Nonlinear compressive reduced basis approximation for parametric PDE's

Hassan Ballout

September 4, 2024

University of Strasbourg

CSMI Master's Program

Supervisors: *Yvon Maday and Christophe Prud'homme*

Contents

1 Introduction

1.1 Model Order Reduction

In the world of computational science, engineering, and industry, the significance of numerical simulation is ever increasing. In particular, numerical simulations of partial differential equations (PDEs) are widely used in many fields such as fluid dynamics, structural mechanics, aerodynamics, and many others. For these types of problems, multiple numerical methods could be employed such as finite element method (FEM), finite volume method (FVM), finite difference method (FDM), spectral methods, etc. Regarded as high-fidelity (or full-order) techniques, this category of methods typically achieves superior accuracy but at the expense of computational intensity, involving up to $O(10^6 - 10^9)$ degrees of freedom and multiple hours (or even days) of CPU time in some cases as highlighted in [15]. This high computational cost is a major obstacle for a wide range of applications that require multiple simulations such as PDE-constrained optimization, inverse problems, uncertainty quantification, etc. In this context, model order reduction (MOR) techniques have been developed to address this issue. The main idea behind MOR is to reduce the dimension of the full-order model (FOM) while preserving the accuracy.

1.2 General context: Reduced basis method for parametric PDEs

In this work, we will focus on the reduced basis method (RBM), which is a MOR technique that is applied to a specific yet very rich class of problems, namely parameter-dependent or parametric PDEs, where the parameters may represent geometrical, physical, or boundary conditions of the problem. Mathematically, a parametric PDE can be written as:
For a given parameter $\mu \in \mathcal{P} \in \mathbb{R}^P$ (the parameter space), find $u(\mu) \in X$ such that:

$$\mathcal{R}(u(\mu); \mu) = 0 \tag{1}$$

where X is some appropriate Hilbert space, chosen for the above problem to be well-posed.
As a linear model reduction technique, the classical RBM aims to find a low-dimensional subspace that approximates the set of solutions of the parametric PDE or what is called the solution manifold in the high-dimensional (full-order) space. In addition to its low dimensionality property, a fundamental difference between the RBM and the FEM (or other multipurpose discretization methods) is that the latter can be used for any type of problem, while the former is problem-dependent, i.e., the features of the problem are learned while constructing the reduced basis space which makes it unuseful for other problems [13]. In practice, RBM is based on a two phases approach:

- **Offline phase**: A computationally expensive phase where the learning of the solution manifold is done.
- **Online phase**: A much cheaper phase where the solution $u(\mu)$ is computed for any new parameter value μ.

In the offline phase, a reduced basis space X_N is constructed, typically using algorithms such as Proper Orthogonal Decomposition (POD) or Greedy method [15]. These algorithms identify a quasi-optimal basis as an alternative to the out-of-reach optimal one. Then, provided that such a quasi-optimal X_N has been identified, a "reduced" basis method can be deployed in the online phase under the form, e.g. of a Galerkin formulation in X_N for the approximation of the solution $u(\mu)$ for some given value μ of the parameter [13]. The online cost of RBM scales like $\mathcal{O}(N^3)$ which makes it very efficient for small values of N. Here and throughout the remainder of this report, an RBM with the traditional linear approximation is referred to as a classical RBM.

1.3 Specific context

1.3.1 Kolmogorov barrier

Linear reduced basis approximation techniques and more specifically the classical RBM rely on the assumption that the solution manifold is well approximated by a low-dimensional subspace. This leads to the notion of Kolmogorov N-width of a set $\mathcal{M}$, denoted $d_N(\mathcal{M})$, which indicates how well $\mathcal{M}$ can be approximated by the best N-dimensional linear subspace. In the context of parametric PDEs, $\mathcal{M} = \{u(\mu) | \mu \in \mathcal{P}\}$ is the solution manifold and we are interested in the decay of $d_N(\mathcal{M})$ as N increases. For many problems, elliptic PDEs for instance, the decay of $d_N(\mathcal{M})$ is exponential which makes the classical RBM very efficient. However, for other classes of problems, typically hyperbolic transport equations with parameter-dependent shock positions, the decay of $d_N(\mathcal{M})$ is very slow. This slow decay alongside the optimality (by definition) of the Kolmogorov width constitutes a barrier for the classical RBM which is called the Kolmogorov barrier. In practice, this barrier is manifested by the need for a large number of basis functions N to achieve a given accuracy which makes the classical RBM less efficient for this type of problem. Additionally, the curse of dimensionality of both the parameter space and the physical space or domain over which the PDE is defined can exacerbate the Kolmogorov barrier.

1.3.2 Nonlinear approximation

In the previous section 1.3.1, we discussed the Kolmogorov barrier and stated that even if the set of parameters lies in a space of small dimension $\mathcal{P} \in \mathbb{R}^P$ ($P \in \mathbb{N}$ small), the dimension N of X_N is not necessarily small. This mismatch between the dimension of the reduced basis space and the dimension of the solution manifold $\mathcal{M}$ (at most of dimension P) is due to the linearity of the first concept while the dependency of the solution $u(\mu)$ on the parameter μ is most often highly non-linear.

A lot of research has been done trying to mitigate the Kolmogorov barrier using nonlinear approximation techniques. There are essentially 2 wide classes of approaches, Lagrangian and Eulerian. Lagrangian approaches (also called registration methods see [16]) involve a map ϕ_μ so that the set of all $v(\mu) = u_\mu \circ \phi_\mu$ defines a manifold better suited to linear approximation methods than the original $\mathcal{M}$ with thus a faster decay of the Kolmogorov width. Examples of Lagrangian approaches have been proposed in [10, 17, 7]. Eulerian approaches have a more nonlinear root: some approaches are based on a partition of the parameter domain in an adaptive way [6], others rely on basis updates [12], or finally based on artificial learning methods [8, 11]. In these last references, the original notion of "learning" of the reduced basis methods is hence pushed further to the extent of investigating the nonlinearity of the manifold. This learning process can be used to the point of not even referring to the problem (1) and at the end propose a method that, to any input parameter $\mu \in \mathcal{P}$, proposes as an output an approximation to $u(\mu)$. Even if this is shown to work in those publications, this approach has some obvious drawbacks :

- the size of the database for the learning stage, needs to be quite large in order to apprehend the complexity of $\mathcal{M}$.
- there is no evaluation of the residual of equation (1) that would allow to state if the problem is well approximated or not and what to do in order to improve a non-sufficient accuracy.

this is why, following [1], the authors in [2, 5] have proposed to "help" the statistical approaches by using intermediate models and still ground the approximation of the approximated resolution of problem (1). This method is called the nonlinear compressive reduced basis method.

1.4 Overview of the project

After introducing the context of the project in sections [1.1,1.2,1.3], we will begin by presenting the test case we will be working on. Next, we will provide a detailed explanation of the classical RBM, covering both theoretical and algorithmic aspects, and present some numerical results on the test case. Following this, we will introduce the nonlinear compressive reduced basis method, discussing its results and practical issues. Finally, we will compare the use of a quadratic model in the nonlinear compressive reduced basis method with the quadratic method proposed in [1] and propose an enhancement to this last method. The implementation details of all the mentioned methods will be covered in a separate section.

2 Problem setting: the multiparameter problem

2.1 Heat transfer in a thermal fin

We consider the problem of designing a thermal fin to effectively remove heat from a surface. The two-dimensional fin, shown in Figure 1, consists of a vertical central **post** and say N_f horizontal **subfins**; the fin conducts heat from a prescribed uniform flux source at the root, $\mathbf{\Gamma}_{\text{root}}$, through the large-surface-area subfins to surrounding flowing air. The fin is characterized by a (N_f+1)-component parameter vector $\mu = (\mu_1, \mu_2, \ldots, \mu_{N_f+1})$, where $\mu_i = k^i$, $i = 1, \ldots, N_f$, and $\mu_{N_f+1} = Bi$; μ may take on any value in a specified design set $\mathcal{P} \subset \mathbb{R}^P$, where $P = N_f + 1$.

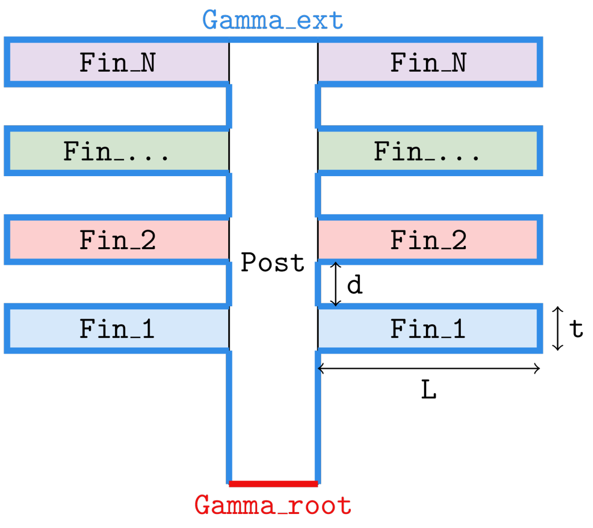

Figure 1: Thermal fin geometry.

Here k^i is the thermal conductivity of the i-th subfin (normalized relative to the post conductivity $k^0 = 1$); and Bi is the Biot number, a nondimensional heat transfer coefficient reflecting convective transport to the air at the fin surfaces (larger Bi means better heat transfer). For example, suppose we choose a thermal fin with $N_f = 4$, $k^1 = 0.4$, $k^2 = 0.6$, $k^3 = 0.8$, $k^4 = 1.2$, and $Bi = 0.1$; for this particular configuration $\mu = (0.4, 0.6, 0.8, 1.2, 0.1)$, which corresponds to a single point in the set of all possible configurations $\mathcal{P}$ (the parameter or design set).

We are interested in the design of this thermal fin, and we thus need to look at certain outputs or cost-functionals of the temperature as a function of μ. We choose for our output T_{root}, the average steady-state temperature of the fin root normalized by the prescribed heat flux into the fin root. The particular output chosen relates directly to the cooling efficiency of the fin — lower values of T_{root} imply better thermal performance. The steady-state temperature distribution within the fin, $u(\mu)$, is governed by the elliptic partial differential equation

$$-k^i \Delta u = 0 \text{ in } \Omega^i,\ i = 0, \dots, N_f, \tag{2}$$

where Δ is the Laplacian operator, and u_i refers to the restriction of u to Ω^i. Here Ω^i is the region of the fin with conductivity k^i, $i = 0, \dots, N_f$; Ω^0 thus the central post, and Ω^i, $i = 1, \dots, N_f$, corresponds to the subfins. The entire fin domain is denoted $\Omega = \bigcup_{i=0}^{N_f} \Omega^i$; the boundary is denoted $\mathbf{\Gamma}$. We must also ensure continuity of temperature and heat flux at the conductivity-discontinuity interfaces $\Gamma^i_{int} \equiv \partial\Omega^0 \cap \partial\Omega^i$, $i = 1, \dots, N_f$, where $\partial\Omega^i$ denotes the boundary of Ω^i, we have on Γ^i, $i = 1, \dots, N_f$:

$$\begin{aligned} [u] &= 0, \\ \left[k^i \frac{\partial u}{\partial n^i} \right] &= 0, \end{aligned} \tag{3}$$

where n^i is the outward normal on $\partial\Omega^i$. Finally, we introduce a Neumann flux boundary condition on the fin root:

$$-k^0 \frac{\partial u}{\partial n^0} = -1 \text{ on } \Gamma_{\text{root}}. \tag{4}$$

which models the heat source; and a Robin boundary condition:

$$-k^i \frac{\partial u}{\partial n^i} = Bi\, u^i \text{ on } \Gamma^i_{\text{ext}},\ i = 1, \dots, N_f, \tag{5}$$

which models the convective heat losses. Here Γ^i_{ext} is that part of the boundary of Ω^i exposed to the flowing fluid; note that $\bigcup_{i=0}^{N_f} \Gamma^i_{\text{ext}} = \Gamma \setminus \Gamma_{\text{root}}$. The average temperature at the root, $T_{\text{root}}(\mu)$, can then be expressed as $l(u(\mu))$, where

$$l(v) = \int_{\Gamma_{\text{root}}} v. \tag{6}$$

2.2 Weak formulation

The weak or variational formulation of the problem (2) is essential to proceed with a Galerkin-type approximation. To derive the weak form, we multiply (2) by a test function $v \in X = H^1(\Omega)$ and integrate over each subdomain:

$$\int_{\Omega^i} -k^i \Delta u\, v = 0, \quad i = 0, \dots, N_f,$$

Using Green's formula, we get:

$$\int_{\Omega^i} k^i (\nabla u^i, \nabla v) dx - \int_{\partial\Omega^i} k^i (\nabla u^i \cdot \mathbf{n}^i) v\, ds = 0 \quad \text{for } i = 0, \dots, N_f$$

Since $\partial\Omega^i = \Gamma^i_{int} \cup \Gamma^i_{ext}$ for $i = 1, \dots, N_f$, using equations (3) and (5), we get:

$$\int_{\Omega^i} k^i (\nabla u^i, \nabla v) dx - \int_{\Gamma^i_{int}} (\nabla u^i \cdot \mathbf{n}^i) v\, ds + B_i \int_{\Gamma^i_{ext}} u^i v\, ds = 0$$

While for $i = 0$, we have $\partial\Omega^0 = \cup_{i=1}^{N_f} \Gamma^i_{int} \cup \Gamma^0_{ext} \cup \Gamma_{root}$, using equations (3), (4) and (5), we get:

$$\int_{\Omega^0} k^0 (\nabla u^0, \nabla v) dx + \sum_{i=1}^{N_f} \int_{\Gamma^i_{int}} (\nabla u^0 \cdot \mathbf{n}^i) v\, ds + \int_{\Gamma^0_{ext}} Bi\, u^0 v\, ds - \int_{\Gamma_{root}} v\, ds = 0$$

Adding the previous equations, we finally get the weak form of the problem: For a given $\mu \in \mathcal{P}$, find $u(\mu) \in X$ such that:

$$\overbrace{\sum_{i=0}^{N_f} \int_{\Omega^i} k^i(\nabla u(\mu), \nabla v) dx + B_i \int_{\Gamma_{ext}} u(\mu)\, v\, ds}^{a(u(\mu),v;\mu)} = \overbrace{\int_{\Gamma_{root}} v\, ds}^{f(v;\mu)}, \quad \forall v \in X \tag{7}$$

We can see that the bilinear form $a(\cdot, \cdot; \mu)$ is symmetric, coercive and continuous, and the linear form $f(\cdot; \mu)$ is continuous. This makes the problem (7) well-posed for any $\mu \in \mathcal{P}$ by the Lax-Milgram theorem. Additionally, $l(v) = f(v)$ for this problem, which makes it a compliant problem.

3 Classical Reduced Basis Method

In this section, we will present the classical RBM. The explanation will begin with general notations and principles, followed by a discussion of the specific application to our test case after each part. We begin by introducing the mathematical formulation of interest, followed by the presentation of the truth or high-fidelity problem. Next, we introduce the reduced basis approximation and discuss some practical aspects and numerical results. Finally, we present the limitations of this method and the motivation for the nonlinear compressive reduced basis method.

3.1 Mathematical formulation

In the context of the reduced basis method, we are specifically interested in parametric PDEs. More precisely, we consider the following steady parametric variational problem:
Let X be a Hilbert space and $\mathcal{P} \subset \mathbb{R}^P$. Given $\mu \in \mathcal{P}$, find $u(\mu) \in X$ such that:

$$a(u(\mu), v; \mu) = f(v; \mu), \quad \forall v \in X \tag{8}$$

and evaluate:

$$s(\mu) = l(u(\mu); \mu) \tag{9}$$

where $a(.,.;\mu)$ is bilinear form, $f(.;\mu)$ and $l(.;\mu)$ are linear forms, and $s(\mu)$ is an output of interest. In this project we restrict ourselves to the simple compliant case where we assume in addition that:

- $l(\cdot; \mu) = f(\cdot; \mu), \quad \forall \mu \in \mathcal{P}$
- the bilinear form $a(\cdot, \cdot; \mu)$ is symmetric for any $\mu \in \mathcal{P}$

These assumptions simplify considerably the formulation and give multiple convergence and error control properties as we will see later. For the well-posedness of the problem(8) for any μ, we assume that:

- $a(.,.;\mu)$ is coercive and continuous for all $\mu \in \mathcal{P}$ with respect to the norm $\|.\|_X$, i.e. there exists $\alpha(\mu) > \alpha > 0$ and $\gamma(\mu) < \gamma < \infty$ such that:

$$a(u, u; \mu) \geq \alpha(\mu)\|u\|_X^2 \quad \text{and} \quad a(u, v; \mu) \leq \gamma(\mu)\|u\|_X\|v\|_X, \quad \forall u, v \in X \tag{10}$$

- $f(\cdot; \mu)$ is continuous with respect to the norm $\|.\|_X$, i.e. there exists $\delta(\mu) < \infty$ such that:

$$f(v; \mu) \leq \delta(\mu)\|v\|_X, \quad \forall v \in X \tag{11}$$

The well-posedness of the problem (8) for any μ is then guaranteed by the Lax-Milgram theorem. From 2.2 we can see that our test case satisfies all the previous assumptions.

3.2 High-fidelity model

Since the exact solution $u(\mu)$ of the problem (8) is not analytically available in general, we approximate it using a high fidelity or truth approximation method such as FEM, FVM, FDM, spectral methods, etc. These methods typically achieve high accuracy but at the expense of high computational cost.

For our elliptic problem, FEM is the most suitable method that allows us to approximate accurately the exact solution $u(\mu)$. The FEM start by discretizing the domain Ω into a mesh of N_{el} non-overlapping elements K_i such that:

$$\bar{\Omega} = \cup_{i=1}^{N_{el}} K_i, \quad \overset{\circ}{K_i} \cap \overset{\circ}{K_j} = \emptyset, \quad i \neq j$$
$$\mathcal{T}_h = \{K_i\}_{i=1}^{N_{el}}$$

and we define the mesh size $h = \max_{K_i \in \mathcal{T}_h} \max_{x,y \in K_i} \|x - y\|$. In figure 2, we show an example of meshing the domain Ω for $N_f = 5$ and $N_{el} = 43622$ elements. This mesh is automatically generated in Feel++ [14] using Gmsh.

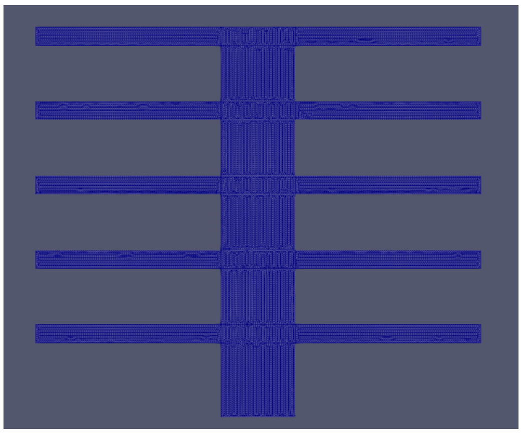

Figure 2: Mesh for $N_f = 5$ and 43622 elements

Then one can define a discrete finite dimensional space $X_h \in X$. As an example, X_h can be constructed based on piece-wise polynomial basis functions. We denote N_h the dimension of X_h also called the number of degrees of freedom (DOF), and $\{\xi_i\}_{i=1}^{N_h}$ a basis of X_h. The FEM approximation $u_h(\mu)$ is an element of X_h and is written as:

$$u_h(\mu) = \sum_{i=1}^{N_h} (u_h^\mu)_i \ \xi_i$$

We consider a Galerkin approach where the test functions are also in X_h. Then the discrete problem is written as follows:
find $u_h(\mu) \in X_h$ such that:

$$a(u_h(\mu), v_h; \mu) = f(v_h; \mu), \quad \forall v_h \in X_h \tag{12}$$

and evaluate:

$$s_h(\mu) = l(u_h(\mu); \mu) \tag{13}$$

By choosing the test functions v_h in (12) to be the basis functions ξ_i, we can rewrite 12:

$$a(\sum_{j=1}^{N_h} (u_h^\mu)_j \ \xi_j, \xi_i; \mu) = f(\xi_i; \mu), \quad \forall i = 1, \ldots, N_h$$

$$\iff \sum_{j=1}^{N_h} (u_h^\mu)_j \ a(\xi_j, \xi_i; \mu) = f(\xi_i; \mu), \quad \forall i = 1, \ldots, N_h$$

Finally, we can write the discrete problem in matrix form:

$$\mathbf{A}_h^\mu \mathbf{u}_h^\mu = \mathbf{f}_h^\mu \tag{14}$$

where $\mathbf{A}_h^\mu = \{a(\xi_j, \xi_i; \mu)\}_{i,j=1}^{N_h}$ is the stiffness matrix and $\mathbf{f}_h^\mu = \{f(\xi_i; \mu)\}_{i=1}^{N_h}$. The output $s_h(\mu)$ can then be evaluated as:

$$s_h(\mu) = (\mathbf{u}_h^\mu)^T \mathbf{f}_h^\mu \tag{15}$$

The coercivity, continuity, and the Galerkin orthogonality give the following error estimate:

$$\|u(\mu) - u_h(\mu)\|_X \leq \frac{\gamma(\mu)}{\alpha(\mu)} \inf_{v_h \in X_h} \|u(\mu) - v_h\|_X \tag{16}$$

This implies that the FEM approximation $u_h(\mu)$ is related to the best approximation of $u(\mu)$ in X_h through the coercivity and continuity constants.

This high fidelity approximation is supposed to be accurate enough i.e. $\|u(\mu) - u_h(\mu)\|_X$ can be made arbitrarily small by refining the mesh $(h \to 0)$. This accuracy even if it comes in general at a high computational cost as N_h is usually large, is necessary, as the RBM approximates $u_h(\mu)$ and not $u(\mu)$. For the total reduced basis error, we have:

$$\underbrace{\|u(\mu) - u_N(\mu)\|_X}_{\text{total error}} \leq \underbrace{\|u(\mu) - u_h(\mu)\|_X}_{\text{FEM error}} + \underbrace{\|u_h(\mu) - u_N(\mu)\|_X}_{\text{RB error}} \tag{17}$$

from (17) we can see that if the FEM error is not well controlled, will dominate the total error even if our RBM is perfect.

For our test case with 5 subfins ($N_f = 5$), the high fidelity approximation is obtained using a second order FEM approximation with $N_{el} = 43622$ elements and $N_h = 89745$ DOF. The FEM L^2-error is estimated by the error with respect to 3rd order FEM approximation over the same mesh ($N_{href} = 200050$ DOF) is $\approx 10^{-4}$. For both FEM solvers, we use the Feel++ library [14]. In figure 3, we show the high fidelity solutions for a fixed value of thermal conductivity $k^i = 1, \forall i$ and varying Biot number $Bi = 0.01, 0.1, 1$. As expected, the temperature at the root decreases as the Biot number increases.

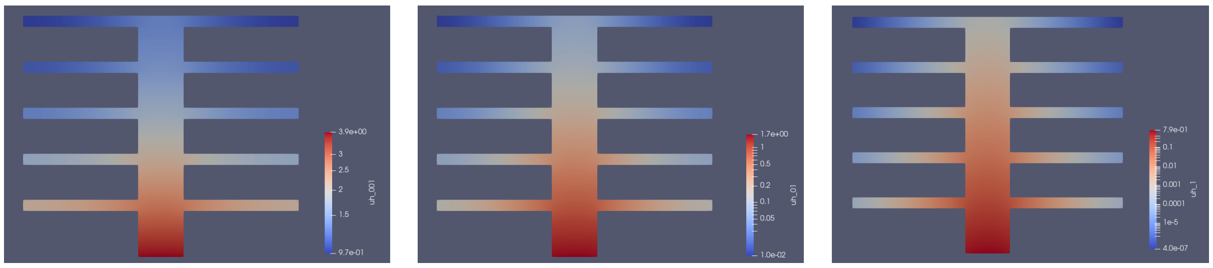

Figure 3: High fidelity solutions for Bi=0.01(left), Bi=0.1(middle) and Bi=1(right)

3.3 Reduced Basis Approximation

As we mentioned in the introduction, the final goal of RB methods is to approximate the solution manifold $\mathcal{M}$ with a low-dimensional subspace X_N. The solution manifold is the set of all solutions of the parametrized problem under variation of the parameter:

$$\mathcal{M} = \{u(\mu) \,|\, \mu \in \mathcal{P}\} \subset X \tag{18}$$

However, as we discussed in section 3.2, the exact solution $u(\mu)$ is not always available, so we replaced it with its high fidelity approximation $u_h(\mu)$, knowing that the error $\|u(\mu) - u_h(\mu)\|_X$

can be made arbitrarily small by refining the mesh. Therefore, we can define the discrete solution manifold $\mathcal{M}_h$ as follows:

$$\mathcal{M}_h = \{u_h(\mu) \,|\, \mu \in \mathcal{P}_h\} \subset X_h, \quad \mathcal{P}_h \subset \mathcal{P} \tag{19}$$

"The final goal of RB methods is to approximate any member of this solution manifold with a low number of, say N, basis functions. A central assumption in the development of any reduced model is that the solution manifold is of low dimension, i.e., that the span of a low number of appropriately chosen basis functions represents the solution manifold with a small error." [9].

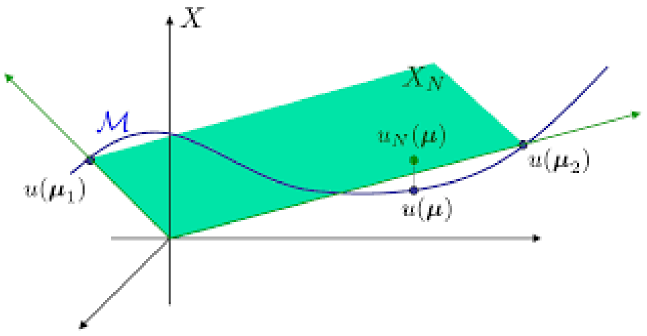

Figure 4: Solution Manifold

We assume for now that we already have the reduced space $X_N = \mathrm{span}\{\zeta_1, \ldots, \zeta_N\} \subset X_h$. The construction of the reduced space is discussed in section 3.4. Given a parameter value μ in $\mathcal{P}$, similarly to what we did in section 3.2, we use a Galerkin approach and we seek a reduced basis approximation $u_N(\mu)$ in X_N such that:

$$a(u_N(\mu), v_N; \mu) = f(v_N; \mu), \quad \forall v_N \in X_N \tag{20}$$

This reduced basis approximation $u_N(\mu)$ is expressed as follows:

$$u_N(\mu) = \sum_{i=1}^{N} (u_N^\mu)_i \; \zeta_i \tag{21}$$

And by testing the previous formulation with $\{\zeta_i\}_{i=1}^N$, we obtain the coordinates vector u_N^μ as the solution of the following linear system:

$$\mathbf{A}_N^\mu \mathbf{u}_N^\mu = \mathbf{f}_N^\mu \tag{22}$$

where $\mathbf{A}_N^\mu = \{a(\zeta_i, \zeta_j; \mu)\}_{i,j=1}^N$ and $\mathbf{f}_N^\mu = \{f(\zeta_i; \mu)\}_{i=1}^N$. Finally, we can evaluate the reduced basis output as:

$$s_N(\mu) = (\mathbf{u}_N^\mu)^T \mathbf{f}_N^\mu \tag{23}$$

3.4 Reduced Basis Construction

Many strategies can be used to construct the reduced basis space X_N. The simplest one consists of sampling the parameter space $\mathcal{P}$ and computing the truth solution $u_h(\mu)$ for each parameter value in the sample. However, these approaches don't guarantee any optimality of the reduced space constructed. Alternatively, methods like Singular Value Decomposition (SVD), Proper Orthogonal Decomposition (POD), Greedy methods, etc. can be used to construct the reduced basis space in a quasi-optimal way. While SVD and POD are related and can be seen as data compression techniques, the Greedy method is an iterative algorithm. A combination of these methods like POD-Greedy can also be used to construct the reduced basis space, especially for parabolic problems. Even if SVD/POD are the methods used in this project, the Greedy method is also discussed in a separate section.

3.4.1 Proper Orthogonal Decomposition

"The Proper Orthogonal Decomposition is an explore-and-compress strategy in which one sample the parameter space, computes the corresponding truth solutions at all sample points and, following compression, retains only the essential information" [9]. Like the SVD, it can be seen as a technique for reducing the dimensionality of a system by representing it in a new orthonormal basis that captures the most important features of the system. This basis is optimal in a least-squares sense. After the transformation, the new uncorrelated variables are called POD modes and are ordered in decreasing importance. A lower-dimensional representation of the data is then obtained by truncating the new basis to retain only the first few modes.

We start by introducing an ordered training set $\mathcal{P}_s = \{\mu_1, \mu_2, ..., \mu_{n_s}\} \subset \mathcal{P}$ of n_s parameter values. For each parameter value $\mu_i \in \mathcal{P}_s$, we compute the corresponding truth solution $u_h(\mu_i) \in \mathcal{M}_h(\mathcal{P}_s) \subset \mathcal{M}_h(\mathcal{P})$ and we define the snapshot matrix $S \in \mathbb{R}^{N_h \times n_s}$ as:

$$S = [u_h^{\mu_1}, u_h^{\mu_2}, ..., u_h^{\mu_{n_s}}],$$

where the vectors $u_h^{\mu_i} \in \mathbb{R}^{N_h}$ represent the degrees of freedom of the solution $u_h(\mu_i) \in X_h$. Then the Singular Value Decomposition (SVD) of S reads:

$$S = U \Sigma Z^T$$

where $U = [\zeta_1, \zeta_2, ..., \zeta_{N_h}] \in \mathbb{R}^{N_h \times N_h}$ and $Z = [\psi_1, \psi_2, ..., \psi_{n_s}] \in \mathbb{R}^{n_s \times n_s}$ are orthogonal matrices, and $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_r) \in \mathbb{R}^{N_h \times n_s}$ with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$. Here $r \leq \min(N_h, n_s)$ denotes the rank of S, which is strictly smaller than n_s if the snapshot vectors are not all linearly independent. Then we can write:

$$S\psi_i = \sigma_i \zeta_i, \quad \text{and} \quad S^T \zeta_i = \sigma_i \psi_i, \quad \forall i = 1, \ldots, r$$

equivalently:

$$S^T S \psi_i = \sigma_i^2 \psi_i, \quad \text{and} \quad S S^T \zeta_i = \sigma_i^2 \zeta_i, \quad \forall i = 1, \ldots, r \tag{24}$$

i.e. σ_i^2 , $i = 1, \ldots, r$, are the nonzero eigenvalues of the matrix $S^T S$ (and also of SS^T), listed in nondecreasing order. While in a pure SVD setting, the "reduced" basis $V \in \mathbb{R}^{N_h \times N}$ of dimension N is defined as the set of the first N left singular vectors ζ_i , $i = 1, \ldots, N$ (or simply the columns U). In the POD setting, we avoid the computation of the SVD of S, and equivalently following (24), we compute the correlation matrix $C = S^T S \in \mathbb{R}^{n_s \times n_s}$, the POD basis V is then given by the set of vectors:

$$\zeta_i = \frac{1}{\sigma_i} S \psi_i, \quad \forall i = 1, \ldots, N$$

where ψ_i, $i = 1, \ldots, N$ are the first N eigenvectors of the correlation matrix C corresponding to the N largest eigenvalues σ_i^2, $i = 1, \ldots, N$. If we define $\mathcal{V}_N = \{W \in \mathbb{R}^{n_s \times N} \mid W^T W = I\}$ the set of all N-dimensional orthonormal bases, the POD basis $V \in \mathcal{V}_N$ by construction and verifies:

$$\sum_{i=1}^{n_s} ||u_h^{\mu_i} - VV^T u_h^{\mu_i}||_2^2 = \min_{W \in \mathcal{V}_N} \sum_{i=1}^{n_s} ||u_h^{\mu_i} - WW^T u_h^{\mu_i}||_2^2 = \sum_{i=N+1}^{r} \sigma_i^2 \tag{25}$$

The second equality in (25) gives us a practical criterion to select the minimal POD dimension $N \leq r$ s.t the projection error is smaller than a given tolerance ϵ_{POD}. In practice, we choose N as the smallest integer such that:

$$I(N) = \frac{\sum_{i=1}^{N} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2} \geq 1 - \epsilon_{\text{POD}} \tag{26}$$

that is the energy retained by the last $r - N$ modes is less than ϵ_{POD}^2 of the total energy. $I(N)$ is called the relative information content (RIC) and it represents the fraction of the total energy retained by the first N POD modes.

In the POD procedure presented above, the minimization of the projection error of the snapshots vectors $\{u_h^{\mu_i}\}_{i=1}^{n_s}$ is done in $||.||_2$ norm. However, since the snapshots functions $\{u_h(\mu_i)\}_{i=1}^{n_s}$ belong to $X_h \in X$, the $||.||_X$ norm is more appropriate (native) for the error minimization. In many cases, this last norm coincides with or is equivalent to the norm induced by the bilinear form for a fixed reference parameter value $\bar{\mu} \in \mathcal{P}$:

$$(u, v)_X = a(u, v; \bar{\mu}), \quad \forall u, v \in X \tag{27}$$

$$||u||_X = \sqrt{(u, u)_X}, \quad \forall u \in X \tag{28}$$

In matrix form, this norm is given by:

$$||u||_X = \sqrt{\mathbf{u}^T \mathbf{A}_h^{\bar{\mu}} \mathbf{u}}, \quad \forall u \in X_h \tag{29}$$

where $\mathbf{u}$ is the vector of the degrees of freedom of the function u. The POD framework can then be adapted to the $||.||_X$ norm by computing the correlation matrix as:

$$C = S^T A_h^{\bar{\mu}} S \tag{30}$$

Finally, the POD algorithm when $ns \leq N_h$ can be summarized as follows (from[15]):

Algorithm 1 POD algorithm with respect to the $||.||_X$ norm

```
function V = POD($S$, $A_h^{\bar{\mu}}$, $\epsilon_{\text{POD}}$)
    form the correlation matrix $C = S^T A_h^{\bar{\mu}} S$
    solve the eigenvalue problem $C\psi_i = \sigma_i^2 \psi_i, \quad i = 1, \dots, r$
    set $\zeta_i = \frac{1}{\sigma_i} S\psi_i$
    define N as the minimum integer such that $I(N) \geq 1 - \epsilon_{\text{POD}}$
    set $V = [\zeta_1, \zeta_2, ..., \zeta_N]$
end function
```

As we already mentioned, in a POD setting, we avoid the computation of the SVD of $S \in \mathbb{R}^{N_h \times n_s}$ (N_h usually very large), and we deal with the correlation matrix $C \in \mathbb{R}^{n_s \times n_s}$ instead. However, solving $ns \gg N$ high-fidelity problems to construct the snapshot matrix S is still computationally expensive. To address this issue, the Greedy method is an alternative that allows the construct of the reduced basis space in an iterative way. This method is discussed in the next section.

3.4.2 Greedy Method

Unlike the proper orthogonal decomposition (POD) method, the greedy algorithm follows an iterative procedure. In each iteration, a new basis function is added, leading to an enhancement in the overall accuracy of the basis space. This process involves computing one high-fidelity solution per iteration, with a total of N high-fidelity solutions required to construct the N-dimensional reduced basis.

The main idea behind the greedy algorithm is to select at each iteration n, the next basis function that maximizes the error, more precisely, we are looking for $u_h(\mu_{n+1})$ such that:

$$\mu_{n+1} = \arg\max_{\mu \in \mathcal{P}} ||u_h(\mu) - V^n u_n(\mu)||_X \tag{31}$$

where $V^n \in \mathbb{R}^{N_h \times n}$ is the reduced basis space at the n-th iteration and $u_n(\mu) \in \mathbb{R}^n$ is the solution of the reduced problem (22). In other words, at each step, the greedy algorithm selects the worst element approximated by the current reduced basis space.

The main goal of the greedy algorithm is to construct a quasi-optimal reduced basis space in a more efficient way (compared to SVD/POD), even if we are talking here about an offline cost. The cost of solving problem (31) is very high for the following reasons:

- Finding the maximum in (31) over the entire parameter space $\mathcal{P}$ is computationally infeasible.
- We need the high fidelity solution $u_h(\mu)$ for each parameter value to evaluate the error.

To address the first point, we introduce a finite training set $\mathcal{P}_h \subset \mathcal{P}$ of cardinality n_{train}. The maximum in (31) is then computed over $\mathcal{P}_h$ instead of $\mathcal{P}$. This turns the problem of finding the maximum of the error bound over $\mathcal{P}$ into a simpler enumeration problem, where we only need to list the values of the error and select the maximum. However, the second point remains a challenge as we still need to compute the high-fidelity solution for each parameter value in $\mathcal{P}_h$. To address this issue, an error estimator $\Delta_n(\mu)$ is used in practice. This a posteriori error estimator verifies:

$$||u_h(\mu) - V^n u_n(\mu)||_X \leq \Delta_n(\mu), \quad \forall \mu \in \mathcal{P} \tag{32}$$

This error estimator can be computed efficiently and independently of N_h. Finally, the maximization problem in (31) can be replaced by:

$$\mu_{n+1} = \arg\max_{\mu \in \mathcal{P}_h} \Delta_n(\mu) \tag{33}$$

and the new basis function ζ_{n+1} is given by applying a Gram-Schmidt orthogonalization process to the high fidelity solution $u_h(\mu_{n+1})$ to ensure some numerical stability properties.

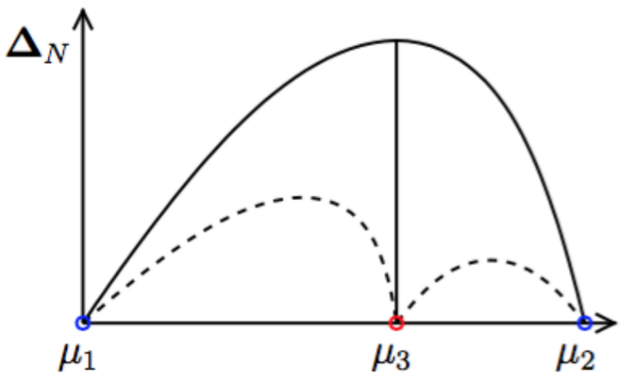

Figure 5: Greedy Procedure

Since the greedy method is not used in this project, we will not go into more detail about how to compute $\Delta_n(\mu)$.

3.5 Affine Decomposition

After constructing the reduced basis space X_N as discussed in 3.4, we return to the reduced basis approximation (20). The matrix $\mathbf{A}_N^\mu$ is full (dense), while the matrix $\mathbf{A}_h^\mu$ is sparse. However, since $N \ll N_h$, solving the reduced problem (22) is much cheaper than solving the high fidelity problem (14). Unfortunately, the assembly of the reduced matrix $\mathbf{A}_N^\mu$ and $\mathbf{f}_N^\mu$ depends on N_h. More precisely, to compute the reduced basis solution matrix $\mathbf{A}_N^\mu$, one has to compute the full order solution matrix $\mathbf{A}_h^\mu$ and projecting it onto the reduced space X_N:

$$\mathbf{A}_N^{\mu} = V^T \mathbf{A}_h^{\mu} V \tag{34}$$

For each new parameter value μ, we need to assemble the matrix $\mathbf{A}_h^{\mu}$. This operation depends on N_h which is generally very large, so it is very expensive and we would like to avoid it, especially in the online stage.

To overcome this issue, we assume that the bilinear form $a(\cdot,\cdot;\mu)$ is affine in the parameter μ, i.e.:

$$a(\cdot,\cdot;\mu) = \sum_{q=1}^{Q_a} \theta_q^a(\mu)\, a_q(\cdot,\cdot) \tag{35}$$

where $a_q : X \times X \to \mathbb{R}$ are bilinear forms independent of μ and $\theta_q^a : \mathcal{P} \to \mathbb{R}$ are scalar quantities depending only on μ.

Now, a series of Q_a matrices $\mathbf{A}_N^q \in \mathbb{R}^{N\times N}$ (associated to a_q) can be precomputed independently of μ in an offline stage. Then, during the online stage, for a given parameter value μ, we compute the reduced basis solution matrix $\mathbf{A}_N^{\mu}$ as follows:

$$\mathbf{A}_N^{\mu} = \sum_{q=1}^{Q_a} \theta_q^a(\mu)\, \mathbf{A}_N^q \tag{36}$$

This operation has a complexity proportional to $\mathcal{O}(Q_a\, N^2)$ which is independent of $N_h >> N$. The same assumption can be made for the linear form $f(\cdot;\mu)$.

The possibility of ensuring an offline-online decomposition of the RBM relies on the affine decomposition assumption. However, in practice, nothing guarantees that this assumption is satisfied. In this case, multiple strategies can be used to ensure an offline-online decomposition like the Empirical Interpolation Method (EIM). In these methods, the bilinear (and linear) forms are approximated by a quantity that verifies the affine decomposition assumption.

Back to our test case 2, both, the bilinear form $a(\cdot,\cdot;\mu)$ and the linear form $f(\cdot;\mu)$ are affine in the parameter μ and verify the affine decomposition assumption. Indeed, the bilinear form can be written as:

$$a(u,v;\mu) = \sum_{i=0}^{N_f} k^i \int_{\Omega^i} \nabla u \cdot \nabla v \, dx + Bi \int_{\Gamma_{\text{ext}}} uv \, ds \tag{37}$$

which is equivalent to (35) with $Q_a = N_f + 2$ and $\{a_q\}_{q=1}^{Q_a}$, $\{\theta_q^a\}_{q=1}^{Q_a}$ are given by:

$$\theta_1^a(\mu) = k^0 = 1, \quad a_1(u,v) = \int_{\Omega^0} \nabla u \cdot \nabla v \, dx, \tag{38}$$

$$\theta_i^a(\mu) = k^{i-1}, \quad a_i(u,v) = \int_{\Omega^{i-1}} \nabla u \cdot \nabla v \, dx, \quad i = 2, \dots, N_f + 1, \tag{39}$$

$$\theta_{N_f+2}^a(\mu) = Bi, \quad a_{N_f+2}(u,v) = \int_{\Gamma_{ext}} u\, v\, ds, \tag{40}$$

3.5.1 Results

The procedure described in the previous sections is implemented in this project. We consider at first a simple numerical experiment where we compare the FEM and RBM solutions for given parameter values. Close to the test case in section 3.2, we consider $N_f = 5$ and $P = 1$, where the conductivity is fixed to $k^i = 0.9\, \forall i$, and the Biot number is varied in the range $Bi \in [0.001, 1]$. The high fidelity problem is exactly as described in section 3.2 with $N_h = 89745$ DOF. For this problem, for $\epsilon_{\text{POD}} = 10^{-5}$, the number of modes required is $N = 8$. In figures 6 and 7, we present the FEM and RBM solutions for $Bi = 0.001$ and $Bi = 1$ respectively.

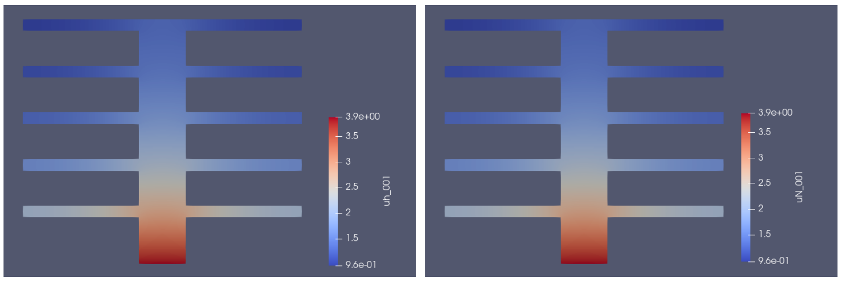

Figure 6: FEM(left) and RB with $N = 8$(right) solutions for $Bi = 0.001$

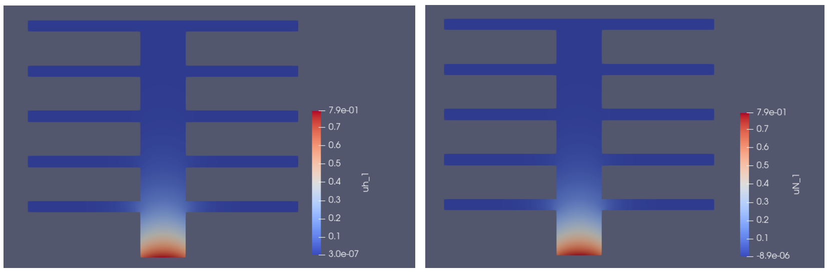

Figure 7: FEM(left) and RB with $N = 8$(right) solutions for $Bi = 1$

An error analysis is shown in figure 8 where we present the maximum relative error in energy norm and the maximum relative output error for different number of modes. We can additionally verify the quadratic property of the output error with respect to the energy error for compliant problems:

$$|s_h(\mu) - s_N(\mu)| = ||u_h(\mu) - u_N(\mu)||_\mu^2 \tag{41}$$

where $||.||_\mu$ is the energy norm induced by the bilinear form $a(\cdot, \cdot; \mu)$.

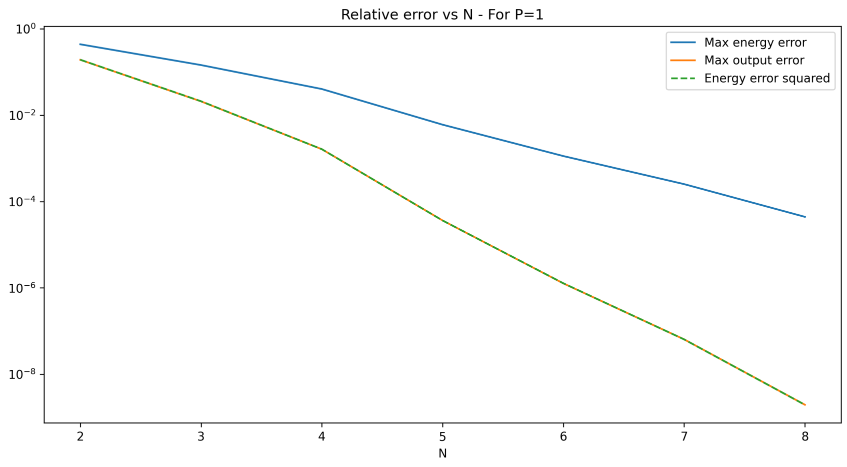

Figure 8: Relative error vs the number of modes for $P = 1$

Finally and most importantly, by reducing the complexity dependence from N_h to $N = 8$, the time required to get a solution is reduced from 5.18 seconds to 1.98×10^{-4} seconds.

3.6 Classical Reduced Basis method Challenges

As we have seen in the previous sections, the reduced basis method aims to guarantee both accuracy and efficiency. However, this efficiency depends on the compactness of the reduced basis, N. For a large class of problems, the number of modes N required to reach a given accuracy could be very large. For instance, as we have mentioned in 1.3, hyperbolic transport problems have a slow decay of the Kolmogorov width which implies that the solution manifold is not well approximated by a low dimensional linear space. Another challenge for the classical reduced basis method is the curse of dimensionality of the space Ω or even the parameter space $\mathcal{P}$. For the latter, we can study it in our test case by increasing the number of subfins N_f and the number of parameters P. Let us consider the same test case with 5 subfins and compare the number of modes required to reach a given accuracy for $P = 1$ and $P = 6$. In figure 9 and with more details in table 1, a comparison is made between $P = 1$ and $P = 6$. The number of modes required to reach the same accuracy is much larger for $P = 6$ than for $P = 1$ and the Kolmogorov width decay is slower. Since our test case is linear, even if the number of modes required is large, the online time is still very small. However, it is chosen as a simple example to illustrate the challenges of the classical reduced basis method and test the nonlinear compressive reduced basis approximation method.

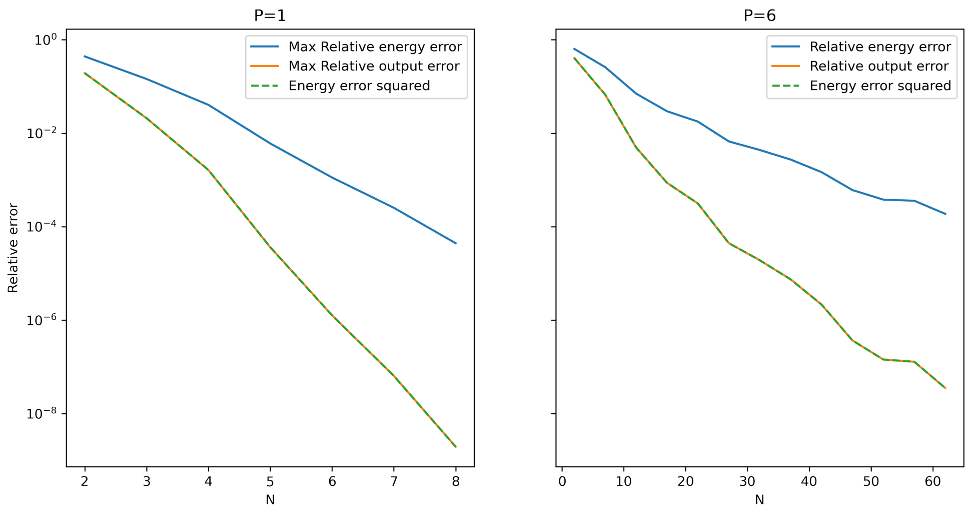

Figure 9: Relative error vs number of POD modes for $P = 1$(left) and $P = 9$(right)

Table 1: Comparison between $P = 1$ and $P = 6$

P	$N_{\max}$	Online time	FEM time	Max L2 error	Mean L2 error
1	8	1.98e-04	5.18e-01	3.17e-05	6.37e-06
6	63	6.42e-04	5.18e-01	2.29e-05	3.68e-06

4 Nonlinear compressive reduced basis approximation

In this section, the nonlinear compressive reduced basis approximation method is presented. As mentioned in section 1.3, this method is proposed in [2, 5] to deal with the particular situation where the Kolmogorov widths decay slowly. We will first introduce some essential mathematical definitions and concepts, then we will present the methodology and the algorithm and finally, we will discuss some numerical results.

4.1 Encoder-decoder framework

Generally speaking, in approximation theory, approximation methods depending on m parameters are built on 2 continuous mappings:

- The encoder:
$$E : X \to \mathbb{R}^m$$
For a given $u \in X$, $E(u)$ is a vector of m parameters that represents (encodes) u.

- The decoder:
$$D : \mathbb{R}^m \to X$$
which is used to reconstruct an approximation of u from $E(u)$.

The approximation of u is then given by the encoder-decoder composition:

$$u \approx D(E(u))$$

The quality of this approximation can be measured by its maximum distortion (worst case) over $\mathcal{M}$:
$$\mathcal{E}^{wc}(\mathcal{M}; E, D) = \max_{v \in \mathcal{M}} \|v - D(E(v))\|$$
This leads us to the notion of widths:

$$\inf_{E,D} \mathcal{E}^{max}(\mathcal{M}; E, D)$$

by optimizing the choice of (E, D) under specific restrictions. In our study, we are particularly interested in the following cases:

- If we assume that the decoder D is linear, we obtain the Kolmogorov m-width:
$$d_m(\mathcal{M}) := \inf_{\dim(X_N) \leq m} \max_{v \in \mathcal{M}} \min_{w \in X_N} \|v - w\|_X \tag{42}$$
which measures how well the set $\mathcal{M}$ can be approximated by an m-dimensional subspace.

- Reciprocally, if we assume that the encoder E is linear, and the decoder D is nonlinear, we obtain the sensing numbers:
$$s_m(\mathcal{M}) := \inf_{D,\lambda_1,\ldots,\lambda_m} \max_{v \in \mathcal{M}} \|v - D(\lambda_1(v), \ldots, \lambda_m(v))\|_X \tag{43}$$
where $\lambda_1, \ldots, \lambda_m$ are linear functionals.

4.2 Sensing number and Parameter identification

After introducing the encoder-decoder framework, each method can now be seen as a particular choice of (E, D). The classical RBM can be seen as a linear encoder-decoder method, which is why it is described as a Kolmogorov-based method. On the other hand, autoencoder methods in [8, 11] can be seen as nonlinear encoder-decoder methods, where the notion of auto-encoder is used to minimize the information needed to reconstruct the identity in $\mathcal{M}$, as a combination of two neural networks $\mathcal{E}$ and $\mathcal{D}$ (i.e. $Id = \mathcal{D} \circ \mathcal{E}$), and where the size of the layer at the output of $\mathcal{E}$ and the input of $\mathcal{D}$ is minimized.. looking thus to identify m. In these publications, neural networks are therefore used to discover $\mathcal{E}$ and $\mathcal{D}$, and further investments are made in the field of machine learning, with two warnings: the amount of data required to train the associated networks, and the lack of understanding underlying machine learning.

In the opposite direction, we can refer to a still timely but much older problem of "parameter reconstruction" or "parameter identification", and ask what minimal knowledge on the elements v of $\mathcal{M}$ is required to identify the value of parameters μ in $\mathcal{P}$ such that $v = u(\mu)$. Indeed, assuming that we have such a reconstruction $R : v \in \mathcal{M} \mapsto \mu \in \mathcal{P}$, then the identity mapping in $\mathcal{M} : Id_{\mathcal{M}}$ would involve the solution operator $\mu \in \mathcal{P} \mapsto u(\mu) \in \mathcal{M}$ so that $Id_{\mathcal{M}} = v \mapsto u(R(v))$.

Our approach stands in the middle of these two extreme encoder-decoder paradigms and relies on the two notions. The first one is the sensing numbers $s_m(\mathcal{M})$ defined in 43, where the encoder is still assumed to be linear but the decoder is assumed to be nonlinear. Then, referring to the parameter reconstruction/identification, we can raise the question: what are the best linear mappings $\lambda_1, \ldots, \lambda_m$ that represent well $\mathcal{M}$, or, if we remember that we are in a Hilbert setting, from Riesz identification between X and X', the question becomes: what are the best elements in $X : v_1, \ldots, v_m$ that are the more aligned with $\mathcal{M}$ (in order to have the largest scalar product with any element of $\mathcal{M}$), so that, for any $w \in \mathcal{M}$ and any j, $1 \leq j \leq m : \lambda_j(w) = (v_j, w)_X$ are the best linear functional appearing above? This is actually what is considered in the POD framework within the notion of Relative Information Content: the first modes on the POD approach are the ones that contain in the vectorial space spanned by $\mathcal{M}$ the largest amount of information. This naturally leads to considering that the first POD functions $v_j = \zeta_j$ are the best elements that carry the information on solutions in $\mathcal{M}$. This is what led the authors in [2, 5] to propose the "ansatz" that a candidate for the optimal encoder in 43 consists in the moments associated with those POD modes. In order to evaluate this claim, let us understand to which extent and how many of these first POD moments allow us to recover the parameters and define a reconstruction R that is written under the form $R = D_R \circ E$ where the encoder E is linear and defined as being the set of moments

$$v \mapsto \{(v, \zeta_j)_X\}_{j=1}^{m} \tag{44}$$

Actually, in dimension $P = 1$ (only the Biot number is varying) the following figure 10 graphically illustrates the simple dependence of the Biot number as a function of the first moment of the solution, which is the sought for decoder D_R. The (Lipschitz) continuity of u with respect to Bi and, thus, the continuity of the decoder $D = u \circ D_R$, is a classical result (see e.g. [15] Sect.5.3 for the proof).

In higher dimensions, we have to investigate the nonlinear decoder. As is presented in [5], where polynomial and ensemble learning regression methods (like a random forest) can be used to define the decoder. Back with the 1-dimensional problem ($P = 1$), the dependence of $\mu = Bi$ with respect to the first coefficient displayed in Figure 10 can very well be approximated as a spline regression problem. By increasing the number of knots (n_{knots}) and the degree of the model, spline regression can achieve machine precision. Decision trees and random forests also deliver excellent accuracy, with mean relative errors reaching up to 10^{-7}. However, polynomial models perform very weakly on this problem and do not achieve the same level of precision. In higher-dimensional problems ($P > 1$), the regression problem becomes harder, and some algorithms suffer from the curse of dimensionality, as is well reported in the literature see [3]. In our case, the reconstruction of the parameter using decision trees (or random forests) gives "acceptable" results up to $P = 3$ (a mean relative error of 10^{-3} on the Biot number reconstruction and 10^{-2} for k_0 and k_1). For the same reason, for $P > 3$, more advanced models need to be employed to achieve a reasonable reconstruction of the parameters. We can conclude from these initial results that indeed $s_m(\mathcal{M}) = 0$ for $m \geq P$, at least for $P = 1, 2, 3$, which sets the connection between the manifold dimension and the complexity of the manifold written in terms of sensing number rather than Kolmogorov m width.

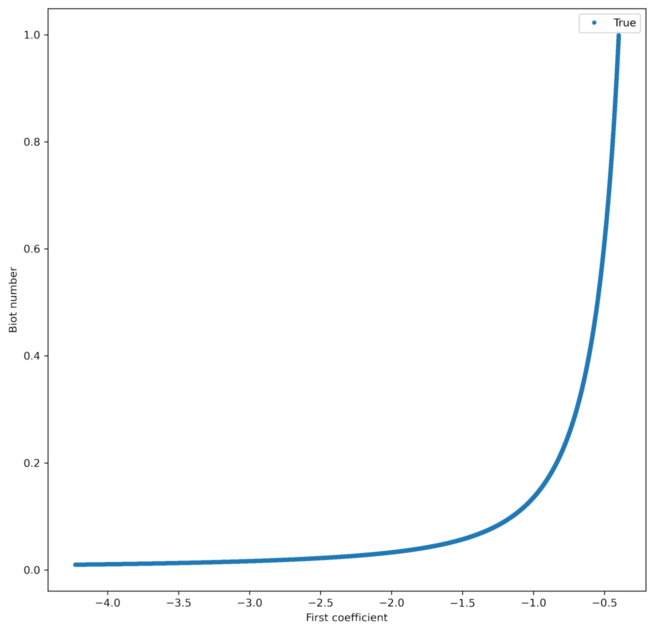

Figure 10: Dependence of $\mu = Bi$ on the first RB coefficient $(u_N^\mu)_1$

4.3 Methodology Overview

As we have discussed in sections 1.3 and 3.6, the linear reduced basis approximation is not always suitable, in particular for problems with a slow Kolmogorov width decay. It is then natural to consider the nonlinear reduced basis approximation. In this section, we will present the nonlinear compressive reduced basis approximation method proposed in [5, 2] which intends to deal with the particular situation where:

- The Kolmogorov widths $d_m(\mathcal{M})$ decay slowly.
- The sensing numbers $s_m(\mathcal{M})$ decay much faster.

These two assumptions tell us, that the same target accuracy ϵ can be reached either by $s_{n(\epsilon)}(\mathcal{K})$ or $d_{N(\epsilon)}(\mathcal{K})$ with $n(\epsilon) \ll N(\epsilon)$. In section 3, we have seen that the classical RBM is based on the Kolmogorov width $d_{N(\epsilon)}(\mathcal{M})$ and that the basis can be constructed using the POD method where $N(\epsilon)$ is chosen using a RIC criterion. On the other hand, the optimal $n(\epsilon)$ measurements $\lambda_1, \dots, \lambda_{n(\epsilon)}$ in the definition of $s_{n(\epsilon)}(\mathcal{M})$ are not available in general. However, in 4.2, we have seen that it is reasonable to consider them as the moments associated with the first $n(\epsilon)$ POD modes where $n(\epsilon) = n \approx P$. It is reasonable then to expect that the $N - n$ remaining coefficients depend somehow on the first n coefficients. Mathematically, if we denote $(\psi_i)_{i=1}^N$ the POD basis computed offline, then by 21, the reduced basis approximation is given by:

$$u_N(\mu) = \sum_{i=1}^{N} (u_N^\mu)_i \ \zeta_i$$

In addition, we expect that :

$$(u_N^\mu)_k = \phi_k((u_N^\mu)_1, \dots, (u_N^\mu)_n), \quad \forall k \in \{n+1, \dots, N\}, \tag{45}$$

where $(\phi_k)_{k=n+1}^N$ are unknown nonlinear transformations. Then, it follows from the reduced basis problem (20) that:

$$a(u_N(\mu), \zeta_i; \mu) = f(\zeta_i; \mu), \quad \forall i \in \{1, \dots, n\} \tag{46}$$

by taking $v_N = \zeta_i$ for $i \in \{1, \dots, n\}$, then from (45) we have:

$$\sum_{j=1}^{n} (u_N^\mu)_j \; a(\zeta_j, \zeta_i; \mu) + \sum_{j=n+1}^{N} \phi_j((u_N^\mu)_1, \dots, (u_N^\mu)_n) \; a(\zeta_j, \zeta_i; \mu) = f(\zeta_i; \mu), \quad \forall i \in \{1, \dots, n\}$$

Which can be written in the matrix form:

$$\mathbf{A}_n^\mu \mathbf{u}_n^\mu = \mathbf{f}_n^\mu - B^\mu \phi(\mathbf{u}_n^\mu) \tag{47}$$

where $\mathbf{A}_n^\mu = (a(\zeta_j, \zeta_i; \mu))_{i,j=1}^n \in \mathbb{R}^{n \times n}$, $\mathbf{f}_n^\mu = (f(\zeta_i; \mu))_{i=1}^n \in \mathbb{R}^n$, $\mathbf{u}_n^\mu = ((u_N^\mu)_i)_{i=1}^n \in \mathbb{R}^n$, $\phi(\mathbf{u}_n^\mu) = (\phi_j(\mathbf{u}_n^\mu))_{j=n+1}^N \in \mathbb{R}^{N-n}$ and $B^\mu = (a(\zeta_j, \zeta_i; \mu))_{i=1, j=n+1}^{n, N} \in \mathbb{R}^{n \times (N-n)}$. The nonlinear transformations $\{\phi_k\}_{k=n+1}^N$ are approximated by a regression nonlinear model $\hat{\phi} : \mathbb{R}^n \to \mathbb{R}^{N-n}$. This model is learned in an offline phase. Finally, the online phase consists of solving the following nonlinear system:

$$\mathbf{A}_n^\mu \mathbf{u}_n^\mu = \mathbf{f}_n^\mu - B^\mu \hat{\phi}(\mathbf{u}_n^\mu) \tag{48}$$

We can see that $\hat{\phi}$ should be learned on the reduced model, which makes the generation of training data much cheaper than generating the snapshots during the basis construction. Additionally, these precomputed snapshots can also be used as a training set for the nonlinear model $\hat{\phi}$ after being projected on the reduced basis space.

The nonlinear compressive RBM is not independent of the classical RBM, instead, it builds on it trying to get the same final approximation (same basis) but in a more efficient way. In practice, the full basis of size N is constructed as in the classical RBM and our final approximation lies in this N-dimensional space. This is why the method is not expected to give a better accuracy than the classical RBM, but instead to be more efficient while keeping the same accuracy.

This being said, we can now present the algorithm of the nonlinear compressive reduced basis approximation method. We start by the offline phase where we construct the reduced basis, learn the nonlinear model $\hat{\phi}$ and assemble the offline quantities that are independent of the parameter.

Algorithm 2 Nonlinear compressive reduced basis offline phase

```
1: Compute the reduced basis of size $N$ using POD.
2: Assemble $\{\mathbf{A}_N^q\}_{q=1}^{Q_a}$, $\{\mathbf{B}^q\}_{q=1}^{Q_a}$ and $\{\mathbf{f}_N^q\}_{q=1}^{Q_f}$
3: Choose a train sample of size $M$: $\{\mu_1, \dots, \mu_M\}$.
4: for $i = 1$ to $M$ do
5:     Assemble $\mathbf{A}_N^{\mu_i} = \sum_{q=1}^{Q_a} \theta_q^a(\mu_i) \mathbf{A}_N^q$ and $\mathbf{f}_N^{\mu_i} = \sum_{q=1}^{Q_f} \theta_q^f(\mu_i) \mathbf{f}_N^q$.
6:     Solve for $\mathbf{u}^i$, $\mathbf{A}_N^{\mu_i} \mathbf{u}^i = \mathbf{f}_N^{\mu_i}$.
7: for $k = n+1$ to $N$ do
8:     Learn $\hat{\phi}_k = \arg\min_{\phi_k \in F} \left\{ \sum_{i=1}^M |\mathbf{u}_k^i - \phi_k(\mathbf{u}_1^i, \dots, \mathbf{u}_n^i)|^2 \right\}$.
```

In the algorithm 2, we described the learning process of the nonlinear model $\hat{\phi}$ as minimizing a mean squared loss to select the best model within a sufficiently rich class F of nonlinear functions. This class could be for example, the set of:

- Quadratic functions as in [1].

- Polynomials of some higher degree $d > 2$.
- Splines.
- Neural networks as in [2].

another class of regression models used in this work is tree based models, such as decision trees and random forests. The choice of the regression model is critical since it affects the accuracy, stability and computational cost (online and offline) of the method.

In the online phase, we solve the nonlinear system 48 for a given parameter value μ. For the sake of simplicity, we consider the Picard iteration method to solve the nonlinear system in the algorithm 3. In practice, more advanced and stable methods can be used to solve the nonlinear system.

Algorithm 3 Nonlinear compressive reduced basis online phase (Picard iteration)

```
1: Assemble $\mathbf{A}_n^\mu = \sum_{q=1}^{Q_a} \theta_q^a(\mu)\mathbf{A}_n^q$, $\mathbf{B}^\mu = \sum_{q=1}^{Q_a} \theta_q^a(\mu)\mathbf{B}^q$ and $\mathbf{f}_n^\mu = \sum_{q=1}^{Q_f} \theta_q^f(\mu)\mathbf{f}_n^q$.
2: **Initialization:** $\mathbf{u_n^0} = \mathbf{u_0}$, s.t. $\mathbf{A}_n^\mu \mathbf{u_0} = \mathbf{f}_n^\mu$.
3: **Iteration** $k$**:**
4:     Compute $\hat{\phi}(\mathbf{u_n^{k-1}})$.
5:     Solve $\mathbf{A}_n^\mu \mathbf{u_n^k} = \mathbf{f}_n^\mu - B^\mu \hat{\phi}(\mathbf{u_n^{k-1}})$.
6: **Stopping criterion:** $\|\mathbf{u_n^k} - \mathbf{u_n^{k-1}}\| < \epsilon$.
7: $\mathbf{u}_N^\mu = [\mathbf{u}_n^k, \hat{\phi}(\mathbf{u}_n^k)]$.
```

4.4 Regression step

One of the fundamental steps in the nonlinear compressive RBM is the regression task, specifically training a model that accurately predicts the high RB coefficients from the first n independent ones. Both accuracy and performance (prediction cost) are crucial in this step. Accuracy is vital because any error in this step introduces a corresponding error in the final RB approximation. Performance is equally important, as it significantly impacts online performance, especially that the prediction is used iteratively. Another fundamental consideration is the stability of these approximations. An unstable model is more likely to cause divergence in the associated nonlinear solvers.

In the following, we present the results of the regression task. We start with a 1-dimensional problem ($P = 1$) and then move to higher dimensions ($P > 1$). We consider the same test case as in section 2 with $N_f = 5$. The data is generated using the classical RBM as described in section 3.

4.4.1 $\mathbf{P = 1}$

Starting with the 1-dimensional case ($P = 1$), we consider the problem of predicting the last $(N - n)$ coefficients from the first n coefficient(s). The value of N is determined such that $\epsilon_{\text{POD}} = 10^{-5}$, which gave us N = 8. Following the discussion in section 4.2, it is natural to expect that $n = 1$ is sufficient for the reconstruction of high modes. This is geometrically illustrated in Figure 11 where we can see the nonlinear dependence of the high RB coefficients on the first one.

For this regression task, we tested several classical ML algorithms, namely decision tree, random forest, spline, and polynomial regression. The training dataset size is 10^7 for decision tree and random forest models and 10^5 for polynomial and spline models. In Figure 12, we can

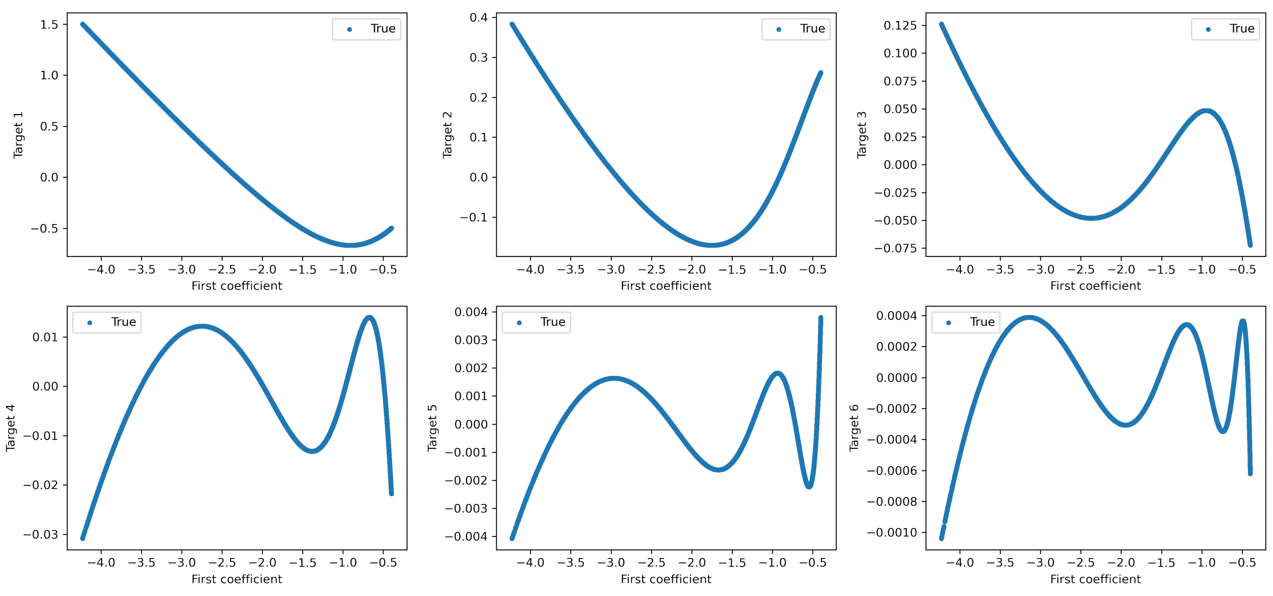

Figure 11: Dependence of the high RB coefficients on the first one for $P = 1$

see that splines provide the best accuracy compared to other models. The accuracy of polynomial regression improves with increasing degree due to the oscillatory behavior of the higher RB coefficients (see Figure 11). The spline model, tuned with hyperparameters (knots=100, degree=10), achieves the best accuracy due to its local adaptability, giving it a significant advantage over polynomials.

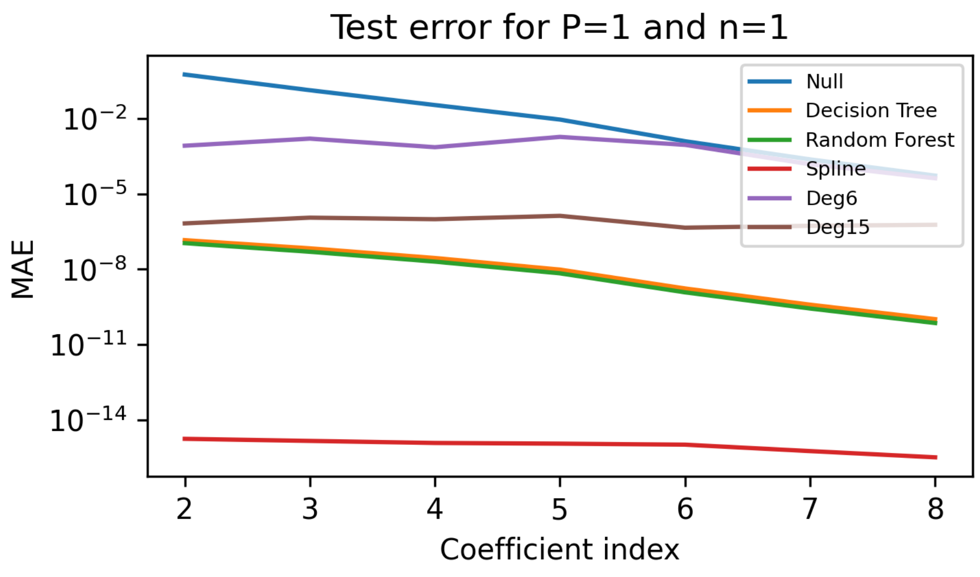

Figure 12: Mean absolute error on a test set for different ML models for $P = 1$ and $n = 1$.

Next, we consider increasing values of n for $P = 1$. The results are depicted in Figure 13. We observe that decision trees (and similarly random forests) do not improve, while polynomial regressors gain in accuracy. This result is expected and can be explained by the fact that the first coefficient is the only independent one. Increasing n does not provide additional information but implicitly increases the degree of polynomial models, which explains the improvement in accuracy. While this approach appears to work well with polynomials, it has significant drawbacks. Firstly, increasing n, significantly raises the prediction cost, given by $\binom{n+d}{d}$ where

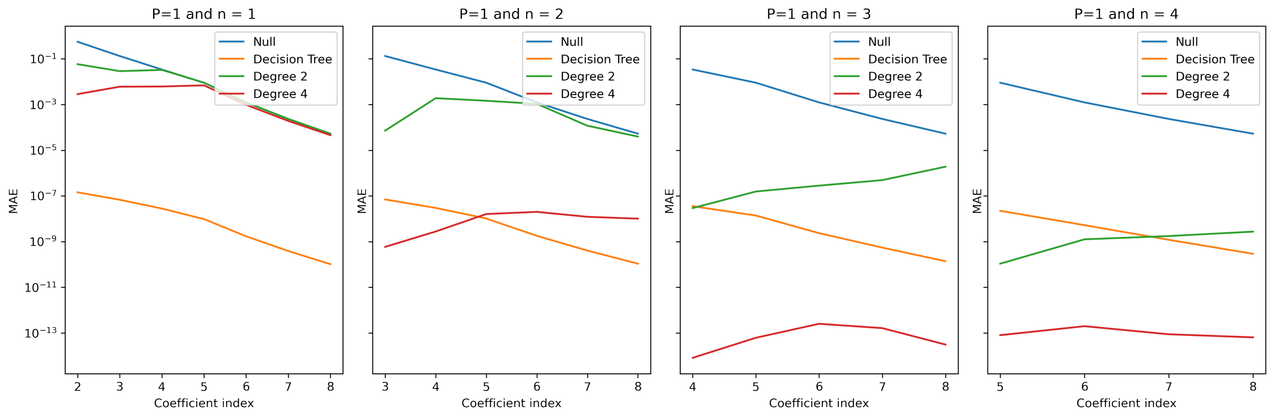

Figure 13: Mean absolute error for different number of features n and $P = 1$

d is the degree of the polynomial. Secondly, it introduces an instability in the model. This is instability can be seen as an extrapolation issue where the model predicts very large values outside the training range. This is particularly problematic in the online phase, especially when using Picard iteration for solving the nonlinear system. Other nonlinear solvers are more robust and can handle such issues.

4.4.2 Multi-parameter case

We now consider the case $P = 2$. The number of POD modes $N(\epsilon_{\text{POD}}) = 25$. Figure 14 shows that for $n = P = 2$, decision trees and random forests perform better than polynomials; however, we do not achieve the same accuracy as for $P = 1$. Similar to $P = 1$, increasing n improves the accuracy of polynomial models but leads to instability.

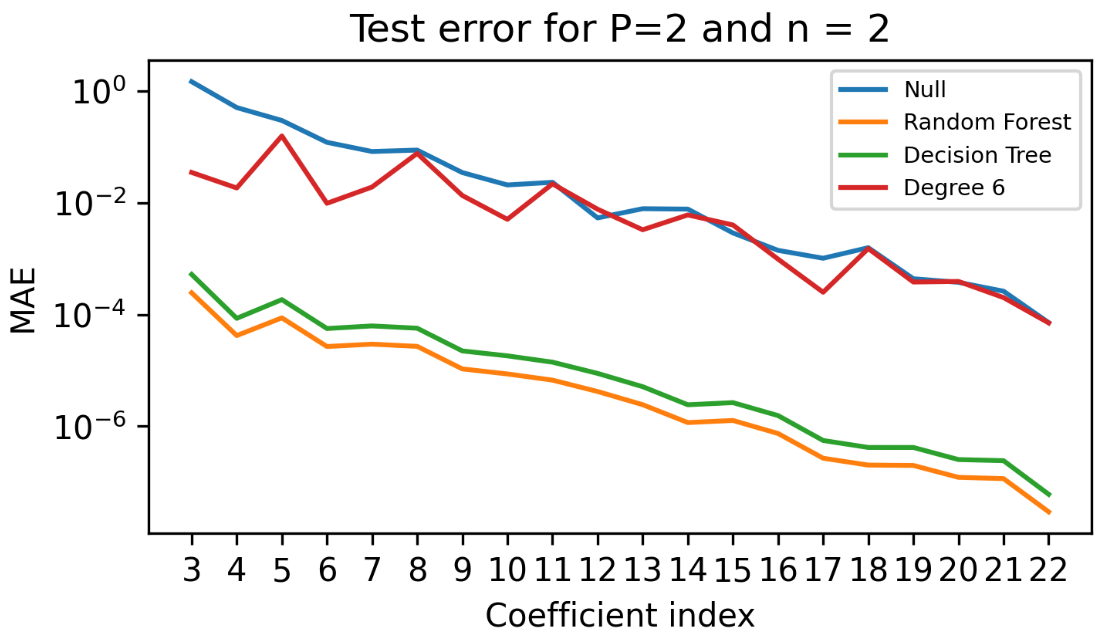

Figure 14: Mean absolute error on a test set for different ML models for $P = 2$ and $n = 2$.

For higher parameter dimensions ($P > 2$), the multivariate regression task becomes more complex. Achieving acceptable accuracy using these classic machine-learning algorithms is challenging and improved approaches need be involved. This is illustrated in Figure 15 for $P = 3$ and in Figure 16 for $P = 6$ where the accuracy is not satisfactory.

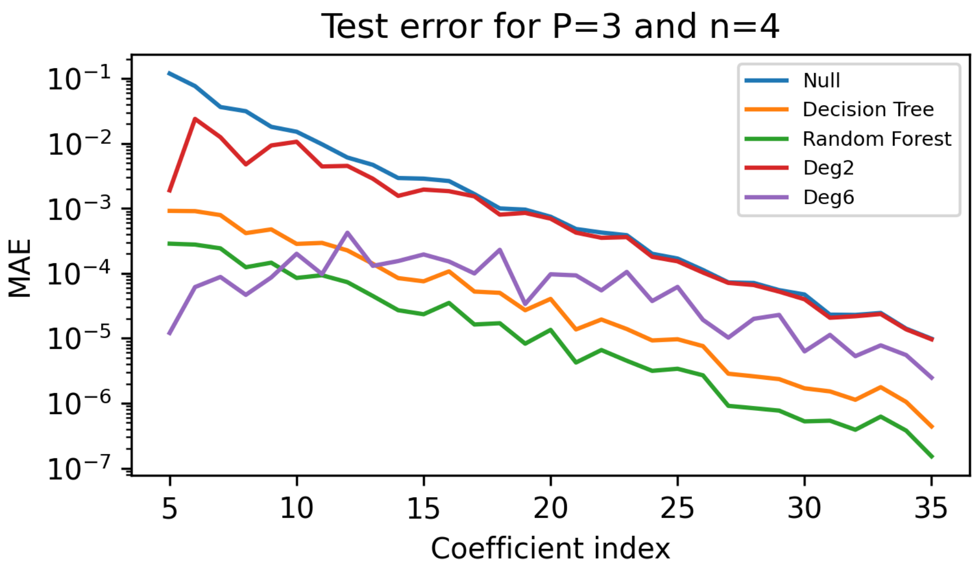

Figure 15: Mean absolute error on a test set for different ML models for $P = 3$ and $n = 4$.

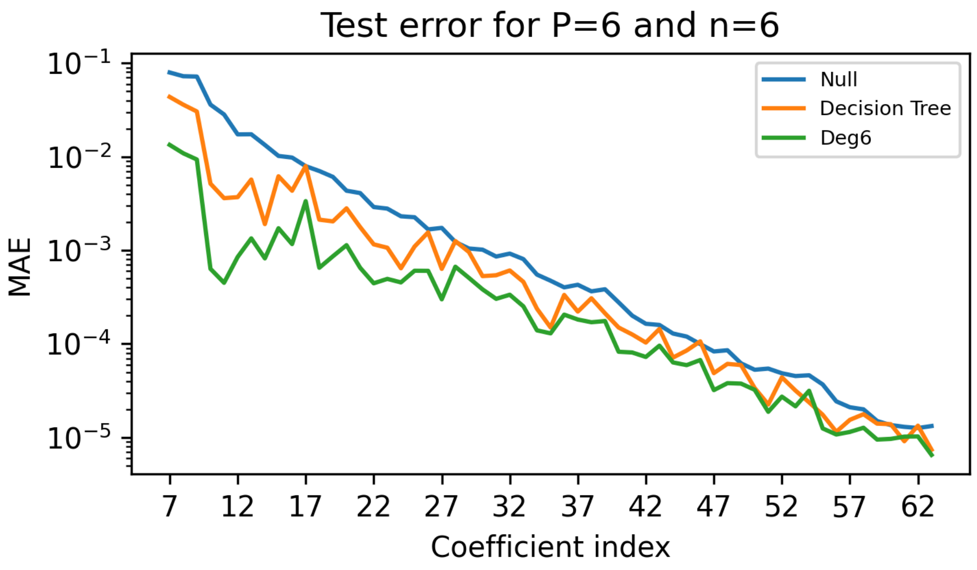

Figure 16: Mean absolute error on a test set for different ML models for $P = 6$ and $n = 6$.

4.5 Nonlinear compressive RBM in action

After generating the reduced basis as described in section 3 and training the regression model as described in section 4.4, we can now move to the online phase. This phase consists of solving the nonlinear system 48 for a given parameter value μ. This can be done using iterative methods such as Picard iteration, Newton's method, or any other nonlinear solver. In our numerical experiments, we use 'fsolve' from the 'scipy.optimize' package in Python, which is based on the Powell hybrid method and is more robust than the Picard iteration, especially for the extrapolation problem of the polynomial regression models discussed in section 4.4.

Our test case 2 is chosen to verify the fundamental dependence and the applicability of the nonlinear compressive RBM. However, the original problem is elliptic and linear which makes the classical RBM very efficient. For this reason, we will not perform time-based comparisons between the classical and the nonlinear compressive RBM. A more relevant test case to show the efficiency of the nonlinear compressive RBM is hyperbolic transport problems.

Theoretically, once N is fixed, the nonlinear compressive RBM should retrieve the same accuracy as the classical RBM. This is true if we have a perfect regression model. However, in practice, as we have seen in section 4.4, we had some issues with the regression model, especially for higher dimensions. This regression error dominates the final error in all our numerical experiments.

For the 1-dimensional case ($P = 1$), we have seen that using a spline regression model gives perfect accuracy. With this regression model, we can now retrieve same accuracy as the classical RBM with $N = 8$ modes. This is illustrated in table 2.

For the 2-dimensional case ($P = 2$), we have seen that using a decision tree regression model gives a good but not sufficient accuracy. To reach the same accuracy as the classical RBM with $N = 22$ modes, we need to increase n to 6 and use a polynomial regression model of degree 4. As we already mentioned, this is not the best choice since the dependence of the high RB coefficients is on the first two coefficients. This is illustrated in table 3. As expected, for $P > 2$, the results become less accurate and the regression error dominates the final error.

Table 2: Nonlinear compressive RBM results for $P = 1$

Model	**Mean Energy Error**	**Mean L2 Error**	**Max L2 Error**
Classical RBM (with 8 modes)	1.22×10^{-5}	6.80×10^{-6}	3.86×10^{-5}
Classical RBM (with 1 mode)	5.96×10^{-1}	6.33×10^{-1}	2.68×10^{0}
Spline	1.22×10^{-5}	6.80×10^{-6}	3.86×10^{-5}
Decision tree	1.22×10^{-5}	6.84×10^{-6}	3.87×10^{-5}
Degree 6 regression	3.78×10^{-3}	1.96×10^{-2}	3.78×10^{-3}

Table 3: Nonlinear compressive RBM results for $P = 2$

Model	**Mean Energy Error**	**Mean L2 Error**	**Max L2 Error**
Classical RBM (with 22 modes)	7.46×10^{-6}	2.59×10^{-6}	5.12×10^{-6}
Classical RBM (with 6 modes)	2.35×10^{-2}	1.14×10^{-2}	1.98×10^{-2}
Classical RBM (with 2 modes)	2.42×10^{-1}	1.67×10^{-1}	3.49×10^{-1}
Degree 4 (n=6) regression	7.46×10^{-6}	2.59×10^{-6}	5.12×10^{-6}
Decision tree (n=2)	1.55×10^{-3}	6.74×10^{-4}	2.84×10^{-3}

4.6 Implementation

In this section, we present the implementation of the classical and the nonlinear compressive RBM. The implementation is done in python language and relies on the **Feel++** library for the finite element method (FEM) computations more specifically on the linear-bilinear forms API. It consists of a generic (problem independent) class **Crb**. This class provides all the methods needed to compute both high fidelity (FEM here) and reduced basis solutions. The main methods are:

- **decompose**: A vitual method that should be implemented in the problem dependent subclass. It is used to specify the weak formulation of the problem in affine decomposition form.

- **solve**: Solve the FEM problem for a given parameter value μ, this method returns the solution $u_h(\mu)$ and the output $s_h(\mu)$.
- **POD**: Generate $N-$dimensional reduced basis space X_N using a given training set. The number of modes N can be chosen by the user or determined using the RIC criterion to achieve a target accuracy ϵ_{POD}.
- **assembleRbOffline**: Assemble all the offline matrices $\mathbf{A}_N^q$ and $\mathbf{f}_N^q$ in order to perform an efficient online phase. This method is called only once in the offline stage.
- **assembleOfflineNonlinear**: Assemble the offline matrices $\mathbf{B}^q$ needed for the nonlinear compressive RBM. This method is called only once in the offline stage (only for the nonlinear compressive RBM).
- **assembleOnlineAN**: Assemble the online matrix $\mathbf{A}_N^\mu$ for a given parameter value μ from the offline matrices $\mathbf{A}_N^q$. This method is called in the online stage. Additionally, it can be called to assemble a submatrix $\mathbf{A}_n^\mu$ of $\mathbf{A}_N^\mu$ for the nonlinear compressive RBM.
- **assembleOnlinefN**: Assemble the online vector $\mathbf{f}_N^\mu$ for a given parameter value μ from the offline vectors $\mathbf{f}_N^q$. This method is called in the online stage. Additionally, it can be called to assemble a subvector $\mathbf{f}_n^\mu$ of $\mathbf{f}_N^\mu$ for the nonlinear compressive RBM.
- **assembleOnlineB**: Assemble the online matrix $\mathbf{B}^\mu$ for a given parameter value μ from the offline matrices $\mathbf{B}^q$. This method is called in the online stage (only for the nonlinear compressive RBM).
- **onlineSolve**: Solve the online problem of the classical RBM for a given parameter value μ, this method returns the solution $u_N(\mu)$ and the output $s_N(\mu)$.
- **onlineSolveNonlinear**: Solve the online problem of the nonlinear compressive RBM for a given parameter value μ. The user can choose the nonlinear solver to use (Picard iterations by default). This method requires a regression model already trained (with a predict method). It returns the solution $u_N(\mu)$ and the output $s_N(\mu)$.

Since the class **Crb** is case-independent, we need to create a subclass for each problem we want to solve and implement a few virtual methods. In the following, we give an example of a subclass for our test case 2.

We create a subclass **Fin** where we set the problem-specific parameters and methods. In particular, in **decompose()** method we precise the weak formulation (in affine decomposition form) using **Feel++** bilinear and linear forms. For our test case, the affine decomposition is already given in section 3.5. This can be implemented as follows:

```
def decompose(self):
# assemble rhs
self.Fq=[]
for i,mat in enumerate(['Gamma_root']):
    self.Fq.append(self.lap.assembleFlux(markers=[mat],coeffs=1))
    self.f_thetas.append(lambda mu: 1)
# assemble the bilinear forms
self.Aq=[]
for i,mat in enumerate(['Post']+[f'Fin_{j}' for j in range(1,self.nfins+1)
    ]):
    C=np.array([[1,0],[0,1]])
    self.Aq.append(self.lap.assembleGradGrad(markers=[mat],coeffs=C))
    if i == 0 or i >= self.nfins+1:
        self.a_thetas.append(lambda mu: 1)
    if i > 0 and i < self.nfins+1:
        self.a_thetas.append((lambda index: lambda mu: mu[index-1])(i))
for i,mat in enumerate(['Gamma_ext']):
    self.Aq.append(self.lap.assembleMass(markers=[mat],coeffs=1))
    self.a_thetas.append(lambda mu: mu[self.nfins])
```

This affine decomposition will be used to ensure efficient reduced basis approximation through the offline-online decomposition. For the truth solver, we can simply reassemble the bilinear and linear forms for each new parameter value μ and solve the problem using **solve()** method. While for the reduced basis solver, we have to perform the offline phase before solving the reduced problem.

5 Quadratic approximation

In 1, we discussed in detail the limitations of the classical RBM and the notion of the Kolmogorov barrier. Then, we briefly introduced some nonlinear approximation techniques before presenting the nonlinear compressive RBM. Here we present the quadratic approximation method proposed in [1], study its advantages and limitations, and compare it to the nonlinear compressive RBM with a quadratic regression model. We will change the notations to be consistent with the ones used in [1]. Here, N and n don't have the same meaning as in the previous sections. We recall that the classical RBM approximates a high fidelity solution (in a vector form) $u \in \mathbb{R}^N$ by linear approximation $\tilde{u} \in \mathbb{R}^N$:

$$\tilde{u} = Vq \tag{49}$$

where $V \in \mathbb{R}^{N\times n}$ is the reduced order basis, $n \ll N$ and $q \in \mathbb{R}^n$ is the vector of coordinates in the reduced basis, known as vector of generalized coordinates.

In the quadratic method, the idea is to add a quadratic correction term to the linear approximation in (49). This approach can be motivated from a Taylor expansion viewpoint. The linear part of the approximation can be computed in the same way as we have done so far, typically using an SVD of the snapshot matrix. Then, the quadratic term is computed by minimizing the error between the high-fidelity solution and the corresponding linear approximation. Following this, the quadratic approximation is given by:

$$\tilde{u} = Vq + H[q \otimes q] \tag{50}$$

where $H \in \mathbb{R}^{N\times n^2}$ and $\otimes$ represents the vectorized Kronecker product such that $q \otimes q \in \mathbb{R}^{n^2}$. Note that if we denote n_1 the number of basis functions needed to achieve a certain accuracy using the classical RBM, then the number of basis functions needed to achieve the same accuracy using the quadratic approximation is $n_2 \approx \sqrt{n_1}$.

5.1 Two-step construction of a quadratic approximation

The main step in the construction of the quadratic approximation is the computation of the matrix H. Here, we present a two-step approach to compute H as proposed in [1].

In the first step, we compute the matrix $V \in \mathbb{R}^{N\times n}$, $n \ll N$, using an SVD of a snapshot matrix $S \in \mathbb{R}^{N\times N_s}$, where N_s is the number of high fidelity solutions collected.

In the second step, $H \in \mathbb{R}^{N\times n^2}$ is determined such that for all solution snapshots in S, the errors of the quadratic approximation (50) are minimized.

The two-step aspect of the approach outlined above for determining the matrix coefficients of the quadratic solution approximation (50) has an advantage over an alternative approach where both V and H are simultaneously determined. Indeed, the two-step approach allows a full column rank and orthogonality of V, for example, $V^T V = I$, which is numerically more convenient.

In practice, for each solution snapshot u_l collected in $S \in \mathbb{R}^{N \times N_s}$, let $e_l \in \mathbb{R}^N$ denote the error associated with its orthogonal projection on the subspace represented by V – that is,

$$e_l = u_l - V q_l, \quad \forall l = 1, \ldots, N_s \tag{51}$$

the vector of generalized coordinates q_l is computed such that the corresponding error e_l is orthogonal to the subspace represented by V and if by construction V is orthogonal, it follows from (51) that

$$q_l = V^T u_l, \quad \forall l = 1, \ldots, N_s \tag{52}$$

Assembling all approximation errors (51) in the matrix $E \in \mathbb{R}^{N \times N_s}$ and all vectors of generalized coordinates (52) in $Q \in \mathbb{R}^{n^2 \times N_s}$ as follows

$$E = [e_1, \ldots, e_{N_s}], \quad Q = [q_1 \otimes q_1, \ldots, q_{N_s} \otimes q_{N_s}] \tag{53}$$

constructing the matrix H can be written as:

$$H = \arg \min_{H' \in \mathbb{R}^{N \times n^2}} \|E - H'Q\|_F \tag{54}$$

Let $[E]_{i,1}$ and $[H]_{i,1}$, $i = 1, \ldots, N$ denote the row partitioning of the matrices E and H respectively. Using this partitioning, the minimization problem (54) can be rewritten as:

$$H = \arg \min_{H' \in \mathbb{R}^{N \times n^2}} \sum_{i=1}^{N} \|[E]_{i,1} - [H']_{i,1} Q\|_2^2 \tag{55}$$

from 55, it follows that the minimization problem is separable in the rows of H and can be solved independently for each row (embarrassingly parallel). We have these N least squares problems:

$$h_i = [H]_{i,1} = \arg \min_{h'_i \in \mathbb{R}^{1 \times n^2}} \|[E]_{i,1} - h'_i Q\|_2^2, \quad i = 1, \ldots, N \tag{56}$$

We can see that each column block $q_l \otimes q_l$ in Q contains n^2 entries. However, due to symmetry, $n(n-1)/2$ of these are repeated entries. The matrix Q is then ill-conditioned. This specific issue can be addressed by simply excluding all duplicate entries from the construction of Q to obtain in principle a redundancy-free matrix $\bar{Q} \in \mathbb{R}^{n(n+1)/2 \times N_s}$. In this case, the corresponding entries in each row vector hi are also excluded from its definition to obtain a new row vector $\bar{h}_i \in \mathbb{R}^{1 \times n(n+1)/2}$.

Depending on the combination of the number of collected solution snapshots Ns and the target dimension of the linear part n, each regularized least squares problem in (56) can be underdetermined, square, or overdetermined. Typically, $Ns > n(n+1)/2$, which leads to an overdetermined system (more equations than unknowns). In this case, an approximate solution of each least squares problem in (56) can be obtained using an SVD of the matrix Q based on the pseudo-inverse. The approximate solution of the least squares problems in (56) is then given by:

$$\bar{h}_i = U \Sigma^{-1} V^T [E]_{i,1}^T \tag{57}$$

where $Q = U \Sigma V^T$ is the SVD of $\bar{Q}$.

5.2 Comparison with the nonlinear compressive RBM

The nonlinear compressive RBM with a general regression model such as random forests or neural networks is more general than both quadratic approaches and reduces the complexity to some $n \approx P$. However, if we choose a quadratic regression model nothing guarantees that the nonlinear dependence of the high RB coefficients on the first n coefficients is quadratic. That is why we need to increase n to make the regression model more flexible and get better accuracy. For the quadratic approximation in [1], the basis functions in H are optimized for a quadratic dependence, which is not the case for the nonlinear compressive RBM where the basis functions associated to the quadratic term are the rest of the POD modes.

Even if none of these two quadratic approaches can be seen as a generalization of the other, we can see that the quadratic term $q \otimes q$ is homogeneous in the coordinates q and don't cover constant or linear terms which is not the case for the quadratic model used in the nonlinear compressive RBM. In this particular sense, the nonlinear compressive RBM with a quadratic regression model is more general than the quadratic approximation in [1].

To illustrate this, we consider a simple test case where the parametric solutions are given by:

$$u(c) = c\phi_1 + (\alpha + \beta c + \gamma c^2)\phi_2 \tag{58}$$

Here the parameter is $\mu = c$. We recall that in both quadratic approaches, an SVD is first performed. It is clear that only 2 modes are needed to capture the solutions in (58). However, nothing guarantees that these 2 modes will be ϕ_1 and ϕ_2 and that the quadratic dependence of the second coefficient will be preserved after the SVD. To make the test more realistic, we add some constraints and we choose ϕ_1, ϕ_2 and α, β, γ in a way that the we get the same expression as in (58) after the SVD. We impose the following constraints:

$$\begin{aligned} &\int_\Omega \phi_1\phi_1 dx = 1, \quad \int_\Omega \phi_2\phi_2 dx = 1, \quad \int_\Omega \phi_1\phi_2 dx = 0, \\ &\int_{c_{min}}^{c_{max}} c^2 dc = 1, \quad \int_{c_{min}}^{c_{max}} (\alpha + \beta c + \gamma c^2)^2 dc = 1, \quad \int_{c_{min}}^{c_{max}} c(\alpha + \beta c + \gamma c^2) dc = 0 \end{aligned} \tag{59}$$

Finally, to control the order of ϕ_1 and ϕ_2 in the SVD, we add some weights λ_1 and λ_2 such that $\lambda_1 > \lambda_2 > 0$. And our test case becomes:

$$u(c) = \lambda_1 c\phi_1 + \lambda_2(\alpha + \beta c + \gamma c^2)\phi_2 \tag{60}$$

where the constraints in (59) are satisfied.

With the test case in (60), applying an SVD to a snapshot matrix will give approximately ϕ_1 and ϕ_2 as the first two modes. To test the limitation of the quadratic approximation in [1] discussed above, we consider $n = 1$. It is clear that the nonlinear compressive RBM with a quadratic regression model will retrieve exactly the second coefficient and thus the exact solution. However, the quadratic approximation in [1] will not be able to retrieve the constant term α and the linear term βc in the second coefficient. This is illustrated in table 4.

Table 4: Comparison between the quadratic approximation in [1] and the nonlinear compressive RBM with a quadratic regression model

Model	**Mean L2 Error**
Classical RBM (with 2 modes)	≈ 0
Classical RBM (with 1 mode)	0.0012
Quadratic approximation	0.0012
NL compressive RBM with quadratic model	3.6933×10^{-12}

5.3 Enhanced quadratic approximation

The quadratic approximation in [1] has some limitations, one of them is that the quadratic term $q \otimes q$ doesn't cover constant or linear terms as we have seen in the previous section. To overcome this limitation, we propose an enhanced quadratic approximation where we add to the formulation in 50 a constant term and a linear term. The enhanced quadratic approximation is given by:

$$\tilde{u} = Vq + H[q \otimes q] + C + Lq \tag{61}$$

where $C \in \mathbb{R}^N$, $L \in \mathbb{R}^{N \times n}$. The entries of C, L, and H are simultaneously determined by minimizing the error between the high-fidelity solution and the corresponding linear approximation, the minimization problem (54) becomes:

$$\min_{H,C,L} \|E - HQ - \mathbf{C} - L\mathbf{q}\|_F \tag{62}$$

where $\mathbf{C} = [C, \ldots, C] \in \mathbb{R}^{N \times N_s}$ and $\mathbf{q} = [q_1, \ldots, q_{N_s}] \in \mathbb{R}^{n \times N_s}$. This problem also can be solved row by row as a least squares problem. If we denote E_i, H_i, L_i the i^{th} row of E, H, L respectively and c_i the i^{th} element of C, the least squares problem for the i^{th} row is:

$$\min_x \|Ax - E_i\|_2 \tag{63}$$

where $A = [1_{N_s}, \mathbf{q}^T, Q^T] \in \mathbb{R}^{N_s \times (1+n+n(n+1)/2)}$ and $x = [c_i, L_i^T, H_i^T]^T \in \mathbb{R}^{1+n+n(n+1)/2}$.

Finally, if we apply the enhanced quadratic approximation to the test case in (60) with $n = 1$, we will retrieve the same performance as the nonlinear compressive RBM with a quadratic regression model. This is illustrated in table 5.

Table 5: Comparison between quadratic methods

Model	**Mean L2 Error**
Classical RBM (with 2 modes)	≈ 0
Classical RBM (with 1 mode)	0.0012
Quadratic approximation	0.0012
NL compressive RBM with quadratic model	3.6933×10^{-12}
Enhanced quadratic approximation	3.7027×10^{-12}

Note that the error in the enhanced quadratic approximation and the nonlinear compressive RBM with a quadratic regression model is not exactly zero due to the fact that we perform an SVD on a finite number of snapshots so the modes are not exactly ϕ_1 and ϕ_2.

6 Conclusion

During this project, we have studied the classical reduced basis method which is the most basic and widely used reduced order modeling technique. We have seen that this method has some limitations, especially when it comes to hyperbolic and/or high-dimensional problems. We have then mentioned some nonlinear approximation techniques to overcome these limitations. We have focused on the nonlinear compressive RBM, a method that combines the reduced basis method with machine learning techniques. We have presented the mathematical formulation, the algorithm, and the implementation of this method. A multi-parameter test case has been used to illustrate all the presented concepts. Finally, we have compared the nonlinear compressive RBM with another nonlinear approximation technique, the quadratic approximation, for which we have proposed an enhanced version.

The next step is to apply this method to more realistic test cases, especially hyperbolic transport problems, and to use more advanced regression models such as neural networks.

References

[1] Joshua Barnett and Charbel Farhat. Quadratic approximation manifold for mitigating the kolmogorov barrier in nonlinear projection-based model order reduction. *Journal of Computational Physics*, 464:111348, 2022. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2022.111348. URL `https://www.sciencedirect.com/science/article/pii/S0021999122004107`.

[2] Joshua Barnett, Charbel Farhat, and Yvon Maday. Neural-network-augmented projection-based model order reduction for mitigating the kolmogorov barrier to reducibility. *Journal of Computational Physics*, 492:112420, 2023. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2023.112420. URL `https://www.sciencedirect.com/science/article/pii/S0021999123005156`.

[3] Gérard Biau. Analysis of a random forests model. *Journal of Machine Learning Research*, 13(38):1063–1095, 2012. URL `http://jmlr.org/papers/v13/biau12a.html`.

[4] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[5] Albert Cohen, Charbel Farhat, Agustín Somacal, and Yvon Maday. Nonlinear compressive reduced basis approximation for PDE's. working paper or preprint, March 2023. URL `https://hal.science/hal-04031976`.

[6] Jens L. Eftang, Anthony T. Patera, and Einar M. Rønquist. An hp certified reduced basis method for parametrized parabolic partial differential equations. In Jan S. Hesthaven and Einar M. Rønquist, editors, *Spectral and High Order Methods for Partial Differential Equations*, pages 179–187, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[7] Andrea Ferrero, Tommaso Taddei, and Lei Zhang. Registration-based model reduction of parameterized two-dimensional conservation laws. *Journal of Computational Physics*, 457:111068, 2022. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2022.111068. URL `https://www.sciencedirect.com/science/article/pii/S0021999122001309`.

[8] Simone Fresca, Luca Dede', and Andrea Manzoni. A comprehensive deep learning-based approach to reduced order modeling of nonlinear time-dependent parametrized pdes. *Journal of Scientific Computing*, 87(61), 2021. doi: 10.1007/s10915-021-01462-7. URL `https://doi.org/10.1007/s10915-021-01462-7`.

[9] Jan Hesthaven, Gianluigi Rozza, and Benjamin Stamm. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations.* 01 2016. ISBN 978-3-319-22470-1. doi: 10.1007/978-3-319-22470-1.

[10] Angelo Iollo and Damiano Lombardi. Advection modes by optimal mass transfer. *Phys. Rev. E*, 89:022923, Feb 2014. doi: 10.1103/PhysRevE.89.022923. URL `https://link.aps.org/doi/10.1103/PhysRevE.89.022923`.

[11] Kookjin Lee and Kevin T. Carlberg. Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. *Journal of Computational Physics*, 404:108973, 2020. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2019.108973. URL `https://www.sciencedirect.com/science/article/pii/S0021999119306783`.

[12] Yvon Maday and Benjamin Stamm. Locally adaptive greedy approximations for anisotropic parameter reduced basis spaces. *SIAM Journal on Scientific Computing*, 35(6):A2417–A2441, 2013. doi: 10.1137/120873868. URL `https://doi.org/10.1137/120873868`.

[13] Christophe Prud'Homme, Yvon Maday, and Hassan Ballout. Nonlinear compressive reduced basis approximation for multi-parameter elliptic problem. working paper or preprint, July 2024. URL `https://hal.science/hal-04628481`.

[14] Christophe Prud'homme et al. feelpp/feelpp: Feel++ v111 preview 9, 2024. URL `https://zenodo.org/records/10837178`. Accessed: 2024-07-22.

[15] Alfio Quarteroni, Andrea Manzoni, and Federico Negri. *Reduced basis methods for partial differential equations: An introduction*. 01 2015. ISBN 978-3-319-15430-5. doi: 10.1007/978-3-319-15431-2.

[16] Tommaso Taddei. A registration method for model order reduction: Data compression and geometry reduction. *SIAM Journal on Scientific Computing*, 42(2):A997–A1027, 2020. doi: 10.1137/19M1271270. URL `https://doi.org/10.1137/19M1271270`.

[17] Taddei, T., Perotto, S., and Quarteroni, A. Reduced basis techniques for nonlinear conservation laws. *ESAIM: M2AN*, 49(3):787–814, 2015. doi: 10.1051/m2an/2014054. URL `https://doi.org/10.1051/m2an/2014054`.