$$

The following orthogonality relation holds

$$\int_0^{+\infty} e^{-x}x^{\alpha}P_m(x)P_n(x)\,dx = \frac{\Gamma(n+\alpha+1)}{n!}\delta_{nm}, \qquad n, m \in \mathbb{N}.$$

A.4 Proof of Theorem 3.2.1

In this section we use the same notation introduced in section 3.2. Before proving theorem 3.2.1, we prove a preliminary result concerning difference operators in tensor product operations.

Proposition A.4.1. *Let $\alpha \in \mathbb{N}^n$ and $\alpha \geq \mathbf{1}$. Then*

$$\bigotimes_{i=1}^{n}\Delta_{\alpha_i}^{(i)} = \sum_{\substack{\gamma\in\{0,1\}^n \\ \alpha-\gamma\geq\mathbf{1}}}(-1)^{|\gamma|_1}\bigotimes_{i=1}^{n}U_{\alpha_i-\gamma_i}^{(i)}.$$

Proof. We apply induction on the dimension n. In the elementary case $n = 1$ we just to verify the following:

$$\Delta_1^{(1)} = U_1^{(1)} = (-1)^0 U_{1-0}^{(1)};$$
$$\Delta_i^{(1)} = U_i^{(1)} - U_{i-1}^{(1)} = (-1)^0 U_{i-0}^{(1)} + (-1)^1 U_{i-1}^{(1)}, \quad i \geq 2$$

Now let us suppose that the claim holds for some $n \geq 1$. Let $\alpha \in \mathbb{N}^{n+1}$ and $\alpha \geq \mathbf{1}$. If $\alpha_{d+1} \neq 1$, then we get by direct computation

$$\begin{aligned}\sum_{\substack{\gamma\in\{0,1\}^{n+1} \\ \alpha-\gamma\geq\mathbf{1}}}(-1)^{|\gamma|_1}\bigotimes_{i=1}^{n+1}U_{\alpha_i-\gamma_i}^{(i)} &= \sum_{\substack{\gamma\in\{0,1\}^{n} \\ \alpha-\gamma\geq\mathbf{1}}}(-1)^{|\gamma|_1+0}\left(\bigotimes_{i=1}^{n}U_{\alpha_i-\gamma_i}^{(i)}\right)\otimes U_{\alpha_{n+1}-0}^{(n+1)} \\ &+ \sum_{\substack{\gamma\in\{0,1\}^{n} \\ \alpha-\gamma\geq\mathbf{1}}}(-1)^{|\gamma|_1+1}\left(\bigotimes_{i=1}^{n}U_{\alpha_i-\gamma_i}^{(i)}\right)\otimes U_{\alpha_{n+1}-1}^{(n+1)} \\ &= \sum_{\substack{\gamma\in\{0,1\}^{n} \\ \alpha-\gamma\geq\mathbf{1}}}(-1)^{|\gamma|_1}\left(\bigotimes_{i=1}^{n}U_{\alpha_i-\gamma_i}^{(i)}\right)\otimes \Delta_{\alpha_{n+1}}^{(n+1)}.\end{aligned}$$

The induction hypotesis implies therefore that

$$\sum_{\substack{\gamma\in\{0,1\}^{n+1} \\ \alpha-\gamma\geq\mathbf{1}}}(-1)^{|\gamma|_1}\bigotimes_{i=1}^{n+1}U_{\alpha_i-\gamma_i}^{(i)} = \left(\bigotimes_{i=1}^{n}\Delta_{\alpha_i}^{(i)}\right)\otimes\Delta_{\alpha_{n+1}^{(n+1)}} = \bigotimes_{i=1}^{n+1}\Delta_{\alpha_i}^{(i)}.$$

If $\alpha_{n+1} = 1$, then we substitute $U^{(n+1)}_{\alpha_{n+1}-1} = 0$ in the computation above and we arrive at the same conclusion. This proves the claim. □

We can now prove theorem 3.2.1.

Theorem 3.2.1. *The operator $\mathcal{Q}^n_q$ defining the Smolyak quadrature rule can be written as*

$$\mathcal{Q}^n_q = \sum_{\substack{\max\{n,q-n+1\}\leq|\alpha|_1\leq k \\ \alpha\in\mathbb{N}^n,\ \alpha\geq\mathbf{1}}} (-1)^{q-|\alpha|_1}\binom{d-1}{k-|\alpha|_1}\bigotimes_{i=1}^{n} U^{(i)}_{\alpha_i}.$$

Proof. Thanks to theorem we have the following representation of operator (3.8):

$$\mathcal{Q}^n_q = \sum_{\substack{|\alpha|_1\leq q \\ \alpha\in\mathbb{N}^n}} \sum_{\substack{\gamma\in\{0,1\}^n \\ \alpha-\gamma\geq\mathbf{1}}} (-1)^{|\gamma|_1}\bigotimes_{i=1}^{n} U^{(i)}_{\alpha_i-\gamma_i}.$$

Changing the order of the summation we get:

$$\mathcal{Q}^n_q = \sum_{\gamma\in\{0,1\}^n} \sum_{\substack{|\alpha|_1\leq q \\ \alpha\in\mathbb{N}^n,\ \alpha-\gamma\geq\mathbf{1}}} (-1)^{|\gamma|_1}\bigotimes_{i=1}^{n} U^{(i)}_{\alpha_i-\gamma_i}.$$

We can now replace the summation variable α by $\beta = \alpha - \gamma$ with the conditions $\beta \geq \mathbf{1}$ and $|\beta|_1 \leq k - |\gamma|_1$. Then we can change the order of summation obtaining

$$\mathcal{Q}^n_q = \sum_{\substack{|\beta|_1\leq q \\ \beta\in\mathbb{N}^n,\ \beta\geq\mathbf{1}}} \sum_{\substack{\gamma\in\{0,1\}^n \\ |\gamma|_1\leq q-|\beta|_1}} (-1)^{|\gamma|_1}\bigotimes_{i=1}^{n} U^{(i)}_{\beta_i}. \tag{A.5}$$

Moreover it holds that

$$\begin{aligned}
\sum_{\substack{\gamma\in\{0,1\}^n \\ |\gamma|_1\leq q-|\beta|_1}} (-1)^{|\gamma|_1} &= \sum_{i=0}^{\min\{n,q-|\beta|_1\}} (-1)^i \sum_{\substack{\gamma\in\{0,1\}^n \\ |\gamma|_1=i}} 1 \\
&= \sum_{i=0}^{\min\{n,q-|\beta|_1\}} (-1)^i \cdot \Big|\{\gamma\in\{0,1\}^n \,:\, |\gamma|_1 = i\}\Big| \\
&= \sum_{i=0}^{\min\{n,q-|\beta|_1\}} (-1)^i \binom{n}{i}.
\end{aligned}$$

The term above vanishes whenever $n \leq q - |\beta|_1$, so we can discard these multi-indixes. Recalling that $\beta \geq \mathbf{1}$ we can equivalently assume $|\beta|_1 \geq \max\{n, q-n+1\}$ in (A.5). Since it holds that (see [49], formula 0.151.4)

$$\sum_{i=0}^{k}(-1)^i\binom{d}{k} = (-1)^k\binom{d-1}{k}, \qquad \text{for } k\geq 0 \text{ and } d > k,$$

we finally get

$$\sum_{\substack{\gamma\in\{0,1\}^n \\ |\gamma|_1\leq q-|\beta|_1}} (-1)^{|\gamma|_1} = (-1)^{q-|\beta|_1}\binom{n-1}{q-|\beta|_1},$$

which concludes the proof. □

A.5 Two analysis results

We report in this section two results used in Chapter 4. The first one is a result of distribution theory. For a proof we remand to [44], p. 60.

Theorem A.5.1. *Let Ω be an open connected subset of $\mathbb{R}^d$. If $m \geq 1$ is an integer and $T \in \mathcal{D}'(\Omega)$ is a distribution such that*

$$\partial^\alpha T = 0, \qquad \forall\, |\alpha|_1 = m$$

then T is a polynomial of degree $\leq m-1$

The second one is a classical results about compact imbeddings of Sobolev spaces. For a proof we remand to [7], p. 285.

Theorem A.5.2 (Rellich-Kondrašov imbedding theorem)**.** *Let Ω be a domain in $\mathbb{R}^d$ and let $p \in [1, +\infty)$. Then the following compact imbeddings hold:*

$$\begin{aligned}
&W^{1,p}(\Omega) \hookrightarrow L^q(\Omega) \qquad \forall\, q \in [1, p^*), \quad \textit{where } \frac{1}{p^*} = \frac{1}{p} - \frac{1}{d}, \quad \textit{if } p < d,\\
&W^{1,p}(\Omega) \hookrightarrow L^q(\Omega) \qquad \forall\, q \in [p, +\infty), \quad \textit{if } p = d,\\
&W^{1,p}(\Omega) \hookrightarrow C^0(\overline{\Omega}) \qquad \textit{if } p > d.
\end{aligned}$$

As a special case of the Rellich-Kondrašov theorem, note that the compact imbedding

$$H^1(\Omega) \hookrightarrow L^2(\Omega)$$

always holds, independently of the dimension d.

Bibliography

[1] R. A. Adams and J. Fournier. *Sobolev Spaces*. Academic Press, 2003.

[2] F. Ballarin, A. Sartori, and G. Rozza. RBniCS - reduced order modelling in FEniCS. http://mathlab.sissa.it/rbnics, 2015.

[3] M. Barrault, Y. Maday, N.C. Nguyen, and A.T. Patera. An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*, 339(9):667–672, 2004.

[4] V. Bathelmann, E. Novak, and K. Ritter. High dimensional polynomial interpolation on sparse grids. *Advances in Computational Mathematics*, 4(12):273–288, 2000.

[5] S. Boyaval, C. Le Bris, T. Leliévre, Y. Maday, N. C. Nguyen, and A. T. Patera. Reduced basis techniques for stochastic problems. *Archives of Computational Methods in Engineering*, 4(17):435–454, 2010.

[6] S. Boyaval, C. Le Bris, Y. Maday, N. C. Nguyen, and A. T. Patera. A reduced basis approach for variational problems with stochastic parameters: Application to heat conduction with variable robin coefficient. *Computer Methods in Applied Mechanics and Engineering*, 41(198):3187–3206, 2009.

[7] H. Brezis. *Functional Analysis, Sobolev Spaces and Partial Differential Equations*. Springer, 2010.

[8] P. Chen. *Model Order Reduction Techniques for Uncertainty Quantification Problems*. PhD thesis, École polytechnique fédérale de Lausanne EPFL, 2014.

[9] P. Chen, A. Quarteroni, and G. Rozza. A weighted reduced basis method for elliptic partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 51(6):3163–3185, 2013.

[10] P. Chen, A. Quarteroni, and G. Rozza. Comparison between reduced basis and stochastic collocation methods for elliptic problems. *Journal of Scientific Computing*, 1(59):187–216, 2014.

[11] P. Chen, A. Quarteroni, and G. Rozza. Reduced order methods for uncertainty quantification problems. Technical Report 03, ETH Zurich, SAM Report, 2015, Submitted.

[12] P. Chen, A. Quarteroni, and G. Rozza. Multilevel and weighted reduced basis method for stochastic optimal control problems constrained by stokes equations. *Numerische Mathematik*, 1(133):67–102, 2016.

[13] Y. Chen, J. S. Hesthaven, Y. Maday, and J. Rodrìguez. A monotonic evaluation of lower bounds for inf-sup stability constants in the frame of reduced basis approximations. *Comptes Rendus Mathematique*, 23(346):1295–1300, 2008.

[14] Y. Chen, J. S. Hesthaven, Y. Maday, and J. Rodrìguez. Certified reduced basis methods and output bounds for the harmonic maxwell's equations. *SIAM Journal on Scientific Computing*, 2(32):970–996, 2010.

[15] Y. Chen, J. S. Hesthaven, Y. Maday, and J. Rodríguez. Improved successive constraint method based a posteriori error estimate for reduced basis approximation of 2D Maxwell's problem. *M2AN Math. Model. Numer. Anal.*, 43(6):1099–1116, 2009.

[16] F. Chinesta, A. Huerta, G. Rozza, and K. Willcox. Model order reduction. In *Encyclopedia of Computational Mechanics*. 2016.

[17] P. G. Ciarlet. *Mathematical Elasticity*. Elsevier, 1993.

[18] P. G. Ciarlet. *The Finite Element Method for Elliptic Problems*. Siam, 2002.

[19] J.W. Demmel. *Applied numerical linear algebra*. Society for Industrial and Applied Mathematics (SIAM), Philadelphi, PA, 1997.

[20] J.L. Eftang, M.A. Grepl, and A.T. Patera. A posteriori error bounds for the empirical interpolation method. *Comptes Rendus Mathematique*, 51(1):28–58, 2010.

[21] G. Fishman. *Monte Carlo: concepts, algorithms and applications*. Springer Science & Business Media, 2013.

[22] F. Gelsomino and G. Rozza. Comparison and combination of reduced-order modelling techniques in 3D parametrized heat transfer problems. *Math. Comput. Model. Dyn. Syst.*, 17(4):371–394, 2011.

[23] T. Gerstner and M. Griebel. Numerical integration using sparse grids. *Numerical Algorithms*, 3-4(18):209–232, 1998.

[24] M.A. Grepl, Y. Maday, N.C. Nguyen, and A.T. Patera. Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations. *M2AN Mathematical Modelling and Numerical Analysis*, 41(3):575–605, 2007.

[25] J. S. Hesthaven, B. Stamm, and S. Zhang. Certified reduced basis method for the electric field integral equation. *SIAM Journal on Scientific Computing*, (32):A1777–A1799, 2012.

[26] J.S. Hesthaven, G. Rozza, and B. Stamm. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. Springer, 2016.

[27] M. Holtz. *Sparse Grid Quadrature in High Dimensions with Applications in Finance and Insurance*. Springer, 2010.

[28] P. Huynh, D. Knezevic, Y. Chen, J. S. Hestaven, and A. Patera. A natural-norm successive constraint method for inf-sup lower bounds. *Computer Methods in Applied Mechanics and Engineering*, (199):1963–1975, 2010.

[29] P. Huynh, G. Rozza, S. Sen, and A. T. Patera. A successive constraint linear optimization method for lower bounds of parametric coercivity and inf-sup stability constants. *Comptes Rendus Mathematique*, 8(345):473–478, 2007.

[30] I.T. Jolliffe. *Principal Component Analysis*. Springer New York, 2002.

[31] Vesa Kaarnioja. Smolyak quadrature. Master's thesis, University of Helsinki, 2013.

[32] Michel Loève. *Probability theory. Vol. II*. Springer-Verlag, 1978.

[33] N. C. Nguyen, G. Rozza, D. B. P. Huynh, and A. T. Patera. *Reduced Basis Approximation and a Posteriori Error Estimation for Parametrized Parabolic PDEs: Application to Real-Time Bayesian Parameter Estimation*, pages 151–177. John Wiley & Sons, Ltd, 2010.

[34] J. A. Nitsche. On korn's second inequality. *RAIRO-Analyse numérique*, 3(15):237–248, 1981.

[35] F. Nobile, R. Tempone, and C. G. Webster. A sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 5(46):2309–2345, 2008.

[36] E. Novak and K. Ritter. High dimensional integration of smooth functions over cubes. *Numerische Mathematik*, 1(75):79–97, 1996.

[37] A. T. Patera and G. Rozza. Reduced basis approximation and a posteriori error estimation for parametrized partial differential equations. version 1.0. augustine.mit.edu, 2007.

[38] A. Quarteroni. *Numerical Models for Differential Problems*, volume 8 of *MS&A*. Springer-Verlag Italia, Milano, 2013.

[39] A. Quarteroni, G. Rozza, and A. Manzoni. Certified reduced basis approximation for parametrized partial differential equations and applications. *Journal of Mathematics in Industry*, 1(3):1–44, 2011.

[40] A. Quarteroni, R. Sacco, and F Saleri. *Numerical mathematics*, volume 37 of *Texts in Applied Mathematics*. Springer-Verlag, Berlin, 2007.

[41] A. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations*. Springer, 1994.

[42] G. Rozza, D.B.P. Huynh, and A.T. Patera. Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations: application to transport and continuum mechanics. *Archives of Computational Methods in Engineering*, 3(15):229–275, 2008.

[43] S. Salsa. *Equazioni a derivate parziali: Metodi, modelli e applicazioni.* Number 98. Springer, 2016.

[44] L. Schwartz. Théorie des distributions. *Actualités Scientifiques et Industrielles, Institut de Mathématique, Université de Strasbourg*, 2(1), 1950.

[45] Sullivan T. *Introduction to Uncertainty Quantification.* Springer, 2015.

[46] L. N. Trefethen. Is gauss quadrature better than clenshaw-curtis? *SIAM review*, 1(50):67–87, 2008.

[47] G. W. Wasilkowski. Explicit cost bounds of algorithms for multivariate tensor product problems. *Journal of Complexity*, 1(11):1–56, 1995.

[48] D. Xiu and J. S. Hesthaven. High-order collocation methods for differential equations with random inputs. *SIAM Journal on Scientific Computing*, 3(27):1118–1139, 2006.

[49] D. Zwillinger, editor. *Table of integrals, series, and products.* Elsevier, 2014.