Trabajo de Fin de Máster:

Métodos de Orden Reducido para Ecuaciones Diferenciales/ Reduced Order Methods for Differential Equations

Alejandro Bandera Moreno
Máster Univesitario en Matemáticas

Tutorizado por:
Soledad Fernández García
Macarena Gómez Mármol
Diciembre 2020

DPTO. ECUACIONES DIFERENCIALES Y ANÁLISIS NUMÉRICO - US

To my parents

Resumen

Gracias al desarrollo tecnológico de los ordenadores durante el siglo XX, las simulaciones numéricas se han convertido en un campo fundamental en la mayoría de las Ciencias, y son particularmente importantes en el campo de las Ecuaciones Diferenciales, como consecuencia de sus múltiples aplicaciones. Las simulaciones nos ofrecen una plataforma virtual para la realización de tests, de gran utilidad para comprender la dinámica de distintos sistemas, o para la construcción de simuladores de todo tipo, entre otras muchas aplicaciones. Para que los modelos sean realistas y puedan ser utilizados de manera efectiva, es necesario resolver problemas complejos con una alta precisión (*high-fidelity*), lo que puede conllevar un alto coste computacional. El modelado de orden reducido entra en juego ya que reemplaza el sistema *high-fidelity* por otro en el que se puede evaluar la solución para cualquier nuevo caso con un coste computacional bajo, mientras se mantienen las propiedades cualitativas y cuantitativas principales de la solución. En este trabajo consideraremos dos métodos para el modelado de orden reducido: *Proper Orthogonal Decomposition* (POD) y una adaptación del método *Greedy*. Como aplicación, comprobaremos en primer lugar los resultados en el caso de las Ecuaciones en Derivadas Parciales (EDPs), donde realizaremos modelado de orden reducido respecto de los parámetros físicos. Por último, aplicaremos el método POD para aproximar los datos obtenidos al resolver Sistemas Diferenciales Ordinarios (SDO) como primer paso para la creación del sistema reducido asociado.

Abstract

Thanks to the technological development of computers in the 20th century, numerical simulations have become a fundamental field in most of the Sciences, and they are especially important in the field of Differential Equations, because of their multiple applications. Simulations provide us a virtual platform to perform tests, with great utility in order to understand the dynamics of different systems, or to build all kind of simulators, among many other applications. For the models to be realistic and can be used effectively, it is necessary to solve complex problems with high precision (*high-fidelity*), this can carry a high computational cost. Reduced order modelling comes into play as it replaces the *high-fidelity* system with another, in which we can evaluate the solution for any new parameter instance with low computational cost, while capturing its main qualitative and quantitative features. In this work, we consider two methods of reduced order modelling, *Proper Orthogonal Decomposition* and an adaptation of the *Greedy* method. As an application, we will first check the results for Partial Differential Equations (PDEs), we apply reduced order modelling with respect to physical parameters. At last, we apply the POD method in order to approximate data obtained by solving Systems of Differential Equations (SDEs), as the first step to build the associated reduced system.

Contents

Introduction

Due to the achievements of numerical analysis and scientific computing, numerical simulations of Differential Equations have gained a great importance. They provide a virtual platform to perform tests, with great utility in order to understand the dynamics of different systems or to build all kind of simulators, among many other applications.
The increase of computational power along with the improvements of algorithms for solving large linear systems, make possible numerical simulations of complex, multiscale and Multiphysics phenomena, through *high-fidelity* approximation techniques, such as the finite element method. A high-fidelity approximation means a solution as faithful as the original solution. However, this can be quite demanding, as they involve up to $O(10^6 - 10^9)$ degrees of freedom and several CPU on powerful hardware parallel architectures. Therefore, they can be prohibitive when expecting to deal quickly and efficiently with repetitive solutions of PDEs, which is the case of parametrized PDEs. There can be different types of parameters involved in a parametrized PDE, physical parameters, those that change the formulation of the PDE, geometric parameters, those that modify the definition of the domain, temporal parameters, etc.
There are problems that show patterns in function of the parameters, with no significant qualitative change inside a range of parameters. For this class of problems, reduced order modelling is any approach aimed at replacing the high-fidelity problem by one with lower numerical complexity. Then, we can evaluate the solution, for any new parameter instance, with a low computational cost independent of the original dimension. This is key to the success of any Reduced Order Model (ROM). Reduced order modelling quickly captures essential features of a structure. In an early stage, the most basic properties must already be present in the smaller approximation and when the reduction process is stopped, all necessary properties of the original model must be captured with sufficient precision.
Proper Orthogonal Decomposition (POD) and Reduced Basis (RB) are remarkable instances, as they exploit the parametric dependence by combining a handful of high-fidelity solutions (or snapshots) computed for a set of parameter values. POD relies in the Singular Value Decomposition and in the fact that the larger eigenvalues

retain more information about the behaviour of the solution, while RB relies in an error estimator and in the local maximum improvement of the approximation. The strong computational speedup achievable by these methods allows to tackle a wide range of problems, due to very short CPU times and limited storage capacities demanded.
In order to study reduced order models, we first need to recall some fundamental results of functional analysis, essential for a correct variational formulation of a broad variety of boundary value problems. The outline of this work is as follows. In Chapter 1, we review three types of boundary problems: strongly coercive, weakly coercive and saddle-point problems.
In Chapter 2, we introduce the high-fidelity discretization techniques for PDEs and give a major example of a high-fidelity approximation technique, the Finite Element Method (FEM). Then, we describe the RB approximation, that approximates the high-fidelity solution of a given elliptic PDE, for any choice of the parameter within a described parameter set and is computationally much cheaper. We also introduce two examples, Galerkin and Least-Squares Reduced Basis methods. Furthermore, we analyse the offline/online decomposition to reduce the computational complexity, and we give an error estimator for the solution of these methods.
After that, Chapter 3 is entirely dedicated to the construction of Reduced Basis Spaces. We present two different methods in order to generate the Reduced Basis spaces, construction by Proper Orthogonal Decomposition (POD) using a Singular Value Decomposition analysis (SVD) and Greedy algorithm, that consists in an iterative sampling from the parameter space fulfilling at each step a suitable optimality criterion, which relies on the a posteriori error estimate.
We apply the theoretical results to a parametrized elliptic PDE in Chapter 4 (POD) and Chapter 5 (Greedy). We consider the 2-dimensional Laplace problem occurring in surfaces of different materials characterized by a particular parameter, with both homogeneous Dirichlet boundary conditions and non-homogeneous Neumann boundary conditions.
Finally, we discuss the use of the POD technique for model order reduction in the field of the Ordinary Differential Equations. More precisely, we apply the POD technique on the temporal variable, considered as the parameter, so we end with a low number of modes, which linear combination gives a representation of the solution, the first step in order to create the reduced order model of a System of Differential Equations. In particular, we study two nonlinear stiff problems: the High Irradiance Responses (HIRES) problem included in [10], and the Intracellular Calcium Concentration (ICC) problem, studied in [7].

Modelos de Orden Reducido para Ecuaciones Diferenciales

Reduced Order Methods for Differential Equations

Chapter 1

Formulation, Analysis and Approximation of Variational Problems

In this chapter, we introduce some basic results of functional analysis, essential for a correct variational formulation of a broad variety of boundary value problems. First, we review three types of boundary problems:

- Strongly coercive problems (section 1.1).
- Weakly coercive problems (section 1.2), where a relaxation of the coercivity property is performed.
- Saddle-point problems (section 1.3), where the solution is not restricted to just one space.

For each of them, we follow the following scheme,

1. Introduction of the functional setting.
2. Well-posedness results.
3. Numerical approximation.
4. Results of existence, uniqueness, stability and convergence.
5. Presentation of an algebraic form.

For the construction of this chapter we have mainly used reference [12].

1.1 Strongly Coercive Problems

First of all, we start studying strongly coercive problems, the simplest of all, in order to present the main ideas for the study of boundary value problems.

1.1.1 Formulation

Let V be a Hilbert space along with its norm. By introducing a bilinear form $a : V \times V \to \mathbb{R}$ and a linear functional $f \in V'$, we consider the following abstract variational problem:

$$\begin{cases} \text{Find } u \in V \text{ such that} \\ \qquad a(u,v) = f(v), \quad \forall v \in V. \end{cases} \tag{1.1}$$

The bilinear form $a(\cdot,\cdot)$ is said to be

- Continuous if there exists $\gamma > 0$ such that

  $$|a(u,v)| \leq \gamma \|u\|_V \|v\|_V, \quad \forall u, v \in V,$$

  where γ is named the continuity constant of $a(\cdot,\cdot)$.

- Coercive if there exists $\alpha > 0$ such that

  $$a(v,v) \geq \alpha \|v\|_V^2, \quad \forall v \in V,$$

  where α is named the coercivity constant of $a(\cdot,\cdot)$.

- Symmetric if

  $$a(u,v) = a(v,u), \quad \forall u, v \in V.$$

The following result states under which assumptions the problem (1.1) is well-posed.

Theorem 1.1 (*Lax-Milgram*)

Let V be a Hilbert space, $a : V \times V \to \mathbb{R}$ a continuous, coercive, bilinear form and $f : V \to \mathbb{R}$ a bounded linear functional.
Then (1.1) has unique solution and satisfies the stability estimate

$$\|u\|_V \leq \frac{1}{\alpha}\|f\|_{V'}.$$

Moreover, if a is symmetric, problem (1.1) is equivalent to a minimization problem for the functional:

$$J(v) = \frac{1}{2}a(v,v) - f(v).$$

■

Now that we know the assumptions necessary for the problem (1.1) to be well-posed, we turn to its numerical approximation.

1.1.2 Approximation

Problem (1.1) as formulated is infinite-dimensional, making it impossible to solve computationally. We build a finite dimensional approximation for the problem (1.1), as follows.
Let for every $h > 0$, $V_h \subset V$ be a finite-dimensional subspace of V, such that $\dim V_h = N_h$; the subscript h is related to a characteristic discretization parameter. The approximate problem or *Galerkin problem* of (1.1) takes the form

$$\begin{cases} \text{Find } u_h \in V_h \text{ such that} \\ \qquad a(u_h, v_h) = f(v_h), \quad \forall v_h \in V_h. \end{cases} \tag{1.2}$$

The solution u_h of this problem is often known as the *Galerkin approximation* of u. From the following property, the *Galerkin problem* (1.2) is said to be *strongly consistent.*

Remark 1.1

Since $V_h \subset V$, the exact solution u satisfies the weak problem (1.1) for each element $v = v_h \in V_h$, hence we have

$$a(u, v_h) = f(v_h), \quad \forall v_h \in V_h.$$

From this equality and the problem statement (1.2), we obtain that u_h satisfies the Galerkin orthogonality

$$a(u - u_h, v_h) = 0, \quad \forall v_h \in V_h.$$

Remark 1.2

Indeed, should $a(\cdot,\cdot)$ be symmetric, the previous equality can be interpreted as the orthogonality condition with respect to the scalar product induced by the form $a(\cdot,\cdot)$ between the approximation error, $u - u_h$, and the subspace V_h. Thus, u_h can be seen as the orthogonal projection of u onto V_h.

With these premises, the following result of existence, uniqueness, stability and convergence of the numerical approximation easily follows.

Theorem 1.2

Let the space V, the bilinear form $a(\cdot,\cdot)$ and the linear functional $f(\cdot)$ satisfy

the hypotheses of Theorem 1.1 and let $V_h \subset V$ be a closed subspace.
Then, $a(\cdot,\cdot)$ is continuous on $V_h \times V_h$ with continuity constant $\gamma_h \leq \gamma$ and coercive on $V_h \times V_h$ with coercivity constant $\alpha_h \geq \alpha$.
Therefore, for every $h > 0$, the discretized problem (1.2) has a unique solution $u_h \in V_h$, that satisfies the stability estimate

$$\|u_h\|_V \leq \frac{1}{\alpha_h}\|f\|_{V'}.$$

Furthermore, if $u \in V$ denotes the unique solution of (1.1), the following optimal error inequality is satisfied

$$\|u - u_h\|_V \leq \frac{\gamma}{\alpha} \inf_{v_h \in V_h} \|u - v_h\|_V,$$

where γ and α are the continuity constant and the coercivity constant of $a(\cdot,\cdot)$, respectively. ■

Remark 1.3

Optimality means that the error in a finite-dimensional approximation $\|u-u_h\|_V$ is bounded from above by the error of the best approximation out of the same finite-dimensional subspace, multiplied by a constant independent of h.

However, this condition is not sufficient to ensure the convergence $\|u - u_h\|_V \to 0$ when $h \to 0$. In fact, an additional property related to the approximability of the discrete spaces is required. In order for the method to converge it will be sufficient to require that, for $h \to 0$, the space V_h tends to "fill" the entire space V, or that V_h is a inner approximation of V.
In the following subsection, we present a technique to simplify calculations when solving problem (1.2)

1.1.3 Algebraic Form

Trying to solve problem (1.2) directly is difficult, so in this subsection, we introduce a simpler way to study the numerical approximation problem (1.2), which is the one we will use in our applications. It is immediate to prove that,

Remark 1.4

The discrete variational problem (1.2) is equivalent to the solution of a linear system of equations.

Indeed, if we denote by $\{\varphi^j\}_{j=1}^{N_h}$ a basis for the finite-dimensional space V_h, then every $v_h \in V_h$ has a unique representation,

$$v_h = \sum_{j=1}^{N_h} v_h^{(j)} \varphi^j, \quad \text{with } \mathbf{v} = (v_h^{(1)}, \ldots, v_h^{(N_h)})^T \in \mathbb{R}^{N_h}.$$

By setting $u_h = \sum_{j=1}^{N_h} u_h^{(j)} \varphi^j$, and denoting by $\mathbf{u}_h$ the vector having as components the unknown coefficients $u_h^{(j)}$, problem (1.2) is equivalent to

$$\begin{cases} \text{Find } \mathbf{u}_h \in \mathbb{R}^{N_h} \text{ such that} \\ \qquad \displaystyle\sum_{j=1}^{N_h} a(\varphi^j, \varphi^i) u_h^{(j)} = f(\varphi^i), \quad \forall\, i = 1, \ldots, N_h, \end{cases}$$

that is,

$$\mathbb{A}_h \mathbf{u}_h = \mathbf{f}_h,$$

where $\mathbb{A}_h \in \mathbb{R}^{N_h \times N_h}$ is the matrix of the system with elements $(\mathbb{A}_h)_{ij} = a(\varphi^j, \varphi^i)$ and $\mathbf{f}_h \in \mathbb{R}^{N_h}$ is the vector with components $(\mathbf{f}_h)_i = f(\varphi^i)$.

Remark 1.5

The coercivity condition states that for every $h > 0$ the matrix of the system is positive definite. Furthermore, if $a(\cdot, \cdot)$ is symmetric so is the matrix of the system.
Other properties, such as the condition number or the sparsity structure of the matrix of the system depend on the chosen basis of V_h.

In the context of *a posteriori* error estimation for reduced basis approximations is required to compute the best discrete coercivity constant,

$$\alpha_h = \inf_{v_h \in V_h} \frac{a(v_h, v_h)}{\|v_h\|_V^2}.$$

Our goal now is to find an algebraic representation of α_h. To this end, let us denote by $\mathbb{X}_h$ the symmetric positive definite matrix associated to the scalar product in V, i.e.

$$(\mathbb{X}_h)_{ij} = (\varphi^i, \varphi^j)_V,$$

so that,

$$\|v_h\|_V^2 = \mathbf{v}^T \mathbb{X}_h \mathbf{v}, \quad \forall v_h \in V_h.$$

We can now rewrite the best discrete coercivity constant as

$$\alpha_h = \inf_{\mathbf{v} \in \mathbb{R}^{N_h}} \frac{\mathbf{v}^T \mathbb{A}_h \mathbf{v}}{\mathbf{v}^T \mathbb{X}_h \mathbf{v}},$$

i.e. α_h is in fact the minimum of a generalized Rayleigh quotient. Since, for any $\mathbf{v} \in \mathbb{R}^{N_h}$, we have,

$$\mathbf{v}^T \mathbb{A}_h \mathbf{v} = \mathbf{v}^T \mathbb{A}_h^S \mathbf{v}, \text{ with } \mathbb{A}_h^S = \frac{1}{2}(\mathbb{A}_h + \mathbb{A}_h^T) \text{ the symmetric part of } \mathbb{A}_h,$$

we obtain that α_h is the smallest eigenvalue λ such that $(\lambda, \mathbf{v}) \in \mathbb{R}^+ \times \mathbb{R}^{N_h}, \mathbf{v} \neq 0$, satisfy

$$\mathbb{A}_h^S \mathbf{v} = \lambda \mathbb{X}_h \mathbf{v}.$$

By left-multiplying this equality by $\mathbb{X}_h^{-1/2}$ and performing the change of variable $\mathbf{w} = \mathbb{X}_h^{1/2}\mathbf{v}$, we then obtain

$$\alpha_h = \lambda_{min}(\mathbb{X}_h^{-1/2} \mathbb{A}_h^S \mathbb{X}_h^{-1/2}).$$

In the following section we relax the coercivity condition in order to consider a more general boundary value problem.

1.2 Weakly Coercive (or Inf-Sup Stable) Problems

Asking a bilinear form to be coercive is a very restrictive assumption, so we relax the coercivity property to its weak formulation. By doing this, we will be able to model a wider variety of boundary value problems. In this section we study weakly coercive problems, also named as inf-sup stable problems.

1.2.1 Formulation

Given two Hilbert spaces V and W along with their dual V' and W', respectively, the bilinear form $a : V \times W \to \mathbb{R}$ and the linear functional $f \in W'$, we consider the following abstract variational problem,

$$\begin{cases} \text{Find } u \in V \text{ such that} \\ \qquad a(u, w) = f(w), \quad \forall w \in W. \end{cases} \tag{1.3}$$

The bilinear form $a(\cdot, \cdot)$ is said to be

- Continuous on $V \times W$ if there exists $\gamma > 0$ such that

  $$|a(v, w)| \leq \gamma \|v\|_V \|w\|_W, \quad \forall v \in V, w \in W,$$

  where γ is named the continuity constant of $a(\cdot, \cdot)$.

- weakly coercive (or inf-sup stable) if there exists a constant $\beta > 0$ such that

$$\inf_{v\in V}\sup_{w\in W}\frac{a(v,w)}{\|v\|_V\|w\|_W} \geq \beta,$$

where β is named the inf-sup stability constant of $a(\cdot,\cdot)$, and

$$\inf_{w\in W}\sup_{v\in V}\frac{a(v,w)}{\|v\|_V\|w\|_W} > 0.$$

Remark 1.6

Weakly coercive conditions can be reformulated as

$$\exists\beta > 0 : \sup_{w\in W}\frac{a(v,w)}{\|w\|_W} \geq \beta\|v\|_V, \quad \forall v \in V.$$

and

$$\textit{if } w \in W \textit{ is such that } a(v,w) = 0, \quad \forall v \in V, \textit{ then } w = 0.$$

The following theorem, known as Nečas theorem [11], shows under which assumptions weakly coercive problems are well-posed.

Theorem 1.3 (*Nečas*)

Let V and W be two Hilbert spaces, $a : V \times W \to \mathbb{R}$ a continuous, weakly coercive bilinear form on $V \times W$, and $f : W \to \mathbb{R}$ a bounded linear functional on W.
Then, the variational problem (1.3) has a unique solution which satisfies the stability estimate

$$\|u\|_V \leq \frac{1}{\beta}\|f\|_{W'}.$$

■

Remark 1.7

Since strong coercivity implies weak coercivity, Lax-Milgram theorem (Theorem 1.1) is a special case of Nečas theorem (Theorem 1.3).

Very often, we deal with noncoercive bilinear forms defined on $V \times V$ (i.e. $W = V$). The following result can be helpful in these cases.

Proposition 1.4

Let us suppose that $H_0^1(\Omega) \subset V \subset H^1(\Omega)$ and the bilinear form $a : V \times V \to \mathbb{R}$ fulfills the following conditions

1. *Gärding inequality. For some constants $\alpha > 0$ and $\lambda > 0$,*

$$a(v,v) \geq \alpha \|v\|^2_{H^1(\Omega)} - \lambda \|v\|^2_{L^2(\Omega)}, \quad \forall v \in V.$$

2. *If $u \in V$ is such that $a(u,v) = 0$ for any $v \in V$, then $u = 0$.*

Then there exists a constant $\beta > 0$ such that

$$\sup_{w \in W} \frac{a(v,w)}{\|w\|_V} \geq \beta \|v\|_V, \quad \forall v \in V,$$

i.e. a is weakly coercive. ■

Thanks to this result, a bilinear form fulfilling Gärding inequality is often referred to as *weakly coercive.*

1.2.2 Approximation

Weakly coercive problems naturally leads to using two different approximation spaces $V_h \subset V$ and $W_h \subset W$. Furthermore, in some cases it may be convenient to introduce different approximation spaces even if $V = W$.
Let $V_h \subset V$ and $W_h \subset W$ be two nontrivial subspaces of V and W, respectively, with $\dim V_h = \dim W_h = N_h < +\infty$. We consider the following variational problem

$$\begin{cases} \text{Find } u_h \in V_h \text{ such that} \\ \qquad a(u_h, w_h) = f(w_h), \quad \forall w_h \in W_h. \end{cases} \tag{1.4}$$

The solution u_h of problem (1.4) is known as the *Petrov-Galerkin approximation* of the solution u of problem (1.3).
The well-posedness of problem (1.4) is guaranteed by the following theorem, known as Babuška theorem [2].

Theorem 1.5 (*Babuška*)

Let the space V and W, the form $a(\cdot,\cdot)$ and the functional $f(\cdot)$ satisfy the hypotheses of Theorem 1.3 and let $V_h \subset V$ and $W_h \subset W$ be two closed subspaces. Then $a(\cdot,\cdot)$ is continuous on $V_h \times W_h$.
Assume also that the bilinear form $a(\cdot,\cdot)$ satisfies the discrete inf-sup condition

$$\beta_h = \inf_{v_h \in V_h} \sup_{w_h \in W_h} \frac{a(v_h, w_h)}{\|v_h\|_V \|w_h\|_W} > 0.$$

Then, for every $h > 0$, problem (1.4) has a unique solution $u_h \in V_h$. Moreover, *that solution satisfies the stability estimate*

$$\|u_h\|_V \leq \frac{1}{\beta_h}\|f\|_{W'},$$

and, if $u \in V$ denotes the unique solution of (1.3), the following optimal error inequality holds

$$\|u - u_h\|_V \leq \frac{\gamma}{\beta_h} \inf_{v_h \in V_h} \|u - v_h\|_V.$$

■

Remark 1.8

While the continuity of the bilinear form $a(\cdot,\cdot)$ over $V_h \times W_h$ (respectively, $V_h \times V_h$) is automatically inherited from the continuity property over $V \times W$ ($V \times V$), we note that:

- *in the coercive case, coercivity of $a(\cdot,\cdot)$ over $V_h \times V_h$ automatically follows from the coercivity property fulfilled over $V \times V$, so that the conformity property $V_h \subset V$ is the only property which is needed to ensure the stability of the discrete variational problem,*

- *in the weakly coercive case, the inclusions $V_h \subset V$ and $W_h \subset W$ are not sufficient to ensure the fulfillment of the discrete inf-sup condition.*

Consequently, the discrete inf-sup assumption must be explicitly required, as done in Theorem (1.5). In other words, V_h and W_h must be built in such a way that the discrete inf-sup condition holds.

1.2.3 Algebraic Form

As done for problem (1.2), we can derive an equivalent algebraic formulation of problem (1.4). We remember that $\{\varphi^j\}_{j=1}^{N_h}$ denotes a basis for the space V_h and we indicate by $\{\phi^j\}_{j=1}^{N_h}$ a basis for the space W_h, so that the functions of V_h and W_h have a unique representation,

$$v_h = \sum_{j=1}^{N_h} v_h^{(j)} \varphi^j, \text{ with } \mathbf{v} = (v_h^{(1)}, \dots, v_h^{(N_h)})^T \in \mathbb{R}^{N_h}, \quad \forall v_h \in V_h,$$
$$w_h = \sum_{j=1}^{N_h} w_h^{(j)} \phi^j, \text{ with } \mathbf{w} = (w_h^{(1)}, \dots, w_h^{(N_h)})^T \in \mathbb{R}^{N_h}, \quad \forall w_h \in W_h.$$

By setting $u_h = \sum_{j=1}^{N_H} u_h^{(j)} \varphi^j$ and denoting by $\mathbf{u}_h$ the vector having as components the unknown coefficients $u_h^{(j)}$, (1.4) is equivalent to

$$\begin{cases} \text{Find } \mathbf{u}_h \in \mathbb{R}^{N_h} \text{ such that} \\ \qquad \displaystyle\sum_{j=1}^{N_h} a(\varphi^j, \phi^i) u_h^{(j)} = f(\phi^i), \quad \forall i = 1, \dots, N_h, \end{cases}$$

that is,

$$\mathbb{A}_h \mathbf{u}_h = \mathbf{f}_h,$$

where $(\mathbb{A}_h)_{ij} = a(\varphi^j, \phi^i)$ and $(\mathbf{f}_h)_i = f(\phi^i)$, for $1 \leq i, j \leq N_h$.

Remark 1.9

The inf-sup condition states that the matrix $\mathbb{A}_h$ is nonsingular for every $h > 0$, without any further implication on the sign of its eigenvalues.

As we did for the coercivity constant, we want to find an algebraic formula suitable to compute the discrete inf-sup constant β_h. To this end, we first introduce the following operator.

Definition 1.10

The discrete supremizer operator $T_h : V_h \to W_h$ is defined as

$$(T_h v_h, w_h)_W = a(v_h, w_h), \quad \forall w_h \in W_h.$$

Then, we obtain the following characterization of the discrete inf-sup constant,

$$\beta_h = \inf_{v_h \in V_h} \frac{\|T_h v_h\|_W}{\|v_h\|_V},$$

or, equivalently,

$$\beta_h^2 = \inf_{v_h \in V_h} \frac{\|T_h v_h\|_W^2}{\|v_h\|_V^2}.$$

Let us denote by $\mathbb{Y}_h$ the matrix asociated with the scalar product in W as we did for the scalar product in V, that is,

$$(\mathbb{Y}_h)_{ij} = (\phi^j, \phi^i)_W,$$

so that,

$$\|w_h\|_W = \mathbf{w}^T \mathbb{Y}_h \mathbf{w}, \quad \forall w_h \in W_h.$$

If we denote by $\mathbf{t}$ the vector of coefficients in the expansion of $T_h v_h \in W_h$ with respect to the basis of W_h, by the definition of the supremizer we obtain,

$$\mathbf{w}^T \mathbb{Y}_h \mathbf{t} = \mathbf{w}^T \mathbb{A}_h^T \mathbf{v}, \quad \forall \mathbf{w} \in \mathbb{R}^{N_h},$$

whence $\mathbf{t} = \mathbb{Y}_h^{-1}\mathbb{A}_h\mathbf{v}$ since $\mathbf{w}$ is arbitrary. Substituting in the characterization of the square of the discrete inf-sup constant, we obtain,

$$\beta_h^2 = \inf_{\mathbf{v}\in\mathbb{R}^{N_h}} \frac{\mathbf{t}^T\mathbb{Y}_h\mathbf{t}}{\mathbf{v}^T\mathbb{X}_h\mathbf{v}} = \inf_{\mathbf{v}\in\mathbb{R}^{N_h}} \frac{\mathbf{v}^T\mathbb{A}_h^T\mathbb{Y}_h^{-T}\mathbb{Y}_h\mathbb{Y}_h^{-1}\mathbb{A}_h\mathbf{v}}{\mathbf{v}^T\mathbb{X}_h\mathbf{v}} = \inf_{\mathbf{v}\in\mathbb{R}^{N_h}} \frac{\mathbf{v}^T\mathbb{A}_h^T\mathbb{Y}_h^{-1}\mathbb{A}_h\mathbf{v}}{\mathbf{v}^T\mathbb{X}_h\mathbf{v}}.$$

We have expressed the square of the discrete inf-sup constant β_h^2 as the minimum of the generalized Rayleigh quotient of the symmetric matrix $\mathbb{A}_h^T\mathbb{Y}_h^{-1}\mathbb{A}_h$. Then, for the inf-sup constant we obtain $\beta_h = \sqrt{\lambda_{min}}$, where λ_{min} is the smallest eigenvalue λ such that $(\lambda, \mathbf{v}) \in \mathbb{R}^+ \times \mathbb{R}^{N_h}$, $\mathbf{v} \neq 0$, satisfy,

$$\mathbb{A}_h^T\mathbb{Y}_h^{-1}\mathbb{A}_h\mathbf{v} = \lambda\mathbb{X}_h\mathbf{v}.$$

Remark 1.11

The singular values of a generic square matrix $\mathbb{B} \in \mathbb{R}^{n\times n}$ are defined as

$$\sigma_i(\mathbb{B}) = \sqrt{\lambda_i(\mathbb{B}^T\mathbb{B})}, 1 \leq i \leq n.$$

By left-multiplying the previous equation by $\mathbb{X}_h^{-1/2}$ and performing the change of variable $\mathbf{w} = \mathbb{X}_h^{1/2}\mathbf{v}$, we then obtain,

$$\beta_h = \sigma_{min}(\mathbb{Y}_h^{-1/2}\mathbb{A}_h\mathbb{X}_h^{-1/2}).$$

Remark 1.12

In particular, if $V_h = W_h$ and the matrix $\mathbb{A}_h$ is symmetric, the discrete inf-sup constant becomes

$$\beta_h = \lambda_{min}(\mathbb{X}_h^{-1/2}\mathbb{A}_h\mathbb{X}_h^{-1/2}),$$

as in the strong coercive case.

Remark 1.13

In an analogous way, we can also characterize the discrete continuity constant as

$$\gamma_h = \sup_{v_h\in V_h}\sup_{w_h\in W_h} \frac{a(v_h, w_h)}{\|v_h\|_V\|w_h\|_W} = \sigma_{max}(\mathbb{Y}_h^{-1/2}\mathbb{A}_h\mathbb{X}_h^{-1/2}).$$

To finish this chapter, we now turn to a very relevant class of problems, called *mixed variational problems* or *saddle point problems*. Several problems can be formulated and analysed within this framework.

1.3 Saddle-Point Problems

The previous results for strongly (1.1) and weakly coercive problems (1.3) do not hold if our solution is not restricted to just one space. So we come to another variety of problems, that we describe subsequently.

1.3.1 Formulation

Given two Hilbert spaces X and Q along with their dual X' and Q', respectively, the bilinear forms $d : X \times X \to \mathbb{R}, b : X \times Q \to \mathbb{R}$, and the linear functionals $f_1 \in X'$ and $f_2 \in Q'$, we consider the following mixed variational (or saddle-point) problem,

$$\begin{cases} \text{Find } (x,p) \in X \times Q \text{ such that} & \\ \qquad d(x,w) + b(w,p) = f_1(w) & \forall w \in X, \\ \qquad b(x,q) = f_2(q) & \forall q \in Q. \end{cases} \tag{1.5}$$

The following theorem due to Brezzi [4] establishes sufficient conditions for the saddle-point problem (1.5) to be well-posed.

Theorem 1.6 (*Brezzi*)

Under the following assumptions:

1. *Continuity of the bilinear form $d(\cdot,\cdot)$,*

   there exists a constant $\gamma_d > 0$ such that

$$|d(x,w)| \leq \gamma_d \|x\|_X \|w\|_X, \quad \forall x, w \in X,$$

2. *Weakly coercivity of the bilinear form $d(\cdot,\cdot)$ on $X_0 \times X_0$,*

   there exists a constant $\alpha_0 > 0$ such that

$$\inf_{x \in X_0} \sup_{w \in X_0} \frac{d(x,w)}{\|x\|_X \|w\|_X} \geq \alpha_0, \quad \inf_{w \in X_0} \sup_{x \in X_0} \frac{d(x,w)}{\|x\|_X \|w\|_X} > 0,$$

3. *Continuity of the bilinear form $b(\cdot,\cdot)$,*

   there exists a constant $\gamma_b > 0$ such that

$$|b(w,q)| \leq \gamma_b \|w\|_X \|q\|_Q, \quad \forall w \in X, q \in Q,$$

4. *The bilinear form $b(\cdot,\cdot)$ satisfies the inf-sup condition*

$$\beta^S = \inf_{q \in Q} \sup_{w \in X} \frac{b(w,q)}{\|w\|_X \|q\|_Q} \geq \beta_0^S > 0,$$

there exists a unique solution $(x, p) \in X \times Q$ to the mixed variational problem (1.5). Moreover the following stability estimates hold:

$$\|x\|_X \leq \frac{1}{\alpha}\left[\|f_1\|_{X'} + \frac{\alpha_0 + \gamma_d}{\beta_0^S}\|f_2\|_{Q'}\right],$$

$$\|p\|_Q \leq \frac{1}{\beta^S}\left[\left(1 + \frac{\gamma_d}{\alpha_0}\right)\|f_1\|_{X'} + \frac{\gamma_d(\alpha_0 + \gamma_d)}{\alpha_0 + \beta_0^S}\|f_2\|_{Q'}\right].$$

■

Where, in order to simplify the notation we have introduced the following subspace of X.

Definition 1.14

We define the subspace of X where the bilinear form $b(\cdot, q)$ is null for every $q \in Q$

$$X_0 = \{w \in X : b(w, q) = 0, \quad \forall q \in Q\} \subset X.$$

There is a connection between saddle point problems (1.5) and weakly coercive problems (1.3), as stated in the following remark.

Remark 1.15

Mixed variational problems are a special case of weakly coercive problems. In fact, by setting $V = W = X \times Q$, defining the bilinear form $a : V \times V \to \mathbb{R}$

$$a((x, p), (w, q)) = d(x, w) + b(w, p) + b(x, q),$$

and the functional

$$f((w, q)) = f_1(w) + f_2(q),$$

we can rewrite (1.5) in the form of (1.3).

1.3.2 Approximation

Let $X_h \subset X$ and $Q_h \subset Q$ be two subspaces of X and Q, respectively. We consider the Galerkin approximation of the mixed variational problem (1.5)

$$\begin{cases} \text{Find } (x_h, p_h) \in X_h \times Q_h \text{ such that} \\ \qquad d(x_h, w_h) + b(w_h, p_h) = f_1(w_h) \quad \forall w_h \in X_h, \\ \qquad b(x_h, q_h) = f_2(q_h) \qquad\qquad\quad\; \forall q_h \in Q_h. \end{cases} \tag{1.6}$$

As done in the continuous case, we define the following subspace of X_h in order to simplify notation.

Definition 1.16

We define the subspace of X_h where the bilinear form $b(\cdot, q_h)$ is null for every $q_h \in Q$

$$X_0^h = \{w_h \in X_h : b(w_h, q_h) = 0, \quad \forall q_h \in Q_h\} \subset X_h.$$

The well-posedness of (1.6) follows by the discrete counterpart of Brezzi theorem [3], that is,

Theorem 1.7 (*Brezzi*)

Let the space X and Q, the bilinear forms $d(\cdot,\cdot), b(\cdot,\cdot)$ and the functionals $f_1(\cdot)$, $f_2(\cdot)$ satisfy the hypotheses of Theorem 1.6 and let $X_h \subset X$ and $Q_h \subset Q$ be two finite dimensional subspaces.
Then $d(\cdot,\cdot)$ and $b(\cdot,\cdot)$ are continuous on $X_h \times X_h$ and $X_h \times Q_h$, respectively.
Assume that the bilinear form $d(\cdot,\cdot)$ is weakly coercive on $X_0^h \times X_0^h$, i.e.

$$\alpha_h = \inf_{x_h \in X_0^h} \sup_{w_h \in X_0^h} \frac{d(x_h, w_h)}{\|x_h\|_X \|w_h\|_X} \geq \hat{\alpha} > 0,$$

$$\inf_{w_h \in X_0^h} \sup_{w_h \in X_0^h} \frac{d(x_h, w_h)}{\|x_h\|_X \|w_h\|_X} > 0.$$

Moreover, suppose that the bilinear form $b(\cdot,\cdot)$ verifies the discrete inf-sup condition for a suitable constant $\hat{\beta}^S > 0$ independent of h,

$$\beta_h^S = \inf_{q_h \in Q_h} \sup_{w_h \in X_h} \frac{b(w_h, q_h)}{\|w_h\|_X \|q_h\|_Q} \geq \hat{\beta}^S.$$

Then, for every $h > 0$, problem (1.6) has a unique solution $(x_h, p_h) \in X_h \times Q_h$, which satisfies the stability estimates

$$\|x_h\|_X \leq \frac{1}{\hat{\alpha}} \left[\|f_1\|_{X'} + \frac{\hat{\alpha} + \gamma_d}{\hat{\beta}^S} \|f_2\|_{Q'} \right],$$

$$\|p_h\|_Q \leq \frac{1}{\hat{\beta}^S} \left[\left(1 + \frac{\gamma_d}{\hat{\alpha}}\right) \|f_1\|_{X'} + \frac{\gamma_d(\hat{\alpha} + \gamma_d)}{\hat{\alpha} + \hat{\beta}^S} \|f_2\|_{Q'} \right].$$

If $(x, p) \in X \times Q$ denotes the unique solution of (1.5), the following optimal error inequality holds

$$\|x - x_h\|_X + \|p - p_h\|_Q \leq C \left(\inf_{w_h \in X_h} \|x - w_h\|_X + \inf_{q_h \in Q_h} \|p - q_h\|_Q \right),$$

where $C = C(\hat{\alpha}, \hat{\beta}^S, \gamma_d, \gamma_b)$ independent of h. ■

As done for problems (1.2) and (1.4), we can derive an equivalent algebraic formulation of the problem (1.6). We finish this chapter by presenting the algebraic formulation in the following subsection.

1.3.3 Algebraic Form

If $\{\varphi^j\}_{j=1}^{N_h}$ and $\{\eta^j\}_{j=1}^{M_h}$ denote two bases for the finite dimensional spaces X_h and Q_h respectively, being $N_h = \dim X_h$ and $M_h = \dim Q_h$, then (1.6) is equivalent to the following linear system,

$$\begin{bmatrix} \mathbb{D}_h & \mathbb{B}_h^T \\ \mathbb{B}_h & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x}_h \\ \mathbf{p}_h \end{bmatrix} = \begin{bmatrix} \mathbf{f}_{1h} \\ \mathbf{f}_{2h} \end{bmatrix},$$

where $i, j = 1, \ldots, N_h$, $k = 1, \ldots, M_h$,

- $x_h = \sum_{j=1}^{N_h} x_h^{(j)} \varphi^j$, $p_h = \sum_{k=1}^{M_h} p_h^{(k)} \eta^k$, $\mathbf{x}_h$ and $\mathbf{p}_h$ are the vectors having as components the unknown coefficients $x_h^{(j)}, p_h^{(k)}$,
- $(\mathbb{D}_h)_{ij} = d(\varphi^j, \varphi^i)$, $(\mathbb{B}_h)_{kj} = b(\varphi^j, \eta^k)$, are the elements of the matrices $\mathbb{D}_h$ and $\mathbb{B}_h$,
- $\mathbf{f}_{1h}$ and $\mathbf{f}_{2h}$ denotes the vectors with components $(\mathbf{f}_{1h})_i = f_1(\varphi^i)$ and $(\mathbf{f}_{2h})_k = f_2(\eta^k)$.

Remark 1.17

Theorem 1.7 ensures that the matrix appearing in the linear system is nonsingular.

Chapter 2

Basic Reduced Order Methods

In the previous chapter, we have seen that in order to solve a problem computationally, it is necessary to obtain a discretization of the working space. So we introduce the high-fidelity discretization techniques for elliptic PDEs and give a major example of a high-fidelity approximation technique, the Finite Element (FE) method. Although this method approximates the solution, in some cases it is computationally excessive to solve, so we study the Reduced Order Model (ROM) approximation, which is a (Petrov-)Galerkin projection onto a lower dimensional space, the Reduced space, that approximates the high-fidelity solution of a given PDE, for any choice of the parameter within a described parameter set where the behaviour of the solutions does not change qualitatively and is computationally much cheaper. We describe the Reduced Basis methods and introduce two examples,

- Galerkin Reduced Basis method and
- Least-Squares Reduced Basis method.

Furthermore, we analyse the offline/online decomposition to reduce the computational complexity and we give an error estimator for the solution of these methods. For the construction of this chapter we have mainly used reference [12].

2.1 High-Fidelity Technique

In all this chapter we consider

- An open subset Ω of $\mathbb{R}^d$, with $d = 1, 2, 3$.
- $V = V(\Omega)$ along with its norm is a Hilbert space and V' denotes its dual.

- $\boldsymbol{\mu} = (\mu_1, \dots, \mu_P)^T$ is the input parameter vector collecting the physical $\boldsymbol{\mu}_{ph}$, and the geometric ones $\boldsymbol{\mu}_g$.[1]
- The set of all possible inputs $\mathcal{P} \subset \mathbb{R}^P$, is compact. We also consider that the behaviour of the solution does not change qualitatively for different values of the parameter $\boldsymbol{\mu} \in \mathcal{P}$.
- For all $\boldsymbol{\mu} \in \mathcal{P}$, $L(\boldsymbol{\mu}) : V \to V'$ is a second order differential operator, usually the laplacian operator.
- For all $\boldsymbol{\mu} \in \mathcal{P}$, $f(\boldsymbol{\mu}) : V \to \mathbb{R}$ is a linear and continuous form on V.

Remark 2.1

If we have geometrical parameters, $\boldsymbol{\mu}_g$, we consider a change of variables that reformulates the domain to a reference domain $\Omega = \widetilde{\Omega}(\boldsymbol{\mu}_{ref})$, where we solve it.

2.1.1 Parametric Formulation

When trying to solve a PDE depending on a parameter vector we normally face the following problem

$$\begin{cases} \text{Given } \boldsymbol{\mu} \in \mathcal{P}, \text{ find the solution } u(\boldsymbol{\mu}) \in V \text{ of} \\ \qquad L(\boldsymbol{\mu})u(\boldsymbol{\mu}) = f(\boldsymbol{\mu}) \text{ in } V'. \end{cases} \tag{2.1}$$

Usually (2.1) is often referred to as the strong formulation of the problem. Defining the following

- $a(\cdot, \cdot; \boldsymbol{\mu}) : V \times V \to \mathbb{R}$ as

$$a(u, v; \boldsymbol{\mu}) = {}_{V'}\langle L(\boldsymbol{\mu})u, v \rangle_V,$$

- and $f(\cdot; \boldsymbol{\mu}) : V \to \mathbb{R}$ as

$$f(v; \boldsymbol{\mu}) = {}_{V'}\langle f(\boldsymbol{\mu}), v \rangle_V,$$

we can reformulate (2.1) as

$$\begin{cases} \text{Given } \boldsymbol{\mu} \in \mathcal{P}, \text{ find } u(\boldsymbol{\mu}) \in V \text{ such that} \\ \qquad a(u(\boldsymbol{\mu}), v; \boldsymbol{\mu}) = f(v; \boldsymbol{\mu}), \quad \forall v \in V. \end{cases} \tag{2.2}$$

this formulation is called the weak formulation of (2.1). We assume that

[1]Each type of parameter require a different mathematical treatment, in this work we will focus on the physical ones.

- $a(\cdot,\cdot;\boldsymbol{\mu})$ is continuous over $V \times V$, that is, there exists $\overline{\gamma} > 0$ such that

$$\gamma(\boldsymbol{\mu}) = \sup_{v\in V}\sup_{w\in V} \frac{a(v,w;\boldsymbol{\mu})}{\|v\|_V\|w\|_V} < \overline{\gamma}, \quad \forall \boldsymbol{\mu} \in \mathcal{P},$$

  where $\gamma(\boldsymbol{\mu})$ is named the continuity factor of $a(\cdot,\cdot;\boldsymbol{\mu})$.

- $f(\cdot;\boldsymbol{\mu})$ is continuous, that is, there exists $\overline{\gamma}_F > 0$ such that

$$\gamma_F(\boldsymbol{\mu}) = \sup_{w\in V} \frac{f(w;\boldsymbol{\mu})}{\|w\|_V} < \overline{\gamma}_F, \quad \forall \boldsymbol{\mu} \in \mathcal{P}. \tag{2.3}$$

  where $\gamma_F(\boldsymbol{\mu})$ is named the continuity factor of $f(\cdot;\boldsymbol{\mu})$.

Regarding stability, we assume that there exists β_0 such that

$$\beta(\boldsymbol{\mu}) = \inf_{v\in V}\sup_{w\in V} \frac{a(v,w;\boldsymbol{\mu})}{\|v\|_V\|w\|_V} \geq \beta_0,$$

and

$$\inf_{w\in V}\sup_{v\in V} \frac{a(v,w;\boldsymbol{\mu})}{\|v\|_V\|w\|_V} > 0.$$

$\beta(\boldsymbol{\mu})$ is the inf-sup stability factor of $a(\cdot,\cdot;\boldsymbol{\mu})$ and we say that $a(\cdot,\cdot;\boldsymbol{\mu})$ is inf-sup stable.

Proposition 2.1

Provided the continuity properties and the stability assumptions are verified, problem (2.2) admits a unique solution thanks to Theorem 1.3 and the following stability estimate holds for all $\boldsymbol{\mu} \in \mathcal{P}$

$$\|u(\boldsymbol{\mu})\|_V \leq \frac{1}{\beta(\boldsymbol{\mu})}\|f(\cdot;\boldsymbol{\mu})\|_{V'} \leq \frac{1}{\beta_0}\|f(\cdot;\boldsymbol{\mu})\|_{V'}.$$

■

2.1.2 High-Fidelity Discretization

When it is not analytically possible to find a solution for (2.2) we can try building a numerical approximation of the solution. However, as we cannot work in a infinite dimensional space with computers, we need to introduce high-fidelity discretizations, where a finite dimensional approximation of the infinite dimensional space is performed.

The Galerkin high-fidelity approximation of a problem under the form (2.2) reads

$$\begin{cases} \text{Find } u_h(\boldsymbol{\mu}) \in V_h \text{ such that} \\ \qquad a(u_h(\boldsymbol{\mu}), v_h; \boldsymbol{\mu}) = f(v_h; \boldsymbol{\mu}), \quad \forall v_h \in V_h. \end{cases} \tag{2.4}$$

We can also recover the strong formulation of the problem in the finite dimensional space.

Remark 2.2

If the problem is under the form (2.1), then problem (2.4) can be equivalently written as

$$\begin{cases} \textit{Find } u_h(\boldsymbol{\mu}) \in V_h \textit{ such that} \\ \qquad L(\boldsymbol{\mu}) u_h(\boldsymbol{\mu}) = f(\boldsymbol{\mu}) \textit{ in } V'. \end{cases} \tag{2.5}$$

For the discrete problem (2.4) to be well-posed, as done for problem (1.4) we need to assume that there exists $\beta_{0,h} > 0$ such that

$$\beta_h(\boldsymbol{\mu}) = \inf_{v_h \in V_h} \sup_{w_h \in V_h} \frac{a(v_h, w_h; \boldsymbol{\mu})}{\|v_h\|_V \|w_h\|_V} \geq \beta_{0,h}, \quad \forall \boldsymbol{\mu} \in \mathcal{P}. \tag{2.6}$$

We also define the discrete continuity factor of $a(\cdot, \cdot; \boldsymbol{\mu})$

$$\gamma_h(\boldsymbol{\mu}) = \sup_{v_h \in V_h} \sup_{w_h \in V_h} \frac{a(v_h, w_h; \boldsymbol{\mu})}{\|v_h\|_V \|w_h\|_V} \leq \gamma(\boldsymbol{\mu}). \tag{2.7}$$

Proposition 2.2

Under the above assumptions, thank to Theorem 1.5 problem (2.4) admits a unique solution and the following stability estimate holds

$$\|u_h(\boldsymbol{\mu})\|_V \leq \frac{1}{\beta_{0,h}} \|f(\cdot; \boldsymbol{\mu})\|_{V'}.$$

Furthermore, $\forall \boldsymbol{\mu} \in \mathcal{P}$

$$\|u(\boldsymbol{\mu}) - u_h(\boldsymbol{\mu})\|_V \leq \frac{\overline{\gamma}}{\beta_{0,h}} \min_{z_h \in V_h} \|u(\boldsymbol{\mu}) - z_h\|_V.$$

■

As stated in Chapter 1 the Galerkin high-fidelity approximation (2.4) is equivalent to the solution of the following linear system

$$\mathbb{A}_h(\boldsymbol{\mu}) \mathbf{u}_h(\boldsymbol{\mu}) = \mathbf{f}_h(\boldsymbol{\mu})$$

where, if $\{\varphi^j\}_{j=1}^{N_h}$ denotes a basis for V_h, then $(\mathbb{A}_h(\boldsymbol{\mu}))_{ij} = a(\varphi^j, \varphi^i; \boldsymbol{\mu})$ and $(\mathbf{f}_h(\boldsymbol{\mu}))_i = f(\varphi^i, \boldsymbol{\mu})$.
The following remark states an easy way to assemble and solve the linear system equivalent to the high-fidelity approximation (2.4) when affine dependence on the parameters is provided for $\mathbb{A}_h(\boldsymbol{\mu})$ and $\mathbf{f}_h(\boldsymbol{\mu})$.

Remark 2.3

When $\mathbb{A}_h(\boldsymbol{\mu})$ and $\mathbf{f}_h(\boldsymbol{\mu})$ both depend affinely on the parameters, i.e.

$$\mathbb{A}_h(\boldsymbol{\mu}) = \sum_{q=1}^{Q_a} \theta_a^q(\boldsymbol{\mu})\mathbb{A}_h^q, \text{ and } \mathbf{f}_h(\boldsymbol{\mu}) = \sum_{q=1}^{Q_f} \theta_f^q(\boldsymbol{\mu})\mathbf{f}_h^q$$

the linear system can be solved using Algorithm 1.

Algorithm 1 High-Fidelity system assembling and solving

```
1: function SolveHF($\mathbb{A}_h^q, \mathbf{f}_h^q, \theta_a^q, \theta_f^q, \boldsymbol{\mu}$)
2:     $\mathbb{A}_h(\boldsymbol{\mu}) = \mathbb{O}$;   $\mathbf{f}_h(\boldsymbol{\mu}) = \mathbf{0}$
3:     for $q = 1 : Q_a$
4:         $\mathbb{A}_h(\boldsymbol{\mu}) \leftarrow \mathbb{A}_h(\boldsymbol{\mu}) + \theta_a^q(\boldsymbol{\mu})\mathbb{A}_h^q$
5:     end for
6:     for $q = 1 : Q_f$
7:         $\mathbf{f}_h(\boldsymbol{\mu}) \leftarrow \mathbf{f}_h(\boldsymbol{\mu}) + \theta_f^q(\boldsymbol{\mu})\mathbf{f}_h^q$
8:     end for
9:     solve linear system $\mathbb{A}_h(\boldsymbol{\mu})\mathbf{u}_h(\boldsymbol{\mu}) = \mathbf{f}_h(\boldsymbol{\mu})$
10:    return $\mathbf{u}_h(\boldsymbol{\mu})$
11: end function
```

2.1.3 Finite Element Method

In this section, we present the most common method in order to obtain the Galerkin high-fidelity approximation of problems under the form (2.2), the Finite Element Method. This method relays on a triangulation on the domain to discretize the evaluation points and obtain a finite dimensional approximation of the problem. First, we define the most common spaces where we find the solution of problem (2.2).
We normally search for the solution inside the Hilbert space $H^1(\Omega)$, i.e.

Definition 2.4

We define the Hilbert space $H^1(\Omega)$ as

$$H^1(\Omega) = \{u \in L^2(\Omega) : \nabla u \in L^2(\Omega)^d\},$$

along with its norm

$$|u|_{H^1(\Omega)} = \|u\|_{L^2(\Omega)} + \|\nabla u\|_{L^2(\Omega)^d}.$$

In many problems we must enforce the solution to be zero over the boundary of Ω, $\partial\Omega$ so we can define the following Hilbert space:

Definition 2.5

We define the Hilbert space $H_0^1(\Omega)$ as

$$H_0^1(\Omega) = \{u \in H^1(\Omega) : u|_{\partial\Omega} = 0\},$$

along with its norm

$$|u|_{H_0^1(\Omega)} = \|\nabla u\|_{L^2(\Omega)^d}.$$

The spaces $H^1(\Omega)$ and $H_0^1(\Omega)$ are infinite dimensional spaces, so we need to define approximations for both spaces. In order to do so, let us introduce first a triangulation of the domain Ω that depend on a parameter h.

Definition 2.6

A triangulation, $\mathcal{T}_h$ of $\Omega \subset \mathbb{R}^d$ is a subdivision of Ω into d-dimensional simplices (triangles for $d = 2$, tetrahedra for $d = 3$), such that

- *$h = \max\limits_{K\in\mathcal{T}} h_K$, where h_K is the diameter of $K \in \mathcal{T}_h$,*
- *any two simplices in $\mathcal{T}_h$ intersect in a simplex of any lower dimension or not at all,*
- *any bounded set in Ω intersects only finitely many simplices in $\mathcal{T}_h$.*

The most natural strategy to define a finite element space is to consider globally continuous functions that are polynomials of degree r on the single triangles of the triangulation $\mathcal{T}_h$ and null anywhere else, that is, to define the following test spaces

Definition 2.7

Let $\Omega \subset \mathbb{R}^d$ be an open set and V a Hilbert space, we define the test spaces of polynomials of degree $r \in \mathbb{N}$ over a triangulation $\mathcal{T}_h$ of Ω as

$$\begin{aligned} X_h^r &= \{v_h \in \mathcal{C}^0(\overline{\Omega}) : v_h|_K \in \mathbb{P}_r, \quad \forall K \in \mathcal{T}_h\}, & V_h &= V \cap X_h^r, \\ X_{0h}^r &= \{v_h \in X_h^r : v_h|_{\partial\Omega} = 0\}, & V_{0h} &= V \cap X_{0h}^r. \end{aligned}$$

We state in the following theorem that the solution of (1.1) and its high-fidelity approximation satisfy a result of convergence which only depends on the discretization parameter h, and the degree of the polynomials r.

Theorem 2.3

Let $u \in V$ be the exact solution of (1.1) and u_h its finite element approximation of degree r. If $u \in V \subset H^{r+1}(\Omega)$ then the following a priori error estimates hold:

$$|u - u_h|_{H^1(\Omega)} \leq C \left(\sum_{K \in \mathcal{T}_h} h_K^{2r} |u|^2_{H^{r+1}(K)} \right)^{1/2},$$

$$|u - u_h|_{H^1(\Omega)} \leq C h^r |u|_{H^{r+1}(\Omega)}.$$

■

The previous result states that in order to increase accuracy, two different strategies can be pursued:

- decreasing h, i.e. refining the grid.
- increasing r, i.e. using finite elements of higher degree (only if u is regular enough).

However, both strategies increase the computational cost for solving the problem (2.4). In the next section, we present a method in order to reduce the computational costs for solving the problem (2.4).

2.2 Reduced Basis Methods

First of all, we state why and how a Reduced Basis Methods must be constructed. Then, we present two major examples of Reduced Basis methods,

- Galerkin Reduced Basis Method (subsection 2.2.1).
- Least-Squares Reduced Basis Method (subsection 2.2.2).

As we said in the introduction of this chapter, solving the discrete problem using a high-fidelity technique is computationally expensive, as it involves solving a N_h-dimensional linear system, and for that reason, in order to reduce the computational costs, the Reduced Basis approximations have been built.
The idea is to choose a set of $N \ll N_h$ of functions that construct the Reduced Basis and solving the problem in this new basis. Setting a Reduced Basis method entails:

1. Construction of a basis of the reduced space $V_N \subset V_h$.

   We start from a set of high-fidelity solutions, usually called snapshots

   $$\{u_h(\boldsymbol{\mu}^1), \dots, u_h(\boldsymbol{\mu}^N)\},$$

   corresponding to a set of N selected parameters

   $$S_N = \{\boldsymbol{\mu}^1, \dots, \boldsymbol{\mu}^N\} \subset \mathcal{P}.$$

   Then, we generate a set of N functions, called the reduced basis,

   $$\{\zeta_1, \dots, \zeta_N\},$$

   by orthonormatization of the snapshots. Finally, we construct the reduced space

   $$V_N = \mathrm{span}(\zeta_1, \dots, \zeta_N) = \mathrm{span}(u_h(\boldsymbol{\mu}^1), \dots, u_h(\boldsymbol{\mu}^N)).$$

   Remark 2.8

   Generally, the reduced basis functions, ζ_m $m \in \{1, \dots, N\}$, are no longer solutions for the high-fidelity problem (2.4).

2. Computation of the Reduced Basis solution $u_N(\boldsymbol{\mu}) \in V_N$.

   We solve the following linear system

   $$u_N(\boldsymbol{\mu}) = \sum_{m=1}^{N} u_N^{(m)}(\boldsymbol{\mu})\zeta_m,$$

   where $\mathbf{u}_N(\boldsymbol{\mu}) = (u_N^{(1)}(\boldsymbol{\mu}), \dots, u_N^{(N)}(\boldsymbol{\mu})) \in \mathbb{R}^N$, are called the Reduced Basis coefficients or generalized coordinates.

3. Setup of a reduced problem for determining the unknown coefficients $\mathbf{u}_N(\boldsymbol{\mu}) \in \mathbb{R}^N$.

   The reduced problem will consist of a set of N equations that are obtained by imposing N independent conditions.

The latter enforce the orthogonality of the residual of the high-fidelity problem (2.4) computed on the Reduced Basis solution,

$$r(\boldsymbol{\mu}) = f(\boldsymbol{\mu}) - L(\boldsymbol{\mu})u_N(\boldsymbol{\mu}),$$

to the functions of a subspace $W_N \subset V_h$, where W_N is called the test subspace. This yields the following Petrov-Galerkin Reduced Basis (PG-RB) problem

$$\begin{cases} \text{Find } u_N(\boldsymbol{\mu}) \in V_N \text{ such that} \\ \qquad {}_{V'}\langle L(\boldsymbol{\mu})u_N(\boldsymbol{\mu}) - f(\boldsymbol{\mu}), w_N\rangle_V = 0, \quad \forall w_N \in W_N, \end{cases} \tag{2.8}$$

or equivalently,

$$\begin{cases} \text{Find } u_N(\boldsymbol{\mu}) \in V_N \text{ such that} \\ \qquad a(u_N(\boldsymbol{\mu}), w_N; \boldsymbol{\mu}) = f(w_N; \boldsymbol{\mu}), \quad \forall w_N \in W_N. \end{cases} \tag{2.9}$$

We study now two examples of the Reduced Basis Method, the Galerkin and the Least-Squares Reduced Basis Method, corresponding to different choices of the test subspace W_N.

Remark 2.9

The Galerkin Reduced Basis (G-RB) problem correspond to the case $W_N = V_N$.

Remark 2.10

The Least-Squares Reduced Basis (LS-RB) problem correspond to the case $W_N = R_{V_h}^{-1} L(\boldsymbol{\mu}) V_N$, where $R_{V_h} : V_h \to V_h'$ is the Riesz map.

2.2.1 Galerkin Reduced Basis Method

When the reduced space V_N, is chosen as the test subspace, i.e. $W_N = V_N$, we obtain what is named the Galerkin Reduced Basis method.
Given $\boldsymbol{\mu} \in \mathcal{P}$, G-RB approximation of (2.2) reads

$$\begin{cases} \text{Find } u_N(\boldsymbol{\mu}) \in V_N \text{ such that} \\ \qquad a(u_N(\boldsymbol{\mu}), v_N; \boldsymbol{\mu}) = f(v_N; \boldsymbol{\mu}), \quad \forall v_N \in V_N. \end{cases} \tag{2.10}$$

When $a(\cdot, \cdot; \boldsymbol{\mu})$ is coercive for any $\boldsymbol{\mu} \in \mathcal{P}$, a straightforward application of the Lax-Milgram theorem (see Theorem 1.1) provides the following,

Theorem 2.4

Under the assumptions of Theorem 1.1 for any $\boldsymbol{\mu} \in \mathcal{P}$ the G-RB problem (2.10) has a unique solution $u_N(\boldsymbol{\mu}) \in V_N$, which satisfies the stability estimate

$$\|u_N(\boldsymbol{\mu})\|_V \leq \frac{1}{\alpha_N(\boldsymbol{\mu})} \|f(\cdot; \boldsymbol{\mu})\|_{V'},$$

where $\alpha_N(\boldsymbol{\mu}) = \inf_{v \in V_N} \dfrac{a(v, v; \boldsymbol{\mu})}{\|v\|_V^2}$ is the stability factor. ■

If the high-fidelity problem (2.4) is well-posed, so is the G-RB problem (2.10) as stated in the following remark,

Remark 2.11

Since $V_N \subset V_h$, then

$$\alpha_N(\boldsymbol{\mu}) \geq \alpha_h(\boldsymbol{\mu}), \quad \forall \boldsymbol{\mu} \in \mathcal{P}.$$

That is, the well-posedness of the G-RB problem (2.10) is therefore inherited from of the high-fidelity problem (2.4).

If we denote the energy norm as $\|\cdot\|_{\boldsymbol{\mu}} = \sqrt{a(\cdot,\cdot;\boldsymbol{\mu})}$, we obtain the following optimal property,

Proposition 2.5

If $a(\cdot,\cdot;\boldsymbol{\mu})$ is symmetric and coercive, then the solution $u_N(\boldsymbol{\mu}) \in V_N$, satisfies,

$$u_N(\boldsymbol{\mu}) = \arg\min_{v \in V_N} \|u_h(\boldsymbol{\mu}) - v\|^2_{\boldsymbol{\mu}}$$

■

We can choose other norms defined over V.

Remark 2.12

By choosing the V-norm instead of the energy norm, we obtain,

$$\|u_h(\boldsymbol{\mu}) - u_N(\boldsymbol{\mu})\|_V \leq \left(\frac{\gamma_h(\boldsymbol{\mu})}{\alpha_N(\boldsymbol{\mu})}\right)^{1/2} \inf_{w \in V_N} \|u_h(\boldsymbol{\mu}) - w\|_V.$$

If $a(\cdot,\cdot;\boldsymbol{\mu})$ is non-coercive over $V_h \times V_h$, then we cannot use Lax-Milgram Theorem (Theorem 1.1). In this situation, we are forced to use the more general Babuška Theorem (Theorem 1.5).

Theorem 2.6

Assume (2.3) and (2.7) hold. Moreover, assume that $a(\cdot,\cdot;\boldsymbol{\mu})$ satisfies the following inf-sup condition, i.e. there exists $\beta_{0,N} > 0$ such that

$$\beta_N(\boldsymbol{\mu}) = \inf_{v_N \in V_N} \sup_{w_N \in W_N} \frac{a(v_N, w_N; \boldsymbol{\mu})}{\|v_N\|_V \|w_N\|_V} \geq \beta_{0,N}.$$

Then, for any $\boldsymbol{\mu} \in \mathcal{P}$ the G-RB problem (2.10) has a unique solution $u_N(\boldsymbol{\mu}) \in V_N$, which satisfies the stability estimate

$$\|u_N(\boldsymbol{\mu})\|_V \leq \frac{1}{\beta_N(\boldsymbol{\mu})} \|f(\cdot;\boldsymbol{\mu})\|_{V'}.$$

■

Once we have a reduced basis, $\{\zeta_1, \ldots, \zeta_N\}$, of V_N, we can study the algebraic form of the Galerkin Reduced Basis case,

$$\sum_{m=1}^{N} a(\zeta_m, \zeta_n; \boldsymbol{\mu}) u_N^{(m)}(\boldsymbol{\mu}) = f(\zeta_n; \boldsymbol{\mu}), \quad 1 \leq n \leq N,$$

if we dentote $\mathbb{A}_N(\boldsymbol{\mu}) \in \mathbb{R}^{N \times N}$, so that, $(\mathbb{A}_N(\boldsymbol{\mu}))_{nm} = a(\zeta_m, \zeta_n; \boldsymbol{\mu})$ and $\mathbf{f}_N(\boldsymbol{\mu}) \in \mathbb{R}^N$, so that, $(f_N(\boldsymbol{\mu}))_n = f(\zeta_n; \boldsymbol{\mu})$, then the algebraic form of the Galerkin Reduced Basis case is equivalent to the following linear system,

$$\mathbb{A}_N(\boldsymbol{\mu}) \mathbf{u}_N(\boldsymbol{\mu}) = \mathbf{f}_N(\boldsymbol{\mu}).$$

Matrix $\mathbb{A}_N$ is full, whereas $\mathbb{A}_h$ is in general sparse. However, since $N \ll N_h$ it is in principle much faster and less computationally intensive to solve.

Remark 2.13

The Reduced Basis matrix $\mathbb{A}_N(\boldsymbol{\mu})$ inherits the properties of symmetry and positivity of $\mathbb{A}_h$.

Unfortunately, the assembly of $\mathbb{A}_N(\boldsymbol{\mu})$ and $\mathbf{f}(\boldsymbol{\mu})$ still involves computations whose complexity depends on N_h. To overcome this drawback we make the affine parametric dependence assumption, so we require a and f to be affine with respect to the parameter $\boldsymbol{\mu}$, that is,

$$a(w, v; \boldsymbol{\mu}) = \sum_{q=1}^{Q_a} \theta_a^q(\boldsymbol{\mu}) a_q(w, v), \quad \forall v, w \in V, \quad \forall \boldsymbol{\mu} \in \mathcal{P}$$

$$f(v; \boldsymbol{\mu}) = \sum_{q=1}^{Q_f} \theta_f^q(\boldsymbol{\mu}) f_q(v), \qquad \forall v \in V, \quad \forall \boldsymbol{\mu} \in \mathcal{P}.$$

Remark 2.14

The affine parametric dependence is inherited by the algebraic problem

$$\mathbb{A}_N(\boldsymbol{\mu}) = \sum_{q=1}^{Q_a} \theta_a^q(\boldsymbol{\mu}) \mathbb{A}_N^q; \quad \mathbf{f}_N(\boldsymbol{\mu}) = \sum_{q=1}^{Q_f} \theta_f^q(\boldsymbol{\mu}) \mathbf{f}_N^q,$$

where $(\mathbb{A}_N^q)_{nm} = a_q(\zeta_m, \zeta_n)$ and $(\mathbf{f}_m^q) = f_q(\zeta_m)$.

Remark 2.15

Often, the affine parametric dependence automatically follows by the definition *of the problem.*

Expanding each ζ_m with respect to the basis functions $\{\varphi^j\}_{i=1}^{N_h}$ of V_h, we get

$$\zeta_m = \sum_{j=1}^{N_h} \zeta_m^{(j)} \varphi^j \text{ for } 1 \leq m \leq N,$$

so $\boldsymbol{\zeta}_m = (\zeta_m^{(1)}, \ldots, \zeta_m^{(N_h)})^T$.

Definition 2.16

We define the transformation matrix $\mathbb{V} \in \mathbb{R}^{N_h \times N}$ *as*

$$\mathbb{V} = [\boldsymbol{\zeta}_1 | \ldots | \boldsymbol{\zeta}_N] \rightarrow (\mathbb{V})_{jm} = \zeta_m^j, \quad 1 \leq m \leq N, 1 \leq j \leq N_h.$$

By the bilinearity of $a(\cdot, \cdot; \boldsymbol{\mu})$ and the linearity of $f(\cdot; \boldsymbol{\mu})$ it follows,

$$a_q(\zeta_m, \zeta_n) = \sum_{i=1}^{N_h} \sum_{j=1}^{N_h} \zeta_m^{(i)} a_q(\varphi^j, \varphi^i) \zeta_n^i; \quad f_q(\zeta_n) = \sum_{i=1} f_q(\varphi^i) \zeta_n^{(i)}.$$

Remark 2.17

Equivalently, in matrix form

$$\mathbb{A}_N^q = \mathbb{V}^T \mathbb{A}_h^q \mathbb{V}; \quad \boldsymbol{f}_N^q = \mathbb{V}^T \boldsymbol{f}_h^q,$$

where $(\mathbb{A}_h^q)_{ij} = a_q(\varphi^j, \varphi^i)$ *and* $(\boldsymbol{f}_h^q)_i = f_q(\varphi^i)$.

We end this section with an observation on the spectral condition number for this method.

Definition 2.18

The spectral condition number of a matrix $\mathbb{B} \in \mathbb{R}^{N \times N}$ *is defined as*

$$\kappa(\mathbb{B}) = \frac{\sigma_{max}(\mathbb{B})}{\sigma_{min}(\mathbb{B})}.$$

Proposition 2.7

The spectral condition number κ*, measures how much the output value can change for a small change in the input argument. According to the spectral condition number a problem can be,*

- *Well-conditioned. If the condition number is low.*

- *Ill-conditioned. If the condition number is high.*

If $a(\cdot,\cdot;\boldsymbol{\mu})$ is symmetric and coercive and the basis functions ζ_m are V-orthogonal, the spectral condition number $\kappa(\mathbb{A}_N(\boldsymbol{\mu}))$ can be bounded uniformly, since,

$$\kappa(\mathbb{A}_N(\boldsymbol{\mu})) \leq \frac{\gamma_h(\boldsymbol{\mu})}{\beta_h(\boldsymbol{\mu})}.$$

Remark 2.19

If ζ_m are not V-orthonormal, we have

$$\kappa(\mathbb{A}_N(\boldsymbol{\mu})) \leq \frac{\gamma_h(\boldsymbol{\mu})}{\beta_h(\boldsymbol{\mu})}\kappa(\mathbb{V}^T\mathbb{X}_h\mathbb{V}).$$

2.2.2 Least-Squares Reduced Basis Method

A special instance of PG-RB problem is the Least-Squares Reduced Basis (LS-RB) method which corresponds to choosing the test space as $W_N = R_{V_h}^{-1}L(\boldsymbol{\mu})V_N$. Here

$$R_{V_h}^{-1}: V_h' \to V_h; \quad (R_{V_h}^{-1}f, y)_V = {}_{V'}\langle f, v\rangle_V, \quad \forall f \in V_h', \quad \forall v \in V_h,$$

is the inverse of the Riesz map $R_{V_h}: V_h \to V_h'$, given by

$${}_{V'}\langle R_{V_h}x, y\rangle_V = (x,y)_V, \quad \forall x, y \in V_h.$$

For a generic Hilbert space V, $w = R_{V_h}^{-1}f$ is the Riesz representative of $f \in V'$ obtained by solving

$$(w,v)_V = {}_{V'}\langle f, v\rangle_V, \quad \forall v \in V.$$

Remark 2.20

The Riesz representative depends on the choice of the inner product over V.

In the case of an inner product produced by a symmetric, coercive bilinear form $a(\cdot,\cdot)$ this yields the variational problem

$$a(w,v) = {}_{V'}\langle f, v\rangle_V, \quad \forall v \in V.$$

Given $\boldsymbol{\mu} \in \mathcal{P}$, the LS-RB approximation of (2.2) reads

$$\begin{cases} \text{Find } u_N(\boldsymbol{\mu}) \in V_N \text{ such that} \\ \qquad a(u_N(\boldsymbol{\mu}), w_N; \boldsymbol{\mu}) = f(w_N;\boldsymbol{\mu}), \quad \forall w_N \in W_N = R_{V_h}^{-1}L(\boldsymbol{\mu})V_N. \end{cases} \tag{2.11}$$

We state in the following proposition when the LS-RB problem (2.11) has a unique solution.

Proposition 2.8

Assume that (2.3), (2.6) and (2.7) hold. If $W_N = R_{V_h}^{-1} L(\boldsymbol{\mu}) V_N$, then

$$\beta_N = \inf_{v_n \in V_N} \sup_{w_N \in W_N} \frac{a(v_N, w_N; \boldsymbol{\mu})}{\|v_N\|_V \|w_N\|_V} \geq \beta_h(\boldsymbol{\mu}) > 0, \quad \forall \boldsymbol{\mu} \in \mathcal{P},$$

and the LS-RB problem (2.11) has a unique solution $u_N(\boldsymbol{\mu}) \in V_N$ for any $\boldsymbol{\mu} \in \mathcal{P}$ which satisfies the stability estimate

$$\|u_N(\boldsymbol{\mu})\|_V \leq \frac{1}{\beta_N(\boldsymbol{\mu})} \|f(\cdot; \boldsymbol{\mu})\|_{V'}.$$

■

Note that $W_N = W_N(\boldsymbol{\mu})$ and that operator $R_{V_h}^{-1} L(\boldsymbol{\mu})$ represents the parametrized version of the supremizer operator, $T_h(\cdot; \boldsymbol{\mu}) : V_h \to V_h$, in this case

$$T_h(v; \boldsymbol{\mu}) = \arg \sup_{w \in V_h} \frac{a(v, w; \boldsymbol{\mu})}{\|w\|_V}.$$

Remark 2.21

Indeed, for any $v, w \in V$, it holds,

$$(R_{V_h}^{-1} L(\boldsymbol{\mu}) v, w)_V = {}_{V'}\langle L(\boldsymbol{\mu}) v, w \rangle_V = a(v, w; \boldsymbol{\mu}) = (T_{V_h}(v; \boldsymbol{\mu}), w)_V.$$

This property provides an indication on how to generate the basis functions of W_N from those of V_h.

$$\begin{cases} \text{Find } \eta_i(\boldsymbol{\mu}) \in W_N \text{ such that} \\ \quad (\eta_i(\boldsymbol{\mu}), z)_V = a(\zeta_i, z; \boldsymbol{\mu}), \quad \forall z \in V_h. \end{cases} \tag{2.12}$$

Therefore, W_N is generated as

$$W_N = \text{span}(\eta_1(\boldsymbol{\mu}), \ldots, \eta_N(\boldsymbol{\mu})).$$

The LS-RB approximation (2.11) enjoys some remarkable optimality properties that we present below.

Proposition 2.9

If $a(\cdot, \cdot; \boldsymbol{\mu})$ is inf-sup stable and $W_N = R_{V_h}^{-1} L(\boldsymbol{\mu}) V_N$ then the solution $u_N(\boldsymbol{\mu}) \in V_N$

to (2.11) satisfies the following optimality property

$$u_N(\boldsymbol{\mu}) = \arg\min_{v\in V_N} \|L(\boldsymbol{\mu})v - f(\boldsymbol{\mu})\|^2_{V'_h}.$$

Moreover, the following best approximation property holds

$$u_N(\boldsymbol{\mu}) = \arg\min_{v\in V_N} \|u_h(\boldsymbol{\mu}) - v(\boldsymbol{\mu})\|^2_{\boldsymbol{\mu}},$$

where $(v,w)_{\boldsymbol{\mu}} = (T_h(v,\boldsymbol{\mu}), T_h(w,\boldsymbol{\mu}))_V = (R^{-1}_{V_h}L(\boldsymbol{\mu})v, R^{-1}_{V_h}L(\boldsymbol{\mu})w)_V.$ ■

Because of the optimality property, the LS-RB method is also referred to as minimum residual method.
Different choices for the norm over V yield different optimal inequalities.

Remark 2.22

By choosing the V-norm instead of the energy norm, we would find the following optimal error inequality thanks to Theorem 1.5,

$$\|u_h(\boldsymbol{\mu}) - u_N(\boldsymbol{\mu})\|_V \le \frac{\gamma_h(\boldsymbol{\mu})}{\beta_N(\boldsymbol{\mu})} \inf_{v\in V_N} \|u_h(\boldsymbol{\mu}) - v\|_V.$$

We can now study the algebraic form of LS-RB problem (2.11). Since W_N is spanned by the basis $\{\eta_n(\boldsymbol{\mu})\}_{n=1}^N$ the problem (2.11) is equivalent to

$$\begin{cases} \text{Find } \mathbf{u}_N(\boldsymbol{\mu}) \in \mathbb{R}^N \text{ such that} \\ \displaystyle\sum_{m=1}^N a(\zeta_m, \eta_n(\boldsymbol{\mu}); \boldsymbol{\mu}) u_N^{(m)}(\boldsymbol{\mu}) = f(\eta_n(\boldsymbol{\mu}); \boldsymbol{\mu}) \; 1 \le n \le N. \end{cases} \tag{2.13}$$

If we denote $\mathbb{A}_N(\boldsymbol{\mu}) \in \mathbb{R}^{N\times N}$ with $(\mathbb{A}_N(\boldsymbol{\mu}))_{nm} = a(\zeta_m, \eta_n(\boldsymbol{\mu}); \boldsymbol{\mu})$ and $\mathbf{f}_N(\boldsymbol{\mu}) \in \mathbb{R}^N$ with $(\mathbf{f}_N(\boldsymbol{\mu}))_m = f(\eta_m(\boldsymbol{\mu}); \boldsymbol{\mu})$, then this is equivalent to the linear system,

$$\mathbb{A}_N(\boldsymbol{\mu})\mathbf{u}_N(\boldsymbol{\mu}) = \mathbf{f}_N(\mu).$$

Remark 2.23

Matrix $\mathbb{A}_N(\boldsymbol{\mu})$ is closely related to the W_N basis and the V-inner product,

$$(\mathbb{A}_N(\boldsymbol{\mu}))_{nm} = (\eta_m(\boldsymbol{\mu}), \eta_n(\boldsymbol{\mu}))_V.$$

Let us introduce now $\boldsymbol{\eta}_n(\boldsymbol{\mu}) \in \mathbb{R}^{N_h}$, solution of the following N_h-dimensional linear system,

$$\mathbb{X}_h \boldsymbol{\eta}_n(\boldsymbol{\mu}) = \mathbb{A}_h(\boldsymbol{\mu})\boldsymbol{\zeta}_n,$$

so that,

$$\boldsymbol{\eta}_n(\boldsymbol{\mu}) = \mathbb{X}_h^{-1}\mathbb{A}_h(\boldsymbol{\mu})\boldsymbol{\zeta}_n.$$

As a result,

$$\mathbb{A}_N(\boldsymbol{\mu}) = \mathbb{V}^T\mathbb{A}_h^T(\boldsymbol{\mu})\mathbb{X}_h^{-1}\mathbb{A}_h(\boldsymbol{\mu})\mathbb{V}.$$

Similary, we obtain,

$$\mathbf{f}_N(\boldsymbol{\mu}) = \mathbb{V}^T\mathbb{A}_h^T(\boldsymbol{\mu})\mathbb{X}_h^{-1}\mathbf{f}_h(\boldsymbol{\mu}).$$

Finally, in this case the affine parametric dependence yields the following decomposition for $\mathbb{A}_N(\boldsymbol{\mu})$ and $\mathbf{f}_N(\boldsymbol{\mu})$,

$$\mathbb{A}_N(\boldsymbol{\mu}) = \sum_{q_1=1}^{Q_a}\sum_{q_2=1}^{Q_a}\theta_a^{q_1}(\boldsymbol{\mu})\theta_a^{q_2}(\boldsymbol{\mu})\mathbb{A}_N^{q_1,q_2},$$

$$\mathbf{f}_N(\boldsymbol{\mu}) = \sum_{q_1=1}^{Q_f}\sum_{q_2=1}^{Q_a}\theta_f^{q_1}(\boldsymbol{\mu})\theta_a^{q_2}(\boldsymbol{\mu})\mathbf{f}_N^{q_1,q_2},$$

where the Q_a^2 matrices $\mathbb{A}_N^{q_1,q_2}$ and the Q_aQ_f vectors $\mathbf{f}_N^{q_1,q_2}$ are given by

$$\mathbb{A}_N^{q_1,q_2} = \mathbb{V}^T\mathbb{A}_h^{q_2T}\mathbb{X}_h^{-1}\mathbb{A}_h^{q_1}\mathbb{V}, \quad \mathbf{f}_N^{q_1,q_2} = \mathbb{V}^T\mathbb{A}_h^{q_2T}\mathbb{X}_h^{-1}\mathbf{f}_h^{q_1}.$$

2.2.3 Petrov-Galerkin Reduced Basis Method

For the more general PG-RB problem (2.9), where the test subspace W_N is free, the uniqueness and existence of the solution is stated in the following theorem.

Theorem 2.10

Assume that conditions (2.3), (2.7) hold and that $a(\cdot,\cdot;\boldsymbol{\mu})$ satisfies the following inf-sup condition, i.e. there exists $\beta_{0,N} > 0$ such that,0

$$\beta_N(\boldsymbol{\mu}) = \inf_{v_N\in V_N}\sup_{w_N\in W_N}\frac{a(v_N,w_N;\boldsymbol{\mu})}{\|v_N\|_V\|w_N\|_V} \geq \beta_{0,N}.$$

Then, for any $\boldsymbol{\mu}\in\mathcal{P}$ the PG-RB problem (2.9) has a unique solution $u_N(\boldsymbol{\mu})\in V_N$ which satisfies the stability estimate

$$\|u_N(\boldsymbol{\mu})\|_V \leq \frac{1}{\beta_N(\boldsymbol{\mu})}\|f(\cdot;\boldsymbol{\mu})\|_{V'}$$

and the optimal error inequality

$$\|u_h(\boldsymbol{\mu}) - u_N(\boldsymbol{\mu})\|_V \leq \frac{\gamma_h(\boldsymbol{\mu})}{\beta_N(\boldsymbol{\mu})}\inf_{v\in V_N}\|u_h(\boldsymbol{\mu}) - v\|_V.$$

■

Remark 2.24

Theorem 2.10 encompasses the Galerkin and the Least-Squares cases as special cases.

2.3 Offline/Online Decomposition

Now that we have presented the theoretical results of the most important RB methods, we proceed to analyse them from a computational standpoint. We seek to take advantage of the affine parametric dependence property by splitting the assembly of the reduced matrix and vector into two different phases.

- An offline phase, where the expensive computations which do not depend on the parameter are performed, such as the assembling of the reduced matrices and vectors.
- An online phase, where the reduced basis system which is parameter-dependent is assembled and solved.

In order to analyse the computational complexity of the operations presented in this section, let us denote by

- n_{vv}: number of operations required to compute a scalar product between two N_h-dimensional vectors.
- n_{mv}: number of operations for a matrix vector product.
- n_{ls}: number of operations to solve a linear system ($\mathbb{X}_h \mathbf{x} = \mathbf{y}$).

2.3.1 Offline Phase

We assemble and store the reduced matrices and vectors according to the results shown in sections 2.2.1 and section 2.2.2, see Algorithm 2.
Analysing the computational complexity of these operations we get

☐ Case G-RB: we must compute Q_a matrices $\mathbb{A}_N^q$ and Q_f vectors $\mathbf{f}_N^q$ with

$$O(Q_f N n_{vv} + Q_a(N n_{nm} + N^2 n_{vv})) \text{ operations.}$$

☐ Case LS-RB: we must compute Q_a^2 matrices $\mathbb{A}_N^{q_1,q_2}$ and $Q_f Q_a$ vectors $\mathbf{f}_N^{q_1,q_2}$ with

$$O(Q_a(N n_{mv} + N n_{ls}) + Q_f Q_a N n_{vv} + Q_a^2(N n_{nm} + N^2 n_{vv})) \text{ operations.}$$

Remark 2.25

Note that the LS-RB method in addition of being more expensive than G-RB method requires more storage.

Algorithm 2 Offline computation of reduced matrices and vectors

```
 1: function OffAssemble($\mathbb{A}_h^q, \mathbf{f}_h^q, \mathbb{V}, \mathbb{X}_h$, method)
 2:     switch method
 3:         case G-RB
 4:             for $q = 1 : Q_a$
 5:                 $\mathbb{A}_N^q = \mathbb{V}^T \mathbb{A}_h^q \mathbb{V}$                    ▷ $O(Nn_{mv} + N^2 n_{vv})$
 6:             end for
 7:             for $q = 1 : Q_f$
 8:                 $\mathbf{f}_N^q = \mathbb{V}^T \mathbf{f}_h^q$                           ▷ $O(Nn_{vv})$
 9:             end for
10:         end case
11:         case LS-RB
12:             for $q_1 = 1 : Q_a$
13:                 compute $\mathbb{Z} = \mathbb{X}_{\mathbb{h}}^{-\mathbb{1}} \mathbb{A}_h^{q_1} \mathbb{V}$          ▷ $O(Nn_{mv} + Nn_{ls})$
14:                 for $q_2 = 1 : Q_a$
15:                     $\mathbb{A}_N^{q_1,1_2} = \mathbb{Z}^T \mathbb{A}_h^{q_2} \mathbb{V}$          ▷ $O(Nn_{ms} + N^2 n_{vv})$
16:                 end for
17:                 for $q_2 = 1 : Q_f$
18:                     $\mathbf{f}_N^{q_1,q_2} = \mathbb{Z}^T \mathbf{f}_h^{q_2}$                   ▷ $O(Nn_{vv})$
19:                 end for
20:             end for
21:         end case
22:     end switch
23:     return $[\mathbb{A}_N^q, \mathbf{f}_N^q]$
24: end function
```

2.3.2 Online Phase

Once we have assembled the reduced matrices and vectors we can start taking advantage of the affine parametric dependence. Given $\boldsymbol{\mu} \in \mathcal{P}$, we first form the RB matrix $\mathbb{A}_N(\boldsymbol{\mu})$ and vector $\mathbf{f}_N(\boldsymbol{\mu})$. This requires

- For G-RB case
$$O(Q_a N^2 + Q_f N) \text{ operations,}$$
- For LS-RB case
$$O(Q_a^2 N^2 + Q_f Q_a N) \text{ operations.}$$

Remark 2.26

G-RB method is always faster than LS-RB method.

Then, we solve the dense N-dimensional RB system with complexity $O(N^3)$. This is all summarized in Algorithm 3.

Algorithm 3 Online RB system assembling and solving

```
function SolveRB($\mathbb{A}_N^q, \mathbf{f}_N^q, \theta_a^q, \theta_f^q, \boldsymbol{\mu}$, method)
    $\mathbb{A}_N(\boldsymbol{\mu}) = \mathbb{O}$, $\mathbf{f}_N(\boldsymbol{\mu}) = \mathbf{0}$
    switch method
        case G-RB
            for $q = 1 : Q_a$
                $\mathbb{A}_N(\boldsymbol{\mu}) \leftarrow \mathbb{A}_N(\boldsymbol{\mu}) + \theta_a^q(\boldsymbol{\mu})\mathbb{A}_N^q$        ▷ $O(N^2)$
            end for
            for $q = 1 : Q_f$
                $\mathbf{f}_N(\boldsymbol{\mu}) \leftarrow \mathbf{f}_N(\boldsymbol{\mu}) + \theta_f^q(\boldsymbol{\mu})\mathbf{f}_N^q$        ▷ $O(N)$
            end for
        end case
        case LS-RB
            for $q_1 = 1 : Q_a$
                for $q_2 = 1 : Q_a$
                    $\mathbb{A}_N(\boldsymbol{\mu}) \leftarrow \mathbb{A}_N(\boldsymbol{\mu}) + \theta_a^{q_1}\theta_a^{q_2}\mathbb{A}_h^{q_1,q_2}$        ▷ $O(N^2)$
                end for
                for $q_2 = 1 : Q_f$
                    $\mathbf{f}_N(\boldsymbol{\mu}) \leftarrow \mathbf{f}_N(\boldsymbol{\mu}) + \theta_a^{q_1}\theta_f^{q_2}\mathbf{f}_N^{q_1,q_2}$        ▷ $O(N)$
                end for
            end for
        end case
    end switch
    solve linear system $\mathbb{A}_N(\boldsymbol{\mu})\mathbf{u}_N(\boldsymbol{\mu}) = \mathbf{f}_N(\boldsymbol{\mu})$        ▷ $O(N^3)$
    return $\mathbf{u}_N(\boldsymbol{\mu})$
end function
```

Remark 2.27

As a result, if N and Q_a are small enough, we can achieve very fast response both for real-time problems and many-query contexts.

In the following section, we study if the solution for these problems (2.10) and (2.11), obtained via Algorithms 2 and 3 is a good approximation to the high-fidelity solution of the high-fidelity problem (2.5) obtained via Algorithm 1.

2.4 A Posteriori Error Estimation

In order to study if the solution of (2.10) and (2.11) is a good approximation to the high-fidelity solution of the high-fidelity problem (2.5), it is necessary to compute an error estimator and it has to be computed in the online phase, so we call it an a posteriori error estimator.
A posteriori error estimators play an essential role at two different stages,

- They guarantee the reliability of the reduction process.
- They guarantee the efficiency of the method.

In the online stage, it allows to bound the error between the RB solution, $u_N(\boldsymbol{\mu})$, and the underlying high-fidelity solution $u_h(\boldsymbol{\mu})$, for each $\boldsymbol{\mu} \in \mathcal{P}$. In this context, an error estimator is required to be:

- Sharp, i.e. as close as possible to the actual error.
- Asymptotically correct, when increasing N this error should tend to zero with the same rate as the actual error.
- Computationally cheap, inexpensive to compute with respect to the total computational costs.

To derive such an estimator, an equivalence will be established between the norm of the error and a corresponding dual norm of the residual. The latter only involves the arrays of the high-fidelity problem together with the computed RB solution, but not the high-fidelity solution.

Remark 2.28

The equivalence between the norm of the error and a corresponding dual norm of the residual is a direct consequence of the stability of the high-fidelity problem.

2.4.1 Error-Residual Relationship

Establishing an error-residual relationship is crucial to derive a posteriori error estimates.
We denote the error between the high-fidelity and reduced solutions by

$$e_h(\boldsymbol{\mu}) = u_h(\boldsymbol{\mu}) - u_N(\boldsymbol{\mu}) \in V_h.$$

From (2.4) and (2.9) we get the error representation

$$a(e_h(\boldsymbol{\mu}), v; \boldsymbol{\mu}) = f(v; \boldsymbol{\mu}) - a(u_N(\boldsymbol{\mu}), v; \boldsymbol{\mu}), \quad \forall v \in V_h. \tag{2.14}$$

Remark 2.29

By setting

$$r(v;\boldsymbol{\mu}) = f(v;\boldsymbol{\mu}) - a(u_N(\boldsymbol{\mu}), v; \boldsymbol{\mu}), \quad \forall v \in V_h,$$

we note that $r(v;\boldsymbol{\mu}) = \langle r(\boldsymbol{\mu}), v\rangle$.

Thanks to the continuity of $a(\cdot,\cdot;\boldsymbol{\mu})$ we get

$$|r(v;\boldsymbol{\mu})| \leq \gamma_h(\boldsymbol{\mu})\|e_h(\boldsymbol{\mu})\|_V\|v\|_V, \quad \forall v \in V_h,$$
$$\|r(\cdot;\boldsymbol{\mu})\|_{V_h'} \leq \gamma_h(\boldsymbol{\mu})\|e_h(\boldsymbol{\mu})\|_V.$$

On the other hand, from the error representation and the stability estimate from Proposition 2.1 we obtain

$$\beta_h(\boldsymbol{\mu})\|e_h(\boldsymbol{\mu})\|_V \leq \|r(\cdot;\boldsymbol{\mu})\|_{V_h'}.$$

$$\frac{1}{\gamma_h(\boldsymbol{\mu})}\|r(\cdot;\boldsymbol{\mu})\|_{V_h'} \leq \|e_h(\boldsymbol{\mu})\|_V \leq \frac{1}{\beta_h(\boldsymbol{\mu})}\|r(\cdot,\boldsymbol{\mu})\|_{V_h'}. \tag{2.15}$$

Remark 2.30

Since $r(\cdot;\boldsymbol{\mu})$ *only involves the high-fidelity arrays and the computed reduced solution* $u_N(\boldsymbol{\mu})$*, but not* $u_h(\boldsymbol{\mu})$*, its norm well serves as an a posteriori error estimator.*

2.4.2 Error Bound

Thanks to (2.15), the quantity,

$$\Delta_N(\boldsymbol{\mu}) = \frac{\|r(\cdot;\boldsymbol{\mu})\|_{V_h'}}{\beta_h(\boldsymbol{\mu})}, \tag{2.16}$$

can play the role of error estimate.

Definition 2.31

We define the associate effectivity factor as

$$\varsigma_N(\boldsymbol{\mu}) = \frac{\Delta_N(\boldsymbol{\mu})}{\|e_h(\boldsymbol{\mu})\|_V}.$$

It is a measure of the quality of the proposed estimator.

For sharpness, we pretend the associate effectivity factor to be as close to 1 as possible.

Remark 2.32

Equivalence (2.15) directly implies

$$1 \leq \varsigma_N(\boldsymbol{\mu}) \leq \frac{\gamma_h(\boldsymbol{\mu})}{\beta_h(\boldsymbol{\mu})}, \quad \forall \boldsymbol{\mu} \in \mathcal{P}.$$

Remark 2.33

Since the stability factors $\beta_h(\boldsymbol{\mu})$ and $\gamma_h(\boldsymbol{\mu})$ are the minimum and maximum generalized singular values of the matrix $\mathbb{A}_h(\boldsymbol{\mu})$ (see Remarks 1.11, 1.12 and 1.13), the effectivity upper bound is in fact the condition number of the high-fidelity problem (2.4).

Remark 2.34

The upper bound is independent of N, i.e. is stable to N-refinement.

However, we can expect large effectiveness when the underlying high-fidelity problem (2.4) is ill-conditioned.

2.4.3 Computation of Error Bounds

To finish this chapter, in order to obtain an algebraic equivalent of the error bound (2.16), we start by deriving the algebraic counterpart of the error representation (2.14). Let us define the discrete error between the reduced basis and high-fidelity solutions,

$$\mathbf{e}_h(\boldsymbol{\mu}) = \mathbf{u}_h(\boldsymbol{\mu}) - \mathbb{V}\mathbf{u}_N(\boldsymbol{\mu}),$$

and the discrete residual,

$$\mathbf{r}_h(\mathbf{u}_N(\boldsymbol{\mu}); \boldsymbol{\mu}) = \mathbf{f}_h(\boldsymbol{\mu}) - \mathbb{A}_h(\boldsymbol{\mu})\mathbb{V}\mathbf{u}_N(\boldsymbol{\mu}).$$

Remark 2.35

Recalling that $\mathbb{A}_h(\boldsymbol{\mu})\mathbf{u}_h(\boldsymbol{\mu}) = \mathbf{f}_h(\boldsymbol{\mu})$, we obtain

$$\mathbb{A}_h(\boldsymbol{\mu})\mathbf{e}_h(\boldsymbol{\mu}) = \mathbf{r}_h(\mathbf{u}_N; \boldsymbol{\mu}),$$

where left-multiplying by $\mathbb{A}_h^{-1}(\boldsymbol{\mu})$ we get

$$\mathbf{e}_h(\boldsymbol{\mu}) = \mathbb{A}_h^{-1}(\boldsymbol{\mu})\mathbf{r}_h(\mathbf{u}_N; \boldsymbol{\mu}).$$

Taking the 2-norm on both sides we obtain the following upper bound,

$$\|\mathbf{e}_h(\boldsymbol{\mu})\|_2 \leq \|\mathbb{A}_h^{-1}(\boldsymbol{\mu})\|_S \|r_h(\mathbf{u}_N(\boldsymbol{\mu}); \boldsymbol{\mu})\|_2 \leq \frac{1}{\sigma_{min}(\mathbb{A}_h(\boldsymbol{\mu}))} \|\mathbf{r}_h(\mathbf{u}_N(\boldsymbol{\mu}); \boldsymbol{\mu})\|_2,$$

where $\sigma_{min}(\mathbb{A}_h(\boldsymbol{\mu}))$ denotes the smallest singular value of $\mathbb{A}_h(\boldsymbol{\mu})$.
Similarly, we can obtain a bound for the error in the V-norm, but first we have to define the $\mathbb{X}_h$-scalar product.

Definition 2.36

We define the $\mathbb{X}_h$-scalar product of two vectors $\mathbf{v}$ and $\mathbf{w}$ as

$$(\mathbf{v}, \mathbf{w})_{\mathbb{X}_h} = \mathbf{w}^T \mathbb{X}_h \mathbf{v},$$

and its associate norm

$$\|\mathbf{v}\|_{\mathbb{X}_h} = \mathbf{v}^T \mathbb{X}_h \mathbf{v}.$$

Taking this into account, we obtain

$$\begin{aligned}
&\mathbb{X}_h^{1/2}\mathbf{e}_h(\boldsymbol{\mu}) = \mathbb{X}_h^{1/2}\mathbb{A}_h^{-1}(\boldsymbol{\mu})\mathbb{X}_h^{1/2}\mathbb{X}_h^{-1/2}\mathbf{r}_h(\mathbf{u}_N(\boldsymbol{\mu});\boldsymbol{\mu}) \rightarrow \\
&\rightarrow \|\mathbf{e}_h(\boldsymbol{\mu})\|_{\mathbb{X}_h} \le \|\mathbb{X}_h^{1/2}\mathbb{A}_h^{-1}(\boldsymbol{\mu})\mathbb{X}_h^{1/2}\|_S \|\mathbf{r}_h(\mathbf{u}_N;\boldsymbol{\mu})\|_{\mathbb{X}_h^{-1}} \rightarrow \\
&\rightarrow \|\mathbf{e}_h(\boldsymbol{\mu})\|_{\mathbb{X}_h} \le \frac{1}{\sigma_{min}(\mathbb{X}_h^{-1/2}\mathbb{A}_h(\boldsymbol{\mu})\mathbb{X}_h^{-1/2})}\|\mathbf{r}_h(\mathbf{u}_N;\boldsymbol{\mu})\|_{\mathbb{X}_h^{-1}},
\end{aligned}$$

Remark 2.37

We recall that

$$\sigma_{min}(\mathbb{X}_h^{-1/2}\mathbb{A}_h(\boldsymbol{\mu})\mathbb{X}_h^{-1/2}) = \sqrt{\lambda_{min}(\mathbb{X}_h^{-1/2}\mathbb{A}_h^T(\boldsymbol{\mu})\mathbb{X}_h^{-1}\mathbb{A}_h(\boldsymbol{\mu})\mathbb{X}_h^{-1/2})} = \beta_h(\boldsymbol{\mu}).$$

This provides the algebraic form of the error bound $\Delta_N(\boldsymbol{\mu})$ and we remark that (2.16) only depends on the basis of V_N, i.e. its definition does not depend on whether we use Galerkin (2.10) or Least-Squares (2.11) projections.
To evaluate the dual norm of the residual, we exploit again the affine decomposition.

Remark 2.38

By the definition of the $\mathbb{X}_h$-scalar product we get

$$\begin{aligned}
\|\mathbf{r}_h(\mathbf{u}_N(\boldsymbol{\mu});\boldsymbol{\mu})\|^2_{\mathbb{X}_h^{-1}} = \; & \mathbf{f}_h(\boldsymbol{\mu})^T\mathbb{X}_h^{-1}\mathbf{f}_h(\boldsymbol{\mu}) - 2\mathbf{f}_h(\boldsymbol{\mu})^T\mathbb{X}_h^{-1}\mathbb{A}_h(\boldsymbol{\mu})\mathbb{V}\mathbf{u}_N(\boldsymbol{\mu}) \\
& + \mathbf{u}_N(\boldsymbol{\mu})^T\mathbb{V}^T\mathbb{A}_h^T\mathbb{X}_h^{-1}\mathbb{A}_h(\boldsymbol{\mu})\mathbb{V}\mathbf{u}_N(\boldsymbol{\mu}).
\end{aligned}$$

Now using the affine decomposition

$$
\begin{aligned}
\|\mathbf{r}_h(\mathbf{u}_N(\boldsymbol{\mu});\boldsymbol{\mu})\|^2_{\mathbb{X}_h^{-1}} = & \sum_{q_1,q_2=1}^{Q_f} \theta_f^{q_1}(\boldsymbol{\mu})\theta_f^{q_2}(\boldsymbol{\mu})\, \underbrace{\mathbf{f}_h^{q_1 T}\mathbb{X}_h^{-1}\mathbf{f}_h^{q_2}}_{c_{q_1,q_2}} \\
& -2\sum_{q_1=1}^{Q_a}\sum_{q_2=1}^{Q_f} \theta_a^{q_1}(\boldsymbol{\mu})\theta_f^{q_2}(\boldsymbol{\mu})\mathbf{u}_N(\boldsymbol{\mu})^T\, \underbrace{\mathbb{V}^T\mathbb{A}_h^{q_1 T}\mathbb{X}_h^{-1}\mathbf{f}_h^{q_2}}_{\mathbf{d}_{q_1,q_2}} \\
& + \sum_{q_1,q_2=1}^{Q_a} \theta_a^{q_1}(\boldsymbol{\mu})\theta_a^{q_2}(\boldsymbol{\mu})\mathbf{u}_N(\boldsymbol{\mu})^T\, \underbrace{\mathbb{V}^T\mathbb{A}_h^{q_1 T}\mathbb{X}_h^{-1}\mathbb{A}_h^{q_2}\mathbb{V}}_{\mathbb{E}_{q_1,q_2}}\,\mathbf{u}_N(\boldsymbol{\mu}).
\end{aligned}
$$

Algorithm 4 Offline computation of $\boldsymbol{\mu}$-independent terms of the dual norm of residual

```
 1: function OffResidual($\mathbb{A}_N^q, \mathbf{f}_N^q, \mathbb{X}_h, \mathbb{V}$)
 2:     for $q_1 = 1 : Q_f$
 3:         $\mathbf{t} = \mathbb{X}_h \mathbf{f}_h^{q_1}$
 4:         for $q_2 = 1 : Q_f$
 5:             $c_{q_1,q_2} = \mathbf{t}^T \mathbf{f}_h^{q_2}$
 6:         end for
 7:     end for
 8:     for $q_1 = 1 : Q_a$
 9:         compute $\mathbb{Z} = \mathbb{X}_h^{-1}\mathbb{A}_h^{q_1}\mathbb{V}$
10:         for $q_2 = 1 : Q_a$
11:             $\mathbb{E}_{q_1,q_2} = \mathbb{Z}^T\mathbb{A}_h^{q_2}\mathbb{V}$
12:         end for
13:         for $q_2 = 1 : Q_f$
14:             $\mathbf{d}_{q_1,q_2} = \mathbb{Z}^T\mathbf{f}_h^{q_2}$
15:         end for
16:     end for
17:     return $\mathbf{c}_{q_1,q_2}, \mathbf{d}_{q_1,q_2}, \mathbb{E}_{q_1,q_2}$
18: end function
```

Remark 2.39

The $\boldsymbol{\mu}$-independent quantities

$$c_{q_1,q_2} \in \mathbb{R}, \quad \boldsymbol{d}_{q_1,q_2} \in \mathbb{R}^N, \quad \mathbb{E}_{q_1,q_2} \in \mathbb{R}^{N\times N},$$

can be precomputed and store offline (see Algorithm 4).

Note that in the LS-RB case

$$\boldsymbol{d}_{q_1,q_2} = \boldsymbol{f}_N^{q_1,q_2}, \quad \mathbb{E}_{q_1,q_2} = \mathbb{A}_N^{q_1,q_2}.$$

For both cases studied before, the offline computational complexity scales as

$$O(Q_f n_{ls} + Q_f^2 n_{vv} + Q_a(N n_{ls} + N n_{mv}) + Q_f Q_a N n_{vv} + Q_a^2(N n_{mv} + N^2 n_{vv})),$$

while the online operation count yields

$$O(Q_f^2 + Q_f Q_a N + Q_a^2 N^2).$$

Finally, we focus now on the stability factor $\beta_h(\boldsymbol{\mu}) = \sigma_{min}(\mathbb{X}_h^{-1/2}\mathbb{A}_h(\boldsymbol{\mu})\mathbb{X}_h^{-1/2})$. For any given $\boldsymbol{\mu} \in \mathcal{P}$, the computation of the stability factor requires to solve the generalized eigenvalue problem.This would require $O(N_h^\alpha)$ operations with $\alpha \in [1,3]$, an unaffordable task in a real-time context. Different strategies have been developed to get rid of this N_h-dependence and enable a fast evaluation of the error bound.

- Successive Constraint Method

  One strategy consists in computing a parameter-dependent lower bound $\beta_h^{LB}(\boldsymbol{\mu})$ to $\beta_h(\boldsymbol{\mu})$ by means of the Successive Constraint Method (SCM).

  For both strongly and weakly coercive problems, the SCM-based lower bound $\beta_h^{LB} : \mathcal{P} \to \mathbb{R}$ such that

  $$0 < \beta_h^{LB}(\boldsymbol{\mu}) \leq \beta_h(\boldsymbol{\mu}), \quad \forall \boldsymbol{\mu} \in \mathcal{P},$$

  whose online evaluation requires the solution of a small linear program with computational complexity independent of N_h.

  The resulting error bound is thus efficiently computable and rigorous. However, the offline-online strategy developed to build the lower bound is applicable only in case of affine parametric dependence.

  The SCM algorithm is rather involved to implement and requires a considerable computational effort, beyond the scope of the current work.

- Interpolatory method

  An alternative strategy, targeted to computational efficiency, relies instead on computing an interpolatory approximation $\beta_I(\boldsymbol{\mu})$ of $\beta_h(\boldsymbol{\mu})$.

  Let us denote by $\Xi_{fine} \in \mathcal{P}$ a sample set whose dimension $n_{fine} = |\Xi_{fine}|$ is sufficiently large. We select a set of parameters, called interpolation points, $\Xi_I = \{\boldsymbol{\mu}^j\}_{j=1}^{n_I}$ and compute the stability factor $\beta_h(\boldsymbol{\mu})$ for each $\boldsymbol{\mu} \in \Xi_I$. Then we compute a suitable interpolant $\beta_I(\boldsymbol{\mu})$, such that,

  $$\beta_I(\boldsymbol{\mu}) = \beta_h(\boldsymbol{\mu}), \quad \forall \boldsymbol{\mu} \in \Xi_I \text{ and } \beta_I > 0, \quad \forall \boldsymbol{\mu} \in \Xi_{fine}.$$

The entire procedure for the online evaluation of the error estimator $\Delta_N(\boldsymbol{\mu})$ is summarized in Algorithm 5.

Algorithm 5 Evaluation of the error estimator $\Delta_N(\boldsymbol{\mu})$

```
 1: function ErrorEstimate($\mathbf{c}_{q_1,q_2}, \mathbf{d}_{q_1,q_2}, \mathbb{E}_{q_1,q_2}, \theta_a^q, \theta_f^q, \mathbf{u}_N(\boldsymbol{\mu}), \boldsymbol{\mu}$)
 2:     compute $\beta(\boldsymbol{\mu})$
 3:     $\varepsilon = 0$
 4:     for $q_1 = 1 : Q_f$
 5:         for $q_2 = 1 : Q_f$
 6:             $\varepsilon \leftarrow \varepsilon + \theta_f^{q_1}(\boldsymbol{\mu})\theta_f^{q_2}(\boldsymbol{\mu})c_{q_1,q_2}$
 7:         end for
 8:     end for
 9:     for $q_1 = 1 : Q_a$
10:         for $q_2 = 1 : Q_a$
11:             $\varepsilon \leftarrow \varepsilon + \theta_a^{q_1}(\boldsymbol{\mu})\theta_a^{q_2}(\boldsymbol{\mu})\mathbf{u}_N(\boldsymbol{\mu})^T\mathbb{E}_{q_1,q_2}\mathbf{u}_N(\boldsymbol{\mu})$
12:         end for
13:         for $q_2 = 1 : Q_f$
14:             $\varepsilon \leftarrow \varepsilon + \theta_a^{q_1}(\boldsymbol{\mu})\theta_f^{q_2}(\boldsymbol{\mu})\mathbf{u}_N(\boldsymbol{\mu})^T\mathbf{d}_{q_1,q_2}$
15:         end for
16:     end for
17:     return $\Delta_N(\boldsymbol{\mu}) = \sqrt{\varepsilon}/\beta(\boldsymbol{\mu})$
18: end function
```

Chapter 3

Construction of Reduced Basis Spaces

As we stated in the previous chapter, setting a Reduced Basis method entails three things,

1. Construction of a Reduced Basis space.
2. Computation of the Reduced Basis solution.
3. Setup of a Reduced Basis problem.

We have already studied the latter two.
In this chapter we focus on the first step, the construction of the Reduced Basis space. We present two different methods in order to generate the Reduced Basis space.

- On the one hand, in section 3.1, we talk about the construction by Proper Orthogonal Decomposition (POD) using a Singular Value Decomposition analysis (SVD).
- On the other hand, in section 3.2, we present the Greedy algorithm, that consist in an iterative sampling from the parameter space fulfilling at each step a suitable optimality criterion that relies on the a posteriori error estimate (see section 2.4).

For the construction of this chapter we have mainly used reference [12].

3.1 SVD-POD

First of all, we recall the basic notions about the Singular Value Decomposition of a matrix. Then we address the Proper Orthogonal Decomposition (POD) and we elaborate on the use of POD to generate a Reduced Basis space.

3.1.1 Basic Notions on Singular Value Decomposition

Singular Value Decomposition of a matrix is a diagonalization process involving left and right multiplication by orthogonal matrices. We summarize here the most important results of the Singular Value Decomposition.

Proposition 3.1

If $\mathbb{A} \in \mathbb{R}^{m\times n}$ is a real matrix, there exist two orthogonal matrices $\mathbb{U} = [\boldsymbol{\zeta}_1|\ldots|\boldsymbol{\zeta}_m] \in \mathbb{R}^{m\times m}$ and $\mathbb{Z} = [\boldsymbol{\Phi}_1|\ldots|\boldsymbol{\Phi}_n] \in \mathbb{R}^{n\times n}$ such that

$$\mathbb{A} = \mathbb{U}\Sigma\mathbb{Z}^T, \tag{3.1}$$

with $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_p) \in \mathbb{R}^{m\times n}$ and $\sigma_1 \geq \cdots \geq \sigma_p \geq 0$, for $p = \min(n, m)$. ■

We have already defined in Remark 1.11 the singular values for a square matrix, here we define them for a generic matrix.

Definition 3.1

We name

- *$\sigma_i = \sigma_i(\mathbb{A})$, the singular values of the matrix $\mathbb{A}$.*
- *$\boldsymbol{\zeta}_i$, are the left singular vectors of $\mathbb{A}$.*
- *$\boldsymbol{\Phi}_j$, are the right singular vectors of $\mathbb{A}$.*

We outline now the relationship between the singular values of a matrix $\mathbb{A}$ and an associate eigenvalue problem.

Remark 3.2

Expression (3.1) implies the spectral decompositions,

$$\mathbb{A}\mathbb{A}^T = \mathbb{U}\Sigma\Sigma^T\mathbb{U}^T \text{ and } \mathbb{A}^T\mathbb{A} = \mathbb{Z}\Sigma^T\Sigma\mathbb{Z}^T,$$

where,

$$\Sigma\Sigma^T = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_p^2, \overbrace{0, \ldots, 0}^{m-p}), \text{ and } \Sigma^T\Sigma = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_p^2, \overbrace{0, \ldots, 0}^{n-p}).$$

Remark 3.3

Since $\mathbb{A}\mathbb{A}^T$ and $\mathbb{A}^T\mathbb{A}$ are symmetric, the left (right) singular vectors of $\mathbb{A}$ turn out to be the eigenvectors of $\mathbb{A}\mathbb{A}^T(\mathbb{A}^T\mathbb{A})$.

Remark 3.4

There is a very close relationship between the SVD of $\mathbb{A}$ and the eigenvalue problems for $\mathbb{A}^T\mathbb{A}$ and $\mathbb{A}\mathbb{A}^T$, since

$$\sigma_i = \sqrt{\lambda_i(\mathbb{A}^T\mathbb{A})}, \ i = 1, \dots, p.$$

This is the same definition as given for square matrices in remark 1.11.

Remark 3.5

If $\mathbb{A} \in \mathbb{R}^{n\times n}$ is a symmetric matrix, then, $\sigma_i(\mathbb{A}) = |\lambda_i(\mathbb{A})|$ with $\lambda_1(\mathbb{A}) \geq \cdots \geq \lambda_n(\mathbb{A})$ being the eigenvalues of $\mathbb{A}$.

Remark 3.6

The singular values of a matrix are related to both its norm and its condition number

$$\|\mathbb{A}\|_S = \sqrt{\lambda_{max}(\mathbb{A}^*\mathbb{A})} = \sigma_{max}, \ \|\mathbb{A}\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2} = \sqrt{\sum_{i=1}^{p}\sigma_i^2}.$$

Remark 3.7

If $\mathbb{A} \in \mathbb{R}^{n\times n}$ is nonsingular, we obtain,

$$\mathbb{A}^{-1} = \mathbb{Z}\Sigma^{-1}\mathbb{U}^T,$$

with $\Sigma^{-1} = \mathrm{diag}(\sigma_1^{-1}, \dots, \sigma_n^{-1})$, this shows that σ_n^{-1}, the inverse of the smallest singular value of $\mathbb{A}$, is the largest singular value of $\mathbb{A}^{-1}$, whence

$$\|\mathbb{A}^{-1}\|_S = \frac{1}{\sigma_n} \to \kappa(\mathbb{A}) = \|\mathbb{A}\|_S\|\mathbb{A}^{-1}\|_S = \frac{\sigma_1}{\sigma_n}.$$

Now that we have presented the most important results of SVD theory, we focus on low-rank approximations to a matrix $\mathbb{A}$.
Since $\mathrm{rank}(\mathbb{A}) = \mathrm{rank}(\Sigma)$ and the latter is equal to the number of its nonzero diagonal entries, if $\mathbb{A} \in \mathbb{R}^{m\times n}$ has r positive singular values, then $\mathrm{rank}(\mathbb{A}) = r$. Not only, we can provide an orthonormal basis for both the kernel and the range of $\mathbb{A}$,

as follows,

$$\ker(\mathbb{A}) = \operatorname{span}(\boldsymbol{\Phi}_{r+1}, \ldots, \boldsymbol{\Phi}_n), \text{ and } \operatorname{range}(\mathbb{A}) = \operatorname{span}(\boldsymbol{\zeta}_1, \ldots, \boldsymbol{\zeta}_r).$$

Proposition 3.2

If $\mathbb{A} \in \mathbb{R}^{m\times n}$ has rank equal to r, then it can be written as the sum of r rank-1 matrices

$$\mathbb{A} = \sum_{i=1}^{r} \sigma_i \boldsymbol{\zeta}_i \boldsymbol{\Phi}_i^T$$

■

The next theorem states some optimality properties for low-rank approximations.

Theorem 3.3 (*Schmidt-Eckard-Young*)

Given a matrix $\mathbb{A} \in \mathbb{R}^{m\times n}$ of rank r, the matrix,

$$\mathbb{A}_k = \sum_{i=1}^{k} \sigma_i \boldsymbol{\zeta}_i \boldsymbol{\Phi}_i^T,$$

satisfy the optimality property,

$$\|\mathbb{A} - \mathbb{A}_k\|_F = \min_{\substack{\mathbb{B}\in\mathbb{R}^{m\times n}\\ \operatorname{rank}(\mathbb{B})\le k}} \|\mathbb{A} - \mathbb{B}\|_F = \sqrt{\sum_{i=k+1}^{r} \sigma_i^2},$$

and

$$\|\mathbb{A} - \mathbb{A}_k\|_2 = \min_{\substack{\mathbb{B}\in\mathbb{R}^{m\times n}\\ \operatorname{rank}(\mathbb{B})\le k}} \|\mathbb{A} - \mathbb{B}\|_2 = \sigma_{k+1}.$$

■

In Reduced Basis construction, the SVD of a matrix $\mathbb{A} \in \mathbb{R}^{m\times n}$ is often realized by picking a rectangular left matrix $\mathbb{U}_1 \in \mathbb{R}^{m\times n}$ instead of a squared matrix $\mathbb{U} \in \mathbb{R}^{m\times m}$. In such case, we obtain

$$\mathbb{A} = \mathbb{U}_1 \Sigma_1 \mathbb{Z}^T \text{ with } \Sigma_1 \in \mathbb{R}^{n\times n}.$$

This represents the so-called thin SVD.

Remark 3.8

In this case $\mathbb{U}_1 = \mathbb{U}(:, 1:n) = [\boldsymbol{\zeta_1}|\ldots|\boldsymbol{\zeta_n}] \in \mathbb{R}^{m\times n}$ *and* $\Sigma_1 = \Sigma(1:n, 1:n) = \operatorname{diag}(\sigma_1, \ldots, \sigma_n) \in \mathbb{R}^{n\times n}$.

3.1.2 Proper Orthogonal Decomposition

Proper Orthogonal Decomposition is a technique for reducing the dimensionality of a given dataset by representig it onto an orthonormal basis which is optimal in a least-squares sense. The original variables are transformed into a new set of uncorrelated variables, a lower dimensional representation of the data is thus obtained by truncating the new basis to retain the first few modes.
For parametrized problems, we consider a set $\Xi_S = \{\boldsymbol{\mu}^1, \dots, \boldsymbol{\mu}^{n_s}\}$ of n_s parameter samples and the corresponding set of snapshots $\{u_h(\boldsymbol{\mu}^1), \dots, u_h(\boldsymbol{\mu}^{n_s})\}$, that is, the solutions of the high-fidelity problem (2.4). We define the snapshots matrix $\mathbb{S} \in \mathbb{R}^{N_h \times n_s}$ as,

$$\mathbb{S} = [\mathbf{u}_1 | \dots | \mathbf{u}_{n_s}],$$

where the vectors $u_i \in \mathbb{R}^{N_h}$, $1 \leq i \leq n_s$, represent the degrees of freedom of the functions $u_h(\boldsymbol{\mu}^i) \in V_h$.
The SVD of $\mathbb{S}$ reads

$$\mathbb{S} = \mathbb{U}\Sigma\mathbb{Z}^T,$$

where

$$\mathbb{U} = [\boldsymbol{\zeta}_1 | \dots | \boldsymbol{\zeta}_{N_h}] \in \mathbb{R}^{N_h \times N_h}, \quad \mathbb{Z} = [\boldsymbol{\Phi}_1 | \dots | \boldsymbol{\Phi}_{n_s}] \in \mathbb{R}^{n_s \times n_s},$$

$$\Sigma = \text{diag}(\sigma_1, \dots, \sigma_r) \in \mathbb{R}^{N_h \times n_s}, \sigma_1 \geq \dots \geq \sigma_r \text{ y } r = \min\{N_h, n_s\} = \text{rank}(\mathbb{S}).$$

Remark 3.9

Then, we can write

$$\begin{aligned} \mathbb{S}\boldsymbol{\Phi}_i &= \sigma_i \boldsymbol{\zeta}_i & &\rightarrow \mathbb{S}^T\mathbb{S}\boldsymbol{\Phi}_i = \sigma_i^2 \boldsymbol{\Phi}_i, & i &= 1, \dots, r, \\ \mathbb{S}^T \boldsymbol{\zeta}_i &= \sigma_i \boldsymbol{\Phi}_i & &\rightarrow \mathbb{S}\mathbb{S}^T \boldsymbol{\zeta}_i = \sigma_i^2 \boldsymbol{\zeta}_i, & i &= 1, \dots, r. \end{aligned}$$

Definition 3.10

The matrix $\mathbb{C} = \mathbb{S}^T\mathbb{S} \in \mathbb{R}^{n_s \times n_s}$ is known as the correlation matrix, and its elements are given by $(\mathbb{C})_{ij} = \mathbf{u}_i^T \mathbf{u}_j$, $1 \leq i, j \leq n_s$.

For any $N \leq n_s$, the POD basis $\mathbb{V} \in \mathbb{R}^{N_h \times N}$ of dimension N is defined as the set of the first N left singular vectors $\{\boldsymbol{\zeta}_1, \dots, \boldsymbol{\zeta}_N\}$ of $\mathbb{U}$, or equivalently, the set of vectors $\{\boldsymbol{\zeta}_j = \frac{1}{\sigma_j}\mathbb{S}\boldsymbol{\Phi}_j, \quad 1 \leq j \leq N\}$, obtained from the first N eigenvectors $\boldsymbol{\Phi}_1, \dots, \boldsymbol{\Phi}_N$ of the correlation matrix.

Remark 3.11

By construction the POD basis is orthonormal.
Moreover, it minimizes, over all possible N-dimensional orthonormal basis $\mathbb{W} = [\mathbf{w}_1 | \dots | \mathbf{w}_N] \in \mathbb{R}^{N_h \times N}$, the sum of squares of the error between snapshots vector $\mathbf{u}_i$ and its projection onto the space spanned by $\mathbb{W}$.

More precisely, recalling that the projection $\Pi_{\mathbb{W}}\mathbf{x}$ of a vector $\mathbf{x} \in \mathbb{R}^{N_h}$ onto span($\mathbb{W}$) is given by

$$\Pi_{\mathbb{W}}\mathbf{x} = \sum_{j=1}^{N} (\mathbf{x}, \mathbf{w}_j)_2 \mathbf{w}_j = \mathbb{W}\mathbb{W}^T\mathbf{x},$$

the following property holds.

Proposition 3.4

Let $\mathcal{V}_N = \{\mathbb{W} \in \mathbb{R}^{N_h \times N} : \mathbb{W}^T\mathbb{W} = \mathbb{1}_N\}$ be the set of all N-dimensional orthonormal bases.
Then,

$$\sum_{i=1}^{n_s} \|\boldsymbol{u}_i - \mathbb{V}\mathbb{V}^T\boldsymbol{u}_i\|_2^2 = \min_{\mathbb{W}\in\mathcal{V}_N} \sum_{i=1}^{n_s} \|\boldsymbol{u}_i - \mathbb{W}\mathbb{W}^T\boldsymbol{u}_i\|_2^2 = \sum_{i=N+1}^{r} = \sigma_i^2. \tag{3.2}$$

■

Remark 3.12

From (3.2), it follows that the error in the POD basis is equal to the sum of the squares of the singular values corresponding to the neglected POD modes.

This result suggests a suitable criterion to select the minimal POD dimension $N \leq r$ such that the projection error is smaller than a desired tolerance ε_{POD}. Indeed, it is sufficient to choose N as the smallest integer such that

$$I(N) = \frac{\sum_{i=1}^{N} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2} \geq 1 - \varepsilon_{POD}^2, \tag{3.3}$$

that is, the energy retained by the last $r - N$ nodes is equal or smaller than ε_{POD}^2.

Remark 3.13

$I(N)$ represents the percentage of energy of the snapshots captured by the first N POD modes, and it is referred to as the relative information content of the POD basis.

Remark 3.14

Equivalently, (3.3) ensures that the relative error between $\mathbb{S}$ and its N-rank approximation $\mathbb{S}_N$ is smaller than ε_{POD}, i.e.

$$\frac{\|\mathbb{S} - \mathbb{S}_N\|_F}{\|\mathbb{S}\|_F} \leq \varepsilon_{POD}.$$

The procedure summarized in Algorithm 6 combines the definition of POD basis, together with the optimality criterion (3.3).

Algorithm 6 POD algorithm

```
 1: function POD($\mathbb{S}, \varepsilon_{POD}$)
 2:     if $n_s \leq N_h$ then
 3:         form the correlation matrix $\mathbb{C} = \mathbb{S}^T\mathbb{S}$
 4:         solve the eigenvalue problem $\mathbb{C}\boldsymbol{\Phi}_i = \sigma_i^2\boldsymbol{\Phi}_i,\ i = 1, \dots, r$
 5:         set $\boldsymbol{\zeta}_i = \frac{1}{\sigma_i}\mathbb{S}\boldsymbol{\Phi}_i$
 6:     else
 7:         form the matrix $\mathbb{K} = \mathbb{S}\mathbb{S}^T$
 8:         solve the eigenvalue problem $\mathbb{K}\boldsymbol{\zeta}_i = \sigma_i^2\boldsymbol{\zeta}_i,\ i = 1, \dots, r$
 9:     end if
10:     define $N$ as the minimum integer such that $I(N) \geq 1 - \varepsilon_{POD}^2$.
11:     return $\mathbb{V} = [\boldsymbol{\zeta}_1|\dots|\boldsymbol{\zeta}_N]$
12: end function
```

Remark 3.15

We remark that computing the POD basis by solving an eigenvalue problem for the correlation matrix $\mathbb{C}$ yields inaccurate results for the modes associated to small singular values. This is due to the round off errors introduced while constructing $\mathbb{C}$ and the fact that $\kappa(\mathbb{C}) = (\kappa(\mathbb{S}))^2$. In such cases, it is recommended to construct the POD basis by means of stable algorithms for the computation of the SVD.

Since the snapshots functions $u_h(\boldsymbol{\mu}_i)$ belong to $V_h \subset V$, it is natural to seek an alternative POD basis which minimizes the $\mathbb{X}_h$-norm of the projection error of the snapshots vectors $\mathbf{u}_i$. In particular, we seek a basis $\mathbb{W} \in \mathcal{V}_N^{\mathbb{X}_h}$ with

$$\mathcal{V}_N^{\mathbb{X}_h} = \{\mathbb{W} \in \mathbb{R}^{V_h \times N} : \mathbb{W}^T\mathbb{X}_h\mathbb{W} = \mathbb{1}_N\},$$

which minimizes the squares of the $\mathbb{X}_h$-norm of the error between each snapshot vector $\mathbf{u}_i$ and its $\mathbb{X}_h$-orthogonal projection onto the subspace spanned by $\mathbb{W}$, i.e.

$$\min_{\mathbb{W}\in\mathcal{V}_N^{\mathbb{X}_h}} \sum_{i=1}^{n_s} \|\mathbf{u}_i - \Pi_{\mathbb{W}}^{\mathbb{X}_h}\mathbf{u}_i\|_{\mathbb{X}_h}^2, \tag{3.4}$$

here $\Pi_{\mathbb{W}}^{\mathbb{X}_h}\mathbf{x} = \sum_{j=1}^{N}(\mathbf{x}, \mathbf{w}_j)_{\mathbb{X}_h}\mathbf{w}_j = \mathbb{W}\mathbb{W}^T\mathbb{X}_h\mathbf{x}$ is the $\mathbb{X}_h$-orthonormal projection of $\mathbf{x} \in \mathbb{R}^{N_h}$ onto span($\mathbb{W}$).

Remark 3.16

We have

$$\sum_{i=1}^{n_s} \|\boldsymbol{u}_i - \Pi_{\mathbb{W}}^{\mathbb{X}_h} \boldsymbol{u}_i\|_{\mathbb{X}_h}^2 = \sum_{i=1}^{n_s} \|\boldsymbol{u}_i - \mathbb{W}\mathbb{W}^T\mathbb{X}_h \boldsymbol{u}_i\|_{\mathbb{X}_h}^2 =$$

$$= \sum_{i=1}^{n_s} \|\mathbb{X}_h^{1/2}\boldsymbol{u}_i - \mathbb{X}_h^{1/2}\mathbb{W}\mathbb{W}^T\mathbb{X}_h\boldsymbol{u}_i\|_2^2 = \|\mathbb{X}_h^{1/2}\mathbb{S} - \mathbb{X}_h^{1/2}\mathbb{W}\mathbb{W}^T\mathbb{X}_h\mathbb{S}\|_F^2.$$

Substituting in (3.4) and setting $\widetilde{\mathbb{S}} = \mathbb{X}_h^{1/2}\mathbb{S}$ and $\widetilde{\mathbb{W}} = \mathbb{X}_h^{1/2}\mathbb{W}$, we finally obtain,

$$\min_{\mathbb{W}\in\mathcal{V}_N^{\mathbb{X}_h}} \sum_{i=1}^{n_s} \|\mathbf{u}_i - \Pi_{\mathbb{W}}^{\mathbb{X}_h}\mathbf{u}_i\|_{\mathbb{X}_h}^2 = \min_{\mathbb{W}\in\mathcal{V}_N} \|\widetilde{\mathbb{S}} - \widetilde{\mathbb{W}}\widetilde{\mathbb{W}}^T\widetilde{\mathbb{S}}\|_F^2.$$

At this stage, Theorem 3.3 and Proposition 3.4 yields the following result,

Proposition 3.5

Let $\mathbb{S} = [\boldsymbol{u}_i|\dots|\boldsymbol{u}_{n_s}] \in \mathbb{R}^{N_h\times n_s}$ *be a given matrix of rank* $r \leq \min(N_h, n_s)$, $\mathbb{X}_h \in \mathbb{R}^{N_h\times N_h}$ *a symmetric positive definite matrix,* $\widetilde{\mathbb{S}} = \mathbb{X}_h^{1/2}\mathbb{S}$ *and* $\widetilde{\mathbb{S}} = \widetilde{\mathbb{U}}\Sigma\widetilde{\mathbb{Z}}^T$ *its singular value decomposition, where*

$$\widetilde{\mathbb{U}} = [\widetilde{\boldsymbol{\zeta}}_1|\dots|\widetilde{\boldsymbol{\zeta}}_{N_h}] \in \mathbb{R}^{N_h\times N_h}, \text{ and } \widetilde{\mathbb{Z}} = [\widetilde{\boldsymbol{\Phi}}_1|\dots|\widetilde{\boldsymbol{\Phi}}_{n_s}] \in \mathbb{R}^{n_s\times n_s},$$

are the orthonormal matrices and $\Sigma = \mathrm{diag}(\sigma_1,\dots,\sigma_r) \in \mathbb{R}^{N_h\times n_s}$, *with* $\sigma_1,\dots,\sigma_r$. *Then, for* $N \leq r$, *the POD basis* $\mathbb{V} = [\mathbb{X}_h^{-1/2}\widetilde{\boldsymbol{\zeta}}_1|\dots|\mathbb{X}_h^{-1/2}\widetilde{\boldsymbol{\zeta}}_N]$, *is such that*

$$\sum_{i=1}^{n_s} \|\boldsymbol{u}_i - \mathbb{V}\mathbb{V}^T\mathbb{X}_h\boldsymbol{u}_i\|_{\mathbb{X}_h}^2 = \min_{\mathbb{W}\in\mathcal{V}_N^{\mathbb{X}_h}} \sum_{i=1}^{n_s} \|\boldsymbol{u}_i - \mathbb{W}\mathbb{W}^T\mathbb{X}_h\boldsymbol{u}_i\|_{\mathbb{X}_h}^2 = \sum_{i=N+1}^{r} \sigma_i^2.$$

■

From a computational perspective, since

$$\widetilde{\mathbb{S}}^T\widetilde{\mathbb{S}}\widetilde{\boldsymbol{\Phi}}_i = \sigma_i^2\widetilde{\boldsymbol{\Phi}}_i, \;\; i = 1,\dots,r,$$

if $n_s \leq N_h$, we can conveniently obtain the POD basis without forming the matrix $\mathbb{X}_h^{1/2}$. Indeed, we first compute the correlation matrix $\widetilde{\mathbb{C}} = \widetilde{\mathbb{S}}^T\widetilde{\mathbb{S}} = \mathbb{S}^T\mathbb{X}_h\mathbb{S}$ and its N eigenvectors $\widetilde{\boldsymbol{\Phi}}_1,\dots,\widetilde{\boldsymbol{\Phi}}_N$. Then, we define the POD basis as $\mathbb{V} = [\boldsymbol{\zeta}_1|\dots|\boldsymbol{\zeta}_N]$, where

$$\boldsymbol{\zeta}_i = \mathbb{X}_h^{-1/2}\widetilde{\boldsymbol{\zeta}}_i = \mathbb{X}_h^{-1/2}\frac{1}{\sigma_i}\widetilde{\mathbb{S}}\widetilde{\boldsymbol{\Phi}}_i = \frac{1}{\sigma_i}\mathbb{S}\widetilde{\boldsymbol{\Phi}}_i, \quad 1 \leq i \leq N.$$

The complete procedure is summarized in Algorithm 7.

Algorithm 7 POD algorithm with respect to the $\mathbb{X}_h$ norm

```
1: function POD($\mathbb{S}, \mathbb{X}_h, \varepsilon_{POD}$)
2:     if $n_s \leq N_h$ then
3:         form the correlation matrix $\widetilde{\mathbb{C}} = \mathbb{S}^T \mathbb{X}_h \mathbb{S}$
4:         solve the eigenvalue problem $\widetilde{\mathbb{C}}\widetilde{\boldsymbol{\Phi}}_i = \sigma_i^2 \widetilde{\boldsymbol{\Phi}}_i,\ i = 1, \ldots, r$
5:         set $\boldsymbol{\zeta}_i = \frac{1}{\sigma_i}\mathbb{S}\widetilde{\boldsymbol{\Phi}}_i$
6:     else
7:         form the matrix $\widetilde{\mathbb{K}} = \mathbb{X}_h^{1/2}\mathbb{S}\mathbb{S}^T\mathbb{X}_h^{1/2}$
8:         solve the eigenvalue problem $\widetilde{\mathbb{K}}\widetilde{\boldsymbol{\zeta}}_i = \sigma_i^2 \widetilde{\boldsymbol{\zeta}}_i,\ i = 1, \ldots, r$
9:     end if
10:    define $N$ as the minimum integer such that $I(N) \geq 1 - \varepsilon_{POD}^2$.
11:    return $\mathbb{V} = [\boldsymbol{\zeta}_1 | \ldots | \boldsymbol{\zeta}_N]$
12: end function
```

$\mathcal{P}$-continuous version

The POD basis is the one which best approximates the set of solution snapshots $\mathcal{M}_h^S = \{\mathbf{u}_h(\boldsymbol{\mu}_1), \ldots, \mathbf{u}_h(\boldsymbol{\mu}_{n_s})\}$. However, we still have to investigate,

- The approximation property of the POD basis with respect to the entire solution set $\mathcal{M}_h = \{\mathbf{u}_h(\boldsymbol{\mu}),\ \boldsymbol{\mu} \in \mathcal{P}\}$.
- How to select the parameter samples, $\Xi_S = \{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{n_s}\}$, so that the corresponding snapshots set is sufficiently representative of the solutions set.

If we are interested to find a POD basis of dimension N that approximates the entire solutions set $\mathcal{M}_h$, we are forced to work in the continuous setting. So we present the continuous analogous for the discrete problems. First, we have to consider the following minimization problem,

$$\min_{\mathbb{W} \in \mathcal{V}_N} \int_{\mathcal{P}} \|\mathbf{u}_h(\boldsymbol{\mu}) - \mathbb{W}\mathbb{W}^T\mathbf{u}_h(\boldsymbol{\mu})\|_2^2 d\boldsymbol{\mu}. \tag{3.5}$$

Let us suppose that $\mathbf{u}_h(\boldsymbol{\mu}) \in L^2(\mathcal{P}; \mathbb{R}^{N_h})$, i.e. $\int_{\mathcal{P}} \|\mathbf{u}_h(\boldsymbol{\mu})\|_2^2 d\boldsymbol{\mu} < \infty$, then $\mathbf{u}_h(\boldsymbol{\mu})$ is called a Hilbert-Schmidt (H-S) kernel.

Remark 3.17

A continuous analogue to the snapshots matrix $\mathbb{S}$ is given by the operator $T : L^2(\mathcal{P}) \to \mathbb{R}^{N_h}$ such that, $\forall g \in L^2(\mathcal{P})$,

$$Tg = \int_{\mathcal{P}} \mathbf{u}_h(\boldsymbol{\mu}) g(\boldsymbol{\mu}) d\boldsymbol{\mu}.$$

Its adjoint $T^* : \mathbb{R}^{N_h} \to L^2(\mathcal{P})$ is defined as

$$(g, T^*\mathbf{w})_{L^2(\mathcal{P})} = (Tg, \mathbf{w})_2 \ \forall g \in L^2(\mathcal{P}), \mathbf{w} \in \mathbb{R}^{N_h}.$$

As a result,

$$T^*\mathbf{w} = (\mathbf{u}_h(\boldsymbol{\mu}), \mathbf{w})_2 \ \forall \mathbf{w} \in \mathbb{R}^{N_h},$$

Remark 3.18

That is, T^* is the continuous analogue of the matrix $\mathbb{S}^T$.

Remark 3.19

Note that $Tg \in \mathbb{R}^{N_h}$ and $\operatorname{rank}(T) = r \leq N_h$, while $T^*\mathbf{w} \in L^2(\mathcal{P})$.

Remark 3.20

Since $\boldsymbol{u}_h(\boldsymbol{\mu})$ is a H-S kernel, T is a H-S operator and thus compact. Moreover, the H-S norm of T coincides with the norm of its kernel, i.e.

$$\|T\|^2_{H-S} = \|\boldsymbol{u}_h(\boldsymbol{\mu})\|^2_{L^2(\mathcal{P};\mathbb{R}^{N_h})}.$$

Remark 3.21

Since T is compact, $K = TT^* : \mathbb{R}^{N_h} \to \mathbb{R}^{N_h}$ and $C = T^*T : L^2(\mathcal{P}) \to L^2(\mathcal{P})$ are self-adjoint, non-negative and compact operators, given by

$$\begin{aligned} Cg &= \int_{\mathcal{P}} (\boldsymbol{u}_h(\boldsymbol{\mu}), \boldsymbol{u}_h(\boldsymbol{\mu}'))_2 g(\boldsymbol{\mu}') d\boldsymbol{\mu}', \quad \forall g \in L^2(\mathcal{P}), \\ K\mathbf{w} &= \int_{\mathcal{P}} \boldsymbol{u}_h(\boldsymbol{\mu})(\boldsymbol{u}_h(\boldsymbol{\mu}), \mathbf{w})_2 d\boldsymbol{\mu}, \qquad \forall \mathbf{w} \in \mathbb{R}^{N_h}. \end{aligned}$$

Remark 3.22

Operators K and C are the $\mathcal{P}$-continuous analogue of the correlation matrices $\mathbb{K} = \mathbb{S}\mathbb{S}^T$ and $\mathbb{C} = \mathbb{S}^T\mathbb{S}$, respectively.

Actually, the following proposition holds.

Proposition 3.6

Since K is a linear mapping from $\mathbb{R}^{N_h}$ to $\mathbb{R}^{N_h}$, it is represented by the $N_h \times N_h$ symmetric, positive definite matrix,

$$K = \int_{\mathcal{P}} \boldsymbol{u}_h(\boldsymbol{\mu}) \boldsymbol{u}_h^T(\boldsymbol{\mu}) d\boldsymbol{\mu},$$

whose eigenvalues $\sigma_1^2 \geq \cdots \geq \sigma_r^2 \geq 0$ and associated orthonormal eigenvectors $\boldsymbol{\zeta}_i \in \mathbb{R}^{N_h}$ satisfy,

$$K\boldsymbol{\zeta}_i = \sigma_i^2 \boldsymbol{\zeta}_i, \ i = 1, \dots, r.$$

Moreover, the functions $\Phi_i \in L^2(\mathcal{P})$ defined by

$$\Phi_i = \frac{1}{\sigma_i} T^* \boldsymbol{\zeta}_i, \ i = 1, \dots, r,$$

are the eigenvectors of C. ■

Finally, the following results holds.

Proposition 3.7

As $\boldsymbol{u}_h(\boldsymbol{\mu})$ admits the expansion,

$$\boldsymbol{u}_h(\boldsymbol{\mu}) = \sum_{i=1}^{r} \sigma_i \boldsymbol{\zeta}_i \Phi_i(\boldsymbol{\mu}) = \sum_{i=1}^{r} \boldsymbol{\zeta}_i (\boldsymbol{u}_h(\boldsymbol{\mu}), \boldsymbol{\zeta}_i)_2,$$

the following decomposition holds for T,

$$T(\cdot) = \sum_{i=1}^{r} \sigma_i \boldsymbol{\zeta}_i (\Phi_i(\boldsymbol{\mu}), \cdot)_{L^2(\mathcal{P})}. \tag{3.6}$$

■

Remark 3.23

As in the matrix context, it can be easily proved that

$$\|T\|_{L(L^2(\mathcal{P});\mathbb{R}^{N_h})} = \sigma_1, \quad \|T\|_{H-S} = \sqrt{\sum_{i=1}^{r} \sigma_i^2}.$$

Moreover, truncating the sum (3.6) to first N terms, we obtain the best rank N approximation to the operator T, as the following theorem states.

Theorem 3.8 (*Schmidt*)

The operator $T_N : L^2(\mathcal{P}) \to \mathbb{R}^{N_h}$ defined by

$$T_N(\cdot) = \sum_{i=1}^{N} \sigma_i \boldsymbol{\zeta}_i (\Phi_i(\boldsymbol{\mu}), \cdot)_{L^2(\mathcal{P})}, \ 0 \leq N \leq r,$$

satisfies the following optimality property,

$$\|T - T_N\|_{H-S} = \min_{B \in \mathcal{B}_N} \|T - B\|_{H-S} = \sqrt{\sum_{i=N+1}^{r} \sigma_i^2},$$

where $\mathcal{B}_N = \{B \in L(L^2(\mathcal{P}); \mathbb{R}^{N_h}) : \text{rank}(B) \leq N\}$. ∎

Proposition 3.9

The POD basis $\mathbb{V} = [\boldsymbol{\zeta}_1 | \dots | \boldsymbol{\zeta}_N] \in \mathbb{R}^{N_h \times N}$ *is such that*

$$\int_{\mathcal{P}} \|\boldsymbol{u}_h(\boldsymbol{\mu}) - \mathbb{V}\mathbb{V}^T \boldsymbol{u}_h(\boldsymbol{\mu})\|_2^2 d\boldsymbol{\mu} = \min_{\mathbb{W} \in \mathcal{V}_N} \int_{\mathcal{P}} \|\boldsymbol{u}_h(\boldsymbol{\mu}) - \mathbb{W}\mathbb{W}^T \boldsymbol{u}_h(\boldsymbol{\mu})\|_2^2 d\boldsymbol{\mu}. \tag{3.7}$$

Moreover,

$$\int_{\mathcal{P}} \|\boldsymbol{u}_h(\boldsymbol{\mu}) - \mathbb{V}\mathbb{V}^T \boldsymbol{u}_h(\boldsymbol{\mu})\|_2^2 d\boldsymbol{\mu} = \sum_{i=N+1}^{r} \sigma_i^2.$$

∎

Remark 3.24

We can generalize Proposition 3.9 to the case where the $\mathbb{X}_h$-norm, rather than the Euclidean norm is considered.

Thanks to the previous analysis of the $\mathcal{P}$-continuous version of the POD, we can now answer the questions raised at the beginning concerning approximation and sampling properties of POD basis functions.

Approximation and sampling properties

The key is to regard the discrete minimization problem (3.2) as an approximation of the continuous minimization problem (3.5). To this end, we introduce a suitable quadrature formula,

$$\int_{\mathcal{P}} f(\boldsymbol{\mu}) d\boldsymbol{\mu} \approx \sum_{i=1}^{n_s} w_i f(\boldsymbol{\mu}_i),$$

to approximate the integrals over $\mathcal{P}$ for any continuous function $f : \mathcal{P} \to \mathbb{R}$, where $w_i > 0$ and $\boldsymbol{\mu}_i$ are weighting coefficients and suitable quadrature points, respectively. Then,

$$\int_{\mathcal{P}} \|\mathbf{u}_h(\boldsymbol{\mu}) - \mathbb{W}\mathbb{W}^T \mathbf{u}_h(\boldsymbol{\mu})\|_2^2 d\boldsymbol{\mu} \approx \sum_{i=1}^{n_s} w_i \|\mathbf{u}_h(\boldsymbol{\mu}_i) - \mathbb{W}\mathbb{W}^T \mathbf{u}_h(\boldsymbol{\mu}_i)\|_2^2. \tag{3.8}$$

The parameter samples location are thus defined by a suitable quadrature formula in the parameter space.

Remark 3.25

Indeed, the right-hand side of (3.8) differs from (3.2) only by the weighting coefficients w_i.

However, by defining $\mathbb{D} = \operatorname{diag}(w_1, \dots, w_{n_s}) \in \mathbb{R}^{n_s \times n_s}$,

$$\sum_{i=1}^{n_s} w_i \|\mathbf{u}_h(\boldsymbol{\mu}_i) - \mathbb{W}\mathbb{W}^T\mathbf{u}_h(\boldsymbol{\mu}_i)\|_2^2 = \|\mathbb{S}\mathbb{D}^{1/2} - \mathbb{W}\mathbb{W}^T\mathbb{S}\mathbb{D}^{1/2}\|_F^{1/2}. \tag{3.9}$$

The POD basis associated to the snapshots set, $\mathcal{M}_h^S = \{\mathbf{u}_h(\boldsymbol{\mu}_1), \dots, \mathbf{u}_h(\boldsymbol{\mu}_{n_s})\}$, is given by

$$\mathbb{V} = [\boldsymbol{\zeta}_1 | \dots | \boldsymbol{\zeta}_N], \text{ with } \boldsymbol{\zeta}_i = \frac{1}{\sigma_i^2}\mathbb{S}\mathbb{D}^{1/2}\widehat{\boldsymbol{\Phi}}_i,$$

and

$$\widehat{\mathbb{S}}^T\widehat{\mathbb{S}}\widehat{\boldsymbol{\Phi}}_i = \sigma_i^2\widehat{\boldsymbol{\Phi}}_i, \text{ with } \widehat{\mathbb{S}} = \mathbb{S}\mathbb{D}^{1/2}.$$

The complete procedure to compute $\mathbb{V}$ is summarized in Algorithm 8.

Algorithm 8 POD algorithm with respect to the energy norm and quadrature weights

```
 1: function POD($\mathbb{S}, \mathbb{X}_h, \mathbb{D}, \varepsilon_{POD}$)
 2:     if $n_s \leq N_h$ then
 3:         set $\widehat{\mathbb{S}} = \mathbb{S}\mathbb{D}^{1/2}$
 4:         form the correlation matrix $\widehat{\mathbb{C}} = \widehat{\mathbb{S}}^T\mathbb{X}_h\widehat{\mathbb{S}}$
 5:         solve the eigenvalue problem $\widehat{\mathbb{C}}\widehat{\boldsymbol{\Phi}}_i = \sigma_i^2\widehat{\boldsymbol{\Phi}}_i,\ i = 1, \dots, r$
 6:         set $\boldsymbol{\zeta}_i = \frac{1}{\sigma_i}\widehat{\mathbb{S}}\widehat{\boldsymbol{\Phi}}_i$
 7:     else
 8:         form the matrix $\widehat{\mathbb{K}} = \mathbb{X}_h^{1/2}\mathbb{S}\mathbb{D}\mathbb{S}^T\mathbb{X}_h^{1/2}$
 9:         solve the eigenvalue problem $\widehat{\mathbb{K}}\widehat{\boldsymbol{\zeta}}_i = \sigma_i^2\widehat{\boldsymbol{\zeta}}_i,\ i = 1, \dots, r$
10:         set $\boldsymbol{\zeta}_i = \mathbb{X}_h^{-1/2}\widehat{\boldsymbol{\zeta}}_i$
11:     end if
12:     define $N$ as the minimum integer such that $I(N) \geq 1 - \varepsilon_{POD}^2$.
13:     return $\mathbb{V} = [\boldsymbol{\zeta}_1 | \dots | \boldsymbol{\zeta}_N]$
14: end function
```

Let us now denote by $\mathbb{V}^\infty$ the POD basis solution of the continuous minimization problem (3.7) and by $\mathbb{V}^{n_s}$ the POD basis minimizing (3.9).
Moreover, we define,

$$\mathcal{E}(\mathbb{W}) = \int_{\mathcal{P}} \|\mathbf{u}_h(\boldsymbol{\mu}) - \mathbb{W}\mathbb{W}^T\mathbf{u}_h(\boldsymbol{\mu})\|_2^2 d\boldsymbol{\mu},$$

$$\mathcal{E}^S(\mathbb{W}) = \sum_{i=1}^{n_s} \|\mathbf{u}_h(\boldsymbol{\mu}_i) - \mathbb{W}\mathbb{W}^T\mathbf{u}_h(\boldsymbol{\mu}_{n_s})\|_2^2.$$

Thanks to the optimality result (3.7), it follows that $\mathcal{E}(\mathbb{V}^\infty) \leq \mathcal{E}(\mathbb{V}^{n_s})$. Furthermore,

$$\mathcal{E}(\mathbb{V}^{n_s}) \leq |\mathcal{E}(\mathbb{V}^{n_s}) - \mathcal{E}^S(\mathbb{V}^{n_s})| + \mathcal{E}^S(\mathbb{V}^{n_s}) \leq E_S + \sum_{i=N+1}^{r} (\sigma_i^S)^2,$$

where E_S denotes the quadrature (or sampling error).

Remark 3.26

The retained energy criterion (3.3) can serve as a reliable estimate of the projection error $\mathcal{E}(\mathbb{V}^{n_s})$, provided that an appropriate sampling of the parameter space is performed.
If $E_S < \sum_{i=N+1}^{r} (\sigma_i^S)^2$ then,

$$\mathcal{E}(\mathbb{V}^{n_s}) \lesssim \sum_{i=N+1}^{r} (\sigma_i^S)^2 \rightarrow \frac{\mathcal{E}(\mathbb{V}^{n_s})}{\|\boldsymbol{u}_h\|^2_{L^2(\mathcal{P};\mathbb{R}^{N_h})}} \lesssim \varepsilon^2_{POD}.$$

Remark 3.27

The quadrature error E_S depends on:

- *The quadrature formula chosen.*
- *The number of quadrature points.*
- *The smoothness of the integrand.*
- *The dimension of the parameter space.*

Once the reduced model is built, we compute the ingredients required by the online evaluation of the a posteriori error estimate, as described in Algorithm 9.

Algorithm 9 RB approximation construction by POD algorithm

```
Require: Tolerance $\varepsilon_{POD}$, train sample $\Xi_S \subset \mathcal{P}$
1: for $\boldsymbol{\mu} \in \Xi_S$
2:     $\mathbf{u}_h(\boldsymbol{\mu}) = \text{SolveHF}(\mathbb{A}_h^q, \mathbf{f}_h^q, \theta_a^q, \theta_f^q, \boldsymbol{\mu})$
3:     $\mathbb{S} \leftarrow [\mathbb{S}\mathbf{u}_h(\boldsymbol{\mu})]$
4: end for
5: $\mathbb{V} = \text{POD}(\mathbb{S}, \mathbb{X}_h, \varepsilon_{POD}, \mathbb{D})$
6: $[\mathbb{A}_N^q, \mathbf{f}_N^q] = \text{OFFAssemble}(\mathbb{A}_h^q, \mathbf{f}_h^q, \mathbb{V}, \mathbb{X}_h, \text{method})$
7: $[c_{q_1,q_2}, \mathbf{d}_{q_1,q_2}, \mathbb{E}_{q_1,q_2}] = \text{OFFResidual}(\mathbb{A}_h^q, \mathbf{f}_h^q, \mathbb{X}_h, \mathbb{V})$
```

Remark 3.28

Different sampling strategies can be employed depending on the dimension of the parameter space,

- *Low dimension: Tensorial (full factorial) sampling.*
- *High dimension: Random sampling, Latin hypercube sampling, Sparse grids.*

3.2 Greedy Algorithm

In the case of parametrized PDEs, the POD method for the construction of the RB space might entail a severe computational cost. If the solution of the high-fidelity problem is computationally demanding, performing POD can be excessively expensive.
Greedy algorithms represent an efficient alternative. The goal of this method is to evaluate N snapshots to construct a RB space of dimension N, by seeking at each step the local optimum. However, there are two critical aspects:

- To be efficient, a greedy algorithm must be supported by an a posteriori error estimate for the error $\|u_h(\boldsymbol{\mu}) - u_N(\boldsymbol{\mu})\|_V$, whose evaluation must be performed in a very inexpensive way for any $\boldsymbol{\mu} \in \mathcal{P}$.
- A greedy algorithm is not necessarily cheaper than POD, since at each step a maximization problem has to be solved.

3.2.1 Behind Greedy Algorithm

Instead than optimizing over all possible N-dimensional subspaces, a greedy algorithm is a procedure for the construction of a subspace by iteratively adding a new basis vector at each step satisfying a optimality condition.
More precisely, at the generic iteration $1 \leq n \leq N-1$, we assume that we are given:

1. A sample set,
$$S_n = \{\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_n\},$$
2. The corresponding subspace,
$$V_n = \text{span}\{u_h(\boldsymbol{\mu}_1), \dots, u_h(\boldsymbol{\mu}_n)\},$$
3. An orthonormal basis for V_n,
$$\mathbb{V} = [\boldsymbol{\zeta}_1 | \dots | \boldsymbol{\zeta}_n] \in \mathbb{R}^{N_h \times n}.$$

Remark 3.29

The latter is obtained by orthonormalization of the snapshots set,

$$\mathbb{S} = [\boldsymbol{u}_h(\boldsymbol{\mu}_1)| \dots |\boldsymbol{u}_h(\boldsymbol{\mu}_n)] \in \mathbb{R}^{N_h \times n},$$

where $(\boldsymbol{u}_n(\boldsymbol{\mu}_i))_j = u_h^{(j)}(\boldsymbol{\mu})$, *for* $1 \leq j \leq N_h$, $1 \leq i \leq n$.

Then, we set,

$$\boldsymbol{\mu}^{n+1} = \arg\max_{\mu \in \mathcal{P}} \|\mathbf{u}_h(\boldsymbol{\mu}) - \mathbb{V}\mathbf{u}_n(\mu)\|_{\mathbb{X}_h}, \tag{3.10}$$

where $\mathbf{u}_n(\boldsymbol{\mu}) \in \mathbb{R}^n$ denotes the solution of the RB problem (2.9).
Computing the Greedy algorithm as described before entails the solution of N optimality problems so we present a simplification of the procedure in order to reduce the computational cost for the construction of the RB space.

3.2.2 The Weak Greedy Algorithm

The weak greedy algorithm is obtained by replacing in expression (3.10),

- The parameter set $\mathcal{P}$, with a very fine sample $\Xi_{train} \subset \mathcal{P}$, of cardinality $|\Xi_{train}| = n_{train}$.
- The approximation error $\|\mathbf{u}_h(\boldsymbol{\mu}) - \mathbb{V}\mathbf{u}_n(\boldsymbol{\mu})\|_{\mathbb{X}_h}$, with the a posteriori error estimator built in the previous chapter, such that,

$$\|\mathbf{u}_h(\boldsymbol{\mu}) - \mathbb{V}\mathbf{u}_n(\boldsymbol{\mu})\|_{\mathbb{X}_h} \leq \Delta_n(\boldsymbol{\mu}), \quad \forall \boldsymbol{\mu} \in \mathcal{P}.$$

Remark 3.30

Introducing the training sample enables to turn the optimality problem into a simpler enumeration problem.

Hence, at each step, $n = 1, \dots, N-1$, of the weak greedy algorithm, we need to:

1. Evaluate the a posteriori error bound $\Delta_n(\boldsymbol{\mu})$, for any $\boldsymbol{\mu} \in \Xi_{train}$.
2. Find, by solving an enumeration problem,

$$\boldsymbol{\mu}_{n+1} = \arg\max_{\mu \in \Xi_{train}} \Delta_n(\boldsymbol{\mu}).$$

In other words, at each step we add the particular candidate snapshot that the a posteriori error bound predicts to be the worst approximated by the RB prediction associated to V_n.

Then, the final size N of the RB space V_N is such that

$$\max \Delta_N(\boldsymbol{\mu}) \leq \varepsilon_g,$$

where ε_g is a prescribed, sufficiently small, stopping tolerance. The algorithm details are reported in Algorithm 10.

Algorithm 10 Weak Greedy Algorithm

```
Require: Maximum number of iterations $N_{max}$, stopping tolerance $\varepsilon_g$, train sample
    $\Xi_{train} \subset \mathcal{P}$, starting point $\boldsymbol{\mu}_1 \in \mathcal{P}$
Ensure: Basis $\mathbb{V} \in \mathbb{R}^{N_h \times n}$
 1: $\Xi_g = []$, $\mathbb{V} = []$
 2: $N = 0$, $\delta_0 = \varepsilon_g + 1$
 3: while $N < N_{max}$ & $\delta_N > \varepsilon_g$ do
 4:     $N \leftarrow N + 1$
 5:     compute $\mathbf{u}_h(\boldsymbol{\mu}_N)$
 6:     $\boldsymbol{\zeta}_N = \textsc{GramSchmidt}(\mathbb{V}, \mathbf{u}_h(\boldsymbol{\mu}_N), \mathbb{X}_h)$
 7:     $\mathbb{V} \leftarrow [\mathbb{V}|\boldsymbol{\zeta}_N]$
 8:     $\Xi_g \leftarrow \Xi_g \cup \{\boldsymbol{\mu}_N\}$
 9:     $[\delta_N, \mu_{N+1}] = \max_{\mu \in \Xi_{train}} \Delta_N(\boldsymbol{\mu}_N)$
10: end while
```

Remark 3.31

The weak greedy algorithm requires only $O(N)$ calls to the high fidelity solver, but yields $O(Nn_{train})$ evaluations of the a posteriori error bound, each one requiring the solution of the RB problem.

As shown in the Algorithm 10, the basis $\mathbb{V}$ is kept orthonormal with respect to the scalar product in V_h by iteratively orthonormalizing the new element appended to the existing basis through a Gram-Schmidt procedure, see Algorithm 11.

Algorithm 11 Gram-Schmidt orthonormalization

```
1: function $\textsc{GramSchmidt}(\mathbb{V}, \mathbf{u}, \mathbb{X}_h)$
2:     if $\mathbb{V} = []$ then
3:         $\mathbf{z} = \mathbf{u}$
4:     else
5:         $\mathbf{z} = \mathbf{u} - \mathbb{V}\mathbb{V}^T\mathbb{X}_h\mathbf{u}$
6:     end if
7:     $\mathbf{z} \leftarrow \mathbf{z}/\|\mathbf{z}\|_{\mathbb{X}_h}$
8: end function
```

Remark 3.32

A similar procedure can be built with the energy norm.

Algorithm 12 details how to build the G-RB approximation by means of the Greedy Algorithm 10.

Algorithm 12 RB approximation construction by greedy algorithm

```
Require: Maximum number of iterations $N_{max}$, stopping tolerance $\varepsilon_g$, train sample
    $\Xi_{train} \subset \mathcal{P}$, starting point $\boldsymbol{\mu}_1 \in \mathcal{P}$
Ensure: Basis $\mathbb{V} \in \mathbb{R}^{N_h \times n}$
 1: $\Xi_g = []$, $\mathbb{V} = []$
 2: $N = 0$, $\delta_0 = \varepsilon_g + 1$
 3: while $N < N_{max}$ & $\delta_N > \varepsilon_g$ do
 4:     $N \leftarrow N + 1$
 5:     $\mathbf{u}_h(\boldsymbol{\mu}_N) = \textsc{SolveHF}(\mathbb{A}_h^q, \mathbf{f}_h^q, \theta_a^q, \theta_f^q, \boldsymbol{\mu}_N)$
 6:     $\boldsymbol{\zeta}_N = \textsc{GramSchmidt}(\mathbb{V}, \mathbf{u}_h(\boldsymbol{\mu}_N), \mathbb{X}_h)$
 7:     $\mathbb{V} \leftarrow [\mathbb{V}|\boldsymbol{\zeta}_N]$
 8:     $\Xi_g \leftarrow \Xi_g \cup \{\boldsymbol{\mu}_N\}$
 9:     $[\mathbb{A}_N^q, \mathbf{f}_N^q] = \textsc{OffAssemble}(\mathbb{A}_h^q, \mathbf{f}_h^q, \mathbb{V}, \mathbb{X}_h, \text{method})$
10:     $[c_{q_1 q_2}, \mathbf{d}_{q_1 q_2}, \mathbb{E}_{q_1 q_2}] = \textsc{OffResidual}(\mathbb{A}_h^q, \mathbf{f}_h^q, \mathbb{X}_h^q, \mathbb{V})$
11:     for $\boldsymbol{\mu} \in \Xi_{train}$
12:         $\mathbf{u}_N(\boldsymbol{\mu}) = \textsc{SolveRB}(\mathbb{A}_N^q, \mathbf{f}_N^q, \theta_a^q, \theta_f^q, \boldsymbol{\mu}, \text{method})$
13:         $\Delta_N(\boldsymbol{\mu}) = \textsc{ErrorEstimate}(c_{q_1 q_2}, \mathbf{d}_{q_1 q_2}, \mathbb{E}_{q_1 q_2}, \theta_a^q, \theta_f^q, \mathbf{u}_N(\boldsymbol{\mu}), \boldsymbol{\mu})$
14:     end for
15:     $[\delta_N, \mu_{N+1}] = \max_{\mu \in \Xi_{train}} \Delta_N(\boldsymbol{\mu}_N)$
16: end while
```

We end this section with an remark concerning the training sample, Ξ_{train}.

Remark 3.33

The choice of a good training sample is a delicate issue.
In fact, Ξ_{train} should be:

- *Small for efficiency reasons, but,*
- *Sufficiently large in order to represent the parameter set as good as possible.*

As a matter of fact, the performance of the greedy, and at a large extent, that of the RB method, crucially depends on how well the training sample is chosen.
Now that we have presented the main theoretical results for the Reduced Basis Methods, and in next chapters we apply them to different equations coming from applications.

Chapter 4

Resolution of Laplace Equation by POD Method

In the following chapters, we are going to apply the theoretical results shown before to different practical applications.
In this chapter, we consider the, classical, 2-dimensional Laplace problem occurring in surfaces of different materials characterized by a particular parameter, this parameter could represent the thermal conductivity, so it is a physical parameter. We have chosen Laplace equation as it is a simple model of PDE, its behaviour is well known and it has a great variety of applications.
To solve it, we apply the POD technique. A similar problem is studied in [12].
We present the scheme of this chapter,

1. Presentation of the problem (section 4.1), including the physical equations and boundary conditions governing the problem, the geometry, and the parameter set.

2. Calculation of the variational formulation of the problem (section 4.2) and its affine decomposition.

3. Computation of the POD technique and analysis of the results (section 4.3).

4.1 Presentation

In this section we introduce the 2-dimensional Laplace problem.

1. First, we state the (parametrized) physical equations governing the problem, along with the boundary conditions.

2. Then, we establish the domain where we solve the problem.

3. Finally, we define the parameters involved in the equations, as well as the parameter set where they belong to.

4.1.1 Physical Equations

We consider the governing parametrized 2-dimensional Laplace equation, along with the mixed boundary conditions, for the unknown $u = u(\boldsymbol{\mu})$ reads,

$$\begin{cases} -\nabla \cdot (\nu(\boldsymbol{\mu})\nabla u) = f(\boldsymbol{\mu}), & \text{in } \Omega, \\ u = u_{ext}(\boldsymbol{\mu}), & \text{on } \Gamma_D, \\ -\nu(\boldsymbol{\mu})\dfrac{\partial u}{\partial n} = g(\boldsymbol{\mu}), & \text{on } \Gamma_N, \end{cases} \tag{4.1}$$

where, $\partial\Omega = \Gamma_D \cup \Gamma_N$ and Γ_D denotes the Dirichlet boundary, while Γ_N denotes the Neumann boundary.

Remark 4.1

So that the problem is well-defined, we need to impose some regularity on the functions and on the domain:

- $\nu \in L^\infty(\mathcal{P})$.
- $f(\boldsymbol{\mu}) \in L^2(\Omega)\ \forall \boldsymbol{\mu} \in \mathcal{P}$.
- $g(\boldsymbol{\mu}) \in L^2(\Gamma_N)\ \forall \boldsymbol{\mu} \in \mathcal{P}$.
- Ω *is an open and connected subset of* R^n *of which the boundary* $\partial\Omega$, *is polygonal.*

Remark 4.2

Reduced Basis scheme does not support nonhomogeneous Dirichlet boundary conditions, this is due to the fact that we compute a solution that is a linear combination of functions that satisfy the nonhomogeneous Dirichlet boundary condition, but unless the sum of coefficients is one, the solution does not satisfy the nonhomogeneous Dirichlet boundary condition.

4.1.2 Geometry

We study the domain of the surface, $\Omega = (0,1) \times (0,1)$, consisting of two different subdomains, both with the same center, these will represent two materials with, for example, two thermal conductivities or two viscosities.

$$\Omega_1 = (1/4, 3/4) \times (1/4, 3/4) \text{ y } \Omega_2 = \Omega/\Omega_1.$$

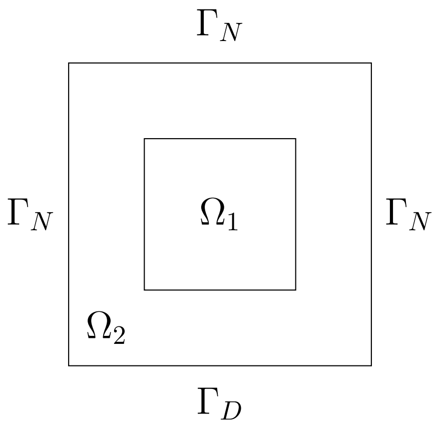


Figure 4.1: *Geometry, subdomains and boundaries for problem (4.1).*

Concerning the boundary of the domain we set, the Dirichlet boundary $\Gamma_D = (0,1)\times\{0\}$ and the Neumann boundary $\Gamma_N = \partial\Omega/\Gamma_D$. See Figure 4.1.

4.1.3 Parameters

Once we have introduced the equations and the domain, we define the parameters involved in the equations (4.1).
Before declaring the parameters, we come with a definition of a characteristic function of a domain.

Definition 4.3

χ_A denotes the characteristic function of the subdomain $A \subset \Omega$.

Since these subregions are characterized by different parameters, $\nu(\boldsymbol{\mu})$ can be expressed as,

$$\nu(\cdot,\boldsymbol{\mu}) = 1 + \mu_1\chi_{\Omega_1}(\cdot), \quad \text{in } \Omega. \tag{4.2}$$

here $1+\mu_1$ is the particular parameter for Ω_1, while for Ω_2 we have normalized the parameter as 1.
Furthermore, we set Ω_1 to be the source element, and we represent it as

$$f(\cdot,\boldsymbol{\mu}) = \mu_f\chi_{\Omega_1}(\cdot), \quad \text{in } \Omega.$$

Concerning the boundary conditions, we have

$$u_{ext}(\boldsymbol{\mu}) = 0 \text{ and } g(\boldsymbol{\mu}) = \mu_N.$$

For the case at hand, the problem depends on 3 parameters:

- μ_1 is the characteristic parameter of Ω_1,
- μ_f is the intensity of the source in Ω_1,
- μ_N represents the flux over the remaining walls of the domain Γ_N.

4.2 Variational Formulation

As declared in the theoretical introduction, we have to obtain the variational formulation of problem (4.1), so that we can exploit the affine decomposition of the problem. This decomposition provides us an efficient offline/online decomposition in order to solve the problem.
Denoting $V = H^1_{\Gamma_D}(\Omega)$[1], we multiply the first equation of (4.1) by a test function $v \in V$ and integrate over the whole domain Ω. We get,

$$\int_\Omega -\nabla \cdot (\nu(\boldsymbol{\mu})\nabla u) v d\Omega = \int_\Omega f(\boldsymbol{\mu}) v d\Omega,$$

now we apply Green's Formula in the first term and we obtain,

$$\int_{\partial\Omega} -\nu(\boldsymbol{\mu})\frac{\partial u}{\partial n} v d\Gamma + \int_\Omega \nu(\boldsymbol{\mu})\nabla u \cdot \nabla v d\Omega = \int_\Omega f(\boldsymbol{\mu}) v d\Omega,$$

splitting the first term in two integrals, one over Γ_D and another over Γ_N, we get,

$$\int_{\partial\Omega} -\nu(\boldsymbol{\mu})\frac{\partial u}{\partial n} v d\Gamma = \underbrace{\int_{\Gamma_D} -\nu(\boldsymbol{\mu})\frac{\partial u}{\partial n} v d\Gamma}_{=0} + \int_{\Gamma_N} -\nu(\boldsymbol{\mu})\frac{\partial u}{\partial n} v d\Gamma = \int_{\Gamma_N} g(\boldsymbol{\mu}) v d\Gamma,$$

where the integral over Γ_D is zero as $v \in V = H^1_{\Gamma_D}(\Omega)$.
Organizing and renaming the terms, we get that the variational formulation of problem (4.1) over the domain Ω reads:

$$\begin{cases} \text{Find } u(\boldsymbol{\mu}) \in V = H^1_{\Gamma_D}(\Omega) \text{ such that} \\ \quad a(u(\boldsymbol{\mu}), v; \boldsymbol{\mu}) = f(v; \boldsymbol{\mu}), \quad \forall v \in V, \end{cases} \tag{4.3}$$

where

$$a(u, v; \boldsymbol{\mu}) = \int_\Omega \nu(\boldsymbol{\mu})\nabla u \cdot \nabla v d\Omega, \tag{4.4}$$

$$f(v; \boldsymbol{\mu}) = \int_\Omega f(\boldsymbol{\mu}) v d\Omega - \int_{\Gamma_N} g(\boldsymbol{\mu}) v d\Gamma. \tag{4.5}$$

[1] $H^1_{\Gamma_D}(\Omega)$ denotes the Hilbert space $H^1_{\Gamma_D}(\Omega) = \{u \in H^1(\Omega) : u|_{\Gamma_D} = 0\}$.

Remark 4.4

By the regularity of the functions and of the domain the variational formulation (4.3) has a solution and it is unique, thanks to Lax-Milgram Theorem (1.1).

Remark 4.5

Note that the bilinear form $a(\cdot,\cdot;\boldsymbol{\mu})$, and the linear form $f(\cdot;\boldsymbol{\mu})$, are parameter dependent, but the domain Ω, and the solution space V, are parameter independent.

The setup, implementation and analysis of RB methods are carried out starting from the weak formulation (4.3).

4.2.1 Affine Decomposition

In order to apply the theoretical results it is necessary to express the bilinear form (4.4) and the linear form (4.5) in their affine expansion.
In the case at hand the bilinear form (4.4) admits an affine expansion. More precisely,

$$a(u,v;\boldsymbol{\mu}) = \sum_{q=1}^{Q_a} \theta_a^q(\boldsymbol{\mu}) a_q(u,v), \tag{4.6}$$

where $Q_a = 2$ and

$$\theta_a^1(\boldsymbol{\mu}) = \mu_1, \quad a_1(u,v) = \int_{\Omega_1} \nabla u \cdot \nabla v d\Omega,$$
$$\theta_a^2(\boldsymbol{\mu}) = 1, \quad a_2(u,v) = \int_{\Omega} \nabla u \cdot \nabla v d\Omega.$$

In the same way, the linear form $f(v;\boldsymbol{\mu})$ can be expressed as

$$f(v;\boldsymbol{\mu}) = \sum_{q=1}^{Q_f} \theta_f^q(\boldsymbol{\mu}) f_q(v), \tag{4.7}$$

where, again, $Q_f = 2$ and

$$\theta_f^1(\boldsymbol{\mu}) = \mu_f, \quad f_1(v) = \int_{\Omega_1} v d\Omega,$$
$$\theta_f^2(\boldsymbol{\mu}) = -\mu_N, \quad f_2(v) = \int_{\Gamma_N} v d\Gamma.$$

We have thus succeeded to write the bilinear and linear forms as linear combinations of $\boldsymbol{\mu}$-independent bilinear or linear forms, respectively, whose coefficients are suitable $\boldsymbol{\mu}$-dependent scalar, real functions. Such a decomposition is at the basis of the offline/online decoupling strategy, which is another distinguishing feature of RB methods for parametrized PDEs.

4.3 Computation

In this section we construct a computational program in order to solve the problem (4.1) in the form (4.3) numerically by means of the POD technique. We take advantage of the online/offline decomposition described in the previous chapter. We split the computation into two phases:

1. Offline Phase. In this section, we compute all the $\boldsymbol{\mu}$-independent elements of the problem (4.3):

   - First of all, we declare the domain of the problem (see Figure (4.1)).
   - Once the domain is declared, we compute the $\boldsymbol{\mu}$-independent bilinear and linear forms of the decompositions (4.6) and (4.7), respectively.
   - Then, we settle the parameter set, $\mathcal{P}$, for the parameters μ_1, μ_f and μ_N.
   - After that, we construct the snapshots matrix, associated to a randomly selected set of parameters vectors. Then, we use the POD algorithm (see Algorithm 6) so that we can obtain the Reduced Basis.
   - Finally, we follow Algorithm 2 to assemble and store the reduced matrices and vectors related to the $\boldsymbol{\mu}$-independent bilinear and linear forms of the decompositions (4.6) and (4.7), respectively.

2. Online Phase. Once we have assembled and stored the reduced matrices and vectors, we can solve the reduced problem instead of the high-fidelity problem, so we reduce the computational complexity and cost. According to the procedure described in Algorithm 3, we need to:

   - Select a parameter vector $\boldsymbol{\mu} \in \mathcal{P}$.
   - Assemble of the $\boldsymbol{\mu}$-dependent matrix and vector, $\mathbb{A}(\boldsymbol{\mu})$ and $\mathbf{f}(\boldsymbol{\mu})$, respectively.
   - Solve of the reduced linear system to obtain the Reduced Basis solution, $\mathbf{u}_N(\boldsymbol{\mu})$.

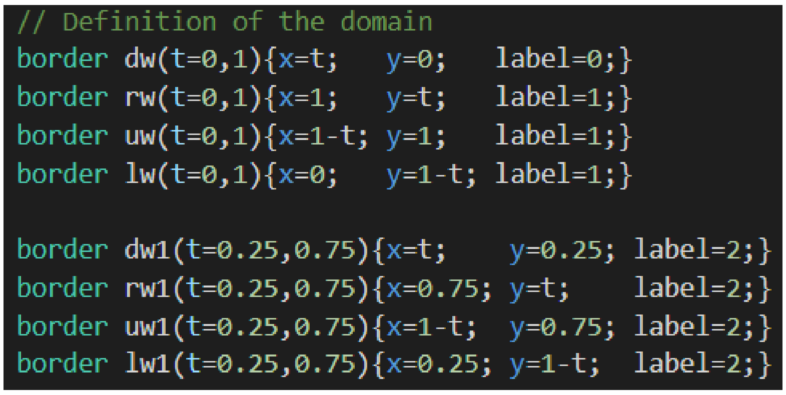
We use the *FreeFEM++* software for the computation, assembling, storage, solving and representation involved in all the the steps declared above [9].

4.3.1 Offline Computation

In this first step, we need to compute all the $\boldsymbol{\mu}$-independent elements of the program, so that in the online phase we can get an actual computational speed improvement. The main goal of this phase is to construct the reduced matrices and vectors associated with the affine decompositions (4.6 and 4.7) of the forms of the variational formulation (4.3).

Miscellaneous computations

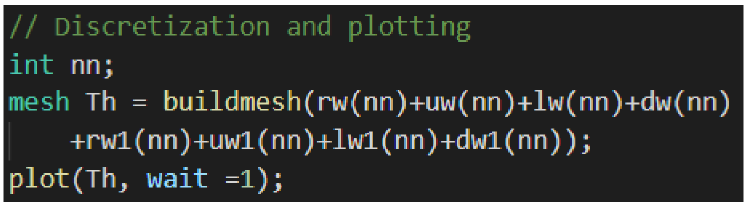
In this introduction we show all the code associated to the computation of several items, such as, the domain, the parameter set and the finite element space definition. First of all, we declare the domain of the problem described in Section 4.1. The procedure is included in Figure 4.2.



```
// Definition of the domain
border dw(t=0,1){x=t;   y=0;   label=0;}
border rw(t=0,1){x=1;   y=t;   label=1;}
border uw(t=0,1){x=1-t; y=1;   label=1;}
border lw(t=0,1){x=0;   y=1-t; label=1;}

border dw1(t=0.25,0.75){x=t;    y=0.25; label=2;}
border rw1(t=0.25,0.75){x=0.75; y=t;    label=2;}
border uw1(t=0.25,0.75){x=1-t;  y=0.75; label=2;}
border lw1(t=0.25,0.75){x=0.25; y=1-t;  label=2;}
```

Figure 4.2: *Commands used for the construction of the domain of the problem (4.1).*

We can as well discretize and plot the previous domain. We divide each side of the boundary of Ω in equal parts, we also do this to the boundary of Ω_1, the discretization of the domain obtained by this mean is called the mesh of the problem. The procedure is shown in Figure 4.3 and the resulting mesh representation (for $nn = 25$) can be seen in Figure 4.4.



```
// Discretization and plotting
int nn;
mesh Th = buildmesh(rw(nn)+uw(nn)+lw(nn)+dw(nn)
    +rw1(nn)+uw1(nn)+lw1(nn)+dw1(nn));
plot(Th, wait =1);
```

Figure 4.3: *Commands used for the discretization and plotting of the domain.*

The discretization of the domain made by triangular elements, result in Nv_h vertices and Nt_h triangles.
Now we focus on the parameter set. We first need to establish the lower and upper limits for each μ_1, μ_f and μ_N. We define:

- Characteristic parameter of Ω_1, μ_1.
- Flux over the Neumann boundary, Γ_N, μ_N.

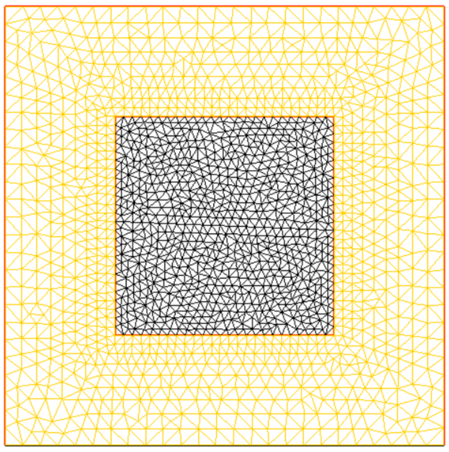


Figure 4.4: *Representation of the mesh obtained after computing the code in Figures 4.2 and 4.3.*

- Intensity of the source, μ_f.

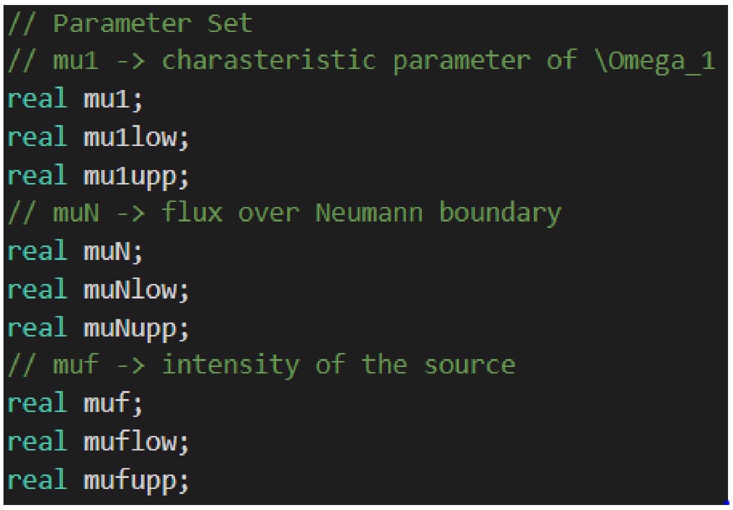
The code used is shown in Figure 4.5.
Finally, we declare the finite element space and its functions, see Figure 4.6. For the high-fidelity approximation we use $\mathbb{P}_1$ finite elements. This yields a high-fidelity space V_h of dimension $N_h = Nv_h$.
In order to simplify the notation in the code, we define the gradient of a function by means of a macro (see Figure 4.7).

Construction of the affine decomposition

At this point, we have all the necessary elements to start computing the $\boldsymbol{\mu}$-independent bilinear and linear forms of the affine decompositions (4.6) and (4.7), respectively. As well, we build the matrix and vector corresponding to the Dirichlet boundary condition. Finally, we declare the matrix, $\mathbb{X}_h$, associated to the scalar product over V_h, see Figure 4.8.

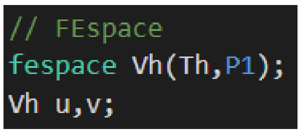
Construction of the Snapshots matrix

In order to take computational advantage of the offline/online decomposition of this problem, we need to construct the associated Reduced Space basis. To this end, we need a set of suitable selected high-fidelity solutions, corresponding to given parameter values, the so-called snapshots.
We have selected a total of n_s snapshots (see Figure 4.9), for each iteration we obtain a random vector $\boldsymbol{\mu}_i = (\mu_1, \mu_N, \mu_f) \in \mathcal{P}$, then, we build the linear system associated



```
// Parameter Set
// mu1 -> charasteristic parameter of \Omega_1
real mu1;
real mu1low;
real mu1upp;
// muN -> flux over Neumann boundary
real muN;
real muNlow;
real muNupp;
// muf -> intensity of the source
real muf;
real muflow;
real mufupp;
```

Figure 4.5: *Commands used for the definition of the parameter set.*



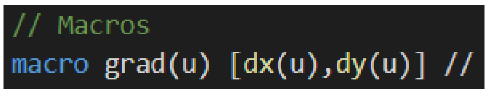
```
// FEspace
fespace Vh(Th,P1);
Vh u,v;
```

Figure 4.6: *Commands used for the definition of the finite element space and its functions.*

to the high-fidelity approximation. We solve the linear system and append the corresponding solution to the snapshots matrix.

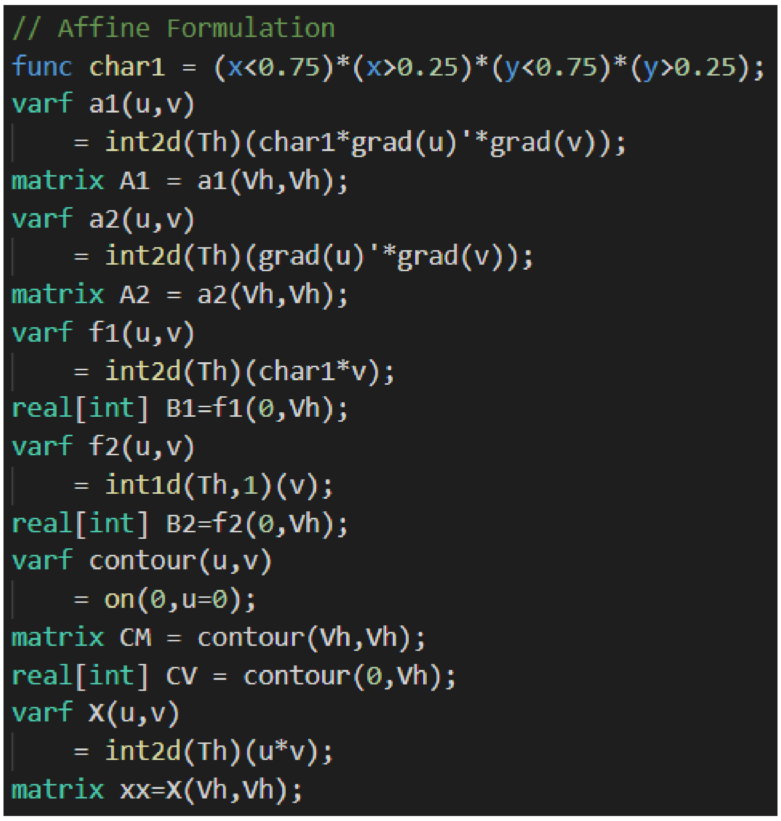
Determination of the Reduced Space Basis

Now that we have the snapshots matrix, we move onto obtaining the Reduced Basis by means of the POD algorithm (see Algorithm 6).
We will built the code such that $n_s < N_h$, so we are set in the first case in the IF-ELSE part of the POD algorithm. First of all, we need to build the correlation matrix associated to the snapshots matrix, the commands used are shown in Figure 4.10.
Now, we have to solve the eigenvalue problem associated to the correlation matrix, see POD Algorithm 6. To this end, we use the command shown in Figure 4.11.
As command dggev is not one of the mainly used commands of *FreeFEM++* lan-



```
// Macros
macro grad(u) [dx(u),dy(u)] //
```

Figure 4.7: *Commands used for the definition of the gradient of a function.*



```
// Affine Formulation
func char1 = (x<0.75)*(x>0.25)*(y<0.75)*(y>0.25);
varf a1(u,v)
    = int2d(Th)(char1*grad(u)'*grad(v));
matrix A1 = a1(Vh,Vh);
varf a2(u,v)
    = int2d(Th)(grad(u)'*grad(v));
matrix A2 = a2(Vh,Vh);
varf f1(u,v)
    = int2d(Th)(char1*v);
real[int] B1=f1(0,Vh);
varf f2(u,v)
    = int1d(Th,1)(v);
real[int] B2=f2(0,Vh);
varf contour(u,v)
    = on(0,u=0);
matrix CM = contour(Vh,Vh);
real[int] CV = contour(0,Vh);
varf X(u,v)
    = int2d(Th)(u*v);
matrix xx=X(Vh,Vh);
```

Figure 4.8: *Commands used for the computation of the elements of the affine decompositions (4.6) and (4.7), the Dirichlet boundary condition and the matrix associated to the scalar product over V_h.*

guage, we explain how we use it in order to obtain the eigenvalues of the correlation matrix.

- This command solves the more general problem

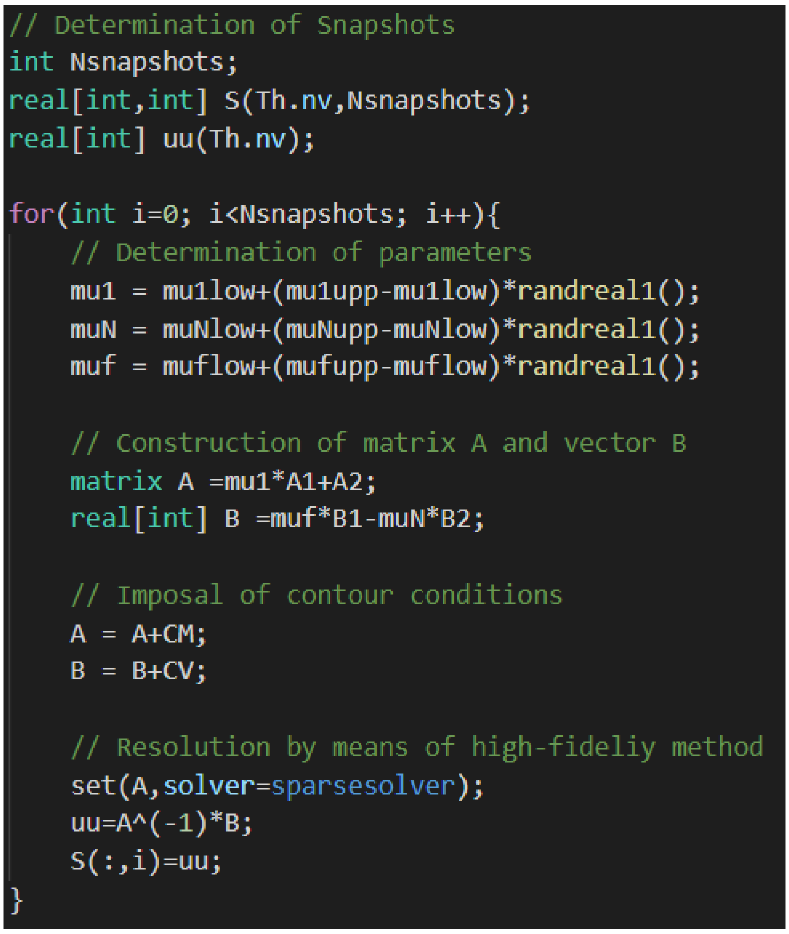
$$\mathbb{C}\boldsymbol{v} = \lambda \mathbb{B}\boldsymbol{v},$$

  so in our case $\mathbb{B} = \mathbb{1}$, i.e. the identity matrix.

- *cev* stores the (complex) eigenvalues associated to the general problem. They are stored in absolute value (modulus) decreasing order.

- *ev* is an auxiliary vector with no further use in this work.

- *eV* stores the (complex) eigenvectors associated to the general problem. They are stored in modulus decreasing order of its associated eigenvalues.

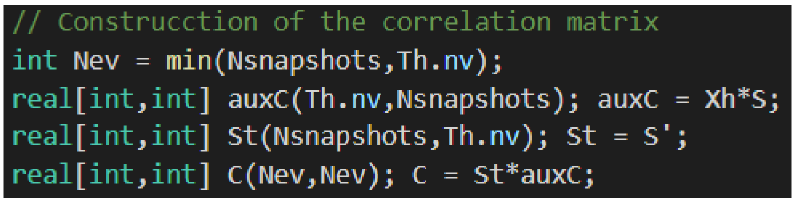


```
// Determination of Snapshots
int Nsnapshots;
real[int,int] S(Th.nv,Nsnapshots);
real[int] uu(Th.nv);

for(int i=0; i<Nsnapshots; i++){
    // Determination of parameters
    mu1 = mu1low+(mu1upp-mu1low)*randreal1();
    muN = muNlow+(muNupp-muNlow)*randreal1();
    muf = muflow+(mufupp-muflow)*randreal1();

    // Construction of matrix A and vector B
    matrix A =mu1*A1+A2;
    real[int] B =muf*B1-muN*B2;

    // Imposal of contour conditions
    A = A+CM;
    B = B+CV;

    // Resolution by means of high-fideliy method
    set(A,solver=sparsesolver);
    uu=A^(-1)*B;
    S(:,i)=uu;
}
```

Figure 4.9: *Commands used for the random election of parameter vectors and the construction of the snapshots matrix.*

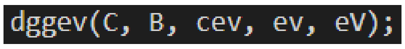
According to POD Algorithm 6, now we have to left-multiply these eigenvectors by the snapshots matrix and the inverse of the root of its associated eigenvalue in order to obtain the left eigenvectors of the SVD. The commands used are shown in Figure 4.12.
At this point, we have all the necessary ingredients to determinate the Reduced Space dimension, N [2]. We set a tolerance ε^2_{POD}, and we use the optimality criterion (3.3), see Figure 4.13.
Obtaining the Reduced Basis is now very simple, as we only have to select the first N left eigenvectors of the SVD snapshots matrix, see Figure 4.14.

[2]In the code shown in Figures 4.13 and 4.14 we name it as Nrb, because N is a protected parameter for *FreeFEM++* language



```
// Construcction of the correlation matrix
int Nev = min(Nsnapshots,Th.nv);
real[int,int] auxC(Th.nv,Nsnapshots); auxC = Xh*S;
real[int,int] St(Nsnapshots,Th.nv); St = S';
real[int,int] C(Nev,Nev); C = St*auxC;
```

Figure 4.10: *Commands used for the construction of the correlation matrix associated to the snapshots matrix.*

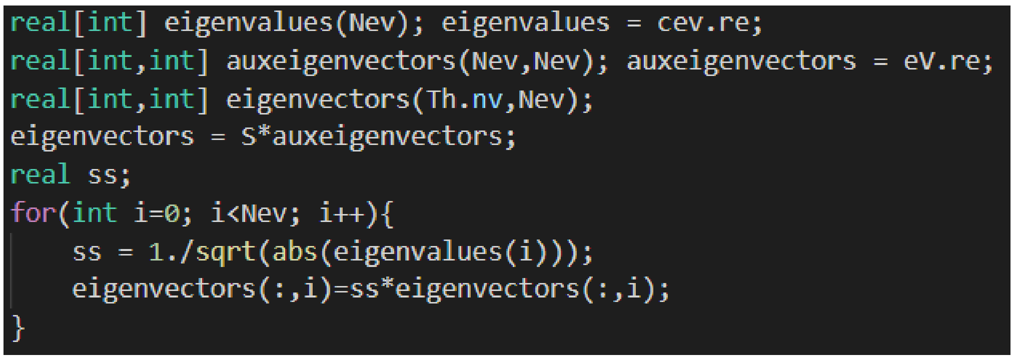


```
dggev(C, B, cev, ev, eV);
```

Figure 4.11: *Command used for solving the eigenvalue problem associated to the correlation matrix, see POD Algorithm 6.*

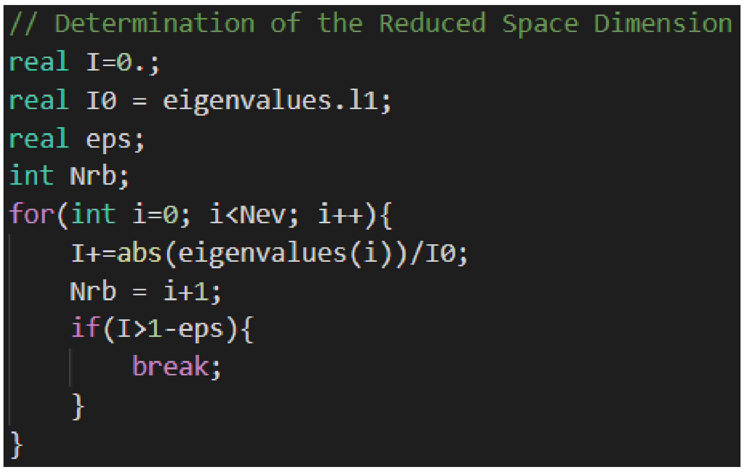
Construction of the Reduced Space elements

Thanks to the Reduced Space Basis, obtaining the reduced matrices and vectors is made by means of simple operations, as shown in Algorithm 2. The commands used for these calculations are collected in 4.15.
As it can be seen in Figure 4.15, not only the reduced matrices and vectors are obtained in this way, but also the reduced counterparts of the boundary conditions. At this point, we have already computed all the $\boldsymbol{\mu}$-dependent, computational expensive, terms. Thanks to these calculations, we can exploit the offline/online decomposition in order to save computational time. In the following section, we focus on the Online part of the problem resolution.



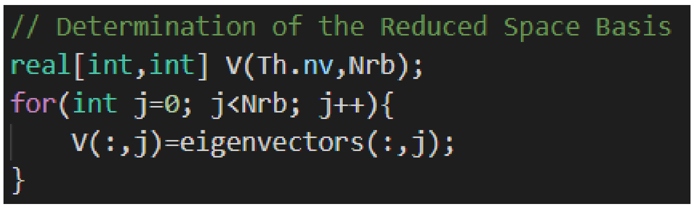
```
real[int] eigenvalues(Nev); eigenvalues = cev.re;
real[int,int] auxeigenvectors(Nev,Nev); auxeigenvectors = eV.re;
real[int,int] eigenvectors(Th.nv,Nev);
eigenvectors = S*auxeigenvectors;
real ss;
for(int i=0; i<Nev; i++){
    ss = 1./sqrt(abs(eigenvalues(i)));
    eigenvectors(:,i)=ss*eigenvectors(:,i);
}
```

Figure 4.12: *Commands used for the computation of the left eigenvectors of the SVD of the snapshots matrix.*



```
// Determination of the Reduced Space Dimension
real I=0.;
real I0 = eigenvalues.l1;
real eps;
int Nrb;
for(int i=0; i<Nev; i++){
    I+=abs(eigenvalues(i))/I0;
    Nrb = i+1;
    if(I>1-eps){
        break;
    }
}
```

Figure 4.13: *Commands used for the determination of the Reduced Space dimension, by means of the optimality criterion (3.3).*



```
// Determination of the Reduced Space Basis
real[int,int] V(Th.nv,Nrb);
for(int j=0; j<Nrb; j++){
    V(:,j)=eigenvectors(:,j);
}
```

Figure 4.14: *Commands used for the construction of the Reduced Basis.*

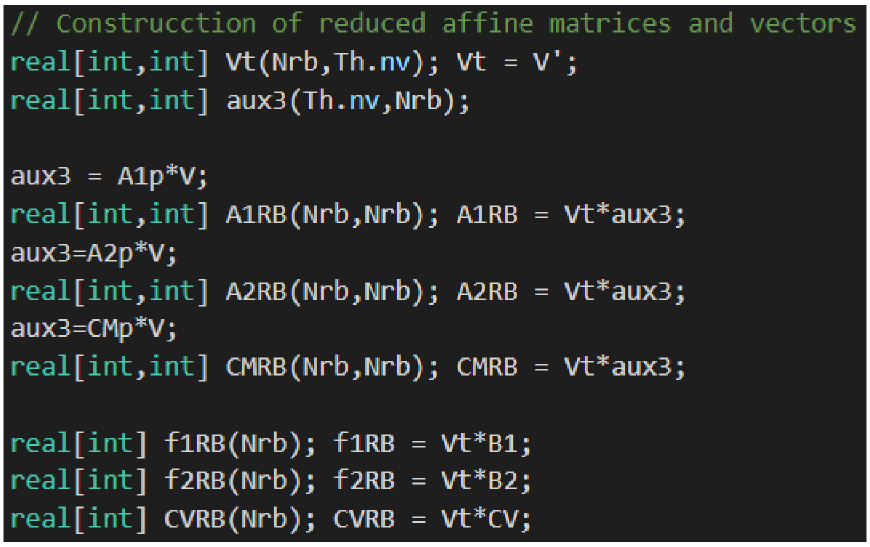
4.3.2 Online Computation

In the previous section, we have managed to assemble and store the reduced versions of the matrices and vectors appearing in the affine decomposition of the bilinear and linear forms (4.6 and 4.7) of the variational form of our problem (4.3), as well as the reduced counterparts of the boundary conditions.
In this section, we follow the procedure described in Algorithm 3, so we need to:

1. Obtain a parameter vector $\boldsymbol{\mu} \in \mathcal{P}$.

2. Assemble the reduced problem by building the reduced $\boldsymbol{\mu}$-dependent matrix and vector, $\mathbb{A}_N(\boldsymbol{\mu})$ and $\mathbf{f}_N(\boldsymbol{\mu})$, respectively.

3. Solve of the reduced linear system to obtain the Reduced Basis solution, $\mathbf{u}_N(\boldsymbol{\mu})$. In order to represent the solution obtained we also need to compute the finite element counterpart of the Reduced Basis solution.

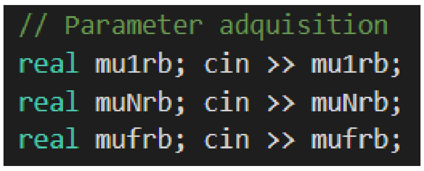


```
// Construcction of reduced affine matrices and vectors
real[int,int] Vt(Nrb,Th.nv); Vt = V';
real[int,int] aux3(Th.nv,Nrb);

aux3 = A1p*V;
real[int,int] A1RB(Nrb,Nrb); A1RB = Vt*aux3;
aux3=A2p*V;
real[int,int] A2RB(Nrb,Nrb); A2RB = Vt*aux3;
aux3=CMp*V;
real[int,int] CMRB(Nrb,Nrb); CMRB = Vt*aux3;

real[int] f1RB(Nrb); f1RB = Vt*B1;
real[int] f2RB(Nrb); f2RB = Vt*B2;
real[int] CVRB(Nrb); CVRB = Vt*CV;
```

Figure 4.15: *Commands used for the construction of the reduced matrices and vectors.*

Parameter acquisition

First of all, we need to acquire a parameter vector $\boldsymbol{\mu} \in \mathcal{P}$. To that end, we ask the user to write in the command window each of the parameters in order to built the parameter vector, commands shown in Figure 4.16

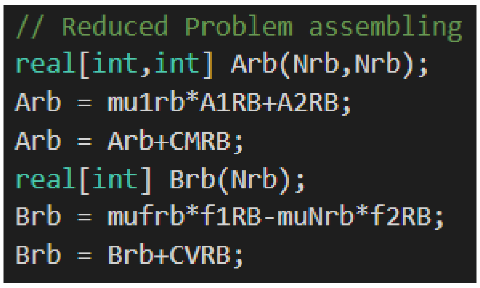


```
// Parameter adquisition
real mu1rb; cin >> mu1rb;
real muNrb; cin >> muNrb;
real mufrb; cin >> mufrb;
```

Figure 4.16: *Commands used for the acquisition of the parameter vector.*

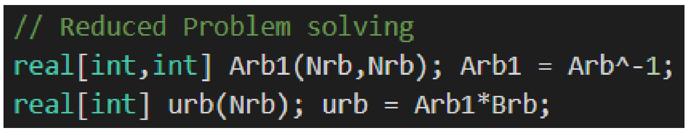
Reduced problem assembling and solving

Once we have the parameter vector for which we are resolving the reduced problem, we can assemble the linear reduced problem. In order to assemble the reduced problem, we build the $\boldsymbol{\mu}$-dependent matrix, $\mathbb{A}_N(\boldsymbol{\mu})$ and vector, $\mathbf{f}_N(\boldsymbol{\mu})$, using the affine decompositions, (4.6) and (4.7), respectively. We note that the boundary conditions are included in this step. Commands are shown in Figure 4.17.
Now solving the reduced matrix is very simple, first we compute the inverse of the matrix $\mathbb{A}_N(\boldsymbol{\mu})$ and then we compute the reduced solution $\mathbf{u}_N(\boldsymbol{\mu})$. Commands shown in Figure 4.18.



```
// Reduced Problem assembling
real[int,int] Arb(Nrb,Nrb);
Arb = mu1rb*A1RB+A2RB;
Arb = Arb+CMRB;
real[int] Brb(Nrb);
Brb = mufrb*f1RB-muNrb*f2RB;
Brb = Brb+CVRB;
```

Figure 4.17: *Commands used for the assembling of the Reduced Problem.*

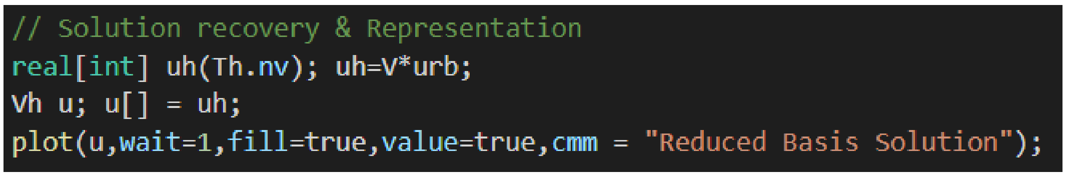


```
// Reduced Problem solving
real[int,int] Arb1(Nrb,Nrb); Arb1 = Arb^-1;
real[int] urb(Nrb); urb = Arb1*Brb;
```

Figure 4.18: *Commands used for the resolution of the Reduced Problem.*

Solution recovery and representation

Finally, we have obtained a reduced vector, $\mathbf{u}_N(\boldsymbol{\mu})$, solution of the reduced problem for the parameter vector $\boldsymbol{\mu}$. In order to represent the reduced solution inside the *FreeFEM++* software we need to compute it in the finite element space. To do so we just left-multiply by the reduced basis, see Figure 4.19.



```
// Solution recovery & Representation
real[int] uh(Th.nv); uh=V*urb;
Vh u; u[] = uh;
plot(u,wait=1,fill=true,value=true,cmm = "Reduced Basis Solution");
```

Figure 4.19: *Commands used for the recovery of the solution and its representation.*

We represent the reduced solution by its isolines over the domain Ω and we ask the software to fill the space between isolines and to give their values. Commands shown in Figure 4.19.
At this point, we have already presented the code that we are using in our calculations for this section. So in the next section we put everything shown in this chapter in action.

4.4 Results

In this section of the chapter, we apply the offline/online decomposition in order to achieve the reduction on the computational costs that we have been seeking, and we also show how it can be used to compare solutions for a wide variety of parameter vectors.

1. First, we compare the reduced solution we obtain for a determinated parameter with the solution we would have obtained if we have used the high-fidelity technique instead.
2. Then, we fix two of the parameters to study the behaviour of the solution over just one of the parameters.
3. Last, we discuss the results obtained.

For each of the cases that we study in this chapter we set the following:

- Number of divisions of each side of the boundaries of Ω and Ω_1, $nn = 25$. This previous discretization yields a Finite Element space of dimension $N_h = 1710$ and $Nt_h = 3318$ triangles.
- Number of snapshots, $n_s = 50$.
- Tolerance $\varepsilon_{POD}^2 = 10^{-12}$.

4.4.1 Comparison with High-Fidelity Solutions

First of all, we compare the reduced solution to the high-fidelity solution obtained for the same parameter vector and the setting established in the previous section, ($nn = 25$, $n_s = 50$, $\varepsilon_{POD} = 10^{-12}$). We study the following parameter set:

- Characteristic parameter of Ω_1, $\mu_1 \in [0, 2]$.
- Flux over the Neumann boundary, Γ_N, $\mu_N \in [0, 1]$.
- Intensity of the source, $\mu_f \in [1, 10]$.

Offline Phase

For the construction of the snapshots matrix we need to randomly select n_s parameter vectors, running the code shown in Figure 4.9, we obtain the parameter vectors shown in Table 4.1.

Randomly Selected Parameter Vectors			
$\boldsymbol{\mu}_1$	(1.91573, 0.972112, 1.32061)	$\boldsymbol{\mu}_2$	(0.866439, 0.129131, 5.66473)
$\boldsymbol{\mu}_3$	(0.719868 0.0870515 1.88298)	$\boldsymbol{\mu}_4$	(1.56529, 0.527958, 3.26208)
$\boldsymbol{\mu}_5$	(0.338631, 0.0993453, 9.66794)	$\boldsymbol{\mu}_6$	(1.21081, 0.389715, 4.43043)
$\boldsymbol{\mu}_7$	(1.94221, 0.110694, 9.54908)	$\boldsymbol{\mu}_8$	(1.30559, 0.750871, 5.07029)
$\boldsymbol{\mu}_9$	(1.54308, 0.80818, 8.19068)	$\boldsymbol{\mu}_{10}$	(1.71812, 0.632508, 5.60191)
$\boldsymbol{\mu}_{11}$	(0.182963, 0.0871058, 7.4865)	$\boldsymbol{\mu}_{12}$	(0.924562, 0.740505, 5.8847)
$\boldsymbol{\mu}_{13}$	(0.872127, 0.746468, 3.29771)	$\boldsymbol{\mu}_{14}$	(0.502091, 0.247963, 2.85143)
$\boldsymbol{\mu}_{15}$	(1.91325, 0.251652, 4.78105)	$\boldsymbol{\mu}_{16}$	(1.97794, 0.876998, 5.56969)
$\boldsymbol{\mu}_{17}$	(0.211672, 0.727466, 4.95877)	$\boldsymbol{\mu}_{18}$	(0.610784, 0.79015, 5.91673)
$\boldsymbol{\mu}_{19}$	(1.87707, 0.90623, 5.84642)	$\boldsymbol{\mu}_{20}$	(0.518796, 0.704476, 8.06587)
$\boldsymbol{\mu}_{21}$	(0.0804587, 0.293897, 9.85975)	$\boldsymbol{\mu}_{22}$	(1.09899, 0.385632, 3.34527)
$\boldsymbol{\mu}_{23}$	(1.46024, 0.732543, 8.6985)	$\boldsymbol{\mu}_{24}$	(0.774337, 0.138993, 7.8516)
$\boldsymbol{\mu}_{25}$	(0.638169, 0.824043, 8.46365)	$\boldsymbol{\mu}_{26}$	(0.437955, 0.0184205, 8.67635)
$\boldsymbol{\mu}_{27}$	(1.8978, 0.0385099, 6.59594)	$\boldsymbol{\mu}_{28}$	(0.491942, 0.866685, 9.80998)
$\boldsymbol{\mu}_{29}$	(1.73521, 0.0581521, 7.13052)	$\boldsymbol{\mu}_{30}$	(0.252263, 0.755113, 8.30539)
$\boldsymbol{\mu}_{31}$	(0.255392, 0.852327, 9.47965)	$\boldsymbol{\mu}_{32}$	(0.922484, 0.78353, 4.40366)
$\boldsymbol{\mu}_{33}$	(1.07813, 0.84071, 6.21647)	$\boldsymbol{\mu}_{34}$	(1.03177, 0.948269, 2.29277)
$\boldsymbol{\mu}_{35}$	(0.417683, 0.442611, 3.1341)	$\boldsymbol{\mu}_{36}$	(1.84362, 0.783143, 2.19388)
$\boldsymbol{\mu}_{37}$	(0.0709919, 0.912566, 7.07242)	$\boldsymbol{\mu}_{38}$	(0.421041, 0.0974507, 8.37658)
$\boldsymbol{\mu}_{39}$	(1.64655, 0.0864343, 5.3516)	$\boldsymbol{\mu}_{40}$	(1.58181, 0.582419, 9.20588)
$\boldsymbol{\mu}_{41}$	(1.61068, 0.620541, 5.2395)	$\boldsymbol{\mu}_{42}$	(1.50519, 0.864642, 8.78801)
$\boldsymbol{\mu}_{43}$	(0.804308, 0.0414592, 6.50109)	$\boldsymbol{\mu}_{44}$	(0.166018, 0.593897, 6.72676)
$\boldsymbol{\mu}_{45}$	(0.166018, 0.593897, 6.72676)	$\boldsymbol{\mu}_{46}$	(1.84689, 0.551924, 2.10753)
$\boldsymbol{\mu}_{47}$	(0.175355, 0.0745348, 8.91859)	$\boldsymbol{\mu}_{48}$	(1.98634, 0.772065, 4.59534)
$\boldsymbol{\mu}_{49}$	(0.764941, 0.438768, 7.76508)	$\boldsymbol{\mu}_{50}$	(1.57032, 0.103838, 3.23487)

Table 4.1: Random parameter vectors used for the construction of the snapshot matrix.

After assembling the Correlation matrix (code shown in Figure 4.10) and obtaining its eigenvalues and eigenvectors (code shown in Figure 4.11), we determine the Reduced Space dimension (by means of the code shown in Figure 4.13). In this case the Reduced Space dimension is $N = 8$. The eigenvectors that build the Reduced Basis are, hence, the ones shown in Figures 4.20 and 4.21.

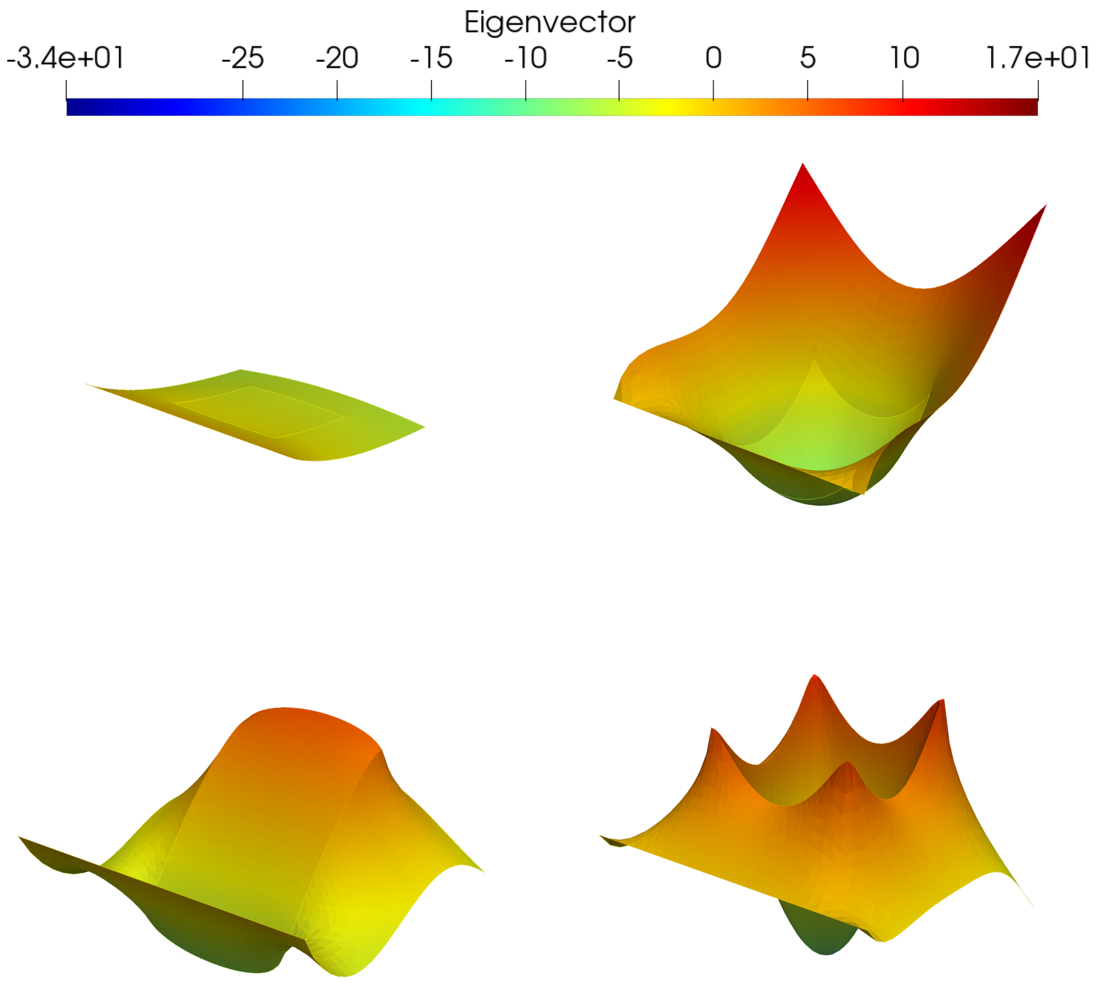


Figure 4.20: *From top to bottom and from left to right, representations of eigenvectors 1, 2, 3, 4 of the Correlation matrix.*

We note that whereas the first eigenvectors are smooth and have representative values over all the domain (see Figure 4.20), the last ones only have representative values in the corners of the inner subdomain (see Figure 4.21). This give us the idea that the corners of the boundary of the inner subdomain is going to be the part of the domain that our program will approximate worse.
We also note that for the high-fidelity scheme we have to solve a linear problem of dimension $N_h = 1710$ which complexity is $O(N_h^3)$, i.e. $\sim 5 \cdot 10^9$ operations, while

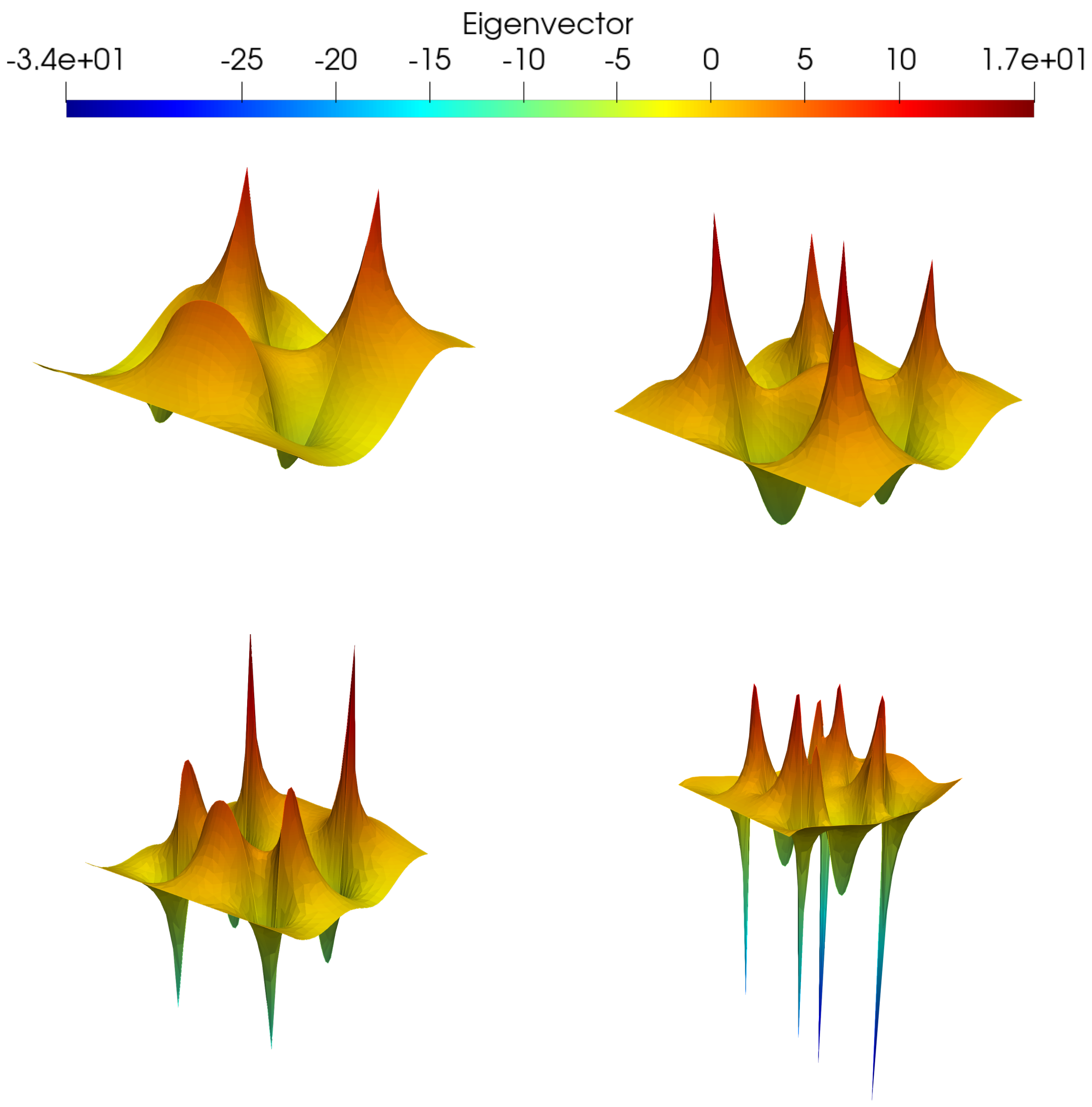


Figure 4.21: *From top to bottom and from left to right, representations of eigenvectors 5, 6, 7, 8 of the Correlation matrix.*

for the reduced scheme we have to solve a linear problem of dimension $N = 8$ which complexity is $O(N^3)$, i.e. $\sim 5 \cdot 10^2$ operations. Although, the high-fidelity problem is a sparse one, its number of operations is still much higher than the reduced problem number of operations. We will discuss this topic at the end of the online phase when we compare the computation times for both schemes.

Online Phase

Now that we have constructed the reduced matrices and vectors we can compare the solutions obtained by both, the Reduced Basis method and the High-Fidelity method.
As we have stated before, we expect that our program will approximate the worse the boundary of the inner subdomain, so we compare the solutions for:

- First, a parameter vector $\boldsymbol{\mu} = (1, 0.5, 2)$, so there is a non-smooth transition between the characteristic parameters of the two subdomains.
- Second, a parameter vector $\boldsymbol{\mu} = (0, 0, 5)$, so the characteristic parameters of the two subdomains are equal, but the source is focused in the inner subdomain.

In the first case, we settle:

- $\mu_1 = 1$, so there is a non-smooth transition between the characteristic parameters of the two subdomains.
- $\mu_N = 0.5$, a midway value for the parameter of the Neumann boundary.
- $\mu_f = 2$, so the source has the same numerical value that the characteristic value of the inner subdomain.

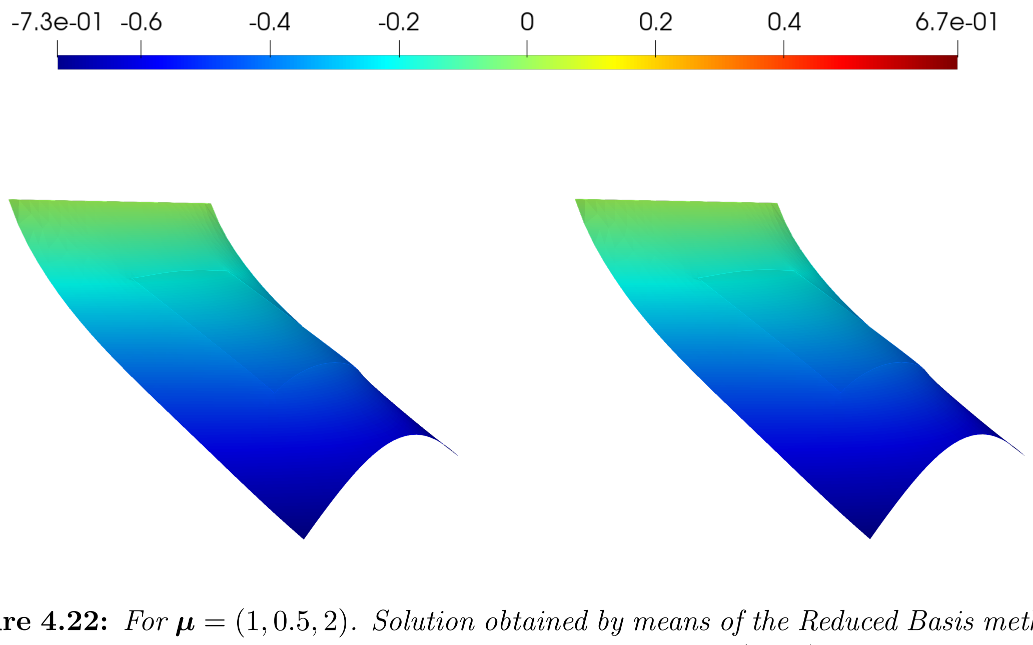


Figure 4.22: *For $\boldsymbol{\mu} = (1, 0.5, 2)$. Solution obtained by means of the Reduced Basis method (left). Solution obtained by means of the High-Fidelity method (right).*

Solving for this parameter vector we obtain:

- For the Reduced Basis method the solution represented in Figure 4.22, left.
- For the High-Fidelity method the solution represented in Figure 4.22, right.

We also represent the difference between both solutions in Figure 4.23. In this representation we can clearly see that the major differences are placed on the corners of the inner domain, as well as the middle points between two adjacent corner points.

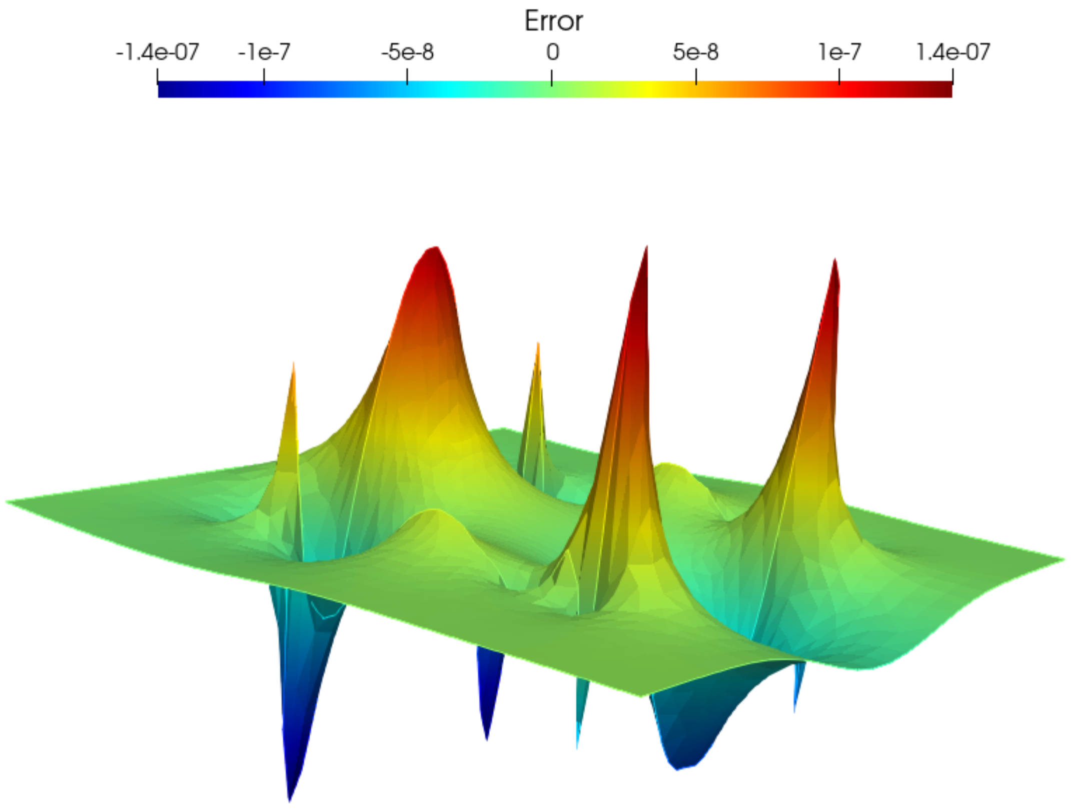


Figure 4.23: *For* $\boldsymbol{\mu} = (1, 0.5, 2)$*. Difference between the Reduced Basis and High-Fidelity methods.*

In the second case, we settle:

- $\mu_1 = 0$, so the characteristic parameters of the two subdomains are equal.
- $\mu_N = 0$, so the Neumann boundary does not affect the solutions.
- $\mu_f = 5$, so the source plays a predominant role in the solution.

Solving for this parameter vector we obtain:

- For the Reduced Basis method the solution represented in Figure 4.24, left.
- For the High-Fidelity method the solution represented in Figure 4.24, right.

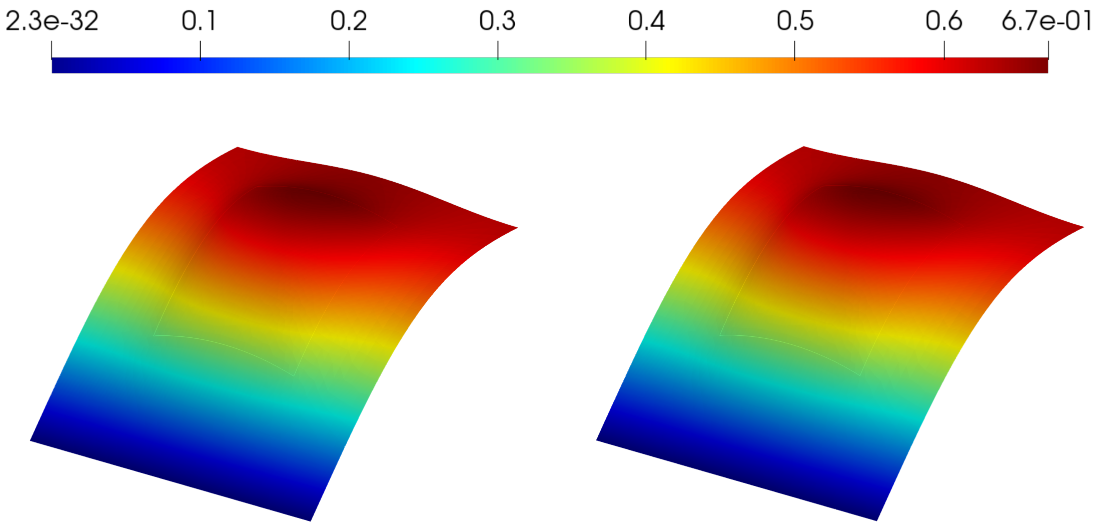


Figure 4.24: *For* $\boldsymbol{\mu} = (0, 0, 5)$. *Solution obtained by means of the Reduced Basis method (left). Solution obtained by means of the High-Fidelity method (right).*

We also represent the difference between both solutions in Figure 4.25. Again, in the representation we can clearly see that the major differences are placed on the corners of the inner domain, as well as the middle points between two adjacent corner points, but the difference are spread over all the boundary not focused only on the corners.

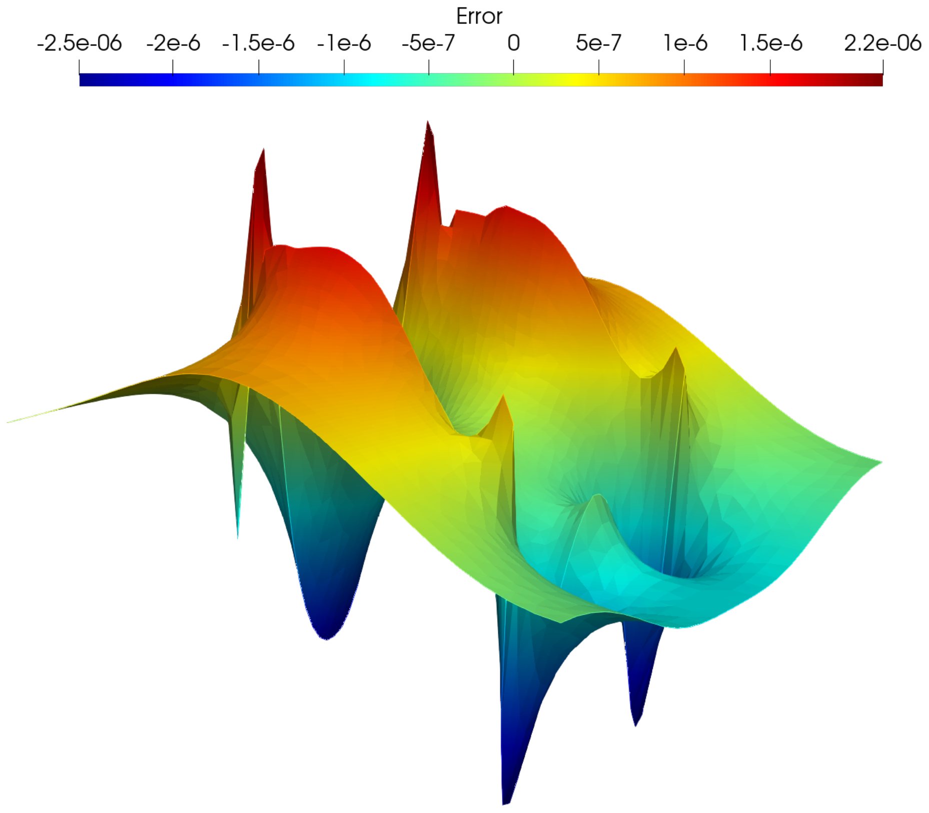


Figure 4.25: *For $\boldsymbol{\mu} = (0, 0, 5)$. Difference between the Reduced Basis and High-Fidelity methods.*

Comparison

In order to compare the Reduced Basis scheme with the High-Fidelity scheme, we calculate the computation times for both of them and then we obtain the quotient. We reproduce the problem for each parameter vector 10^4 times and we get:

- For $\boldsymbol{\mu} = (1, 0.5, 2)$, we obtain $t_{POD} = (0.0401 \pm 0.201) \cdot 10^{-3}$ s and $t_{HF} = (7.50 \pm 0.59) \cdot 10^{-3}$ s, so the quotient is:

$$\rho = \frac{t_{HF}}{t_{POD}} > 28.67.$$

- For $\boldsymbol{\mu} = (0, 0, 5)$, we obtain $t_{POD} = (0.0398 \pm 0.201) \cdot 10^{-3}$ s and $t_{HF} = (7.49 \pm 0.59) \cdot 10^{-3}$ s, so the quotient is:

$$\rho = \frac{t_{HF}}{t_{POD}} > 28.65.$$

Roughly speaking, both quotients are bounded from below by $\rho^* = 25$, so we can conclude that the high-fidelity scheme is twenty five times slower than the Reduced Basis scheme. In other words, in the time we spend assembling and solving one high-fidelity problem we could have assembled and solved 25 reduced basis problems. This online time saving property is what makes the reduced basis technique a powerful tool when solving real time or many-query problems.
We have compared the solutions obtained by means of both schemes, high-fidelity and reduced basis when all the parameters are free. In the following of this section, we fix two of the three parameters and study the dependence of the solution on the free parameter.

4.4.2 Homogeneous Domain and Boundary Condition

We start fixing the characteristic parameter of the inner subdomain, Ω_1 and the Neumann boundary condition parameter. We set both parameters to be zero, so the parameter set we are studying is:

- Intensity of the source, $\mu_f \in [-1, 1]$.

We use the same setting established in the previous section, ($nn = 25$, $n_s = 50$, $\varepsilon^2_{POD} = 10^{-12}$).

Offline Phase

For the construction of the snapshots matrix we need to randomly select n_s source parameters. Running the code shown in Figure 4.9 we obtain the source parameters shown in Table 4.2.
After assembling the Correlation matrix (code shown in Figure 4.10) and obtaining its eigenvalues and eigenvectors (code shown in Figure 4.11), we determine the Reduced Space dimension (see Figure 4.13). In this case, the Reduced Space dimension is $N = 1$. The eigenvector that build the Reduced Basis is, hence, the one shown in Figure 4.26.

Online Phase

In this special case, where $N = 1$, the reduced matrices and vectors degenerate to be a scalar. So we expect the solutions to be qualitatively equivalent. In order to compare some solutions, we settle μ_f to be -1, -0.5, 0, 0.5 and 1. The results are shown in Figure 4.27.

Randomly Selected Source Parameters							
$(\mu_f)_1$	-0.92874	$(\mu_f)_2$	0.0366076	$(\mu_f)_3$	-0.803781	$(\mu_f)_4$	-0.497314
$(\mu_f)_5$	0.926209	$(\mu_f)_6$	-0.237682	$(\mu_f)_7$	0.899795	$(\mu_f)_8$	-0.0954916
$(\mu_f)_9$	0.597929	$(\mu_f)_{10}$	0.0226461	$(\mu_f)_{11}$	0.441445	$(\mu_f)_{12}$	0.08549
$(\mu_f)_{13}$	-0.489397	$(\mu_f)_{14}$	-0.588571	$(\mu_f)_{15}$	-0.159767	$(\mu_f)_{16}$	0.0154877
$(\mu_f)_{17}$	-0.120273	$(\mu_f)_{18}$	0.0926069	$(\mu_f)_{19}$	0.0769812	$(\mu_f)_{20}$	0.570194
$(\mu_f)_{21}$	0.96883	$(\mu_f)_{22}$	-0.478829	$(\mu_f)_{23}$	0.710778	$(\mu_f)_{24}$	0.522578
$(\mu_f)_{25}$	0.658588	$(\mu_f)_{26}$	0.705856	$(\mu_f)_{27}$	0.243543	$(\mu_f)_{28}$	0.957774
$(\mu_f)_{29}$	0.362339	$(\mu_f)_{30}$	0.623421	$(\mu_f)_{31}$	0.884366	$(\mu_f)_{32}$	-0.243631
$(\mu_f)_{33}$	0.159215	$(\mu_f)_{34}$	-0.712718	$(\mu_f)_{35}$	-0.525756	$(\mu_f)_{36}$	-0.734694
$(\mu_f)_{37}$	0.349427	$(\mu_f)_{38}$	0.639241	$(\mu_f)_{39}$	-0.0329784	$(\mu_f)_{40}$	0.82353
$(\mu_f)_{41}$	-0.0578883	$(\mu_f)_{42}$	0.73067	$(\mu_f)_{43}$	0.222464	$(\mu_f)_{44}$	0.272613
$(\mu_f)_{45}$	-0.801326	$(\mu_f)_{44}$	-0.753882	$(\mu_f)_{47}$	0.759687	$(\mu_f)_{48}$	-0.201036
$(\mu_f)_{49}$	0.503352	$(\mu_f)_{50}$	-0.503362				

Table 4.2: Random source parameters used for the construction of the snapshot matrix.

In the representation, it can be clearly seen that all the solutions behave in a similar way but they have a scale factor that depends on the source parameter.

- For $\mu_f > 0$, we obtain non-negative solutions.
- For $\mu_f < 0$, we obtain non-positive solutions.
- For $\mu_f = 0$, we obtain the null solution, as expected.

4.4.3 Homogeneous Boundary and Fixed Source

Now, we fix the Neumann boundary condition parameter to be zero and the source parameter to be the unity. So the parameter set now is:

- Characteristic parameter of Ω_1, $\mu_1 \in [-0.5, 0.5]$.

We use the same setting established in the previous section, ($nn = 25$, $n_s = 50$, $\varepsilon^2_{POD} = 10^{-12}$).

Offline Phase

For the construction of the snapshots matrix we need to randomly select n_s characteristic parameters. Running the code shown in Figure 4.9 we obtain the source parameters shown in Table 4.3.

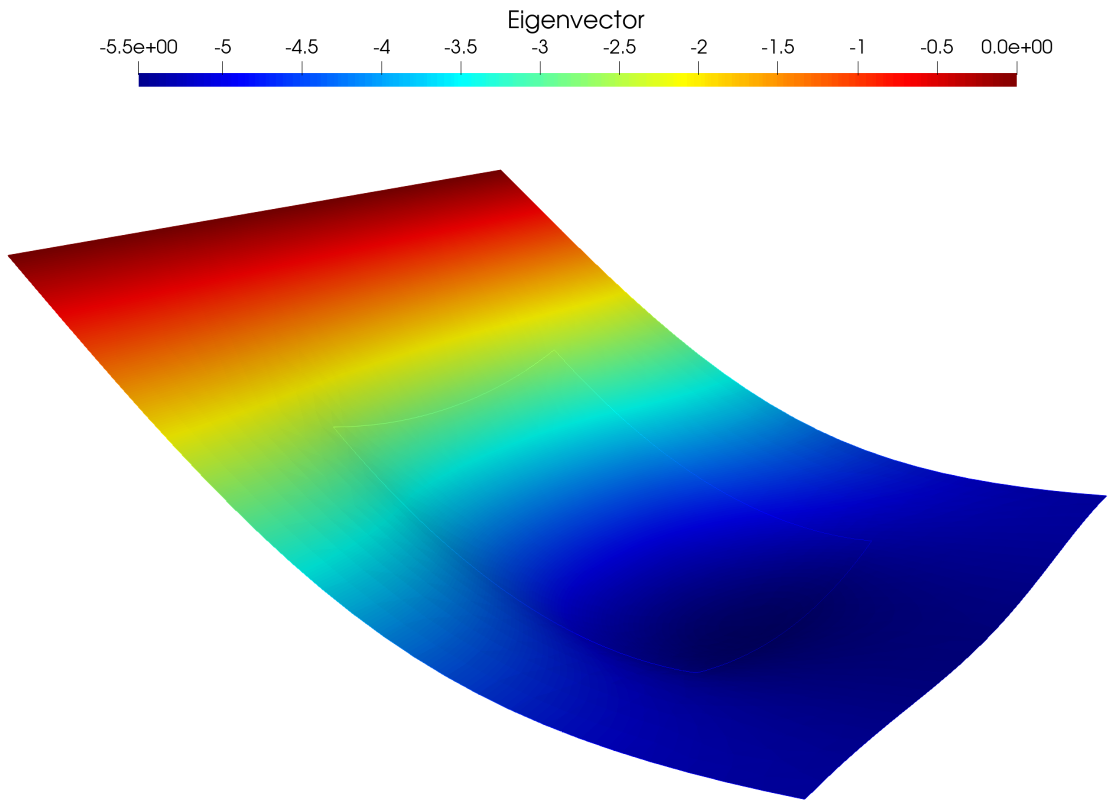


Figure 4.26: *Representation of the first eigenvector of the Correlation Matrix.*

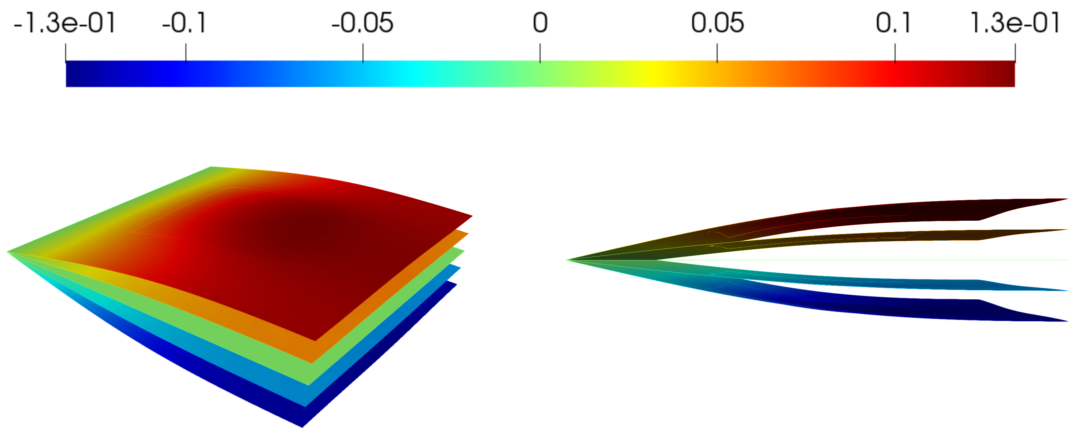


Figure 4.27: *Comparison of the solutions obtained (from up to bottom) for* $\mu_f = 1, 0.5, 0, -0.5, -1$. *Oblique projection, left. Lateral view, right.*

Randomly Selected Characteristic Parameters							
$(\mu_1)_1$	0.457867	$(\mu_1)_2$	-0.0667806	$(\mu_1)_3$	-0.140066	$(\mu_1)_4$	0.282645
$(\mu_1)_5$	-0.330684	$(\mu_1)_6$	0.105404	$(\mu_1)_7$	0.471104	$(\mu_1)_8$	0.152793
$(\mu_1)_9$	0.271542	$(\mu_1)_{10}$	0.359059	$(\mu_1)_{11}$	-0.408519	$(\mu_1)_{12}$	-0.0377191
$(\mu_1)_{13}$	-0.0639365	$(\mu_1)_{14}$	-0.248955	$(\mu_1)_{15}$	0.456627	$(\mu_1)_{16}$	0.48897
$(\mu_1)_{17}$	-0.394164	$(\mu_1)_{18}$	-0.194608	$(\mu_1)_{19}$	0.438535	$(\mu_1)_{20}$	-0.240602
$(\mu_1)_{21}$	-0.459771	$(\mu_1)_{22}$	0.0494963	$(\mu_1)_{23}$	0.230122	$(\mu_1)_{24}$	-0.112831
$(\mu_1)_{25}$	-0.180916	$(\mu_1)_{26}$	-0.281022	$(\mu_1)_{27}$	0.448902	$(\mu_1)_{28}$	-0.254029
$(\mu_1)_{29}$	0.367604	$(\mu_1)_{30}$	-0.373868	$(\mu_1)_{31}$	-0.372304	$(\mu_1)_{32}$	-0.038758
$(\mu_1)_{33}$	0.0390668	$(\mu_1)_{34}$	0.0158867	$(\mu_1)_{35}$	-0.291159	$(\mu_1)_{36}$	0.421812
$(\mu_1)_{37}$	-0.464504	$(\mu_1)_{38}$	-0.289479	$(\mu_1)_{39}$	0.323277	$(\mu_1)_{40}$	0.290907
$(\mu_1)_{41}$	0.30534	$(\mu_1)_{42}$	0.252597	$(\mu_1)_{43}$	-0.0978458	$(\mu_1)_{44}$	-0.416991
$(\mu_1)_{45}$	0.266539	$(\mu_1)_{44}$	0.423444	$(\mu_1)_{47}$	-0.412323	$(\mu_1)_{48}$	0.493172
$(\mu_1)_{49}$	-0.117529	$(\mu_1)_{50}$	0.285161				

Table 4.3: Random characteristic parameters used for the construction of the snapshot matrix.

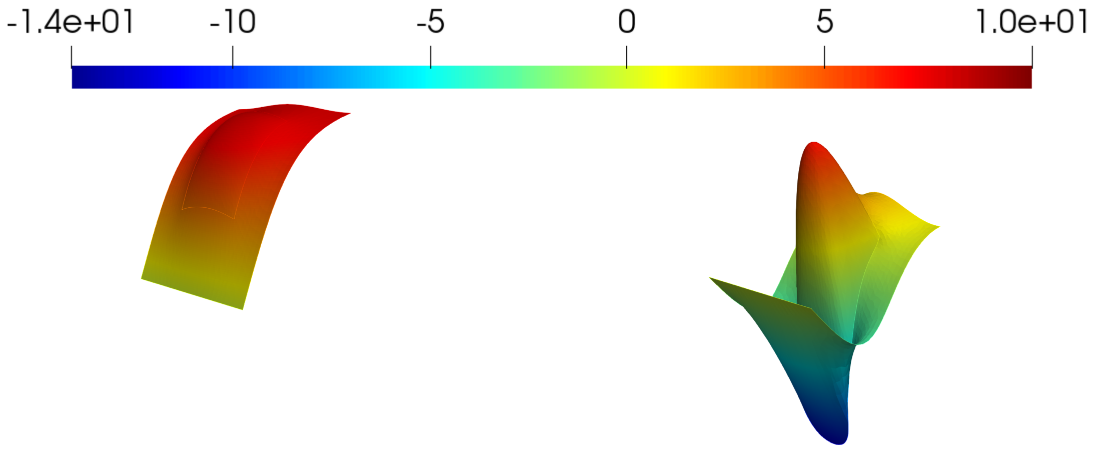


Figure 4.28: *From left to right, representations of eigenvectors 1, 2 of the Correlation matrix.*

After assembling the Correlation matrix (see Figure 4.10) and obtaining its eigenvalues and eigenvectors (code shown in Figure 4.11), we determine the Reduced Space dimension (see Figure 4.13). In this case the Reduced Space dimension is $N = 4$. The eigenvectors that build the Reduced Basis are, hence, the ones shown in Figures 4.28 and 4.29 .

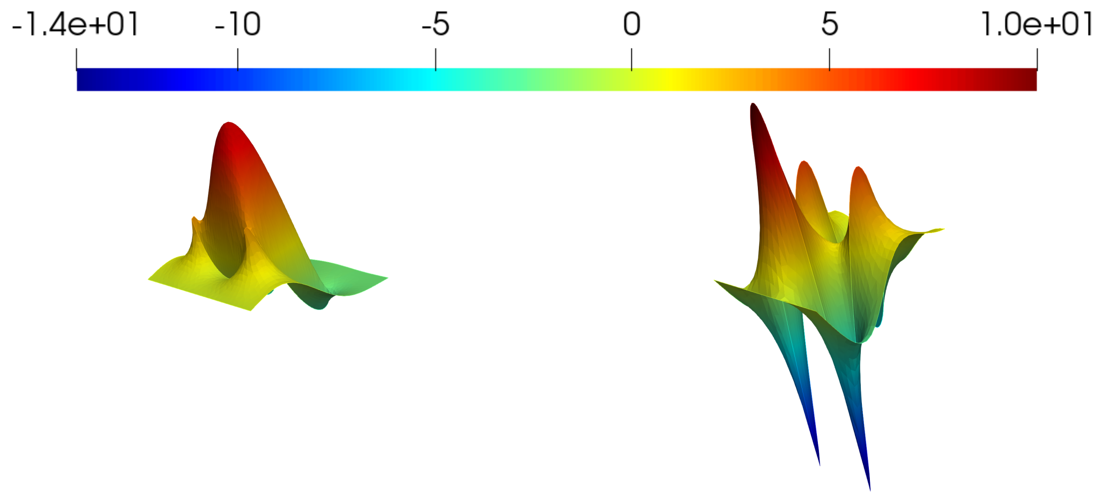


Figure 4.29: *From left to right, representations of eigenvectors 3, 4 of the Correlation matrix.*

Online Phase

Now that we have constructed the reduced matrices and vectors, we can compute the reduced solutions for a subset of the parameter space. We choose the following characteristic parameters of Ω_1, $\mu_1 \in \{-0.5, -0.2, 0, 0.2, 0.5\}$ and we obtain the following solutions, see Figure 4.30 and 4.31.

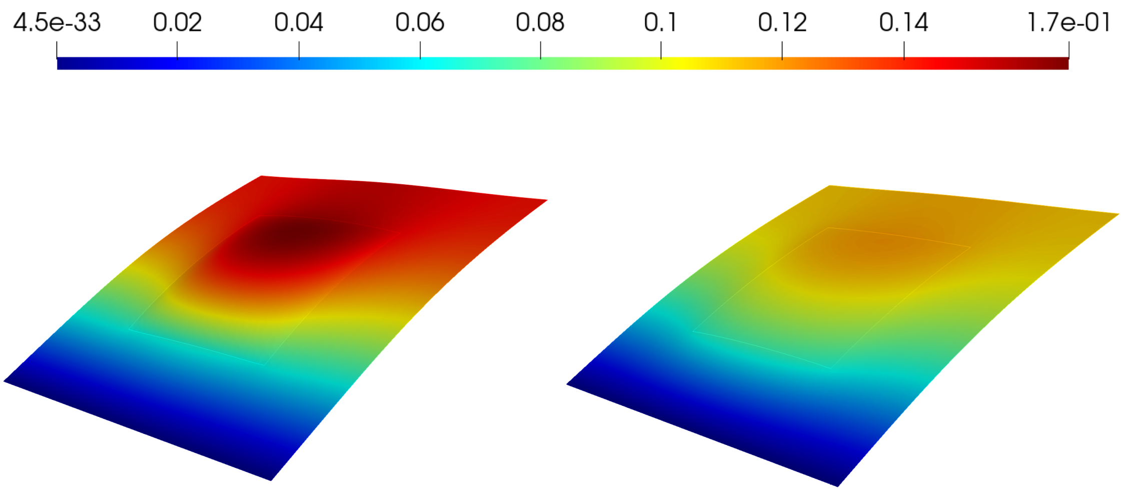


Figure 4.30: *Reduced Basis Solutions for the characteristic parameters of* Ω_1 *selected as* $\mu_1 = -0.5$ *(left) and* $\mu_1 = 0.5$ *(right).*

Although we need more eigenvectors to capture the information about the behaviour of the system, there is no big significant difference between both solutions.

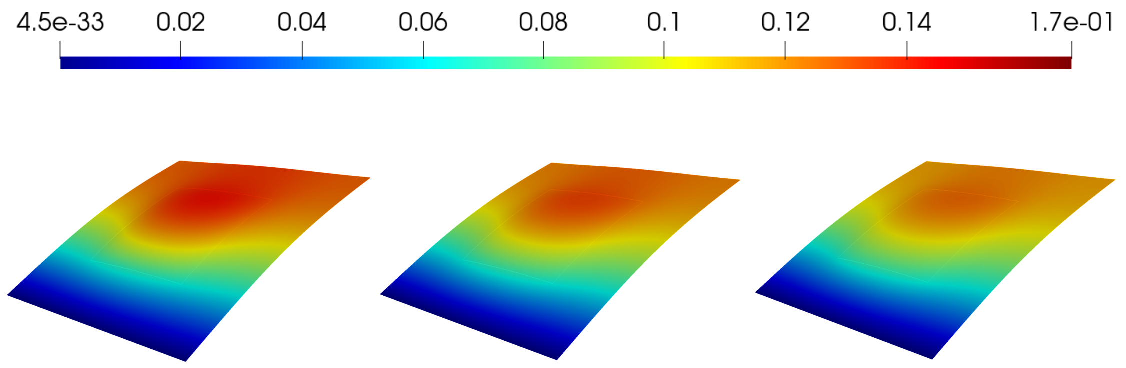


Figure 4.31: *Reduced Basis Solutions for the characteristic parameters of* Ω_1 *selected as* $\mu_1 = -0.2$ *(left),* $\mu_1 = 0$ *(center) and* $\mu_1 = 0.2$ *(right).*

4.4.4 Fixed Source and Homogeneous Domain

At last, we fix the characteristic parameter of the inner subdomain to be zero and the source parameter to be the unity. So the parameter set now is:

- Flux over the Neumann boundary, $\mu_N \in [-1, 1]$.

We use the same setting established in the previous section, ($nn = 25$, $n_s = 50$, $\varepsilon^2_{POD} = 10^{-12}$).

Offline Phase

For the construction of the snapshots matrix we need to randomly select n_s characteristic parameters, running the code shown in Figure 4.9 we obtain the source parameters shown in Table 4.4.
After assembling the Correlation matrix (see Figure 4.10) and obtaining its eigenvalues and eigenvectors (code shown in Figure 4.11), we determine the Reduced Space dimension (see Figure 4.13). In this case the Reduced Space dimension is $N = 2$. The eigenvectors that build the Reduced Basis are hence the ones shown in Figures 4.32.

Randomly Selected Characteristic Parameters							
$(\mu_1)_1$	0.944225	$(\mu_1)_2$	-0.741738	$(\mu_1)_3$	-0.825897	$(\mu_1)_4$	0.0559165
$(\mu_1)_5$	-0.801309	$(\mu_1)_6$	-0.220571	$(\mu_1)_7$	-0.778613	$(\mu_1)_8$	0.501741
$(\mu_1)_9$	0.61636	$(\mu_1)_{10}$	0.265016	$(\mu_1)_{11}$	-0.825788	$(\mu_1)_{12}$	0.48101
$(\mu_1)_{13}$	0.492937	$(\mu_1)_{14}$	-0.504074	$(\mu_1)_{15}$	-0.496697	$(\mu_1)_{16}$	0.753997
$(\mu_1)_{17}$	0.454932	$(\mu_1)_{18}$	0.580301	$(\mu_1)_{19}$	0.81246	$(\mu_1)_{20}$	0.408953
$(\mu_1)_{21}$	-0.412205	$(\mu_1)_{22}$	-0.228736	$(\mu_1)_{23}$	0.465086	$(\mu_1)_{24}$	-0.722015
$(\mu_1)_{25}$	-0.883696	$(\mu_1)_{26}$	0.510226	$(\mu_1)_{27}$	0.704653	$(\mu_1)_{28}$	0.56706
$(\mu_1)_{29}$	0.681421	$(\mu_1)_{30}$	0.896539	$(\mu_1)_{31}$	-0.114778	$(\mu_1)_{32}$	0.566286
$(\mu_1)_{33}$	0.825131	$(\mu_1)_{34}$	-0.805099	$(\mu_1)_{35}$	-0.827131	$(\mu_1)_{36}$	0.164838
$(\mu_1)_{37}$	0.241083	$(\mu_1)_{38}$	0.729283	$(\mu_1)_{39}$	-0.917082	$(\mu_1)_{40}$	0.187794
$(\mu_1)_{41}$	0.233022	$(\mu_1)_{42}$	0.103847	$(\mu_1)_{43}$	-0.85093	$(\mu_1)_{44}$	0.54413
$(\mu_1)_{45}$	0.648086	$(\mu_1)_{44}$	-0.963159	$(\mu_1)_{47}$	-0.92298	$(\mu_1)_{48}$	0.73337
$(\mu_1)_{49}$	-0.122464	$(\mu_1)_{50}$	-0.792323				

Table 4.4: Random boundary parameters used for the construction of the snapshot matrix.

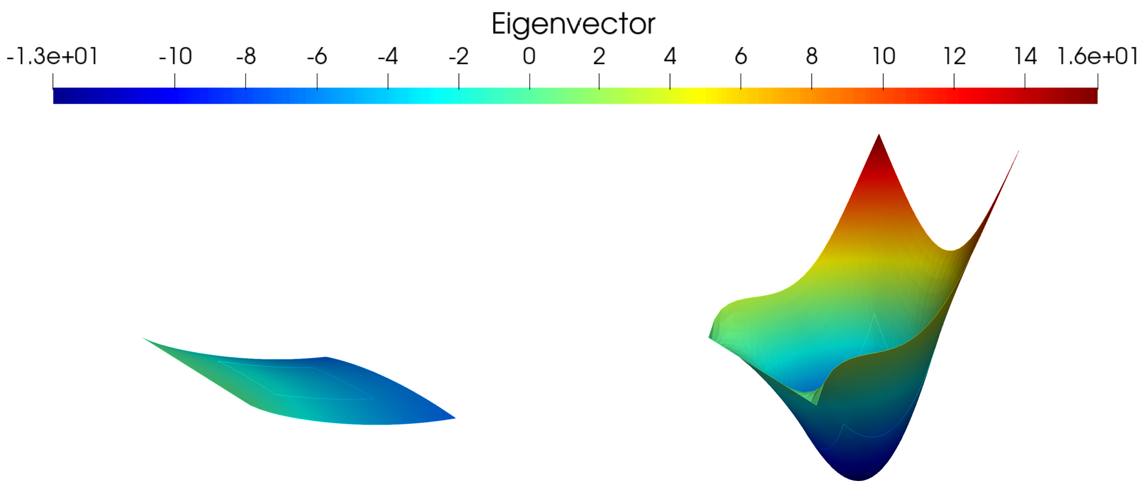


Figure 4.32: *Representations of eigenvectors 1 (left), 2 (right) of the Correlation matrix.*

Online Phase

Now that we have constructed the reduced matrices and vectors, we can compute the reduced solutions for a subset of the parameter space. We choose the following parameters over the Neumann boundary $\mu_N \in \{-0.5, -0.2, 0, 0.2, 0.5\}$ and we obtain the following solutions, see Figure 4.33.

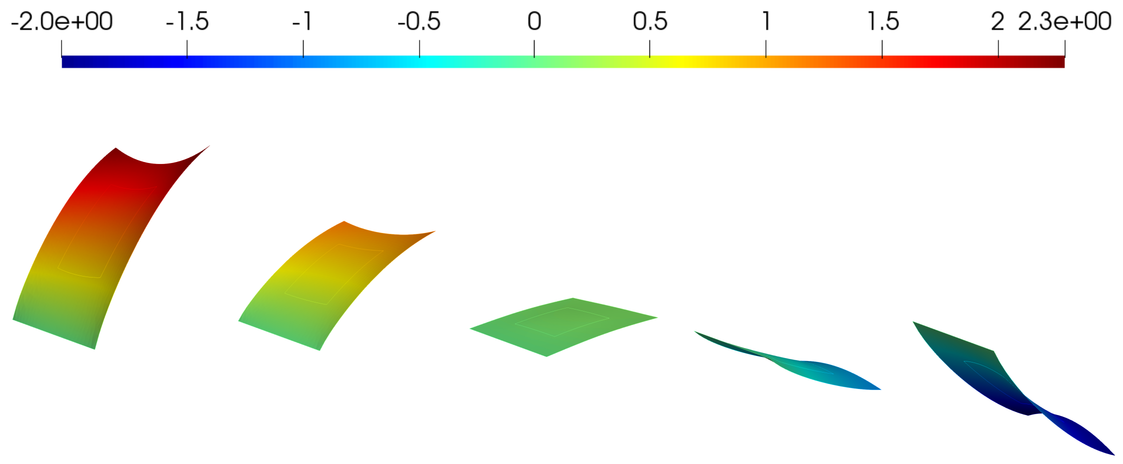


Figure 4.33: *Reduced Basis Solutions for different Neumann boundary parameter. From left to right,* $\mu_N = \{-1, -0.5, 0, 0.5, 1\}$.

As can be seen the curvature of the solutions is reversed as the parameter varies from negative to positive values.

4.5 Conclusion

We end this chapter outlying the main results of the numerical simulations that we have performed.

Comparison with High-Fidelity Solutions

First of all, we have compared both solutions, the one obtained via a high-fidelity method and the other one obtained via a reduced basis method. As can be seen in Figures 4.23 and 4.25 the main differences are located in the corners of boundary of the inner subdomain. This can be due to the distribution of the selected mesh in that boundary.
We have calculated the quotient between the computation times obtained via both methods. We have found that the calculated quotients are bounded from below by $\rho^* = 25$, so we can conclude that the high-fidelity method is twenty times slower that the Reduced Basis scheme.
This time saving property is the main advantage of the reduced order techniques and makes these methods an essential tool when solving real time or many-query problems. In problems where the resolution of the high-fidelity problem is more intricate the quotient is expected to rise so this property becomes more important.

Homogeneous Domain and Boundary Condition

In this case, we have obtained that the reduced basis dimension is $N = 1$.
This means that the solution dependence on the source parameter is simple, different parameters give the same solution multiplied by a factor that depends on the source parameter, as shown in Figure 4.27.

Homogeneous Boundary and Fixed Source

In this case, we have obtained that the reduced basis dimension is $N = 4$, the higher of all the cases. However, our simulations suggest that the solutions does not really change much for different characteristic parameter of the inner subdomain, as shown in Figures 4.30 and 4.31.
The reduced model need so many eigenvectors to approximate the solution because the differences between the solutions are located in the corners of the inner subdomain, where our method performs the worst. We can see the latter in the representation of the forth eigenvector in Figure 4.29. The main values are located in the boundary of the inner subdomain and exhibits high-amplitude oscillations.

Fixed Source and Homogeneous Domain

In this case, we have obtained that the reduced basis dimension is $N = 2$. Moreover, the solutions calculated are the ones that show more differences between them, with highlights in the reversion of curvature as shown in Figure 4.33.

Chapter 5

Resolution of Laplace Equation by Greedy Method

In this chapter, we continue with the, academical, 2-dimensional Laplace problem occurring in surfaces of different materials characterized by a particular parameter. To solve it, we apply the Greedy technique. A similar problem is studied in [12].

5.1 Computation

The only difference between the POD technique and the Greedy technique is how the Reduced Basis is constructed. While in the previous chapter we have constructed the snapshots matrix and then computed its singular values in order to select those with more weight until a tolerance criterion is satisfied, in the Greedy case we need:

1. A parameter sample $\Xi_{train} \subset \mathcal{P}$.

2. An estimator of the approximation error.

5.1.1 Offline Phase

This phase is mainly the same as the presented in Section 4.3.1. The changes arise after the construction of the affine decomposition.

Remark 5.1

The construction of the Reduced Space elements is identical for both methods.

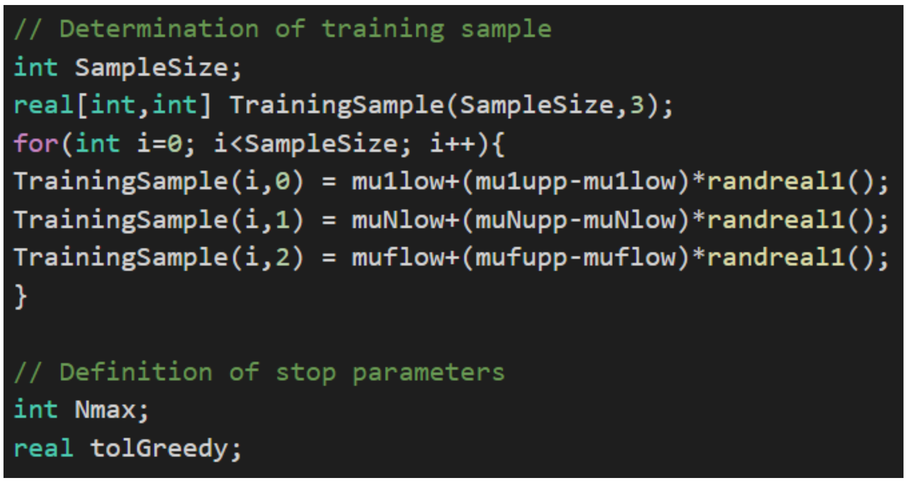
In the case of the application of the Greedy procedure after the construction of the affine decomposition, we continue with:

Initialization

First, we need to define some aspects of the Greedy method, such as,

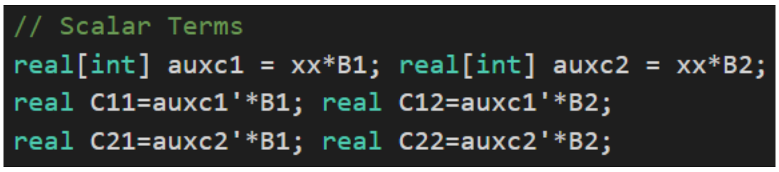
1. Train sample, Ξ_{train}, which is selected randomly from the parameter set.
2. Stopping criteria, this is, a maximum number of iterations, and a tolerance.

The commands are shown in Figure 5.1.



```
// Determination of training sample
int SampleSize;
real[int,int] TrainingSample(SampleSize,3);
for(int i=0; i<SampleSize; i++){
TrainingSample(i,0) = mu1low+(mu1upp-mu1low)*randreal1();
TrainingSample(i,1) = muNlow+(muNupp-muNlow)*randreal1();
TrainingSample(i,2) = muflow+(mufupp-muflow)*randreal1();
}

// Definition of stop parameters
int Nmax;
real tolGreedy;
```

Figure 5.1: *Commands used for the definitions of the train sample and the stopping criteria.*



```
// Scalar Terms
real[int] auxc1 = xx*B1; real[int] auxc2 = xx*B2;
real C11=auxc1'*B1; real C12=auxc1'*B2;
real C21=auxc2'*B1; real C22=auxc2'*B2;
```

Figure 5.2: *Commands used for the computing of the $\boldsymbol{\mu}$-independent scalar terms of the residual estimator.*

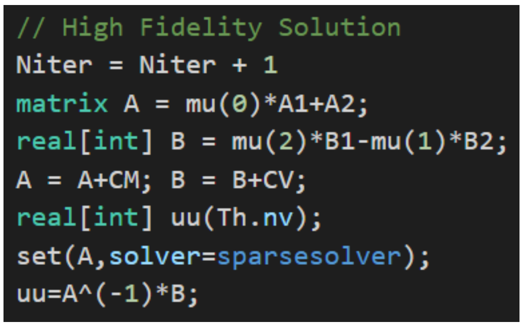
Before we start to follow the Greedy algorithm, Algorithm 10, we need to compute some of the $\boldsymbol{\mu}$-independent terms of the residual estimator, which will be needed when evaluating the error estimator, see code shown in Figure 5.2. This should be done after the affine decomposition is done, and before we start the Greedy algorithm.
We also need a starting parameter vector of the parameter set, $\boldsymbol{\mu}_1$, so we select the middle point of $\mathcal{P}$.
Following the Algorithm 10, we need to follow these steps:

1. First, we need to solve the High-Fidelity system to obtain the vector solution, $\boldsymbol{u}_h(\boldsymbol{\mu})$.
2. Secondly, we apply the Gram-Schmidt orthonormalization to the vector obtained, following Algorithm 11. So we obtain a vector $\boldsymbol{\zeta}$.
3. Then, we append $\boldsymbol{\zeta}$ to the reduced basis, $\mathbb{V}$.
4. With this new basis, we evaluate the error estimator for the parameter vectors in Ξ_{train}.
5. Finally, we select the maximum error estimator and the corresponding parameter vector.
6. Repeat all the steps if the tolerance criteria are not satisfied.

The code presented in Figures 5.3-5.11 is, indeed, the software corresponding to the Greedy procedure so, all of it is enclosed between the lines "while(Niter<Nmax & deltaG>tolGreedy){" and "}".

Greedy Procedure. High Fidelity Solving

Once we have selected a parameter vector of $\mathcal{P}$, our first step is to compute the high-fidelity solution of problem 4.3, by means of the Finite Element Method.



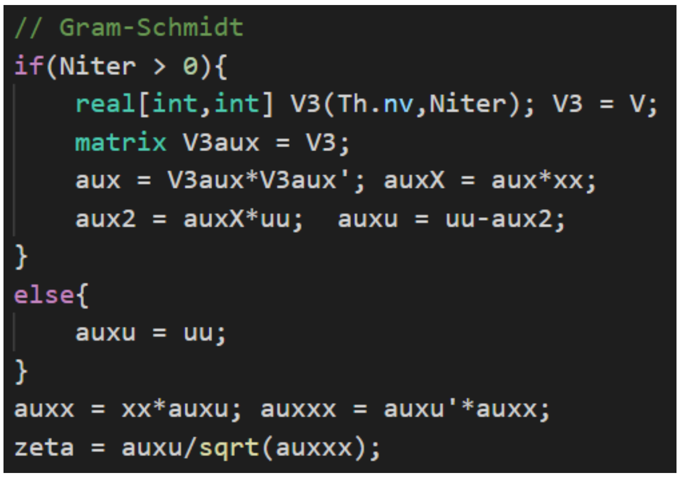
```
// High Fidelity Solution
Niter = Niter + 1
matrix A = mu(0)*A1+A2;
real[int] B = mu(2)*B1-mu(1)*B2;
A = A+CM; B = B+CV;
real[int] uu(Th.nv);
set(A,solver=sparsesolver);
uu=A^(-1)*B;
```

Figure 5.3: *Commands used for the beginning of the while loop and the obtaining of the high-fidelity solution.*

We assemble the involved associated matrices and vectors and solve the linear system, as can be seen in Figure 5.3. The corresponding matrix A of the system is sparse, so we select the sparse solver seeking speed and precision.

Greedy Procedure. Gram-Schmidt Orthonormalization

We have obtained the high fidelity solution by means of the Finite Element Method, but we cannot append it to the reduced basis right away. We first need to apply the Gram-Schmidt orthonormalization in order to keep the reduced basis as a projector into the reduced space.

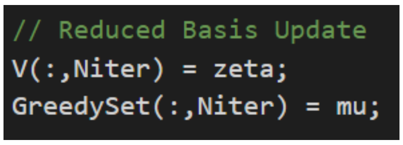


```
// Gram-Schmidt
if(Niter > 0){
    real[int,int] V3(Th.nv,Niter); V3 = V;
    matrix V3aux = V3;
    aux = V3aux*V3aux'; auxX = aux*xx;
    aux2 = auxX*uu;  auxu = uu-aux2;
}
else{
    auxu = uu;
}
auxx = xx*auxu; auxxx = auxu'*auxx;
zeta = auxu/sqrt(auxxx);
```

Figure 5.4: *Commands used for the application of the Gram-Schmidt orthonormalization with respect to the $\mathbb{X}_h$-norm.*

The Gram-Schmidt algorithm with respet to the $\mathbb{X}_h$-norm is resumed in Algorithm 11 and the code we are using is shown in Figure 5.4.

Greedy Procedure. Basis Update



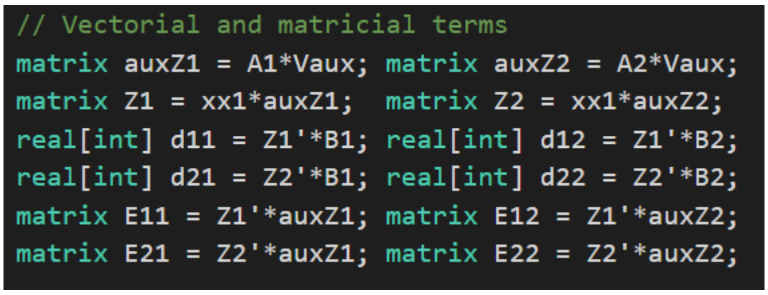
```
// Reduced Basis Update
V(:,Niter) = zeta;
GreedySet(:,Niter) = mu;
```

Figure 5.5: *Commands used for the update of the reduced basis and the set of parameter vectors.*

With the new vector orthonormalized with respect to the previous reduced basis, we can update both, the reduced basis, $\mathbb{V}$ and the set of parameter vectors selected in the Greedy algorithm, Ξ_g.
Commands for this step are shown in Figure 5.5.

Greedy Procedure. Evaluation of the Error Estimator

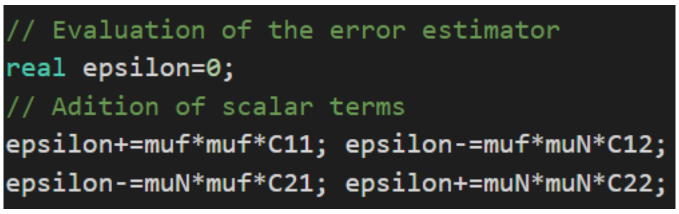
The next step is to evaluate the error estimator for each of the remaining parameter vectors of the training set. Before that, we need to compute the remaining of the $\boldsymbol{\mu}$-independent terms of the residual estimator, which will be needed when evaluating the error estimator, see Figure 5.2.



```
// Vectorial and matricial terms
matrix auxZ1 = A1*Vaux; matrix auxZ2 = A2*Vaux;
matrix Z1 = xx1*auxZ1;  matrix Z2 = xx1*auxZ2;
real[int] d11 = Z1'*B1; real[int] d12 = Z1'*B2;
real[int] d21 = Z2'*B1; real[int] d22 = Z2'*B2;
matrix E11 = Z1'*auxZ1; matrix E12 = Z1'*auxZ2;
matrix E21 = Z2'*auxZ1; matrix E22 = Z2'*auxZ2;
```

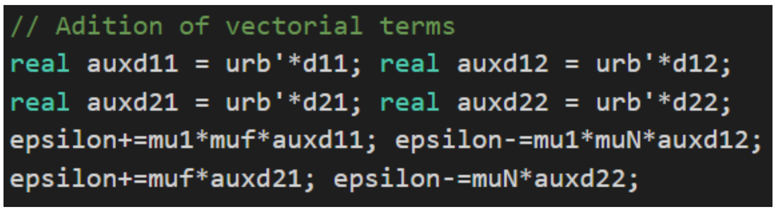
Figure 5.6: *Commands used for the computation of the vectorial and matricial terms involved in the residual error estimator.*

Following Algorithm 5, we need to:



```
// Evaluation of the error estimator
real epsilon=0;
// Adition of scalar terms
epsilon+=muf*muf*C11; epsilon-=muf*muN*C12;
epsilon-=muN*muf*C21; epsilon+=muN*muN*C22;
```

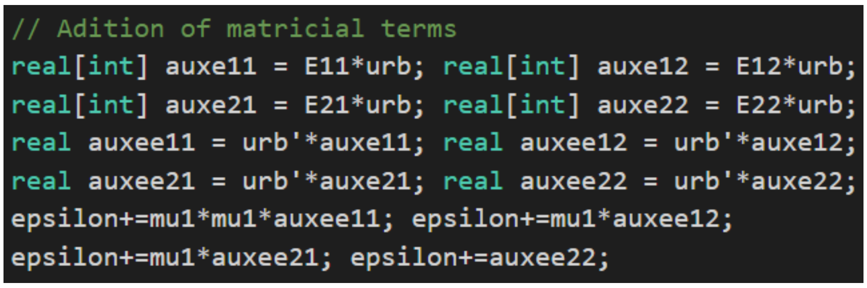
Figure 5.7: *Commands used for the evaluation of the scalar terms involved in the residual error estimator.*



```
// Adition of vectorial terms
real auxd11 = urb'*d11; real auxd12 = urb'*d12;
real auxd21 = urb'*d21; real auxd22 = urb'*d22;
epsilon+=mu1*muf*auxd11; epsilon-=mu1*muN*auxd12;
epsilon+=muf*auxd21; epsilon-=muN*auxd22;
```

Figure 5.8: *Commands used for the evaluation of the vectorial terms involved in the residual error estimator.*

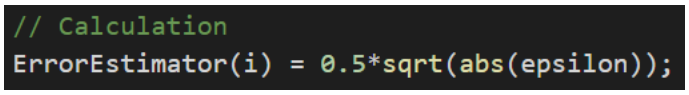
1. First, we add an auxiliar term, ε the corresponding scalar terms. See Figure 5.7.
2. Then, we add the vectorial terms, see Figure 5.8.
3. Finally, we add the corresponding matricial terms, see Figure 5.9.



```
// Adition of matricial terms
real[int] auxe11 = E11*urb; real[int] auxe12 = E12*urb;
real[int] auxe21 = E21*urb; real[int] auxe22 = E22*urb;
real auxee11 = urb'*auxe11; real auxee12 = urb'*auxe12;
real auxee21 = urb'*auxe21; real auxee22 = urb'*auxe22;
epsilon+=mu1*mu1*auxee11; epsilon+=mu1*auxee12;
epsilon+=mu1*auxee21; epsilon+=auxee22;
```

Figure 5.9: *Commands used for the evaluation of the matricial terms involved in the residual error estimator.*

4. To compute the error estimator, $\Delta_N(\boldsymbol{\mu})$, we just need to divide the auxiliar term ε, by the inf-sup constant, $\beta_h(\boldsymbol{\mu})$. The code is shown in Figure 5.10.

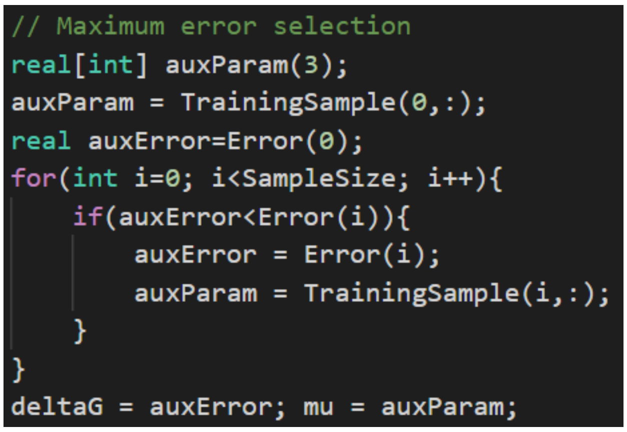


```
// Calculation
ErrorEstimator(i) = 0.5*sqrt(abs(epsilon));
```

Figure 5.10: *Commands used for the computation of the residual error estimator.*

Remark 5.2

In this case the inf-sup constant can be calculated to be $\boldsymbol{\mu}$-independent.



```
// Maximum error selection
real[int] auxParam(3);
auxParam = TrainingSample(0,:);
real auxError=Error(0);
for(int i=0; i<SampleSize; i++){
    if(auxError<Error(i)){
        auxError = Error(i);
        auxParam = TrainingSample(i,:);
    }
}
deltaG = auxError; mu = auxParam;
```

Figure 5.11: *Commands used for the definitions of the train sample and the stopping criteria.*

Greedy Procedure. Maximum Error Selection

With all the error estimators calculated, what is left is to select the greatest one and the corresponding parameter vector, as we have done in the code shown in Figure

5.11.

5.1.2 Online Phase

This phase is the one presented in the section 4.3.2.

5.2 Results

In this section, we present the results obtained when compiling the code shown in the previous section. We are using the same set than the one used in Chapter 4 for the POD case:

- Number of divisions of each side of the boundaries of Ω and Ω_1, $nn = 25$. This previous discretization yields a Finite Element space of dimension $N_h = 1710$ and $Nt_h = 3318$ triangles.

Thanks to the results obtained in the previous chapter, we can consider the cardinality of the parameter sample, $|\Xi_{train}| = n_{train} = 50$, but the stopping criteria to be much less, $N_{max} = 10$. In this first approach to Greedy method, we will consider the approximation error. The computation of the residual norm estimator, computationally more efficient, is left for the near future.
We compare the reduced solution obtained by means of the Greedy method with the solution obtained by means of the high fidelity technique. As done in the previous chapter, we compare the solutions for $\boldsymbol{\mu} = (1, 0.5, 2)$ and $\boldsymbol{\mu} = (0, 0, 5)$.

5.2.1 Offline Phase

For the application of the Greedy technique, we need to select the training sample, Ξ_{train}. As stated in the previous section we will only need $|\Xi_{train}| = n_{train} = 10$ in order to have a good approximation.

Randomly Selected Parameter Vectors			
$\boldsymbol{\mu}_1$	(1, 0.5, 5.5)	$\boldsymbol{\mu}_2$	(1.874488, 0.915550, 4.036574)
$\boldsymbol{\mu}_3$	(0.124890, 0.917799, 6.181785)	$\boldsymbol{\mu}_4$	(0.453448, 0.364568, 9.233515)
$\boldsymbol{\mu}_5$	(1.963821, 0.606405, 9.360574)	$\boldsymbol{\mu}_6$	(0.740735, 0.615922, 2.104873)
$\boldsymbol{\mu}_7$	(1.181710, 0.567371, 1.788721)	$\boldsymbol{\mu}_8$	(1.780477, 0.382845, 7.416643)

Table 5.1: Parameters obtained after applying the Greedy algorithm.

We run the code shown in Figures 5.1 to 5.5 and we obtain the parameters vectors shown in 5.1. We can now assemble the reduced matrices and vectors, following the code shown in Figure 4.15.

Now, we have every ingredient to step into the online phase.

5.2.2 Online Phase

In this section, we present the results obtained in the online phase of the Greedy technique application. We focus on the comparison between the solutions obtained via the Greedy technique and those obtained via the High-Fidelity technique. We study the same cases as in the previous chapter, $\boldsymbol{\mu} = (1, 0.5, 2)$ and $\boldsymbol{\mu} = (0, 0, 5)$, so eventually we can compare both solutions obtained by means of an order reduction technique.

Results for $\boldsymbol{\mu} = (1, 0.5, 2)$.

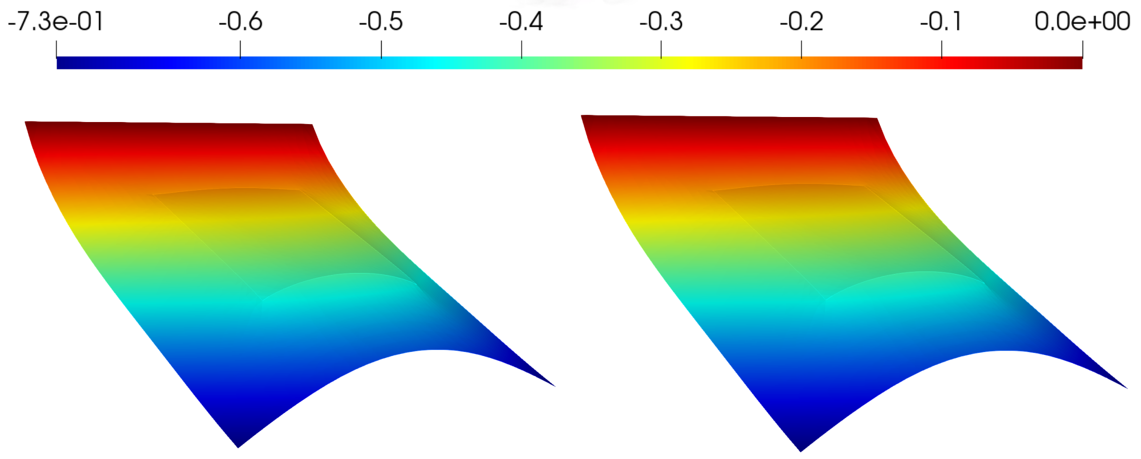


Figure 5.12: *For $\boldsymbol{\mu} = (1, 0.5, 2)$. Solution obtained by means of the Greedy method (left) and solution obtained by means of the High-Fidelity method (right).*

In the first case, we settle:

- $\mu_1 = 1$, so that there is a non-smooth transition between the characteristic parameters of the two subdomains.
- $\mu_N = 0.5$, a midway value for the parameter of the Neumann boundary.
- $\mu_f = 2$, so that the source has the same numerical value that the characteristic value of the inner subdomain.

Solving for this parameter vector, we obtain:

- For the Greedy method, the solution represented in Figure 5.12, left.

- For the High-Fidelity method, the solution represented in Figure 5.12, right.

We also represent the difference between both solutions in Figure 5.13.

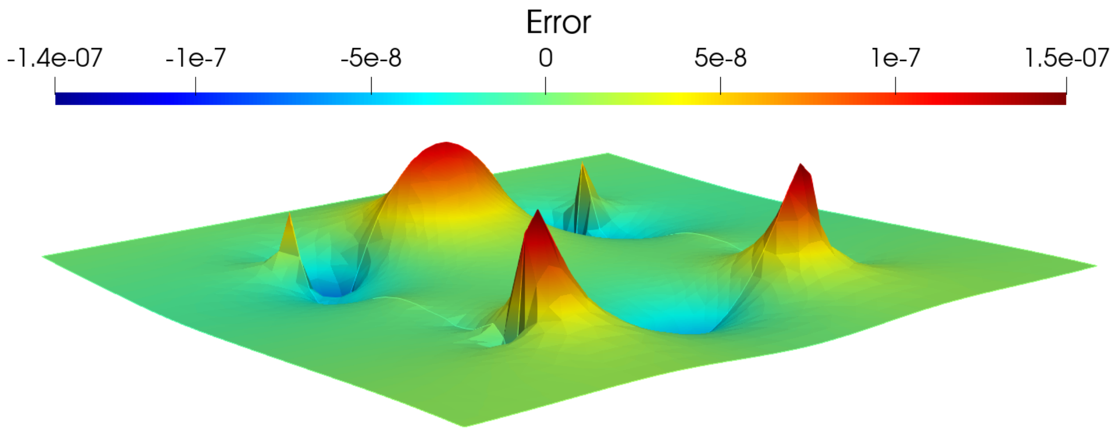


Figure 5.13: *For* $\boldsymbol{\mu} = (1, 0.5, 2)$. *Difference between the Reduced Basis and High-Fidelity methods.*

In this representation we can clearly see that the major differences are placed on the corners of the inner domain, as well as the middle points between two adjacent corner points.

Results for $\boldsymbol{\mu} = (0, 0, 5)$.

In the second case, we settle:

- $\mu_1 = 0$, so the characteristic parameters of the two subdomains are equal.
- $\mu_N = 0$, so the Neumann boundary does not affect the solutions.
- $\mu_f = 5$, so the source plays a predominant role in the solution.

Solving for this parameter vector we obtain:

- For the Greedy method, the solution represented in Figure 5.14, left.
- For the High-Fidelity method, the solution represented in Figure 5.14, right.

We also represent the difference between both solutions in Figure 5.15.

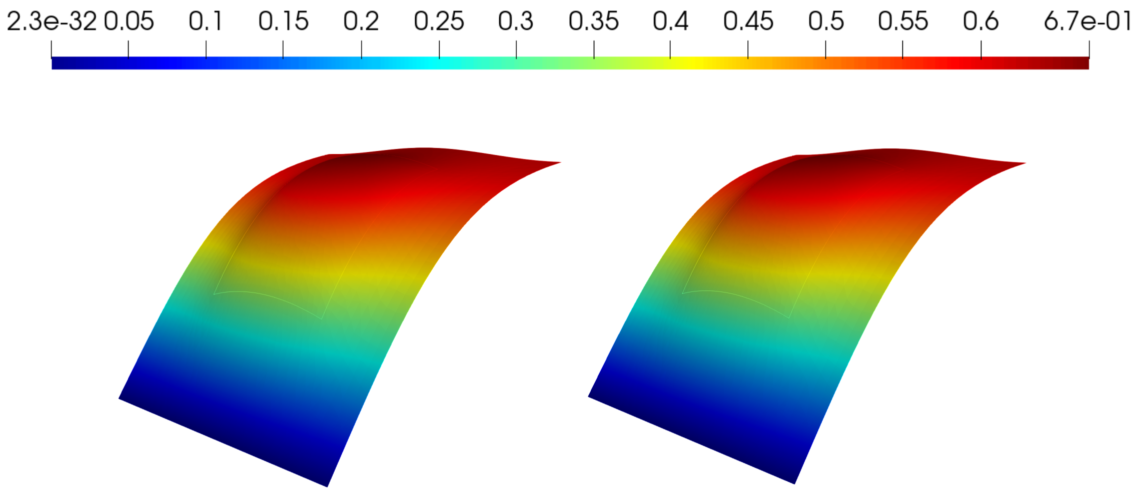

2.3e-32 0.05 0.1 0.15 0.2 0.25 0.3 0.35 0.4 0.45 0.5 0.55 0.6 6.7e-01

Figure 5.14: *For* $\boldsymbol{\mu} = (0, 0, 5)$. *Solution obtained by means of the Greedy method (left) and solution obtained by means of the High-Fidelity method (right).*

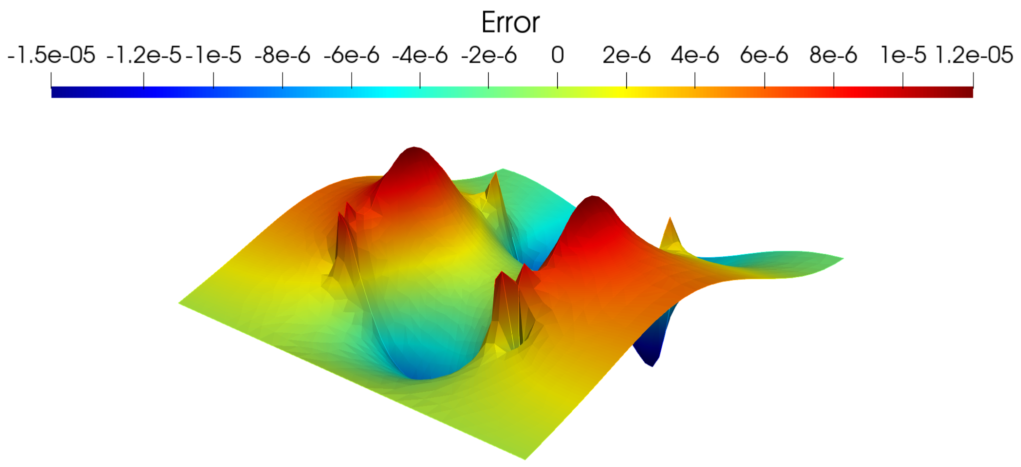

Error
-1.5e-05 -1.2e-5 -1e-5 -8e-6 -6e-6 -4e-6 -2e-6 0 2e-6 4e-6 6e-6 8e-6 1e-5 1.2e-05

Figure 5.15: *For* $\boldsymbol{\mu} = (0, 0, 5)$. *Difference between the Reduced Basis and High-Fidelity methods.*

Again, in the representation we can clearly see that the major differences are placed on the corners of the inner domain, as well as the middle points between two adjacent corner points, but the differences are spread over all the boundary not focused only on the corners.

Comparison

In order to compare the Greedy technique with the High-Fidelity technique, we calculate the computation times for both of them and then we obtain the quotient. We reproduce the problem for each parameter vector 10^4 times and we get

- For $\boldsymbol{\mu} = (1, 0.5, 2)$, we obtain $t_{Greedy} = (0.037 \pm 0.190) \cdot 10^{-3}$ s and $t_{HF} = (7.55 \pm 0.58) \cdot 10^{-3}$, so the quotient is:

$$\rho = \frac{t_{HF}}{t_{Greedy}} > 29.2.$$

- For $\boldsymbol{\mu} = (0, 0, 5)$, we obtain $t_{Greedy} = (0.039 \pm 0.201) \cdot 10^{-3}$ s and $t_{HF} = (7.57 \pm 0.59) \cdot 10^{-3}$, so the quotient is:

$$\rho = \frac{t_{HF}}{t_{Greedy}} > 29.$$

Roughly speaking, both quotients are bounded from below by $\rho^* = 25$, so we can conclude that the high-fidelity scheme is twenty five times slower than the Reduced Basis scheme. In other words, in the time we spend assembling and solving one high-fidelity problem we could have assembled and solved 25 reduced basis problems. This online time saving property is what makes the reduced basis technique a powerful tool when solving real time or many-query problems.
We now compare the computation times for the POD scheme and the Greedy scheme. We use the computations time shown above and in the POD application chapter.

- For $\boldsymbol{\mu} = (1, 0.5, 2)$, we obtain $t_{Greedy} = (0.037 \pm 0.190) \cdot 10^{-3}$ s and $t_{POD} = (0.040 \pm 0.201) \cdot 10^{-3}$, so the quotient is:

$$\rho = \frac{t_{Greedy}}{t_{POD}} \sim 1.$$

- For $\boldsymbol{\mu} = (0, 0, 5)$, we obtain $t_{Greedy} = (0.039 \pm 0.201) \cdot 10^{-3}$ s and $t_{POD} = (0.039 \pm 0.201) \cdot 10^{-3}$, so the quotient is:

$$\rho = \frac{t_{Greedy}}{t_{POD}} \sim 1.$$

This yields that for this particular problem both methods behave the similarly.

5.3 Conclusion

We end this chapter with some conclusions. Some of them are similar to the ones of the POD technique.

Comparison with the High-Fidelity technique

First of all, we have compared both solutions, the one obtained via a high-fidelity method and the other one obtained via a Greedy method, as can be seen in Figures 5.13 and 5.15 the main differences are set in the corners of boundary of the inner subdomain, this can be due to the distribution of the selected mesh in that boundary. We have calculated the quotient between the computation times obtained via both methods, we have found that the calculated quotients are bounded from below by $\rho^* = 25$ so we can conclude that the high-fidelity method is twenty-five times slower that the Greedy scheme.
This time saving property is the main advantage of the reduced order techniques and makes these methods an essential tool when solving real time or many-query problems. In problems where the resolution of the high-fidelity problem is more intricate the quotient is expected to rise so this property becomes more important.

Comparison with the POD technique

We have also compared both schemes, Greedy and POD, and we have found that for this particular problem both problems behave in a similar way in the Online Phase. The differences are thus in the Offline Phase, and due to the fact that we have considered the actual approximation error in the Greedy algorithm, the Offline Phase of the POD scheme is faster that the one of the Greedy scheme.

Chapter 6

ODE Application

In this last chapter, we discuss the use of the POD technique for model order reduction in the field of the Ordinary Differential Equations.
More precisely, we apply the POD technique on the temporal parameter, so we end with a low number of modes, which linear combination gives a representation of the solution such that a priori error is below a prescribed tolerance. This is the first step in the construction of the reduced system.
We study, the High Irradiance Responses (HIRES) problem [10], and the Intracellular Calcium Concentration (ICC) problem, studied in [7].
For the construction of this chapter we have consulted [1], [13], and for the applications [10] for the HIRES problem and [7] for the ICC problem.

6.1 Presentation

In this section we present the following:

- First of all, the POD method adapted to ODEs.
- The High Irradiance Responses (HIRES) problem.
- The Intracellular Calcium Concentration (ICC) problem.

6.1.1 POD method for ODEs

Once we have defined the problems, its parameters and associated vector field, we can apply the POD method, as described in [1]. The steps are:

1. Solve the system over a time interval $[t_0, T_{max}]$ by means of a high-precision solver, for instance, forth order Runge-Kutta solver or RADAU5.

2. Select a set of N_{snap} data points for different time values $\{t_1, \dots, t_{N_{snap}}\}$ in the time interval $[t_0, T_{max}]$. The set of data points is then

$$\{\boldsymbol{y}(t_1), \dots, \boldsymbol{y}(t_{N_{snap}})\},$$

where $\boldsymbol{y} \in \mathbb{R}^n$, for a fixed n.

From now on, we denote $\boldsymbol{y}_i = \boldsymbol{y}(t_i)$ and the j-th component of the vector $\boldsymbol{y}_i$ as $(\boldsymbol{y}_i)_j$.

3. Construction of the correlation matrix C, whose elements are defined by

$$C_{i,j} = (\boldsymbol{y}_i - \bar{\boldsymbol{y}}, \boldsymbol{y}_j - \bar{\boldsymbol{y}}),$$

where $\bar{\boldsymbol{y}} = \dfrac{1}{N_{snap}} \displaystyle\sum_{i=1}^{Ns} \boldsymbol{y}_i$, is the mean of the data points and $(\cdot, \cdot)$ is a scalar product to be defined. In the following, we denote $\tilde{\boldsymbol{y}}_i = \boldsymbol{y}_i - \bar{\boldsymbol{y}}$.

4. Then, we solve the eigenvalues problem for the correlation matrix and select the higher eigenvalues until the relative weight of the selected eigenvalues is above a prescribed tolerance as done for PDEs in Algorithm 6. At the end of this step, we will have obtained a set of modes $\{\boldsymbol{\phi}_1, ..., \boldsymbol{\phi}_{N_{rb}}\}$.

5. Finally, we obtain the temporal coefficients from the following projection

$$\boldsymbol{a}_{r,i} = (\tilde{\boldsymbol{y}}_i, \boldsymbol{\phi}_r), \quad \text{for } i = 1, \dots, N_{snap}.$$

Now that we have outlined the main steps of the POD method for the time parameter in ODEs, we present the problems we are going to study.

6.1.2 HIRES Problem

For the construction of this section we have followed [10].
The problem we present in this section originates from plant physiology and describes how light is involved in morphogeneisis. To be more precise, it explains the *High Irradiance Responses* (HIRES) of photomorphogenesis on the basis of phytochrome, by means of a chemical reaction involving eight reactants. HIRES is a stiff system of 8 ODEs, for a detailed description, consult [8].
This problem is of the following form,

$$\frac{d\boldsymbol{y}}{dt} = \boldsymbol{f}(\boldsymbol{y}); \qquad \boldsymbol{y}(0) = \boldsymbol{y}_0,$$

with $\boldsymbol{y} \in \mathbb{R}^8$ and the function $\boldsymbol{f}(\boldsymbol{y})$ is defined by

$$\boldsymbol{f}(\boldsymbol{y}) = \begin{bmatrix} -k_1y_1 + k_2y_2 + k_6y_3 + o_{k_s} \\ k_1y_1 - (k_2 + k_3)y_2 \\ -(k_1 + k_6)y_3 + k_2y_4 + k_5y_5 \\ k_3y_2 + k_1y_3 - (k_2 + k_4)y_4 \\ -(k_1 + k_5)y_5 + k_2y_6 + k_2y_7 \\ -k_+y_6y_8 + k_4y_4 + k_1y_5 - k_2y_6 + k_-y_7 \\ k_+y_6y_8 - (k_2 + k^* + k_-)y_7 \\ -k_+y_6y_8 + (k_2 + k^* + k_-)y_7 \end{bmatrix}. \tag{6.1}$$

Values of the parameters are given in Table 6.1, and the initial vector is

$$\boldsymbol{y}_0 = [1, 0, 0, 0, 0, 0, 0, 0.057]^T.$$

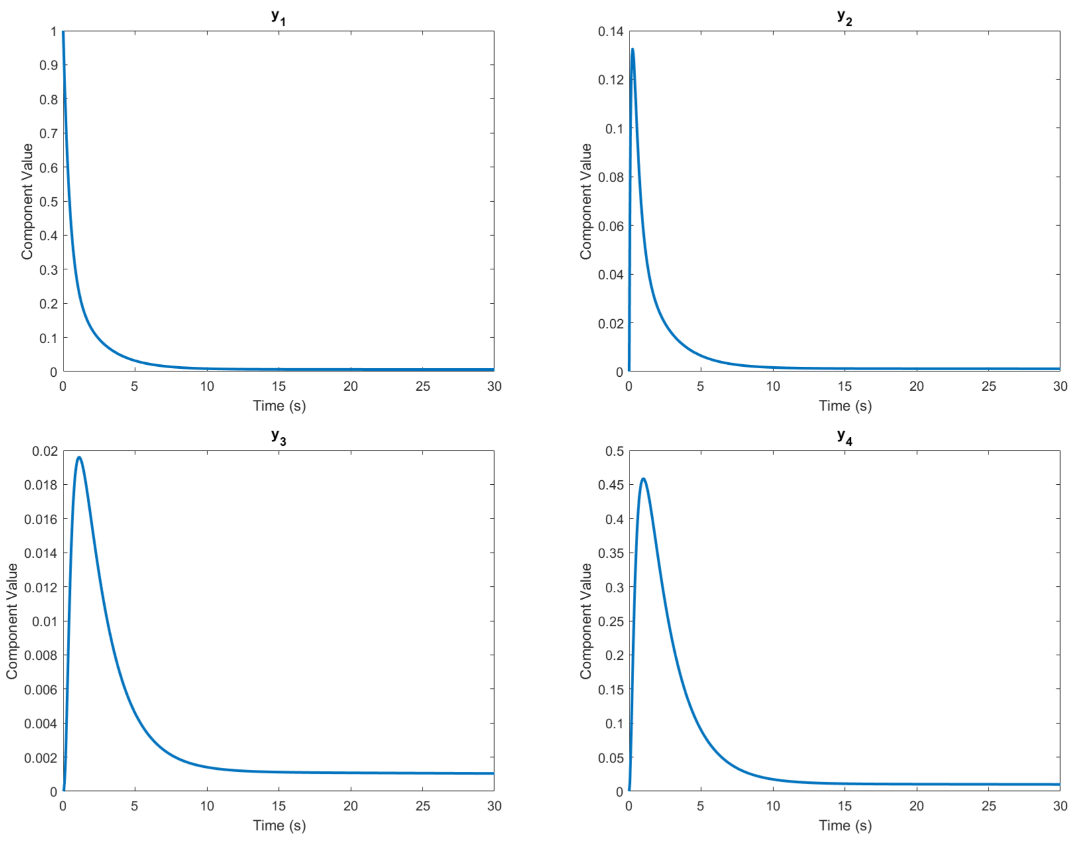


Figure 6.1: *Components 1 to 4 of the HIRES solution in [0, 30].*

We consider a time interval, namely $[0, 30]$, that captures the main behaviour of the solution. After $t = 30$, the solution changes at a much slower rate. A plot showing

Model Parameters				
$k_1 = 1.71$	$k_3 = 8.32$	$k_5 = 0.035$	$k_+ = 280$	$k^* = 0.69$
$k_2 = 0.43$	$k_4 = 0.69$	$k_6 = 8.32$	$k_- = 0.69$	$o_{k_s} = 0.0007$

Table 6.1: Model Parameters for the HIRES Model

the solution behaviour over the time interval is presented in Figures 6.1 and 6.2. The solution is obtained by means of a forth order Runge-Kutta solver.

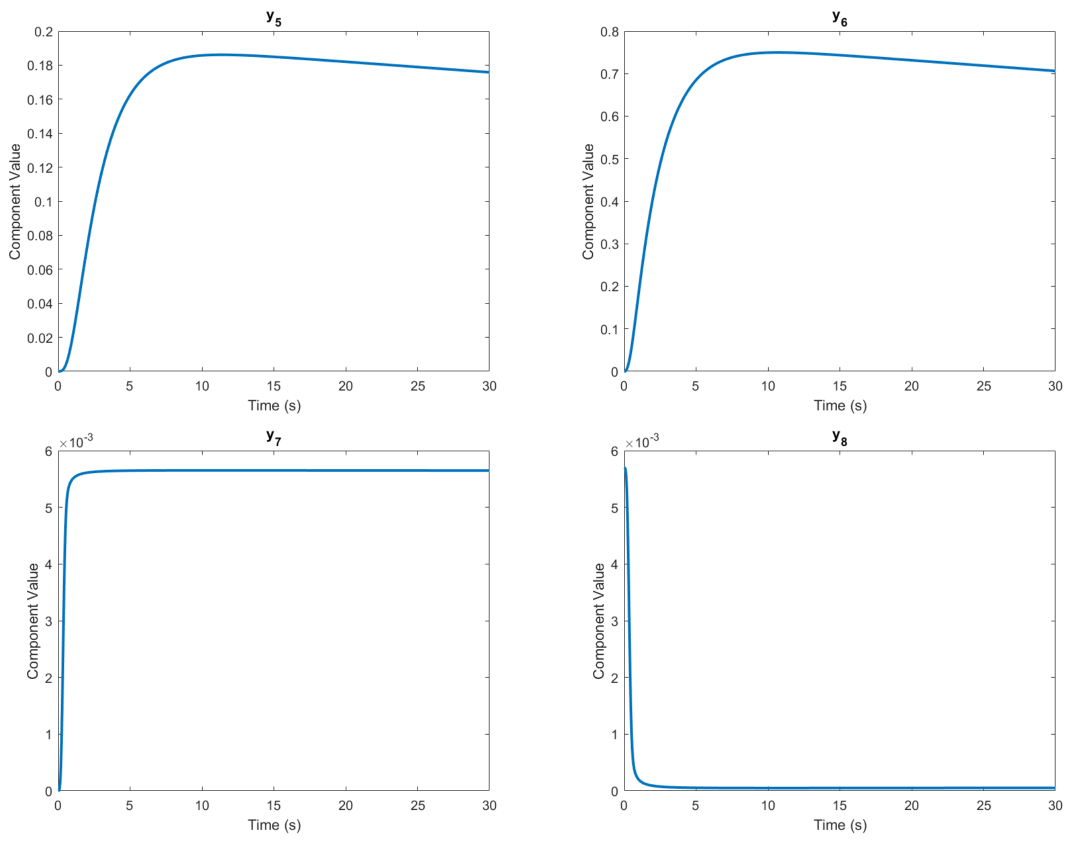


Figure 6.2: *Components 5 to 8 of the HIRES solution in [0, 30].*

6.1.3 ICC Problem

For the construction of this section we have followed [7].
The problem we present in this section originates from neural synchronization and describes oscillations in intracellular calcium concentrations (ICC) of two coupled neurons. To be more precise, the one cell model has been originally designed for

reproducing the changes in ICC of a single gonadotropin-releasing hormone (GnRH) expressing neuron. For a more detailed explanation consult [7].
Again, the problem is of the following form

$$\frac{d\boldsymbol{w}}{dt} = \boldsymbol{f}(\boldsymbol{w}); \qquad \boldsymbol{w}(0) = \boldsymbol{w}_0,$$

with $\boldsymbol{w} = \{x, y, z, X, Y, Z\} \in \mathbb{R}^6$ and the function $\boldsymbol{f}(\boldsymbol{w})$ defined by

$$\boldsymbol{f}(\boldsymbol{w}) = \begin{bmatrix} \tau(-y + g(x) - \phi_f(z)), \\ \tau\epsilon(z + a_1 y + a_2 + c(x - X)), \\ \tau\epsilon\left(\phi_r(x) - \dfrac{z - Ca_b}{\tau_{Ca}}\right) \\ \tau(-Y + g(X) - \phi_f(Z)), \\ \tau\epsilon(X + a_1 Y + a_2 + c(X - x)), \\ \tau\epsilon\left(\phi_r(X) - \dfrac{Z - Ca_b}{\tau_{Ca}}\right) \end{bmatrix}, \tag{6.2}$$

where g is an odd cubic function such that $r = g(x)$ is an cubic-shaped curve, g admits a local minimum at $x = -x_f < 0$ and a local maximum at $x = x_f > 0$, and

$$\phi_f(z) = \frac{\mu z}{z + Ca_0}, \quad \phi_r(x) = \frac{\lambda}{1 + \exp(-\rho(x - x_{on}))}.$$

Moreover, the first three equations correspond to one of the neurons while the latter three correspond to the other one.
For the simulations shown, in [7], they have used $g(x) = -x^3 + 4x$, and the parameter values are summarized in Table 6.2.

Model Parameters					
$a_1 = -0.1$	$a_2 = 0.8$	$\tau = 37$	$Ca_b = 1$	$Ca_0 = 5$	$\tau_{Ca} = 2$
$\varepsilon = 0.06$	$\mu = 2.4$	$x_{on} = -0.45$	$\lambda = 1.75$	$\rho = 4.5$	$c \in [-1, 1]$

Table 6.2: Model Parameters for the ICC Model

We consider a time interval, namely $[0, 40]$, that captures the main behaviour of the solution. After $t = 40$, the solution is periodic in most of the cases. A plot showing the solution behaviour for different values of c over the time interval is presented in Figure 6.3. The solution is obtained by means of a forth order Runge-Kutta solver.

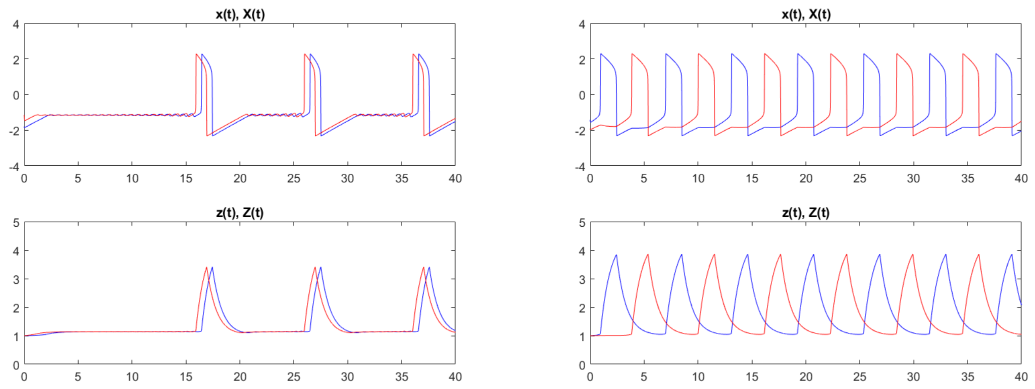
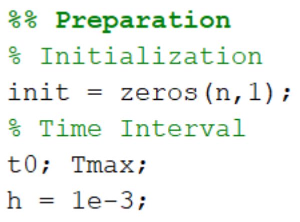


Figure 6.3: *Comparison of the first and third components of each cell of ICC model for* $c = 0$ *(uncoupling, left) and* $c = -0.25$ *(antiphase synchronization, right)*

In the next section, we present the code used for the resolution of these problem.

6.2 Computation

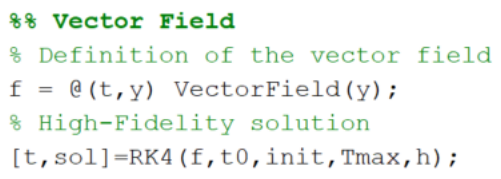
In this section, we present the code used for the POD method application. We use the MATLAB software, version 2020b.
First of all, we define the parameters involved in our ODE problem. After thatm the code will be shown in each particular example.
First, we define the initial condition and the time interval where we solve the problem, as shown in Figure 6.4.



```
%% Preparation
% Initialization
init = zeros(n,1);
% Time Interval
t0; Tmax;
h = 1e-3;
```

Figure 6.4: *Code used for the definition of the initial condition (it will be changed for each problem) and the time interval.*

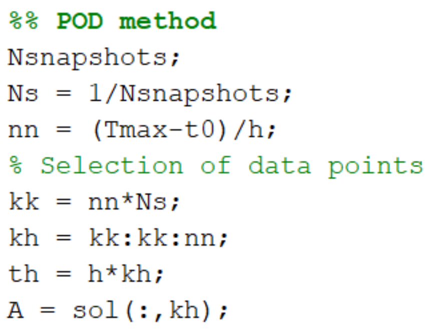
We now define the vector field (one for each problem) and solve over the time interval with a high precision solver, in his case a forth order Runge-Kutta solver. This can be seen in the code shown in Figure 6.5.



```
%% Vector Field
% Definition of the vector field
f = @(t,y) VectorField(y);
% High-Fidelity solution
[t,sol]=RK4(f,t0,init,Tmax,h);
```

Figure 6.5: *Code used for the definition of the vector field, and for the resolution of the problem over the time interval by means of a high precision method.*

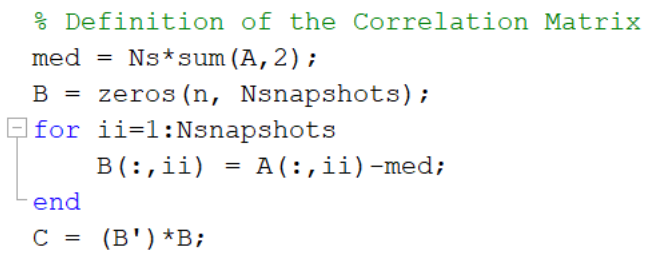
We can now select a set of data points from the high-precision solution, if the points are selected equally spaced in time, we use the code shown in Figure 6.6.



```
%% POD method
Nsnapshots;
Ns = 1/Nsnapshots;
nn = (Tmax-t0)/h;
% Selection of data points
kk = nn*Ns;
kh = kk:kk:nn;
th = h*kh;
A = sol(:,kh);
```

Figure 6.6: *Code used for the data points selection.*

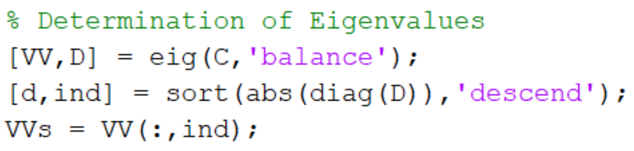
The next step is to build the correlation matrix. We use the code shown in Figure 6.7.



```
% Definition of the Correlation Matrix
med = Ns*sum(A,2);
B = zeros(n, Nsnapshots);
for ii=1:Nsnapshots
    B(:,ii) = A(:,ii)-med;
end
C = (B')*B;
```

Figure 6.7: *Code used for the construction of the correlation matrix.*

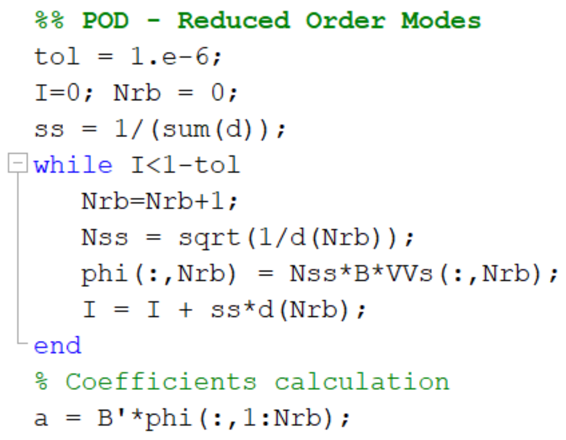
Now, we compute the eigenvalues and eigenvectors of the correlation matrix by means of the code shown in Figure 6.8.



```
% Determination of Eigenvalues
[VV,D] = eig(C,'balance');
[d,ind] = sort(abs(diag(D)),'descend');
VVs = VV(:,ind);
```

Figure 6.8: *Code used for the determination of the eigenvalues and eigenvectors of the correlation matrix.*

Now, we select the eigenvalues until a minimum tolerance criterion is satisfied by means of the code shown in Figure 6.9.



```
%% POD - Reduced Order Modes
tol = 1.e-6;
I=0; Nrb = 0;
ss = 1/(sum(d));
while I<1-tol
   Nrb=Nrb+1;
   Nss = sqrt(1/d(Nrb));
   phi(:,Nrb) = Nss*B*VVs(:,Nrb);
   I = I + ss*d(Nrb);
end
% Coefficients calculation
a = B'*phi(:,1:Nrb);
```

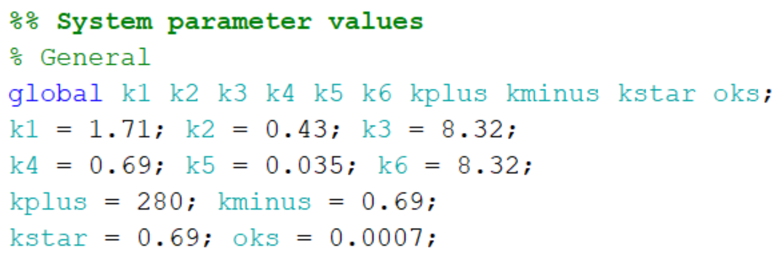
Figure 6.9: *Code used for the selection of eigenvalues until a minimum tolerance criterion is prescribed.*

We have already presented the general software used for both problems. In the following sections we present the proper software used for each problem in particular. The differences are

- The dimension of the problem. We work inside $\mathbb{R}^8$ for HIRES problem and inside $\mathbb{R}^6$ for the ICC problem.
- The parameters values involved in each model.
- The definition of the vector field.

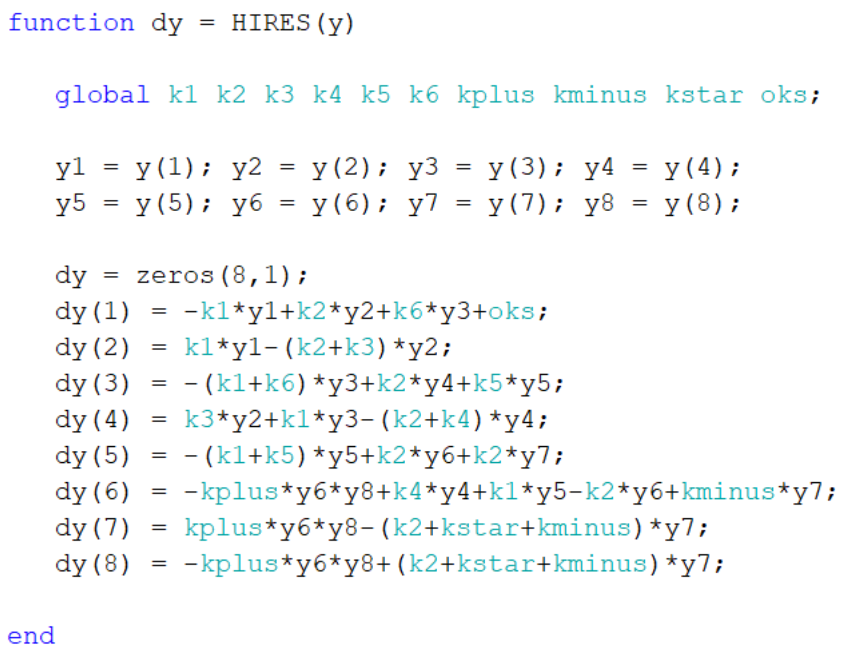
6.2.1 HIRES Problem

We define the parameters involved in the HIRES model (see Table 6.1) and its associated vector field (see Equation 6.1) as shown in Figures 6.10 and 6.11.



```
%% System parameter values
% General
global k1 k2 k3 k4 k5 k6 kplus kminus kstar oks;
k1 = 1.71; k2 = 0.43; k3 = 8.32;
k4 = 0.69; k5 = 0.035; k6 = 8.32;
kplus = 280; kminus = 0.69;
kstar = 0.69; oks = 0.0007;
```

Figure 6.10: *Code used for the definition of the parameters involved in the HIRES problem.*

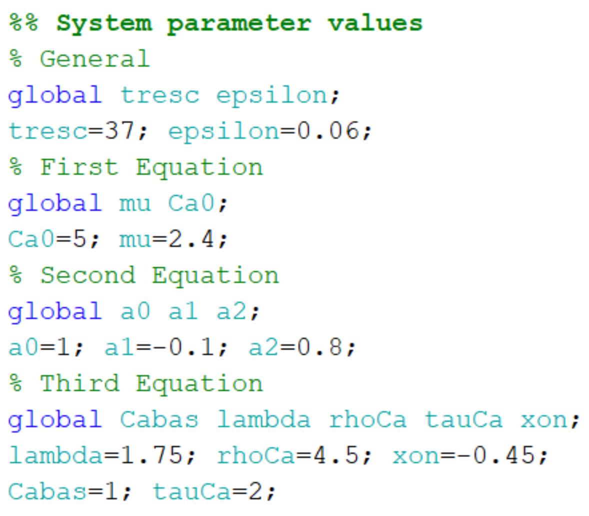


```
function dy = HIRES(y)

   global k1 k2 k3 k4 k5 k6 kplus kminus kstar oks;

   y1 = y(1); y2 = y(2); y3 = y(3); y4 = y(4);
   y5 = y(5); y6 = y(6); y7 = y(7); y8 = y(8);

   dy = zeros(8,1);
   dy(1) = -k1*y1+k2*y2+k6*y3+oks;
   dy(2) = k1*y1-(k2+k3)*y2;
   dy(3) = -(k1+k6)*y3+k2*y4+k5*y5;
   dy(4) = k3*y2+k1*y3-(k2+k4)*y4;
   dy(5) = -(k1+k5)*y5+k2*y6+k2*y7;
   dy(6) = -kplus*y6*y8+k4*y4+k1*y5-k2*y6+kminus*y7;
   dy(7) = kplus*y6*y8-(k2+kstar+kminus)*y7;
   dy(8) = -kplus*y6*y8+(k2+kstar+kminus)*y7;

end
```

Figure 6.11: *Code used for the definition of the vector field in HIRES problem.*

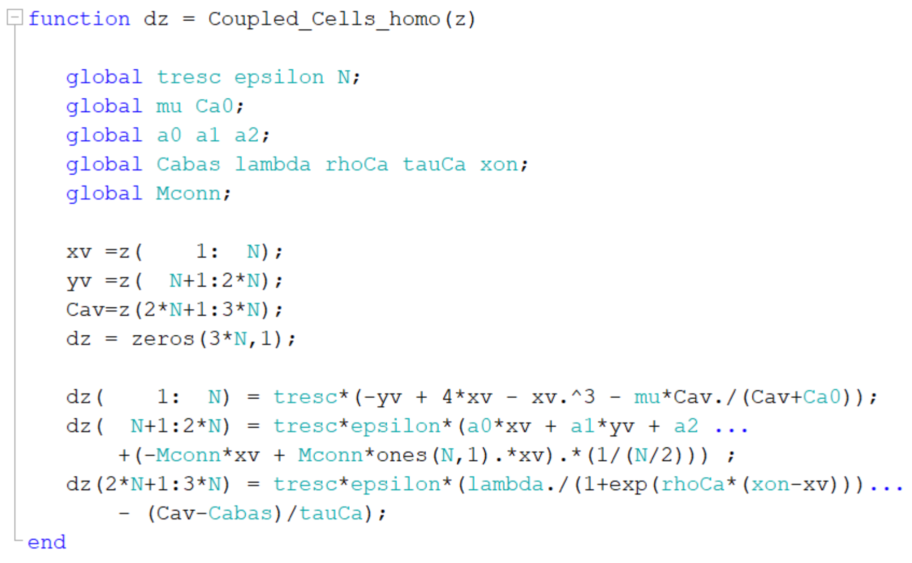
6.2.2 ICC Problem

We define the parameters involved in the ICC model (see Table 6.2) and its associated vector field (see Equation 6.2) as shown in Figures 6.12 and 6.13.



```
%% System parameter values
% General
global tresc epsilon;
tresc=37; epsilon=0.06;
% First Equation
global mu Ca0;
Ca0=5; mu=2.4;
% Second Equation
global a0 a1 a2;
a0=1; a1=-0.1; a2=0.8;
% Third Equation
global Cabas lambda rhoCa tauCa xon;
lambda=1.75; rhoCa=4.5; xon=-0.45;
Cabas=1; tauCa=2;
```

Figure 6.12: *Code used for the definition of the parameters involved in the ICC problem.*



```
function dz = Coupled_Cells_homo(z)

    global tresc epsilon N;
    global mu Ca0;
    global a0 a1 a2;
    global Cabas lambda rhoCa tauCa xon;
    global Mconn;

    xv =z(     1:   N);
    yv =z(   N+1:2*N);
    Cav=z(2*N+1:3*N);
    dz = zeros(3*N,1);

    dz(     1:   N)  = tresc*(-yv + 4*xv - xv.^3 - mu*Cav./(Cav+Ca0));
    dz(   N+1:2*N)  = tresc*epsilon*(a0*xv + a1*yv + a2 ...
        +(-Mconn*xv + Mconn*ones(N,1).*xv).*(1/(N/2))) ;
    dz(2*N+1:3*N)  = tresc*epsilon*(lambda./(1+exp(rhoCa*(xon-xv)))...
        - (Cav-Cabas)/tauCa);
end
```

Figure 6.13: *Code used for the definition of the vector field in ICC problem.*

6.3 Results

In this section, we present the numerical results obtained while and after the use of the POD method for ODEs.
For each problem, we follow the same path:

1. First, we set the initial condition and the time interval as well as the particular elements of the problem.
2. Second, we select the number of selected data points, N_{snap}, from the high precision solution obtained via a forth order Runge-Kutta solver.
3. Third, we obtain the eigenvalues and eigenvectors of the correlation matrix, built from the data points. Then, we select the higher eigenvalues until a tolerance criterion is satisfied.
4. Eventually, we present some graphic representations and comparisons.

6.3.1 HIRES Problem

Computation scheme

1. First of all, we set the time interval $[t_0, T_{max}] = [0, 30]$, the step for the high-precision solution $h = 10^{-3}$ and the initial condition y_0. This is modified in the code shown in Figure 6.4. Then we define the vector field, code shown in Figure 6.11 and solve the problem by means of a high-precision method.
2. As done in [10] for the construction of the POD projection matrix, we select $N_{snap} = 100$ data points equally spaced in the interval $[t_0, T_{max}] = [0, 30]$ by means of the code shown in Figure 6.6.
3. The next step is to build the correlation matrix, code shown in Figure 6.7, and compute its eigenvalues and eigenvectors, code shown in Figure 6.8. Now we apply the POD technique and select the higher eigenvalues until the minimum tolerance criterion is satisfied, code shown in Figure 6.9. We also compute the coefficients of each mode.

Comparison

For the tolerance $\varepsilon = 10^{-6}$, we obtain a reduced order space of dimension $N_{rb} = 5$. We now present the differences between the Runge-Kutta solution and the solutions obtained for $N_{rb} \in \{3, 4, 5\}$. We also compare the solutions with the one obtained via another high-precision solver, RADAU solver, see Table 6.3.

Comparisons					
	RK4	$N_{rb} = 3$	$N_{rb} = 4$	$N_{rb} = 5$	RADAU
$y_1(0.9)$	2,82755E-01	2,83380E-01	2,82906E-01	2,83052E-01	2,82754E-01
$y_1(30)$	5,77881E-03	5,73001E-03	5,79412E-03	5,77882E-03	5,78424E-03
$y_2(0.9)$	6,35866E-02	6,19568E-02	6,43788E-02	6,36594E-02	6,35863E-02
$y_2(30)$	1,12975E-03	1,38189E-03	1,05442E-03	1,12979E-03	1,12976E-03
$y_3(0.9)$	1,90327E-02	1,91969E-02	1,89498E-02	1,90359E-02	1,90327E-02
$y_3(30)$	1,04302E-03	1,01897E-03	1,05238E-03	1,04336E-03	1,04302E-03
$y_4(0.9)$	4,56473E-01	4,55706E-01	4,56392E-01	4,56421E-01	4,56473E-01
$y_4(30)$	1,00132E-02	1,01090E-02	1,00163E-02	1,00132E-02	1,00133E-02
$y_5(0.9)$	1,57766E-02	9,64160E-03	1,54152E-02	1,57349E-02	1,57766E-02
$y_5(30)$	1,75780E-01	1,76594E-01	1,75814E-01	1,75780E-01	1,75780E-01
$y_6(0.9)$	1,55536E-01	1,56523E-01	1,55343E-01	1,55268E-01	1,55536E-01
$y_6(30)$	7,06247E-01	7,06082E-01	7,06241E-01	7,06249E-01	7,06246E-01
$y_7(0.9)$	5,46310E-03	5,15447E-03	5,60250E-03	5,46587E-03	5,46306E-03
$y_7(30)$	5,64830E-03	5,69456E-03	5,63399E-03	5,64830E-03	5,64830E-03
$y_8(0.9)$	2,36900E-04	5,45527E-04	9,74971E-05	2,34134E-04	2,36937E-04
$y_8(30)$	5,16981E-05	5,43681E-06	6,60129E-05	5,16982E-05	5,16992E-05

Table 6.3: Comparison of the solutions obtained via POD method for different N_{rb} and RK4 and RANDAU solvers.

In Table 6.3, we can clearly see that for $N_{rb} = 5$ the reduced solutions approximates very faithfully the data set. In Figure 6.14, we represent the relative error of the solutions for $N_{rb} \in \{3, 4, 5\}$ and we see that for $N_{rb} = 5$ a great improvement is achieved.
We represent in Figure 6.15, the coefficients obtained after applying the POD technique. As it can be seen near $T_{max} = 30$ all the modes are near to zero, that shows that at that time the solutions doesn't change radically.
In Figures 6.16 - 6.19, we represent the comparisons between the Runge-Kutta solution and the ones obtained for different values of N_{rb}. Also a zoom of the time interval $[0, 3]$ is represented so we can spot more easily the differences between the solutions.

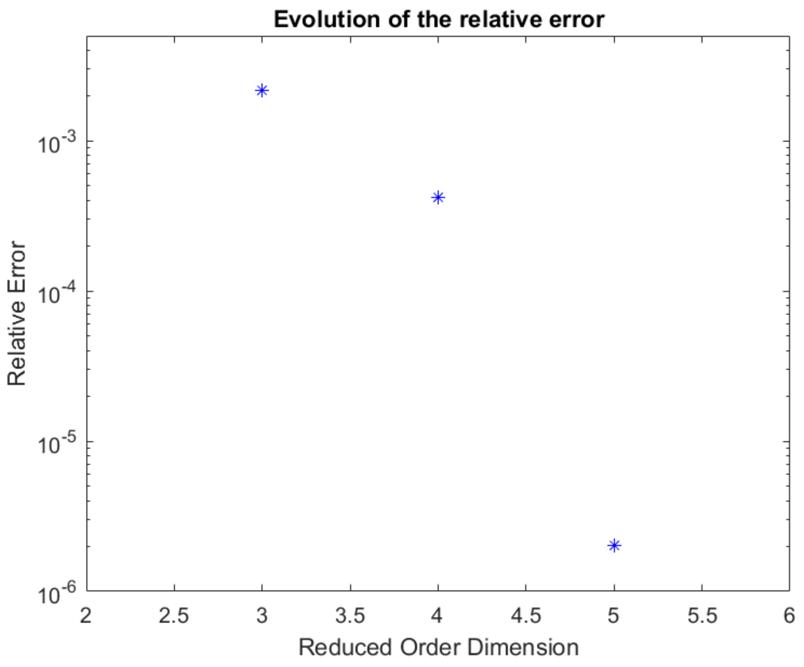


Figure 6.14: *Comparison of the relative error for different reduced order dimensions.*

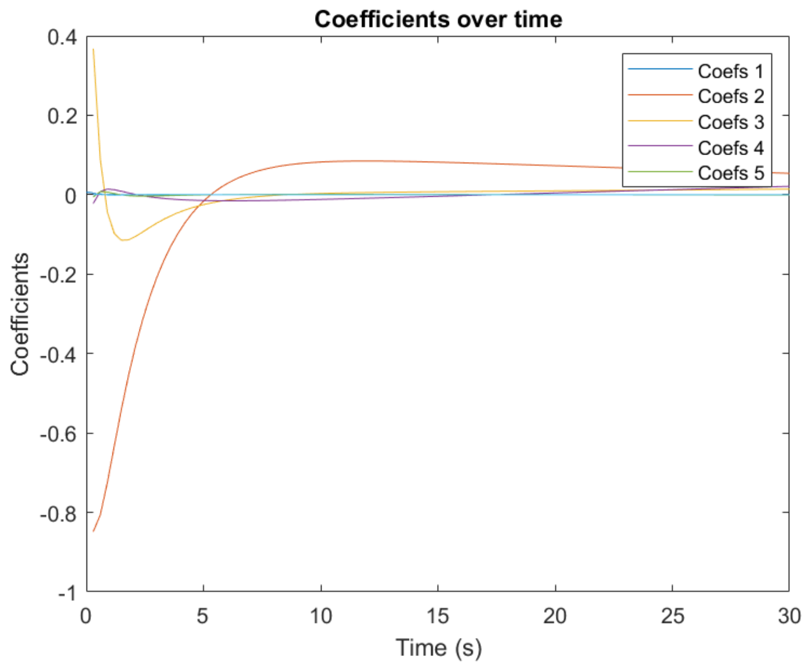


Figure 6.15: *Representations of the coefficients over time obtained in the POD method.*

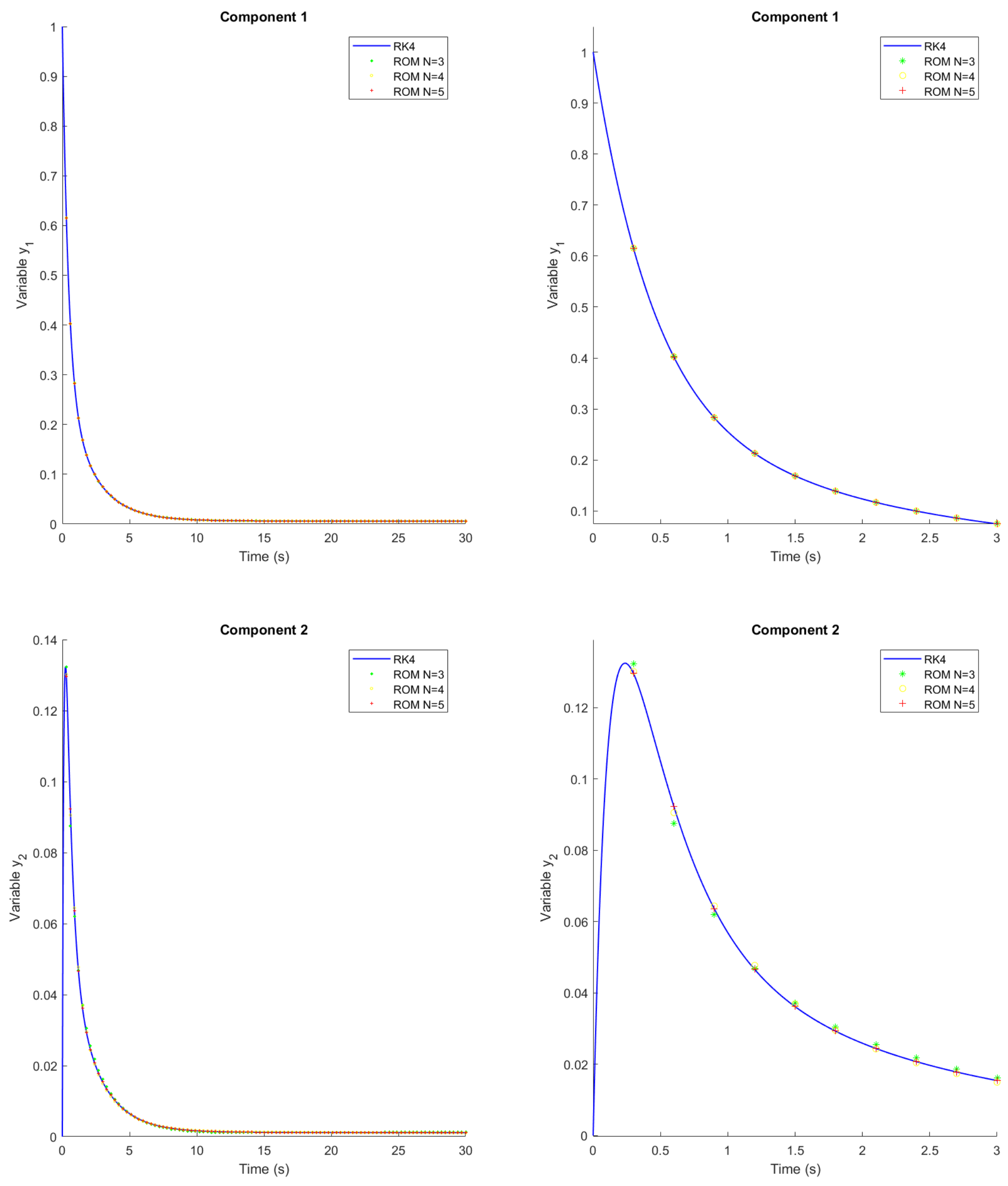


Figure 6.16: *Comparison of the first and second component of the HIRES model solutions: Runge-Kutta solution (blue, continuous), $N_{rb} = 3$ solution (green, stars), $N_{rb} = 4$ solution (yellow, circles) and $N_{rb} = 5$ solution (red, cross).*

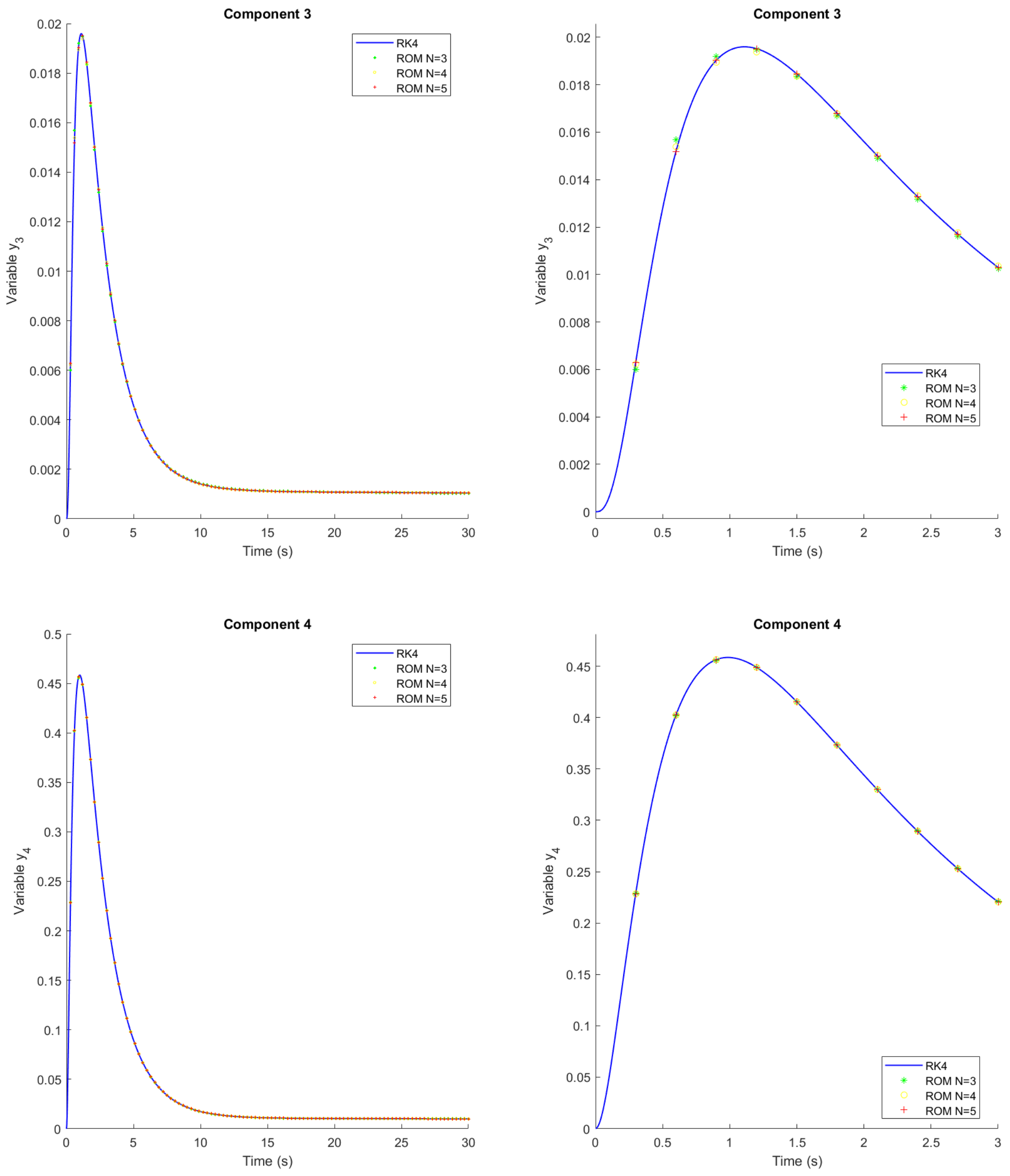


Figure 6.17: *Comparison of the third and forth component of the HIRES model solutions: Runge-Kutta solution (blue, continuous), $N_{rb} = 3$ solution (green, stars), $N_{rb} = 4$ solution (yellow, circles) and $N_{rb} = 5$ solution (red, cross).*

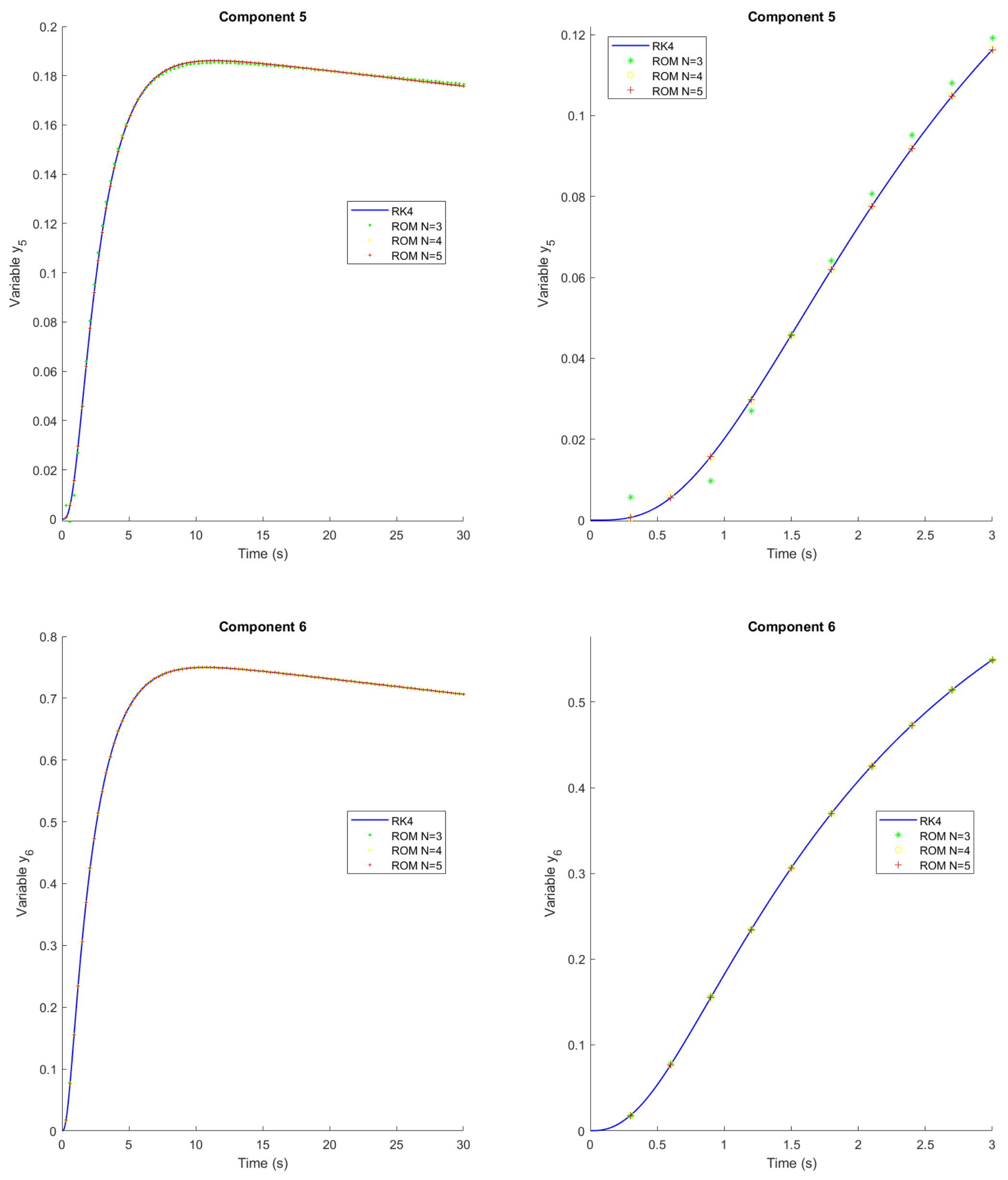


Figure 6.18: *Comparison of the fifth and sixth component of the HIRES model solutions: Runge-Kutta solution (blue, continuous), $N_{rb} = 3$ solution (green, stars), $N_{rb} = 4$ solution (yellow, circles) and $N_{rb} = 5$ solution (red, cross).*

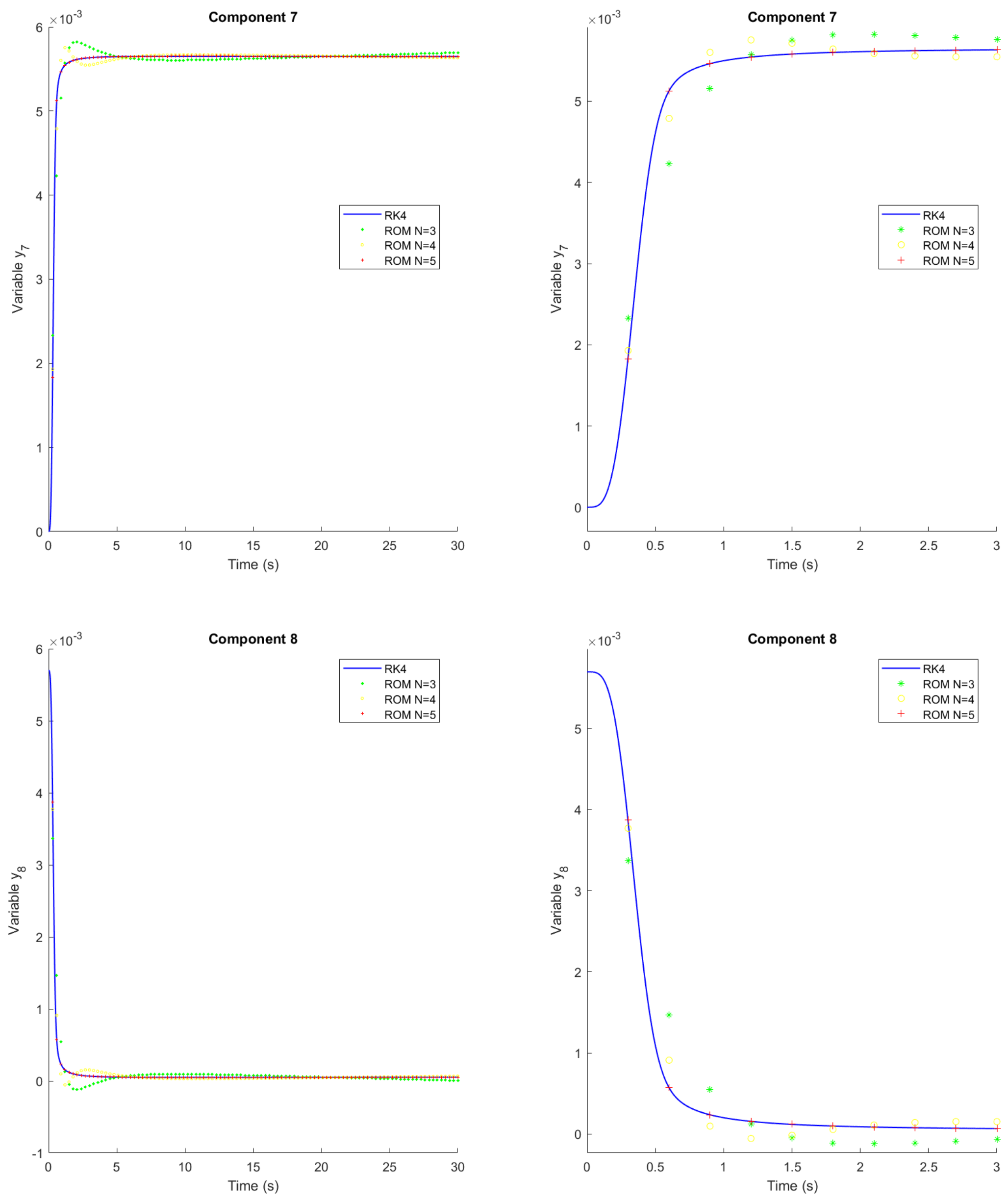


Figure 6.19: *Comparison of the seventh and eighth component of the HIRES model solutions: Runge-Kutta solution (blue, continuous), $N_{rb} = 3$ solution (green, stars), $N_{rb} = 4$ solution (yellow, circles) and $N_{rb} = 5$ solution (red, cross).*

6.3.2 ICC Problem

Computation scheme

1. First of all, we set the time interval $[t_0, T_{max}] = [0, 40]$, the step for the high-precision solution $h = 10^{-3}$ and the initial condition w_0. This is modified in the code shown in Figure 6.4. Then we define the vector field, code shown in Figure 6.13 and solve the problem by means of a high-precision method.

2. Due to the slow-fast nature of this model, we need more data points in order to obtain a good approximation, so for the construction of the POD projection matrix, we select $N_{snap} = 400$ data points equally spaced in the interval $[t_0, T_{max}] = [0, 40]$ by means of the code shown in Figure 6.6.

3. The next step is to build the correlation matrix, code shown in Figure 6.7, and compute its eigenvalues and eigenvectors, code shown in Figure 6.8. Now we apply the POD technique and select the higher eigenvalues until the minimum tolerance criterion is satisfied, code shown in Figure 6.9. We also compute the coefficients of each mode.

In this case we will solve the system for different values of the coupling parameter $c \in [-1, 1]$. We select those values, which are representative of each different synchronization pattern that appears in the article [7].

Uncoupled case. $c = 0$

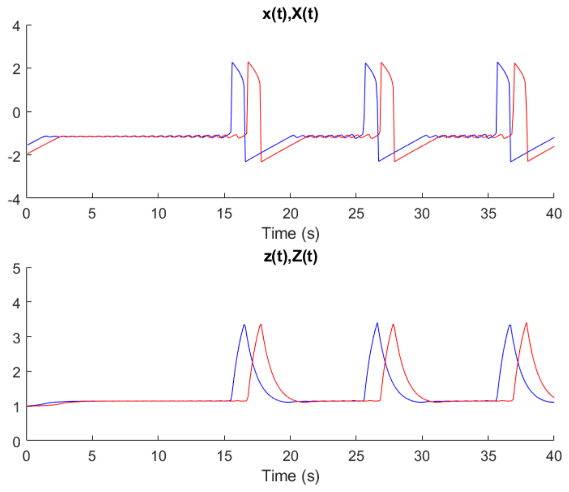


Figure 6.20: *Reduced order solution for the uncoupled case.*

We start with the case in which the two identical cell are uncoupled so they should behave similarly but independently (by taking different initial conditions).
In Figure 6.20, we represent the reduced solution obtained after the application of the POD method to the data set.
In Figure 6.21, we represent the relative error of the solutions for $N_{rb} \in \{4, 5, 6\}$ and we see that for $N_{rb} = 6$ a great improvement is achieved.

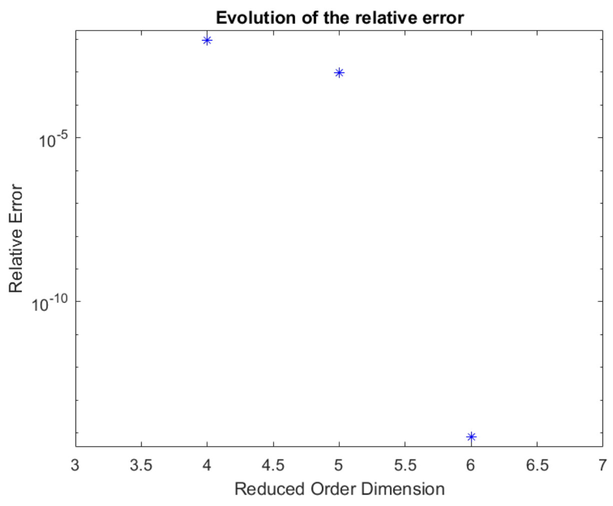


Figure 6.21: *Comparison of the relative error for different reduced order dimensions.*

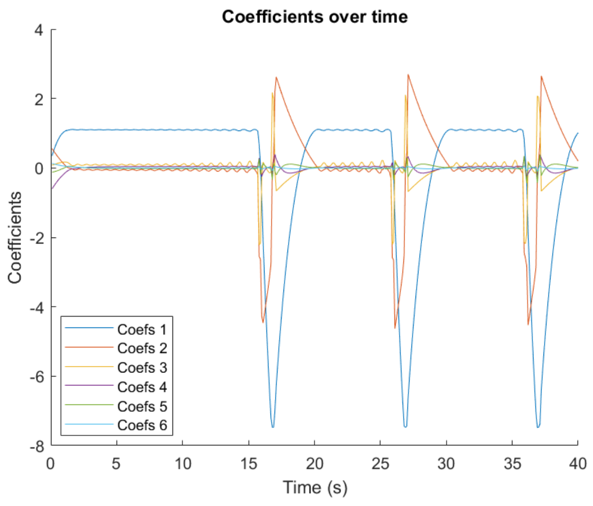


Figure 6.22: *Representation of the coefficients over time obtained in the POD method.*

We represent in Figure 6.22, the coefficients obtained after applying the POD technique.

Total oscillation death. $c = -0.7$

In Figure 6.23, we represent the reduced solution obtained after the application of the POD method to the data set.

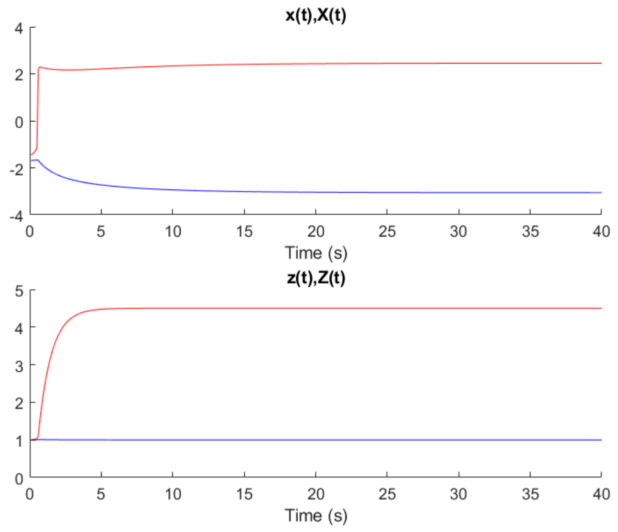


Figure 6.23: *Reduced order solution for the total oscillation death case.*

In Figure 6.24, we represent the relative error of the solutions for $N_{rb} \in \{2, 3, 4\}$ and we see that for $N_{rb} = 4$ a no such great improvement is achieved .This can be due to the static behaviour of the solution.

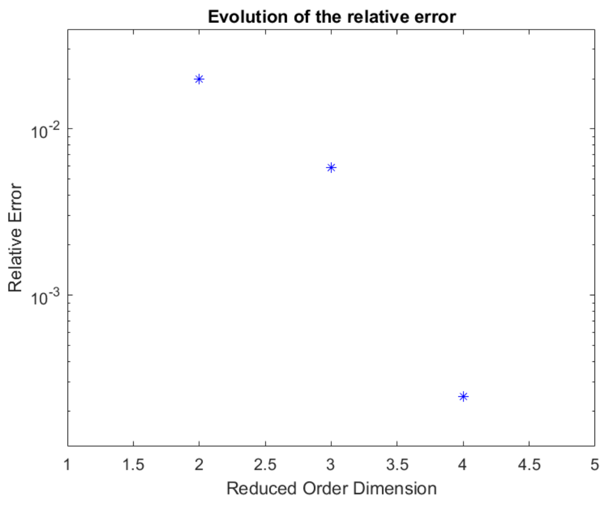


Figure 6.24: *Comparison of the relative error for different reduced order dimensions.*

We represent in Figure 6.25, the coefficients obtained after applying the POD technique.

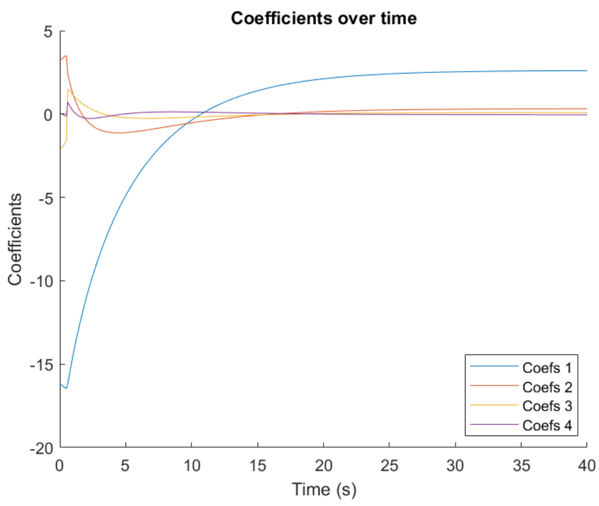


Figure 6.25: *Representation of the coefficients over time obtained in the POD method.*

Relaxation loss. $c = -0.502$

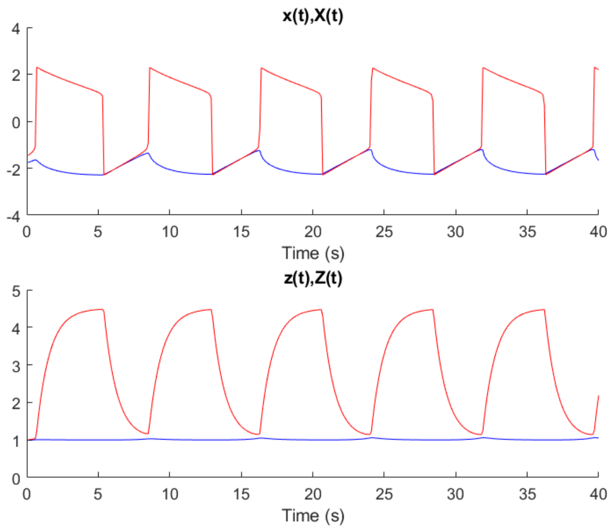


Figure 6.26: *Reduced order solution for the relaxation loss case.*

In Figure 6.26, we represent the reduced solution obtained after the application of the POD method to the data set.

In Figure 6.27, we represent the relative error of the solutions for $N_{rb} \in \{4, 5, 6\}$ and we see that for $N_{rb} = 6$ a great improvement is achieved.

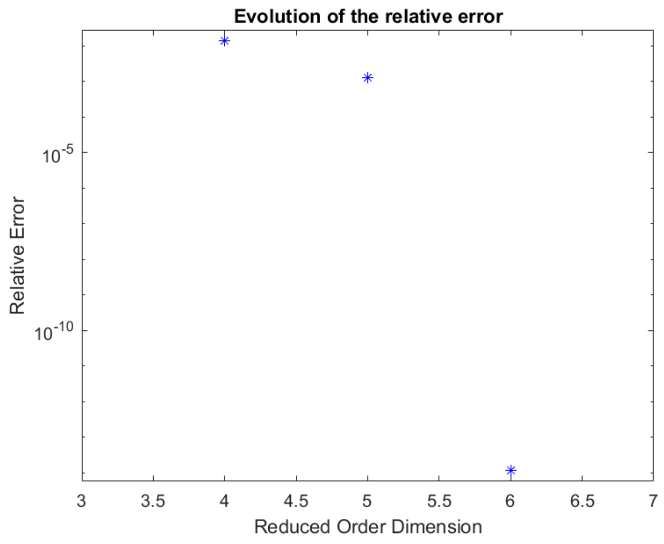


Figure 6.27: *Comparison of the relative error for different reduced order dimensions.*

We represent in Figure 6.28, the coefficients obtained after applying the POD technique.

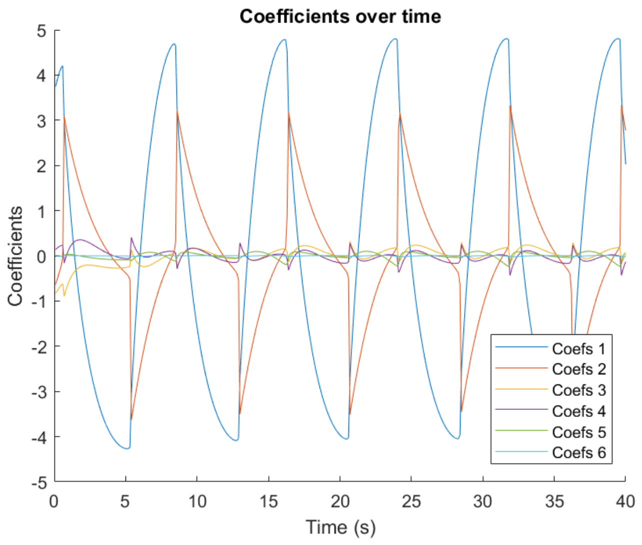


Figure 6.28: *Representation of the coefficients over time obtained in the POD method.*

Antiphase synchronization. $c = -0.25$

In Figure 6.29, we represent the reduced solution obtained after the application of the POD method to the data set.

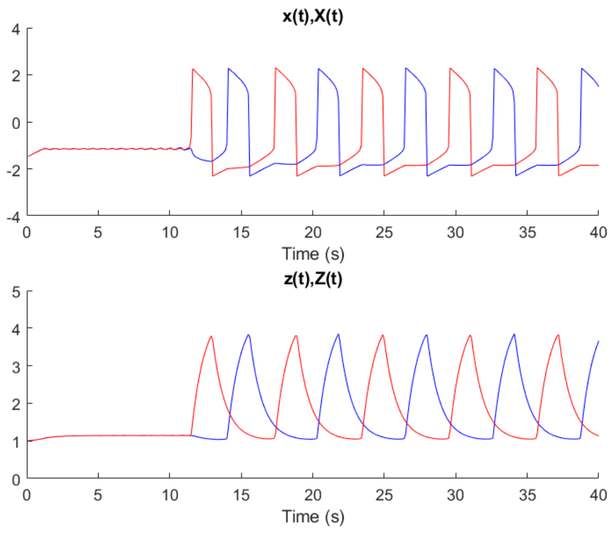


Figure 6.29: *Reduced order solution for the antiphase synchronization case.*

In Figure 6.30, we represent the relative error of the solutions for $N_{rb} \in \{4, 5, 6\}$ and we see that for $N_{rb} = 6$ a great improvement is achieved.

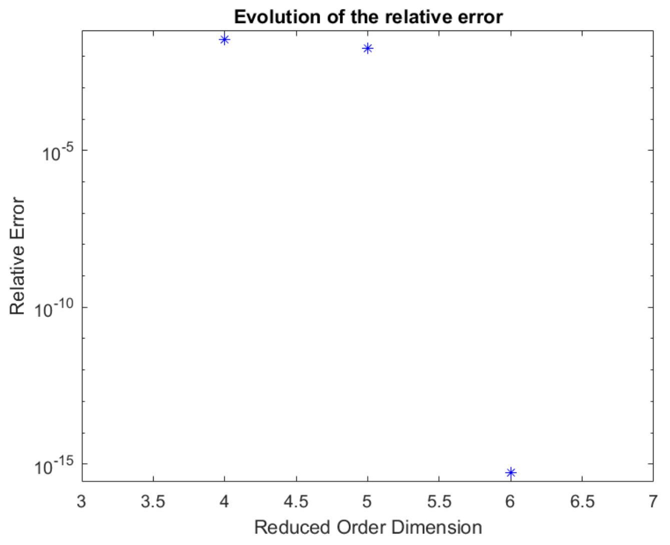


Figure 6.30: *Comparison of the relative error for different reduced order dimensions.*

We represent in Figure 6.31, the coefficients obtained after applying the POD technique.

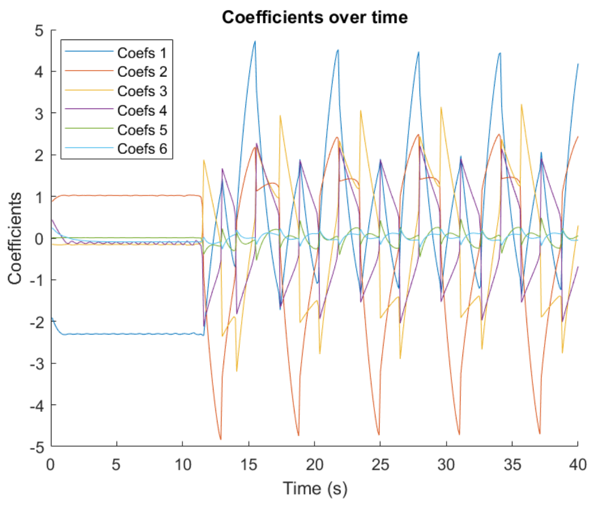


Figure 6.31: *Representation of the coefficients over time obtained in the POD method.*

Almost-in-phase synchronization. $c = 0.1$

In Figure 6.32, we represent the reduced solution obtained after the application of the POD method to the data set.

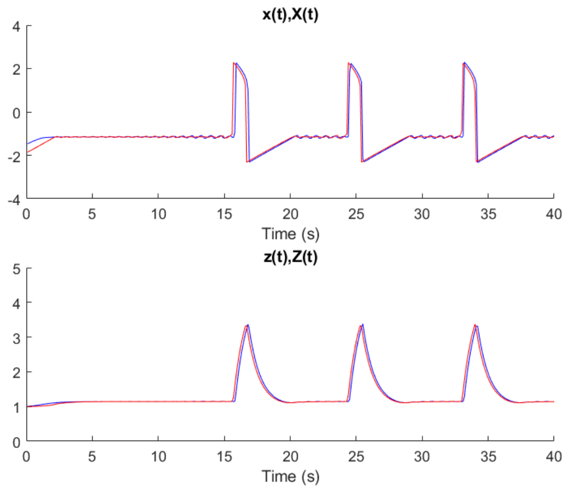


Figure 6.32: *Reduced order solution for the almost-in-phase synchronization case.*

In Figure 6.33, we represent the relative error of the solutions for $N_{rb} \in \{4, 5, 6\}$ and we see that for $N_{rb} = 6$ a great improvement is achieved.

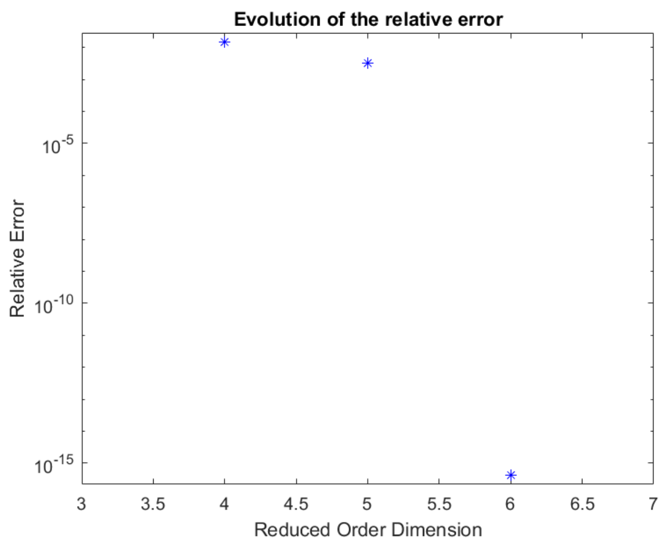


Figure 6.33: *Comparison of the relative error for different reduced order dimensions.*

We represent in Figure 6.34, the coefficients obtained after applying the POD technique.

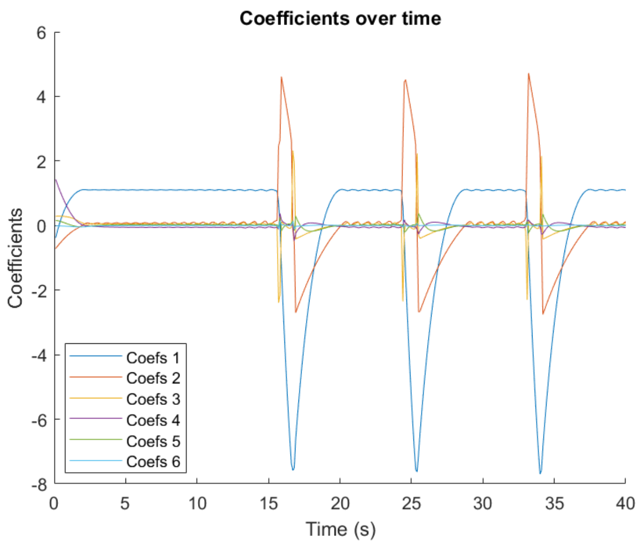


Figure 6.34: *Representation of the coefficients over time obtained in the POD method.*

In-phase locking synchronization. $c = 1$

In Figure 6.35, we represent the reduced solution obtained after the application of the POD method to the data set.

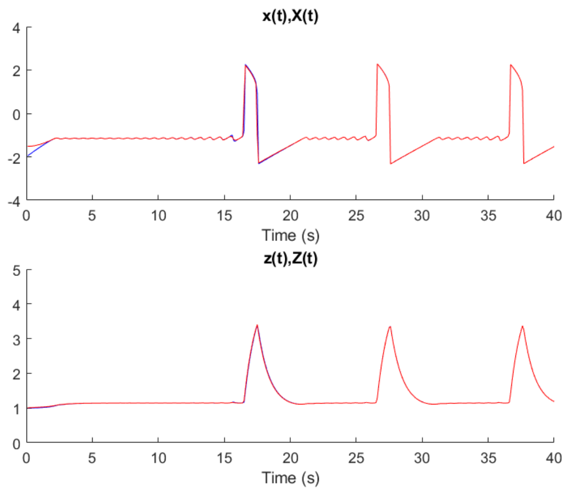


Figure 6.35: *Reduced order solution for the in-phase locking synchronization case.*

In Figure 6.36, we represent the relative error of the solutions for $N_{rb} \in \{4, 5, 6\}$ and we see that for $N_{rb} = 6$ a great improvement is achieved.

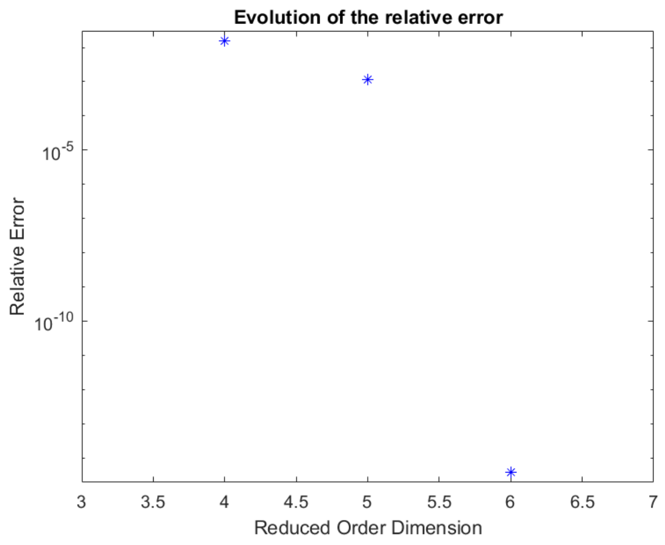


Figure 6.36: *Comparison of the relative error for different reduced order dimensions.*

We represent in Figure 6.37, the coefficients obtained after applying the POD technique.

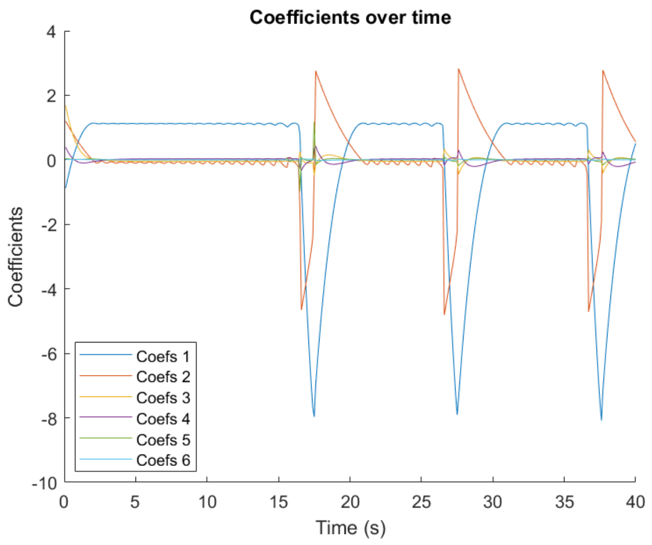


Figure 6.37: *Representation of the coefficients over time obtained in the POD method.*

6.4 Conclusion

In this chapter, we have applied the POD method in order to obtain the best projection subspace that approximates a set of data points. These data points are the evaluation of the solution of a particular ODEs system in a given time.
We have obtained a reduced number of modes or coefficients that represents the main behaviour of the solution over the integration interval, $[t_0, T_{max}]$.
We outline the main results obtained for each problem.

6.4.1 HIRES Problem

We start with the HIRES problem. We have obtained that for a tolerance of $\varepsilon = 10^{-6}$ for the POD method, we obtain a reduced basis of dimension $N_{rb} = 5$. We have represented the values of the reduced solutions in two particular times $t = 0.9$ and $t = 30$ and we have compared them with the solutions obtained by means of two high-precision solvers, forth order Runge-Kutta and RANDAU. In Table 6.3 we can see clearly that the greater the reduced dimension is, the better is the approximation and that for $N_{rb} = 5$ the differences are relatively low.
In Figure 6.14, it can be clearly seen that the relative error drops when we select a reduced order dimension $N_{rb} = 5$. Also, in Figure 6.15 we can see that the modes

tend to zero when t increases. This means that the solution of the problem changes in a much slower rate after $t = 30$, as said in the presentation of the HIRES problem. To sum up, for this model we have reduced a set of a hundred of data points to just five modes that approximates well the solution.

6.4.2 ICC Problem

In the case of the ICC problem we have solved the problem for different values of the coupling parameter.
In the most of the cases we have obtained that for a tolerance of $\varepsilon = 10^{-6}$ the best reduced order dimension is $N_{rb} = 6$ with a great reduction in the relative error, as can be seen in Figures 6.21, 6.27, 6.30, 6.33 and 6.36. In order to obtain a reduced dimension $N_{rb} = 6$ in the total oscillation death case we need to increase the POD tolerance to $\varepsilon = 10^{-12}$ and again a great improvement is obtained when selecting a reduced basis dimension $N_{rb} = 6$. We conclude that we only will need six modes in order to have a great approximation of the solution, no matter what the coupling parameter is.
We also have seen that the reduced solution behaves exactly in the same way as we expected (see [7]). The modes seem to behave with great amplitude oscillations and this can be due to the slow-fast behaviour of the system.

Bibliography

[1] M. Arienti, M. C. Soteriou, *Time-resolved proper orthogonal decomposition of liquid jet dynamics*, Phys. Fluids 21, 112104, 2009.

[2] I. Babuška, *Error-bounds for finite element method*, Numer. Math. 16, 322–333, 1971.

[3] D. Boffi, F. Brezzi, M. Fortin, *Mixed Finite Elements and Applications*, 2013, Springer-Verlag.

[4] F. Brezzi, *On the existence, uniqueness, and approximation of saddle point problems arising from Lagrangian multipliers. R.A.I.R.O.*, Anal. Numér. 2, 129–151, 1974.

[5] F. Chinesta, R. Keunings, A. Leygue, *The Proper Generalized Decomposition for Advanced Numerical Simulations*, 2014, Springer.

[6] M. D'Elia, L. Dedé, A. Quarteroni, *Reduced Basis Method for Parametrized Differential Algebraic Equations*, 2009, Politecnico di Milano.

[7] S. Fernández-García, A. Vidal, *Symmetric coupling of multiple timescale systems with Mixed-Mode Oscillations and synchronization*, Physica D: Nonlinear Phenomena, 401: 1321 29, 2020.

[8] E. Hairer, G. Wanner, *Solving ordinary differntial equations II: Stiff and differential Algebraic problems*, 1987, Springer.

[9] F. Hecht, *FreeFEM Documentation*, (Release 4.6) 2020, Laboratoire Jacques-Louis Lions, Université Pierre et Marie Curie, Paris.

[10] C. Homescu, L. Petzold, R. Serban, *Error Estimation for Reduced Order Models of Dynamical Systems*, SIAM J. Numer. Anal. 43: 4, 1693–1714, 2005.

[11] J. Nečas, *Les Methodes Directes en Theorie des Equations Elliptiques*, 1967, Masson, Paris.

[12] A. Quarteroni, A. Manzoni, F. Negri, *Reduced Order Methods for Partial Differential Equations*, 2016, Springer.

[13] M. Rathinam, L. Petzold, *A New Look at Proper Orthogonal Decomposition*, SIAM J. Numer. Anal. 41: 5, 1893–1925, 2003.

[14] W. Schilders, H. van der Vorst, J. Roomes, *Model Order Reduction*, 2008, Springer.

Notation

General

- $\mathbb{N}$ denotes the set of natural numbers.
- $\mathbb{R}$ denotes the set of real numbers.
- $\mathbb{R}^+$ denotes the set of positive real numbers.
- $\mathbb{R}^d$ is the euclidean space with dimension d.
- $\mathbb{P}_r$ denotes the ring of polynomials of degree up to r.
- Ω is an open subset of $\mathbb{R}^d$.
- $\mathbb{A}^S = \frac{1}{2}(\mathbb{A} + \mathbb{A}^T)$ denotes the symmetric part of the matrix $\mathbb{A}$.
- $\lambda_i(\mathbb{A})$ denotes the i-th eigenvalue of the matrix $\mathbb{A}$.
- $\sigma_i(\mathbb{A})$ denotes the i-th singular value of the matrix $\mathbb{A}$.
- $\mathbb{1}$ is the identity matrix.
- $\mathbb{O}$ is the null matrix.
- $\mathbf{0}$ is the null vector.
- $||\cdot||_2$ denotes the euclidean vector norm.
- $||\cdot||_S$ denotes the spectral matrix norm.
- $||\cdot||_F$ denotes the file matrix norm.
- rank($\mathbb{A}$) denotes the rank of the matrix $\mathbb{A}$,i.e. the dimension of the vector space spanned by its column vectors.
- ker($\mathbb{A}$) denotes the kernel of the matrix $\mathbb{A}$, i.e. the set of solutions to the equation $\mathbb{A}\mathbf{x} = \mathbf{0}$.

- range($\mathbb{A}$) denotes the range of the matrix $\mathbb{A}$, i.e. the span of its column vectors.
- $(\cdot, \cdot)_V$ is the inner product defined over V.
- $\|\cdot\|_V$ is the norm induced by $(\cdot, \cdot)_V$, $\|v\|_V = (v, v)_V^{1/2}$ for any $v \in V$.
- V is a Hilbert space on $\mathbb{R}$ along with its norm $\|\cdot\|_V$.
- V' is the dual space of V, i.e. the space of linear and continuous functionals over V.
- $\langle G, v\rangle = {}_{V'}\langle G, v\rangle_V$, $\forall G \in V', v \in V$ denotes the duality pairing between V' and V.
- $a : V \times V \to \mathbb{R}$ is a bilinear form.
- $f \in V'$ is a linear functional.
- $L^p(\Omega)$ denotes the function space such that $u \in L^p(\Omega) \iff \int_\Omega |u|^p dx < +\infty$.
- $H^1(\Omega)$ denotes the Hilbert space $H^1(\Omega) = \{u \in L^2(\Omega) : \nabla u \in L^2(\Omega)\}$.
- $|\cdot|_{H^1(\Omega)}$ is the norm induced in the Hilbert space $H^1(\Omega)$.
- $H_0^1(\Omega)$ denotes the Hilbert space $H_0^1(\Omega) = \{u \in H^1(\Omega) : u|_{\partial\Omega} = 0\}$.
- $|\cdot|_{H_0^1(\Omega)}$ is the norm induced in the Hilbert space $H_0^1(\Omega)$.
- $h > 0$ is the characteristic discretization parameter.
- $V_h \subset V$ is a finite-dimensional subspace of V, such that $\dim V_h = N_h$.
- $\{\varphi^j\}_{j=1}^{N_h}$ denotes a basis for the finite-dimensional space V_h.
- $\mathbb{A}_h \in \mathbb{R}^{N_h \times N_h}$ is the stiffness matrix, i.e. $(\mathbb{A}_h)_{ij} = a(\varphi^j, \varphi^i)$.
- $\mathbf{f}_h \in \mathbb{R}^{N_h}$ is the vector with components $(\mathbf{f}_h)_i = f(\varphi^i)$.
- $\mathbb{X}_h \in \mathbb{R}^{N_h \times N_h}$ is the symmetric positive definite matrix associated to the scalar product in V, i.e. $(\mathbb{X}_h)_{ij} = (\varphi^i, \varphi^j)_V$.
- u_h denotes the high fidelity solution.

1 Formulation, Analysis and Approximation of Variational Problems

- α denotes the coercivity constant.
- γ denotes the continuity constant.
- β denotes the inf-sup stability constant.
- α_h denotes the discrete coercivity constant.
- γ_h denotes the discrete continuity constant.
- β_h denotes the discrete inf-sup stability constant.
- $\|\cdot\|_a$ is the energy norm, the norm induced by a symmetric bilinear form $a(\cdot,\cdot)$, $\|v\|_a = \sqrt{a(v,v)}$ for any $v \in V$.

2 Reduced Basis Methods

- $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_P)^T$ is the input parameter vector collecting the physical, $\boldsymbol{\mu}_{ph}$, and the geometric ones, $\boldsymbol{\mu}_g$.
- $\mathcal{P}$ is the set of all possible inputs.
- $L(\boldsymbol{\mu}) : V \to V'$ is a second order differential operator, for all $\boldsymbol{\mu} \in \mathcal{P}$.
- $\gamma(\boldsymbol{\mu})$ denotes the continuity factor.
- $\beta(\boldsymbol{\mu})$ denotes the inf-sup stability factor.
- $\mathcal{T}_h$ is a triangulation of Ω.
- $\{\zeta_1, \ldots, \zeta_N\}$ is the reduced basis.
- $V_N = \mathrm{span}(\zeta_1, \ldots, \zeta_N)$ is the reduced space.
- $u_N(\boldsymbol{\mu})$ denotes the reduced solution.
- $r(\boldsymbol{\mu}) = f(\boldsymbol{\mu}) - L(\boldsymbol{\mu})u_N(\boldsymbol{\mu})$ denotes the residual of the high fidelity problem computed on the Reduced Basis solution.
- W_N is the test subspace.
- $\|\cdot\|_{\boldsymbol{\mu}} = \sqrt{a(\cdot,\cdot;\boldsymbol{\mu})}$ denotes the energy norm.

- $\mathbb{A}_N \in \mathbb{R}^{N\times N}$ is reduced the stiffness matrix, i.e. $(\mathbb{A}(\boldsymbol{\mu})_h)_{ij} = a(\zeta^j, \zeta^i; \boldsymbol{\mu})$.
- $\mathbf{f}_N(\boldsymbol{\mu}) \in \mathbb{R}^N$ is the reduced vector with components $(\mathbf{f}_h)_i = f(\zeta^i; \boldsymbol{\mu})$.
- $\theta_a^q(\boldsymbol{\mu})$ are the affine parametric coefficients of $\mathbb{A}_N(\boldsymbol{\mu})$.
- Q_a denotes the number of affine parametric coefficients of $\mathbb{A}_N(\boldsymbol{\mu})$.
- $\theta_f^q(\boldsymbol{\mu})$ are the affine parametric coefficients of $\mathbf{f}_N(\boldsymbol{\mu})$.
- Q_f denotes the number of affine parametric coefficients of $\mathbf{f}_N(\boldsymbol{\mu})$.
- $\mathbb{V} \in \mathbb{R}^{N_h\times N}$ is the transformation matrix, i.e. $(\mathbb{V})_{jm} = \zeta_m^j$.
- $\kappa(\mathbb{A})$ denotes the spectral condition number of the matrix $\mathbb{A}$.
- $R_{V_h}^{-1}$ denotes the inverse of the Riesz map.
- n_{vv}: number of operations required to compute a scalar product between two N_h-dimensional vectors.
- n_{mv}: number of operations for a matrix vector product.
- n_{ls}: number of operations to solve a linear system ($\mathbb{X}_h\mathbf{x} = \mathbf{y}$).
- $e_h(\boldsymbol{\mu}) = u_h(\boldsymbol{\mu}) - u_N(\boldsymbol{\mu}) \in V_h$ denotes the error between the high fidelity and reduced solutions.
- $\Delta_N(\boldsymbol{\mu}) = \|r(\cdot;\boldsymbol{\mu})\|_{V_h'}/\beta_h(\boldsymbol{\mu})$ denotes the error estimate.
- $\varsigma_N(\boldsymbol{\mu}) = \Delta_N(\boldsymbol{\mu})/\|e_h(\boldsymbol{\mu})\|_V$ denotes the associate effectivity factor.
- $(\cdot,\cdot)_{\mathbb{X}_h}$ denotes the $\mathbb{X}_h$-scalar product, i.e. $(v,w)_{\mathbb{X}_h} = \mathbf{w}^T\mathbb{X}_h\mathbf{v}$.
- $\|\cdot\|_{\mathbb{X}_h}$ denotes the $\mathbb{X}_h$-norm.
- $c_{q_1,q_2} = \mathbf{f}_h^{q_1 T}\mathbb{X}_h^{-1}\mathbf{f}_h^{q_2} \in \mathbb{R}$.
- $\mathbf{d}_{q_1,q_2} = \mathbb{V}^T\mathbb{A}_h^{q_1 T}\mathbb{X}_h^{-1}\mathbf{f}_h^{q_2} \in \mathbb{R}^N$.
- $\mathbb{E}_{q_1,q_2} = \mathbb{V}^T\mathbb{A}_h^{q_1 T}\mathbb{X}_h^{-1}\mathbb{A}_h^{q_2}\mathbb{V} \in \mathbb{R}^{N\times N}$.
- $\beta_h^{LB}(\boldsymbol{\mu})$ denotes a parameter-dependent lower bound to $\beta_h(\boldsymbol{\mu})$.
- $\Xi_{fine} \subset \mathcal{P}$ denotes a sample set whose dimension is sufficiently large.
- Ξ_I denotes a set of parameters, called, interpolation points.
- $\beta_I(\boldsymbol{\mu})$ denotes an interpolatory approximation of $\beta_h(\boldsymbol{\mu})$.

3 Construction of Reduced Basis Spaces

- $\boldsymbol{\zeta}_i$ denotes the left singular vectors of a matrix.
- $\boldsymbol{\Phi}_i$ denotes the right singular vectors of a matrix.
- $\mathbb{U} \in \mathbb{R}^{m\times m}$ denotes the matrix with columns $\mathbb{U} = [\boldsymbol{\zeta}_1|\dots|\boldsymbol{\zeta}_m]$.
- $\mathbb{Z} \in \mathbb{R}^{n\times n}$ denotes the matrix with columns $\mathbb{Z} = [\boldsymbol{\Phi}_1|\dots|\boldsymbol{\Phi}_m]$.
- $\Sigma = \operatorname{diag}(\sigma_1,\dots,\sigma_p) \in \mathbb{R}^{m\times n}$ and $\sigma_1 \geq \dots \geq \sigma_p \geq 0$ for $p = \min(m,n)$.
- $\mathbb{A}_k$ denotes the best approximation matrix with $\operatorname{rank}(\mathbb{A}_k) \leq k$.
- n_s denotes the number of parameter samples.
- $\Xi_S = \{\boldsymbol{\mu}_1\dots,\boldsymbol{\mu}_{n_s}\}$ is the set of parameter samples.
- $\mathbb{S} \in \mathbb{R}^{N_h\times n_s}$ denotes the snapshot matrix, i.e. $\mathbb{S} = [\mathbf{u}_1|\dots|\mathbf{u}_{n_s}]$.
- $\mathbb{C} = \mathbb{S}^T\mathbb{S} \in \mathbb{R}^{n_s\times n_s}$ is the correlation matrix, its elements are given $(\mathbb{C})_{ij} = \mathbf{u}_i^T\mathbf{u}_j,\ 1 \leq i,j \leq n_s$.
- $\mathbb{V} = [\boldsymbol{\zeta}_1|\dots|\boldsymbol{\zeta}_N] \in \mathbb{R}^{N_h\times N}$ is the POD basis.
- $\mathbb{W} \in \mathbb{R}^{N_h\times N}$ denotes a possible N-dimensional orthonormal basis.
- $\mathcal{V}_N = \{\mathbb{W} \in \mathbb{R}^{N_h\times N} : \mathbb{W}^T\mathbb{W} = \mathbb{1}_N\}$.
- $\Pi_{\mathbb{W}}\mathbf{x} = \displaystyle\sum_{j=1}^{N}(\mathbf{x},\mathbf{w}_j)_2\mathbf{w}_j = \mathbb{W}\mathbb{W}^T\mathbf{x}$ is the projection of a vector $\mathbf{x} \in \mathbb{R}^{N_h}$ onto $\operatorname{span}(\mathbb{W})$.
- ε_{POD} denotes a tolerance.
- $I(N) = \dfrac{\sum_{i=1}^{N}\sigma_i^2}{\sum_{i=1}^{r}\sigma_i^2}$ represents the percentage of energy of the snapshots captured by the first N POD modes.
- $\mathcal{V}_N^{\mathbb{X}_h} = \{\mathbb{W} \in \mathbb{R}^{N_h\times N} : \mathbb{W}^T\mathbb{X}_h\mathbb{W} = \mathbb{1}_N\}$.
- $\Pi_{\mathbb{W}}^{\mathbb{X}_h}\mathbf{x} = \sum_{j=1}^{N}(\mathbf{x},\mathbf{w}_j)_{\mathbb{X}_h}\mathbf{w}_j = \mathbb{W}\mathbb{W}^T\mathbb{X}_h\mathbf{x}$.
- $\widetilde{\mathbb{S}} = \mathbb{X}_h^{1/2}\mathbb{S}$.
- $\widetilde{\boldsymbol{\zeta}}_i = \mathbb{X}_h^{1/2}\boldsymbol{\zeta}_i$.

- $\mathcal{M}_h = \{\mathbf{u}_h(\boldsymbol{\mu})\ \boldsymbol{\mu} \in \mathcal{P}\}$ is solution set.
- $\mathcal{M}_h^S = \{\mathbf{u}_h(\boldsymbol{\mu}_1), \dots, \mathbf{u}_h(\boldsymbol{\mu}_{n_s})\}$ is the set solution snapshots.
- $T : L^2(\mathcal{P}) \to \mathbb{R}^{N_h}$ is the continuous analogue to the snapshots matrix $\mathbb{S}$.
- $T^* : \mathbb{R}^{N_h} \to L^2(\mathcal{P})$ is the continuous analogue of the matrix $\mathbb{S}^T$.
- $\| \cdot \|_{H-S}$ is the Hilbert-Schmidt operator norm.
- $C = T^*T : L^2(\mathcal{P}) \to L^2(\mathcal{P})$ is the continuous analogue of the correlation matrix $\mathbb{C} = \mathbb{S}^T\mathbb{S}$
- $K = TT^* : \mathbb{R}^{N_h} \to \mathbb{R}^{N_h}$ is the continuous analogue of the correlation matrix $\mathbb{K} = \mathbb{S}\mathbb{S}^T$.
- T_N is the best rank N approximation to the operator T.
- $\mathcal{B}_N = \{B \in L(L^2(\mathbb{P}); \mathbb{R}^{N_h}) : \text{rank}(B) \leq N\}$.
- $w_i > 0$ are the weighting coefficients.
- $\mathbb{D} = \text{diag}(w_1, \dots, w_{n_s}) \in \mathbb{R}^{n_s \times n_s}$.
- $\widehat{\mathbb{S}} = \mathbb{S}\mathbb{D}^{1/2}$.
- $\widehat{\boldsymbol{\Phi}}_i$ are the eigenvectors of $\widehat{S}^T\widehat{S}$.
- $\mathbb{V}^\infty$ denotes the POD basis solution of the continuous minimization problem (3.7).
- $\mathbb{V}^{n_s}$ denotes the POD basis solution of the discrete minimization problem (3.9).
- $\mathcal{E}(\mathbb{W}) = \displaystyle\int_{\mathcal{P}} \|\mathbf{u}_h(\boldsymbol{\mu}) - \mathbb{W}\mathbb{W}^T\mathbf{u}_h(\boldsymbol{\mu})\|_2^2 d\boldsymbol{\mu}$.
- $\mathcal{E}^S(\mathbb{W}) = \displaystyle\sum_{i=1}^{n_s} \|\mathbf{u}_h(\boldsymbol{\mu}) - \mathbb{W}\mathbb{W}^T\mathbf{u}_h(\boldsymbol{\mu})\|_2^2$.
- E_S denotes the quadrature or sampling error.
- $\Xi_{train} \subset \mathcal{P}$ denotes a training sample set.
- n_{train} denotes the cardinality of the training sample set Ξ_{train}.
- N_{max} is the maximum number of iterations for the greedy algorithm 10.
- ε_g denotes the fixed tolerance for the greedy procedure.
- $\delta_N = \max_{\boldsymbol{\mu}}$ is the maximum of error estimators over Ξ_{train}.

4 & 5 PDE Application

- ∇ denotes the laplacian operator.
- $\dfrac{\partial u}{\partial n}$ denotes the normal derivative of the function u.
- χ_Ω is the characteristic function of the domain Ω.
- Ω_1 denotes the inner subdomain $\Omega_1 = (1/4, 3/4) \times (1/4, 3/4)$.
- $\partial\Omega$ denotes the boundary of the domain Ω.
- $\Gamma_D = (0, 1) \times \{0\}$ is the part of the boundary with Dirichlet boundary condition.
- $\Gamma_N = \partial\Omega/\Gamma_D$ is the part of the domain with Newmann boundary condition.
- μ_1 is the characteristic parameter of Ω_1,
- μ_f is the intensity of the source in Ω_1,
- μ_N represents the flux over the other walls of the domain Γ_N.
- nn is the number of divisions of the boundary for the construction of the mesh.
- t_{RB} is the computational time spent in the assembling and solving of the reduced order model.
- t_{HF} is the computational time spent in the assembling and solving of the high-fidelity model.
- ρ is the ratio of t_{HF} to t_{RB} and ρ^* is the low boundary of ρ.

6 ODE Application

- $\{y(t_1), \ldots, y(t_{N_{snap}})\}$, where $y \in \mathbb{R}^n$ is the set of data points.
- $\bar{y} = \dfrac{1}{N_{snap}} \displaystyle\sum_{i=1}^{Ns} y_i$, is the mean of the data points.
- $C_{i,j} = (y_i - \bar{y}, y_j - \bar{y})$ is the correlation matrix, where $\tilde{y}_i = y_i - \bar{y}$.
- $\phi_i \in \{\phi_1, ..., \phi_{N_{rb}}\}$ is the i-th mode.
- $a_{r,i} = (\tilde{y}_i, \phi_r)$ are the temporal coefficients.
- $c \in [-1, 1]$ is the coupling parameter.

List of Algorithms