Structural Wing Model Reduction in Fluid-Structure Interactions

STUDENT: Oriol CHANDRE-VILA
TUTORS: Joseph MORLIER (ISAE-SUPAERO) and Sylvain DUBREUIL (ONERA)

June 27, 2018

Abstract

This paper focuses the use of Reduced Order Models to lighten the computational requirements in fluid-structure problem simulations. Proper Orthogonal Decomposition and Proper Generalized Decomposition methodologies are studied and applied in a Steady Advection-Diffusion problem for 2D and non-homogeneous Dirichlet boundary conditions. Thanks to the above-mentioned approaches, important reductions in computing time have been achieved.

Keywords: Reduced Order Models; Proper Orthogonal Decomposition; Proper Generalized Decomposition; Fluid-Structure Interactions.

1 Introduction

1.1 Motivations

In front of a more and more demanding industrial world, it is expected to compute more precise and wider designs. Consequently, the computational power required is increasingly important. In this connection, Model Order Reduction (hereafter MOR) is a technique for reducing the computational complexity of mathematical models in numerical simulations. By a reduction of the model's associated state space dimension or degrees of freedom, an approximation to the original model is computed, which normally is less expensive [1]. Consecutively, two approaches exist to reduce the number of random variables under consideration by obtaining a set of principal variables: ***(a)*** feature selection and ***(b)*** feature extraction. In this analysis, we are interested in the latter which involves reducing the amount of resources required to describe a large set of data. General dimensionality reduction techniques are used in different methodologies [2].

1.2 Background

1.2.1 Several Reduced Order Models (ROM) techniques

The first category of ROM analyzed -which is widely used- is the *Principal Component Analysis* (PCA). PCA is a statistical procedure which converts a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables -called principal components- by using an orthogonal transformation. This transformation is defined such the first principal component has the largest variance (how much the data could variate), and the succeeding components must be orthogonal to their precedent while having the highest variance [3]. Furthermore, depending on the field of application, this technique has received different names. Just the fields in which this paper is interested are listed: ***(a)*** singular value decomposition (SVD) in linear algebra [4], ***(b)*** proper orthogonal decomposition (POD) in mechanical engineering [5] [6], ***(c)*** discrete Karhunen–Loève transform (KLT) in signal processing [7], ***(d)*** robust PCA -which uses the L^1 instead of the norm L^2- [4] and, ***(e)*** factor analysis in psychometrics [8]. The different fields usually aspire to evolve their problem to an algebraic issue in order to use SVD technique. The aforementioned ROM is defined as a factorization of a matrix into a number of constitutive components all of which have a specific meaning in applications [4]: $\mathbf{A} = \mathbf{U} \cdot \mathbf{\Sigma} \cdot \mathbf{V}^*$, with $\mathbf{U} \in \mathbb{C}(mxm)$, $\mathbf{\Sigma} \in \mathbb{R}(mxn)$ where $(\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_{p=min(m,n)} \geq 0)$, and $\mathbf{V} \in \mathbb{C}(nxn)$. In modern partial SVD algorithms, the properties of random vectors are exploited to efficiently build a subspace that captures the column space of a matrix [9].

Some limitations can be appreciate in PCA. Firstly, the results depend on the scaling of the variables. Secondly, the applicability of PCA is limited by certain assumptions made in its derivation. Thirdly, in fields where all the signals are non-negative and the mean-removal process forces the mean of some exposures to be zero (which consequently creates unphysical negative fluxes), it is necessary a forward modelisation to recover the true magnitude of the signals. Finally, since PCA does not explicitly attempt to model difference between classes of data, *Linear Discriminant Analysis* (LDA) was

developed [10]. The above-mentioned works by creating one or more linear combinations of predictors, creating a new latent variable for each discriminant function. In general terms, the first function created maximizes the differences between groups on that function; and the following functions must not be correlated with the previous functions while maximizing the differences [11].

The second category of ROM analyzed is the *Independent Component Analysis* (ICA). This computational method separates a multivariate signal into additive subcomponents, and it is a special case of blind source separation. On one hand, ICA is based on two assumptions: ***(a)*** the source signals are independent of each other and ***(b)*** the values in each source signal have non-Gaussian distributions. On the other hand, three effects may appear due to mixing the source signals: ***(a)*** *independence*: the source signals are independent but their signal mixtures are not; ***(b)*** *normality*: the distribution of a sum of independent random variables with finite variance yields to a Gaussian distribution, and, ***(c)*** *complexity*: the complexity in time of any signal mixture is greater than that of its simplest constituent source signal [12].

The third category is not a ROM it-self. *Dynamic Mode Decomposition* (DMD) is a matrix decomposition technique that is highly versatile and builds upon the power of SVD. DMD appears as an alternative ***(a)*** when the model equations are beyond their range of validity or ***(b)*** when the governing equations simply are not known or well-formulated [4]. By computing the eigenvalues and eigenvectors of a linear model that approximated the underlying dynamics -even if the system is nonlinear-, experimental data itself drives the understanding of the system [4]. DMD could be perceived as minimizing the norm of the difference between the stored experimental data in $k+1$ and the approximative dynamics obtained by the linear expression $\mathbf{x}_{k+1} = \mathbf{A} \cdot \mathbf{x}_k$; over $k = 1, 2, 3, ..., m-1$ snapshots. The rank q of a POD-projected matrix $\tilde{\mathbf{A}}$ would be iterated until achieving the required precision while accomplishing the lowest q [13]. Despite the fact that it fails immediately if the data matrix is full rank and the data has no suitable low dimensional structure, it is possible to highlight three advantages of DMD: ***(a)*** no equations are needed and ***(b)*** the future state is always known.

The fourth (and last) category analyzed is not a ROM it-self too. Since solving decoupled problems is computationally much less expensive than solving multidimensional problems, *Proper Generalized Decomposition* (PGD) is usually considered a dimensionality reduction algorithm. Thus, this numerical method for solving boundary value problems assumes that the solution of a multidimensional (or multiparametric) problem can be expressed in a separated representation, of the form:

$$\mathbf{u}(x_1 ... x_d) = \sum_{i=1}^{N} \mathbf{F}_i^1(x_1) \times \cdots \times \mathbf{F}_i^D(x_d) \qquad (1)$$

where the number of terms N and the functions $\mathbf{F}_i$ are a priori unknown. At a particular enrichment step $n+1$ of (1), the functions are known for $i \leq n$ from the previous steps. So it should be only computed the new product involving the D unknown functions $\mathbf{F}_{n+1}^j(x_j)$ [14]. In a problem defined as $\mathbf{u}(x,y) = \sum_{i=1}^{N} \mathbf{X}_i(x) \cdot \mathbf{Y}_i(y)$, PGD uses the weak problem formulation (see Algorithm 1).

Algorithm 1 PGD. [14]

Input: Data stored in $u^{n,p}(x,y)$.

1. Calculating $\mathbf{X}_n^p(x)$ from $\mathbf{Y}_n^{p-1}(y)$:

$$u^{n,p}(x,y) = \sum_{i=1}^{n-1} \mathbf{X}_i(x) \cdot \mathbf{Y}_i(y) + \mathbf{X}_n^p(x) \cdot \mathbf{Y}_n^{p-1}(y) \quad (2)$$

2. Calculating $\mathbf{Y}_n^p(y)$ from $\mathbf{X}_n^p(x)$.

$$u^{n,p}(x,y) = \sum_{i=1}^{n-1} \mathbf{X}_i(x) \cdot \mathbf{Y}_i(y) + \mathbf{X}_n^p(x) \cdot \mathbf{Y}_n^p(y) \quad (3)$$

Return: $\mathbf{X}_n^p(x)$ and $\mathbf{Y}_n^p(y)$

Within PGD strategy, $(d \cdot M \cdot N)$ unknowns factors are concerned instead of M^d unknowns of a traditional finite element (FE) method (M is the number of nodes, d is the dimension of the problem and N is the number of summands). In the Table 1, it can be seen the saving in computational costs of PGD. The column M^d shows the cost of a FEM, the column $(d \cdot M \cdot N)$ of PGD and the column $max(N)$ represents the number of summands required for achieving $M^d \equiv (d \cdot M \cdot N)$ [15].

Table 1: *Cost computational comparison [15].*

d	M^d **(M=100)**	$(d \cdot M \cdot N)$	$max(N)$
3	10^6	$300 \cdot N$	3333
5	10^{10}	$500 \cdot N$	$2{\times}10^4$
⋮	⋮	⋮	⋮
40	10^{80}	$4{\times}10^3 \cdot N$	$2,5{\times}10^{76}$

The most advantageous application of PGD is the evaluation of design parameters as new coordinates of the problem. So that, in real-time, it will be possible to evaluate the effect of any modification. This approach is possible thanks to robust offline calculations: a general, static and expensive solution is computed and stored in $\{\mathbf{R}, \mathbf{S}, \mathbf{T}, \cdots, \mathbf{Z}\}$ vectors of the Greedy algorithm -shown in (4).

$$\mathbf{u}^{N+1}(x) = \mathbf{u}^N(x) + \mathbf{R}(x_1)\mathbf{S}(x_2)\mathbf{T}(x_3)\cdots\mathbf{Z}(x_d) \qquad (4)$$

Then, by multiplying and adding the components of these vectors, an online solution is reconstructed.

1.2.2 Computing Performance Analysis of ROM

Reduced Basis (RB) methods, which represent a remarkable instance of ROM techniques, were used to exploit the parametric dependence of the PDE solution by combining a handful of high-fidelity (HF) solutions computed for a set of parameter values. By this approach, a very large algebraic system is replaced by a much smaller one. If N and affine operator components Q_a are small enough, very fast response can be achieved for real-time problems using a Galerkin reduced basis (G-RB). Quaternoni et al., took advantage of a suitable offline/online computational splitting formulation to compute error bounds efficiently: ***(a)*** computing the norm of the residual (see Algorithm 3.4, p. 62 in [16]); and, ***(b)*** computing the stability factor β_h. In the latter, a *Interpolatory Radial Basis Functions* was selected. In order to guarantee the positivity of the interpolant, it was interpolated the logarithm of β_h rather than factor itself. This leads to a symmetric linear system which was solved in the offline phase to yield the interpolation weights (see Algorithm 3.7, p. 67 in [16]).

Thereupon, the G-RB was tested on a steady heat conduction-convection problem satisfied by the non-dimensional temperature u [16]. In addition, the problem was discretized by piecewise linear FE using a mesh made of $\approx 2,2\times10^5$ tetrahedral elements. Thanks to the offline/online decomposition, for any given parameter value $\boldsymbol{\mu}$ at the online stage, the solution can be obtained very rapidly (as can be observed in the Table 2). The offline/online decomposition is a direct consequence of the affine parametric dependence property, which allows to split the assembly of the reduced matrices and vectors.

Table 2: *Computational details for the HF and RO models of the advection-diffusion problem. [16]*

High-fidelity model		Reduced-order model	
Number of FE dofs N_h	44171	Number of RB dofs	29
Affine operator components Q_a	2	Dofs reduction	1520:1
Affine RHS components Q_f	6	Offline CPU time	≈ 5 *min*
FE solution time (assembly + solution)	$\approx 3,5$ *s*	Online CPU time	1 *ms*

The analysis is concluded with several comments. Firstly, it is said that the solution $u_h(\boldsymbol{\mu})$ is highly dependent of $\boldsymbol{\mu}$. Secondly, as can be seen in Figure 1, the estimator $\Delta_N(\boldsymbol{\mu})$ is very sharp and its effectivity $\eta_N(\boldsymbol{\mu})$ is very closed to 1. And thirdly, the error present an exponential decay, which is a remarkable property of the RB approximations. Consequently, they are very accurate and efficient.

Continuing analyzing the current problem, a POD was used to construct a RB approximation. POD approach uses the error estimator $\Delta_N(\boldsymbol{\mu})$ to acquire a more accurate result. Quarteroni et al., used *Latin Hypercube Sampling* (LHS) because it is a very good sampling technique

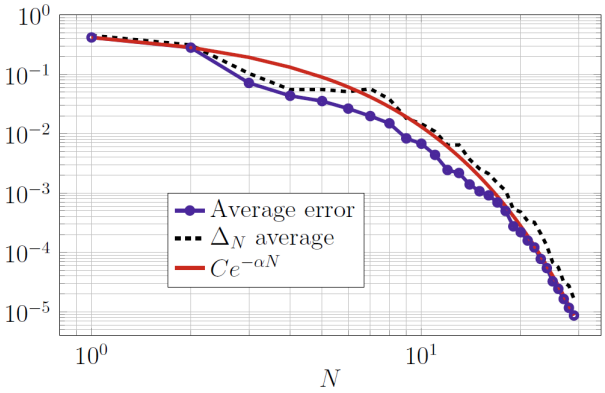

Figure 1: *Comparison of the average error $\|u_h(\boldsymbol{\mu}) - u_N(\boldsymbol{\mu})\|_V$ and bound $\Delta_N(\boldsymbol{\mu})$ computed on a set of 350 random parameter values.[16]*

for initial points. It was reported in Figure 2 the spectrum obtained using LHS with $n_s = 25, 50, 100, 200, 5000$. The snapshot of $n_s = 100$ was used to extract a RB because the spectrum obtained is already sufficiently accurate. The RB extracted required a dimension of $N = 26$ to achieve $\varepsilon_{POD} = 10^{-5}$.

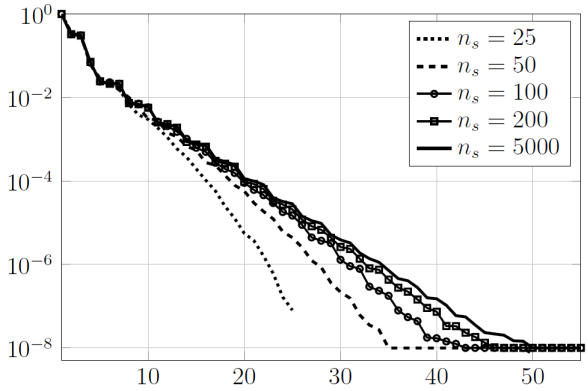

Figure 2: *First 55 singular values of the correlation matrix obtained by LHS.[16]*

To construct RB spaces, greedy algorithms might be also used. This option can be evaluated if POD method entails a severe computational cost. Greedy algorithms represent an efficient alternative as they allow the construction of the reduced space by minimizing the amount of snapshots to be evaluated (by seeking at each step the local optimum). Nevertheless, two potentially critical aspects have to be considered. A greedy algorithm: ***(a)*** must be supported by a posteriori error estimated with $\|u_h(\boldsymbol{\mu}) - u_N(\boldsymbol{\mu})\|_V$ in a very inexpensive way (which has demonstrated to be less accurate); and, ***(b)*** is not necessarily cheaper than a POD. In that point, it is essential to make greedy algorithms competitive with POD.

Remaining in the advection-convection problem, a greedy algorithm was introduced to construct a RB approximation. Firstly, it was performed a preprocessing step to build the RBF interpolant of the stability factor

$\beta_h(\boldsymbol{\mu})$. Since the problem is coercive and it depends only on the μ_4 (number of sampling points) direction, equidistributed interpolation points along the above-mentioned direction were chosen. Using the Algorithm 7.3 (p. 146 in [16]), the convergence history of the greedy algorithm had been reported (see Figure 3). This historical evolution had been performed with different train sample (Ξ_{train}) made of $n_{train} = 10^2, 10^3, 10^4$ points selected by LHS; stopping at $N = 27$ in the first case and at $N = 29$ in the other cases.

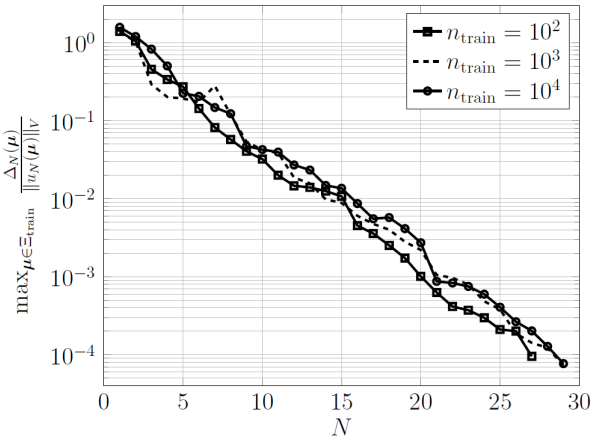

Figure 3: *Convergence history of the greedy algorithm.[16]*

1.2.3 ROM in Aeroelasticity

The final objective of the study concerned within this paper, is the definition of a strategy based on ROM to attack Aeroelastic problems. In the paper published by Lucia et al., a POD produced an accurate model for the coupled system of an aeroelastic panel in crossflow [17]. In concert with the POD, a domain decomposition (DD) was considered necessary in order to commit the overcoming difficulties in translating discontinuities in the flowfield.

For the **Structural dynamics equations**, the panel dynamics were computed with von Karman's large-deflection plate equation (for the theoretical development: [17]); which yields to the structural state $\mathbf{Y}_s = \begin{bmatrix} w(x) \\ s(x) \end{bmatrix}$:

$$\dot{\mathbf{Y}}_s = \begin{bmatrix} 0 & L_N \\ I & 0 \end{bmatrix} \mathbf{Y}_S + \begin{bmatrix} \mu\mathbf{P} \\ 0 \end{bmatrix} \tag{5}$$

$$L_N = -\frac{\mu}{\lambda}\frac{\partial^4}{\partial x^4} + N_x \frac{\partial^2}{\partial x^2} \tag{6}$$

Where: $w(x)$ are the panels deflections, $s(x) = \dot{w}(x)$ are the velocities, $\mathbf{I}$ is the identity matrix, L_N is a nonlinear matrix operator (where $\partial/\partial x$ represents central-difference, spatial discretization), the array $\mathbf{P}$ collecting the values of $\left(\frac{1}{\gamma M_\infty^2} - P\right)$ at the panel grid points, μ is the mass ratio, λ is the non-dimensional dynamic pressure, γ is the ratio of specific heats and M_∞ is the Mach number.

Equations (5) and (6) were integrated in time using an Euler implicit method (1st order accurate in time). Besides, using a large number of structural grid points reduced the effects of translating fluid values to structural nodes.

For the **fluid equations**, the dynamics are governed by the 2-D Euler equations for inviscid flow. Assuming no grid deformation and the integral form of the Euler equations reduce to the following, an overall vector of flow variables $\mathbf{U}(t)$ was produced by collocating variables at each spatial cell location to the corresponding column vector location, using row-by-row ordering (for the theoretical development: [17]). After separating the spatial and the time derivatives, spatial terms were grouped to form a nonlinear operator R acting on the set of fluid variables:

$$\frac{d\mathbf{U}(t)}{dt} = R[\mathbf{U}(t)] \tag{7}$$

While the structural grid used 141 nodes along the solid surface, 116 nodes were extended to the freestream. The spacing of the grid in the normal direction was determined to increase geometrically from the solid wall, while in the streamwise direction was determined as constant ($\Delta_{wall} = 0,0125$).

Through the transpiration boundary condition, the structural system of (5) was loosely coupled to the fluid system (7). In order to avoid time-lagging errors that could arise in the coupling scheme, an extrapolated panel pressure value was used ($\mathbf{P}_{ex}^{n+1}$):

$$\mathbf{P}_{ex}^{n+1} = \mathbf{P}^n + \Delta\mathbf{P}^n = 2\mathbf{P}^n - \mathbf{P}^{n-1} \tag{8}$$

Here, the fluid system dominated the computational time required to produce solutions for the coupled system. Thus, a ROM of the flowfield was coupled to the full-order structural model in order to produce an accurate aeroelastic panel response with less computational expense.

As dynamic cases moving grids require special treatment in the POD/ROM formulation, the solution domain was divided into three sections to facilitate the use of POD with the moving transonic shock. The solution domain was divided into three regions: *section I* or far field, *section II* or near field, and, *section III* or shock region. The section II had been solved more frequently than the far field to update the internal boundary shared with section III. In its turn, the shock region contained the flow over the panel and confines entirely the moving shock of interest. No overlap is included in this decomposition.

Thus, to analyze the POD/ROM/DD three smaller fluid problems were solved (see Figure 4). These problems

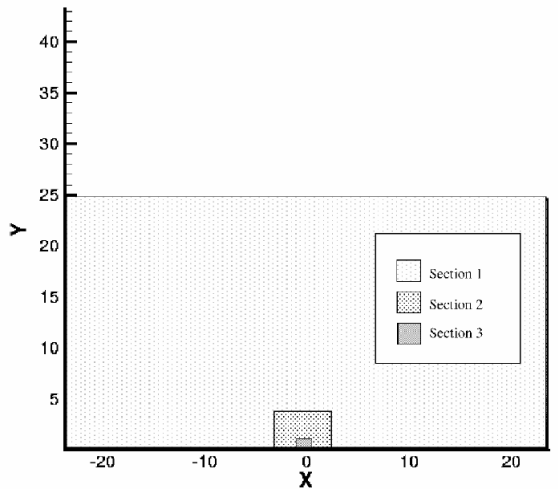

Figure 4: *Domain Decomposition (DD) used in the POD/ROM/DD approach. [17]*

were linked by internal boundaries. For this fluid variable, the full-system dynamics in (7) are depicted hereafter: $\frac{d\mathbf{w}}{dt} = R(\mathbf{w})$. POD produced a linear transformation Ψ between the full-system solution $\mathbf{w}$ and the reduced-order solution $\hat{\mathbf{w}}$:

$$\mathbf{w}(t) = \mathbf{W}_0 + \Psi\hat{\mathbf{w}}(t) \tag{9}$$

Where: $\Psi = SV \rightarrow S^T SV = V\Lambda$ (Λ is the diagonal matrix of eigenvalues). In the study presented, both implicit and explicit time-accurate methodologies were used: ***(a)*** *implicit* for the nonshocked regions (sections I and II), and, ***(b)*** *explicit* for the shock region because the moving discontinuity was a too strong nonlinearity for the first scheme. Nevertheless, some constraints were modeled with a implicit POD/ROM scheme. They were introduced to enforce smoothness on the internal boundary between adjacent domains ($\mathbf{U}_{S1}$ and $\mathbf{U}_{S2}$ with a shared internal boundary Γ). Two constraints options were considered:

$$C^1(\mathbf{U}) = \sum_{k\in\Gamma} \mathbf{U}(k)_{S1} - \mathbf{U}(k)_{S2} = 0 \tag{10}$$

$$C^2(\mathbf{U}) = \sum_{k\in\Gamma} [\mathbf{U}(k)_{S1} - \mathbf{U}(k)_{S2}]^2 = 0 \tag{11}$$

On one hand, the first constraint was similar to requiring the mean difference to be zero on the boundary (approximating the L^1 norm). On the other hand, the second constraint was similar to requiring the variance to be zero on the boundary (approximating the L^2 norm). While the first constraint could be precomputed and does not require updates during the time integrations, the second constraint required that the Jacobian was reformed for each iteration.

This article is considered of interest because it exemplifies a possible treatment of an Aeroelastic problem through ROM. Although, this is an old article which means that the topic has probably seen multiple updates, it can easily be seen that the strategy of ROM does not need to apply the whole problem.

2 Methodology

In order to exemplify this section, the problem presented in 3.1.

2.1 Reference Solution

Extracted from Amsallem et al., a code for this problem has been acquired. This code is necessary in order to validate and compare the processes that are tested (see algorithm 2).

Algorithm 2 Steady Advection-Diffusion reference case algorithm. [18]

Input: Value of the non-homogeneous Dirichlet boundary conditions, value of the parameters and mesh data.

1. Compute matrix $\mathbf{A}_1$. It is a matrix of the second-order derivatives using a centered scheme.
2. Compute matrix $\mathbf{A}_2$. It is a matrix of the first-order derivatives using an upwind scheme.
3. Compute $\mathbf{A}$: $\kappa \cdot \mathbf{A}_1 - \mathbf{u} \cdot \mathbf{A}_2 = \mathbf{A}$.
4. Compute $\mathbf{b}$. It is a vector where the Dirichlet boundary conditions are forced. The other nodes are defined as zero.
5. Solve the problem: $\mathbf{w} = \mathbf{A}^{-1} \cdot \mathbf{b}$

Return: Sample vector $\mathbf{w}$

As it can be seen in the algorithm just presented, the problem is solve at once. No iterations are needed.

2.2 POD case

To perform the approach of the POD in our test case, it has been used the algorithm 3 where a Greedy algorithm is added to the POD treatment. This algorithm can be found in Amsallem et al.

The algorithm presented, it does not apply a completely random first sample. In fact, each initial parameter value has been fixed to the mean value of its range. Besides, the Neumann boundary conditions have been forced within the matrix $\mathbf{A}_2$.
Since, it is a general algorithm, algorithm 3 has been applied to each candidate solution after solving the problem using the algorithm 2 at each candidate.

The goal of this algorithm is to train the model. Here, *train* means to build the best ROB possible in order to, then, create approximative (but faster) solutions. To

Algorithm 3 POD training algorithm. [18]

Input: Parameters range. Here, $\boldsymbol{\mu}$ represents the parameters vector.

1. Select randomly a first sample: $\boldsymbol{\mu}^{(1)}$
2. Solve the HDM: $f(\mathbf{w}(\boldsymbol{\mu}^{(1)}, \boldsymbol{\mu}^{(1)}) = 0$
3. Build a corresponding ROB: $\mathbf{V}$
4. For $i = 2, \cdots, s$
   (a) Solve: $\boldsymbol{\mu}^{(1)} = \underset{\boldsymbol{\mu} \in \{\mu_1, \cdots, \mu_C\}}{\arg\max} ||\mathbf{r}(\boldsymbol{\mu})||$
   (b) Solve the HDM: $f(\mathbf{w}(\boldsymbol{\mu}^{(i)}, \boldsymbol{\mu}^{(i)}) = 0$
   (c) Build a ROB $\mathbf{V}$ based on the samples: $\{\mathbf{w}(\boldsymbol{\mu}^{(1)}), \cdots, \mathbf{w}(\boldsymbol{\mu}^{(i)})\}$

Return: $\mathbf{V}$

achieve this objective, 1000 candidates have been used to create a huge amount of data (cases). In our case, the POD has been trained using 10 iterations (the variable s in the step 4 of the algorithm 3 is defined as 10). In Figure 5 this enrichment process is presented. The idea, as explained in algorithm 3, is to create the first sample *randomly* and from the second sample until the s sample, just add a column to the RB $\boldsymbol{V}$. At each iteration s, a SVD needs to be applied to the system. For instance, the fourth iteration is adding a column to the left-singular vectors resulting from the third POD. Thus, the Greedy algorithm is optimizing the data each iteration using a ROM.

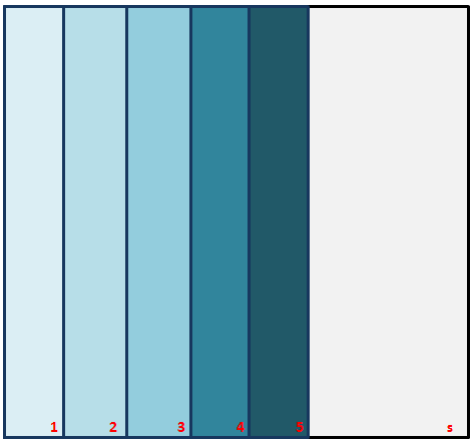

Figure 5: *Scheme for the enrichment process for POD presented in the algorithm 3.*

Once the model has been trained, the desired case can be run. The way to proceed is basically transforming the solution of each case (solved with algorithm 2) by multiplying the $\boldsymbol{V}$ resulting from the algorithm 3.

2.3 PGD case

The second method tested has been the PGD method.

2.3.1 Offline stage

In this case, two approaches have been applied. The first one is the algorithm 1, already presented. Nevertheless, it is a too intrusive method and it would force us to code and to define a new PGD script for each new problem. Besides, in our specific problem, the convective terms introduce non-symmetry to the solution, which obscure the methodology.
Thus, a new approach has been used in order to achieve a more general method of the PGD methodology. Presented in the sixth chapter of Chinesta et al. book [14], the *Residual Minimization for Non-Symmetric Operators* accelerates the convergence of the problem. Briefly, in a problem defined as $\mathbf{A} \cdot \mathbf{X} = \mathbf{F}$, it is sought the solution $\mathbf{X}$ that minimizes the residual norm $||\mathbf{A} \cdot \mathbf{X} - \mathbf{F}||^2$. Notably, an Alternating Direction Scheme is also applied, see Algorithm 4. Here, $\mathbf{Re}$ represents the residual.

Algorithm 4 Alternating Direction Scheme Algorithm for a two-parameters PGD. [14]

Input: The two vectors $\mathbf{R}$ and $\mathbf{S}$ for the previous iteration $n-1$.

1. Using $\mathbf{S}^{n-1}$, $\mathbf{R}^n$ is computed using a symmetric linear algebraic system $\frac{\partial ||\mathbf{Re}||^2}{\partial \mathbf{R}} = 0$.
2. Using $\mathbf{R}^n$, $\mathbf{S}^n$ is computed using a symmetric linear algebraic system $\frac{\partial ||\mathbf{Re}||^2}{\partial \mathbf{S}} = 0$.

Return: Vectors $\mathbf{R}$ and $\mathbf{S}$ for the current iteration n.

The current offline methodology requires defining the problem in a matrix way, and then a solver -which must never be modified- is used. This solver needs five constructors to work and it is provided by Chinesta et al.

Firstly, in a Matlab environment, these 5 constructors are defined as *cell structures* and are the following:

- **AA**: it is called *the operator* and its dimension are $[5 \times 4]$ matrices. The dimensions can be easily justified. On one hand, in the current Advection-Diffusion Problem, five coordinates are defined (x, y, u_1, u_2, κ). On the other hand, two different phenomena are found (advection and diffusion) for two different coordinates (x and y); so that, the problem is divided in x-advection, y-advection, x-diffusion and y-diffusion.
- **BB**: here, the source term are defined. As the source term is redefined in a separated form, the extra-coordinates terms (in our case u_1, u_2, κ) must not modify the solution. Hence, the cell structure understands five vectors, three of which are filled with

ones. Finally, as in our problem no source term is defined in the space domain, the source terms which apply for x and y are represented by an array filled with zeros.

- **N_NT**: for a more general problem, the mass matrices need to be considered within the solver. In our particular case, this cell structure has 5 eye matrices which dimensions depends on the number of each parameter discretization.
- **Dirichlet**: this structure defines the index where the Dirichlet boundary conditions are found. Normally, it has as cells as the number of coordinates of the problem. If no Dirichlet BC are found in a coordinate, the array is left as empty.
- **GG**: here, 5 vectors are defined to force Dirichlet BC. As in the cell **BB**, arrays corresponding to direction where no Dirichlet BC are defined are filled with ones. Nevertheless, if some coordinate needs Dirichlet BC, they must be specified here. For instance, in our problem, the second cell must contain the non-homogeneous boundary temperature.

Then, the whole problem is solved and stored in a cell structure named **FF**. The application of Dirichlet BC in the final results can be seen when representing each column of the different cell's matrix (see Annex: Figures 12, 13, 14, 15 and 16). If there is any Dirichlet BC in any parameter, the first column of the ***FF*** correspondent cell is the correspondent ***GG*** array.

2.3.2 Online stage

With the offline stage computed, it is only needed to launch as many solutions as desired following the algorithm 5.

Algorithm 5 Online PGD.

Input: The solved offline problem saved within **FF**.

1. Definition of the value of the parameters u_1, u_2, κ. This values can be found in the indexes iU, iV, iK of the corresponding **FF**'s cell.
2. Computation the weight of the solution: $\alpha = \mathbf{FF}\{3\}(iU) \cdot \mathbf{FF}\{4\}(iV) \cdot \mathbf{FF}\{5\}(iK)$.
3. Computation of the solution for the whole domain: $\mathbf{T} = \mathbf{FF}\{2\}' \cdot \alpha \cdot \mathbf{FF}\{1\}$.

Return: The temperature field **T**.

2.4 Numerical aspects

On one hand, POD runs the algorithm a predefined number of times. Thus, no stopping criterion is needed. The predefined number of POD computation is 1000.

On the other hand, PGD process applies stopping criterion in two different loops. The loop ranked lower is the one for the fixed-point algorithm (inside the enrichment process). Here, a tolerance of 10^{-8} is sought for the following Stopping Criterion[14] (**FF** answers for the solution):

$$\frac{||\mathbf{FF}_n^p - \mathbf{FF}_n^{p-1}||}{||\mathbf{FF}_n^{p-1}||} \tag{12}$$

Then, for the Enrichment loop, a simplified Stopping Criterion[14] has been applied with a sought tolerance of 10^{-5}. The simplified criterion is defined as the norm of the current enrichment iteration n divided by the norm of the first enrichment iteration:

$$\frac{||\mathbf{FF}_n||}{||\mathbf{FF}_1||} \tag{13}$$

In both criterion, a limit number of iterations is defined in order to force the exit.

3 RESULTS

3.1 Steady Advection-Diffusion Problem

One and only problem has been treated with the ROM. This problem has been extracted from Amsallem's notes [18]. The problem is a Steady Advection-Diffusion problem (14) applied in a closed domain with non-homogeneous Dirichlet boundary conditions (see Figure 6).

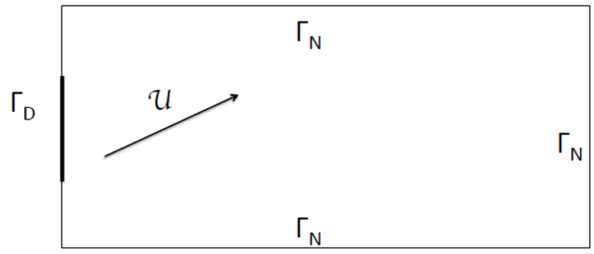

Figure 6: *Steady Advection-Diffusion Problem.[18]*

$$\mathbf{u} \cdot \nabla T - \kappa \cdot \Delta T = 0 \tag{14}$$

Where the range of the parameters are defined below:

$$\begin{cases} \mathbf{x} = (x, y) \in [0, 1] \times 0, 1] \\ \mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}; u_1, u_2 \in [0, 0.5] \\ \kappa \in [0, 0.025] \end{cases} \tag{15}$$

The boundary conditions of the problem are expressed in (16).

$$\begin{cases} T(\{x, y\} \in \Gamma_D) = T_D(y, \bar{y}) \\ \nabla T(\{x, y\} \in \Gamma_N) \cdot \mathbf{n}(\mathbf{x}) = 0 \end{cases} \tag{16}$$

With the non-homogeneous boundary temperature defined as below:

$$T_D(y,\bar{y}) = \begin{cases} 300, & \text{if } y \in [0, 1/3] \\ 300 + 325 \cdot (sin(3\pi \cdot |y-\bar{y}| + 1), & \text{if } y \in [1/3, 2/3] \\ 300, & \text{if } y \in [2/3, 1] \end{cases} \quad (17)$$

Where $\bar{y} = 0.4$.

3.2 Reference Solution

Due to the need of validating the solution while solving both POD and PGD strategies, one case has been used. To define it, the values selected are $u_1 = 0.11\ ^m/_s$, $u_2 = 0\ ^m/_s$ and $\kappa = 0.025\ ^m/_{s^2}$.
As it has been explained in algorithm 2, firstly a huge matrix were all the cases are solved is computed. For the current solution, it has been solved for a mesh of 61×61 elements, the dimensions of the matrix used to compute the solution are 3481×3481.
In Figure 7, the case solution is presented. In order to

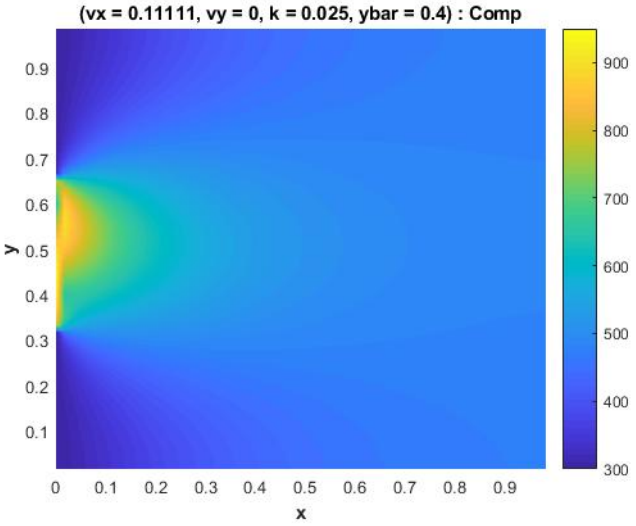

Figure 7: *Computational reference solution for* $u_1 = 0.11\ ^m/_s$, $u_2 = 0\ ^m/_s$ *and* $\kappa = 0.025\ ^m/_{s^2}$ *with a mesh of* 61×61 *elements.*

understand the physics of the problem, in the Figure 10 in the Appendix, the reader can find several cases where different parameters have been changed: u_1, κ and $\bar{y}$. This solution has been extracted from [18].

3.3 POD Solution

As in the reference solution, all the cases have been computed. Nevertheless, as it is the goal of the POD, the matrix used to compute the solution has been reduced. In the POD case, the matrix has a dimension of 3481×10. The size reduction is consistently translated to a reduction in the computation time and data storing space, but it also implies that an error due to the loss of information. In pursuance of considering an average behavior, the POD process has been performed 20 times and an average of the maximal error, an average of the average error and an average of the computation time have been determined. The results have revealed the following results: $E_{max}^{avg} = 7.13\%$ and $E_{avg}^{avg} = 0.94\%$. Besides, the time used to compute the results has been reduced by a 85%. Thus, the results justify the use of a POD strategy if a maximum error of 7.13% is affordable.
In Figure 8, the case solution is presented.

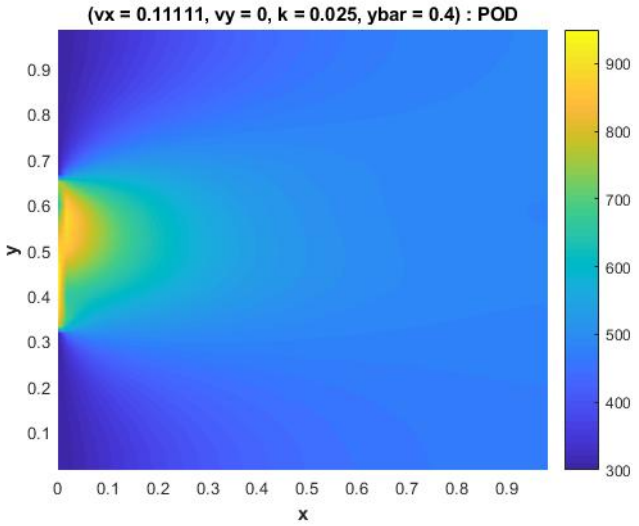

Figure 8: *POD solution for* $u_1 = 0.11\ ^m/_s$, $u_2 = 0\ ^m/_s$ *and* $\kappa = 0.025\ ^m/_{s^2}$ *with a mesh of* 61×61 *elements.*

To enlarge the study, the reader can find represented in Figure 11 of the Appendix the different modes of the POD based RB $\boldsymbol{V}$.

3.4 PGD Solution

The PGD methodology must be analyzed twice (offline and online stage). Firstly, the phase online is evaluated.

Once the problem is solved offline for a wide range of values for all the extra-coordinates it is really fast to get results. Particularly, each solution can be solved with less than half of a second, which represents a reduction of nearly 95%.

Secondly, the offline phase must be also analyzed. First of all the computation time, which has increased the time from nearly 10 seconds to 45 minutes[1]. Furthermore, the dimensions of the data for this case varies depending on the number of iterations needed to reach the tolerance of the enrichment process. However, it is easily seen that the amount of data is bigger than in the

[1]Please note that these values are referred to the used computer. This is not absolute result.

other cases since it must be added the dimensions of all the solution matrices. There are 5 matrices of dimension $[N_{elements}^{parameter} \times n_{iterations}]$.

Nevertheless, the results of the offline phase cannot be accepted because of an issue in the Stopping Criterion of the enrichment process: the tolerance is not reached. Although, the evaluation of the whole methodology could be conducted, the validation of the results could not. Hence, while the fixed-point algorithm reaches the desired tolerance, the enrichment process does never reach the specified tolerance before 1500 iterations -the trigger value for stopping the enrichment iterations if the tolerance is not achieved. Since lowering the exigency in the tolerance has not reached the required level of accuracy of the solution, a proper Stopping Criterion for the critical loop should be defined in order to achieve the success.
Just for exemplifying this issue, in Figure 9 the PGD case solution is presented.

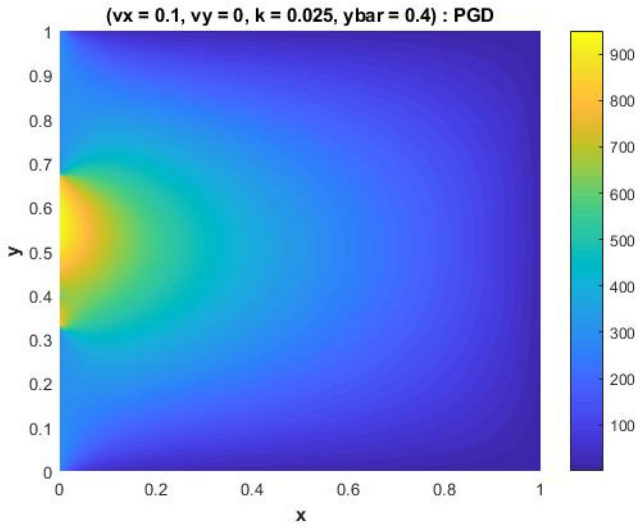

Figure 9: *PGD solution for $u_1 = 0.10$ m/s, $u_2 = 0$ m/s and $\kappa = 0.025$ m/s^2 with a mesh of 61 elements for x and y; 21 elements for u_1 and u_2; and 31 elements for κ.*

4 CONCLUSIONS

After the study performed during this article workout, one can conclude that the success of the POD can be easily proved. Besides, since the solver used to compute a problem by POD is not modified, it is just needed to add some lines of code to reduce the problem. Thus, it is calmly introduced in a already coded problem. In addition, Azaïez et al. in [19] have shown with some relevant numerical experiments that the recursive POD is computationally more accurate than the PGD for multivariate functions, recursively expanding the modes retained in the previous step.

On the contrary, some more efforts are needed to modify a problem to use the PGD. Furthermore, although it is really fast to achieve solutions in the online stage, it can be considered necessary only if a lot of cases of one problem are expected to be computed. Another really interesting application could be the use of the PGD to perform interactive computations or interactive trainings [15]. If the problem is not considered to be highly demanded, it could be better to consider a POD approach.[20]
Extending the PGD analysis, it is really interesting its application by a Parametric approach. The above-mentioned approach with the inclusion of the boundary conditions as extra-coordinates of the problem (either Neumann and Dirichlet) let the user to study a whole generic problem with only one robust offline computation. Nevertheless, the study of the effectiveness of the Stopping Criterion must be deeply analyzed in order to ensure the convergence.

Talking about the approach DMD, it is been seen that it is necessary a film of snapshots to create an effective reduced model. So that, the DMD can be an interesting alternative to POD or PGD when the solution has to be solved also in a time domain. The biggest advantage of the DMD is its ability to define the dimensions of the reduced model to achieve a predefined tolerance.

After the analysis, a final question can be considered: a strategy for Aeroelasticity problems using ROM is possible. Particularly, introducing a general ROM to a multidisciplinary problem could not be the most effective methodology. But, introducing ROM to each (or selected) one-discipline subproblem could worth the extra work of the engineer.
An interesting approach has been presented by Xia et al., in [21], producing a high quality representation of the indicator function of different phases of the material using a combination of POD and PGD.

To conclude, this project has been a first work in the reduced order models by the author and, during a summer internship in ONERA, this methodologies will be studied to be introduced to the open-source Aeroelastic solver *OpenAeroStruct*. To succeed with our goal, the study and definition of errors and Stopping Criterion will be a key aspect to deeply develop and adequately define.

For the codes and more information see the PIR's GitHub or go to the next *url*: `http://github.com/mid2SUPAERO/PIR_CHANDRE_ROM`.

References

[1] Baur, U., Benner, P. and Feng, L., "Model order reduction for linear and nonlinear systems: a system-theoretic perspective", Max-Planck-Institut für Dynamik Komplexer Technischer Systeme Magdeburg, MPIMD/14-07, March 2014.

[2] Reality AI Blog, "Its all about the future", Septembre 2017. [http://reality.ai/it-is-all-about-the-features/].

[3] Einasto, M., Liivamägi, L. J., Saar, E., Einasto, J., Tempel, E., Tago, E. and Martínez, V. J., "Principal component analysis", Astronomy & Astrophysics manuscript, August 2011.

[4] Kutz, J. N., "Data-Driven Modeling & Scientific Computation: Methods for Complex Systems & Big Data", 1st ed., Oxford University Press, Oxford, 2013.

[5] Chatterjee, A., "An introduction to the proper orthogonal decomposition", Current Science, April 2000.

[6] Megretski, A., "Proper Orthogonal Decomposition", Massachusetts Institute of Technology - Department of Electrical Engineering and Computer Science, October 2004.

[7] Ghoman, S.S., Wang, Z., Chen, P.C. and Kapania, R., "A POD-based Reduced Order Design Scheme for Shape Optimization of Air Vehicles", AIAA 2012-1808, April 2012.

[8] Child, D., "The Essentials of Factor Analysis", 3rd ed., Bloomsbury Academic Press, Juin 2006.

[9] Erichson, B., Brunton, S. L., Kutz, J. N., Voronin, S., "Randomized Matrix Decompositions using R", Journal of Statistical Software, April 2018.

[10] Martinez, A.M. and Kak, A.C., "PCA versus LDA", IEEE transactions on pattern analysis and machine intelligence VOL. 23 NO. 2, February 2001.

[11] Abdi, H., "Discriminant Correspondence Analysis", Encyclopedia of Measurement and Statistics, The University of Texas at Dallas, 2007.

[12] Isomura, T. and Toyoizumi, T., "A local learning rule for independent component analysis", Scientific Reports, June 2016.

[13] Kutz, J.N., Brunton, S.L., Brunton, B.W. and Proctor, J.L., "Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems", 1st ed., SIAM, Philadelphia, 2016.

[14] Chinesta, F., Keunings, R. and Leygue, A., "The Proper Generalized Decomposition for Advanced Numerical Simulations: A primer", 1st ed., Springer, 2014.

[15] Cours by Dr. Elias CUETO (Aragon Institute of Engineering Research, Universidad de Zaragoza), "Model Order Reduction. Proper Generalized Decomposition (PGD) methods", in Facultad de Matemáticas de la Universidad de Santiago de Compostela (USC), 2015.

[16] Quarteroni, A., Manzoni, A. and Negri, F., "Reduced Basis Methods for Partial Differential Equations. An Introduction", Unitext, vol. 92. Springer, 2016.

[17] Lucia, D.J., Beran, P.S. and King, P.I., "Reduced-Order Modeling of an Elastic Panel in Transonic Flow", Journal of Aircraft, vol. 40, No. 2, March-April 2003.

[18] Amsallem, D., "An Adaptive and Efficient Greedy Procedure for the Optimal Training of Parametric Reduced-Order Models", International Journal for Numerical Methods in Engineering, 2014.

[19] Azaïez, M., Ben Belgacem, F. and Chacon Rebello, T., "Recursive POD expansion for reaction-diffusion equation", Advanced Modeling and Simulation in Engineering Sciences, 2016.

[20] Valsecchi, L.M., "Reduced Order Methods for PDEs: a comparison between Proper Orthogonal Decomposition and Proper Generalized Decomposition", Corso di Laurea Magistrale in Ingegneria Aeronautica, Politecnico de Milano, 2015.

[21] Xia, L., Raghavan, B., Breitkopf, P. and Zhang, W., "A POD/PGD Reduction Approach for an Efficient Parameterization of Data-driven Material Microstructure Models", Computer Methods in Material Science, 2013.

Appendices

A Some solutions for the Problem

In order to understand the physics of the problem, using the reference code from Amsallem, several cases have been tested and are presented in Figure 10.

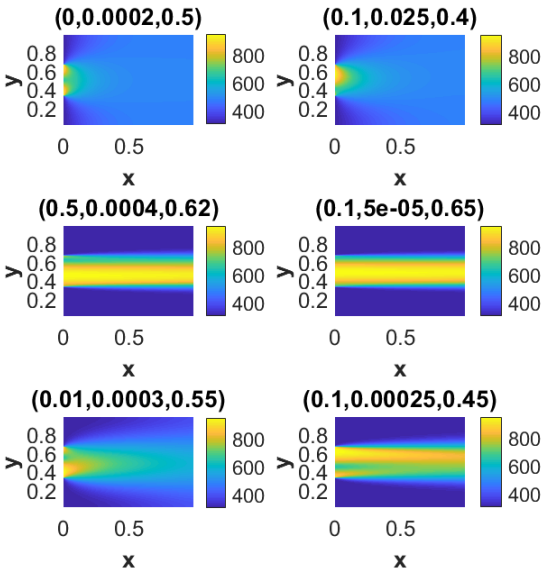

Figure 10: *Several solutions of the Advection-Diffusion problem. In the title: $(u_1,\ \kappa,\ \bar{y})$; $u_2 = 0$ for all the test cases.*

B POD Modes

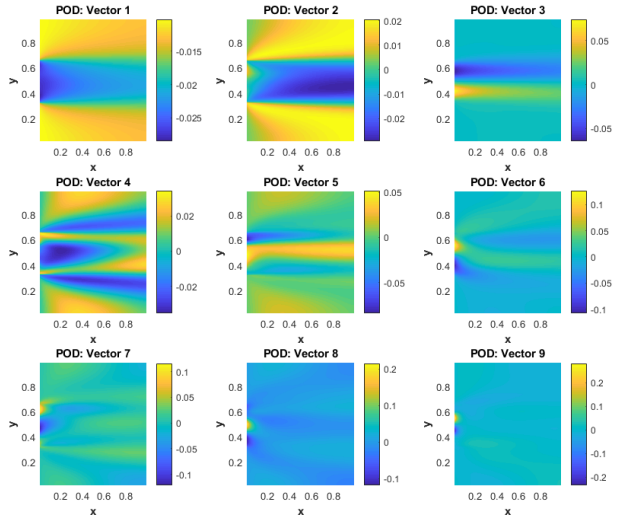

Figure 11: *The nine resulting modes of the Greedy+POD algorithm.*

C PGD solution normalized functions

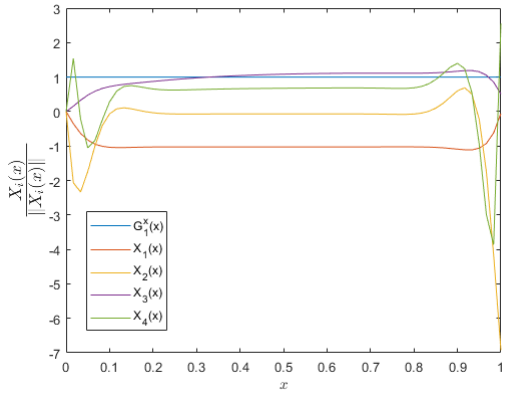

Figure 12: *Normalized functions $X_i(x)$ for $i = 1, \cdots, 5$ produced by the PGD solution of the Advection-Diffusion problem.*

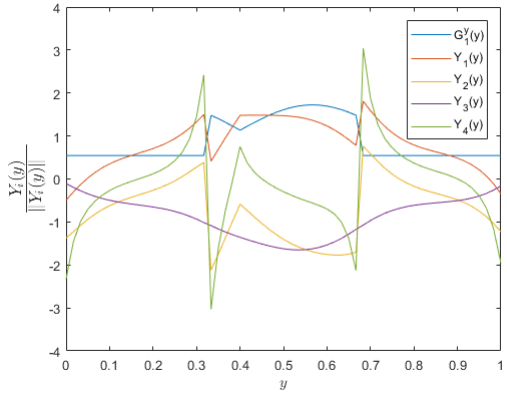

Figure 13: *Normalized functions $Y_i(y)$ for $i = 1, \cdots, 5$ produced by the PGD solution of the Advection-Diffusion problem.*

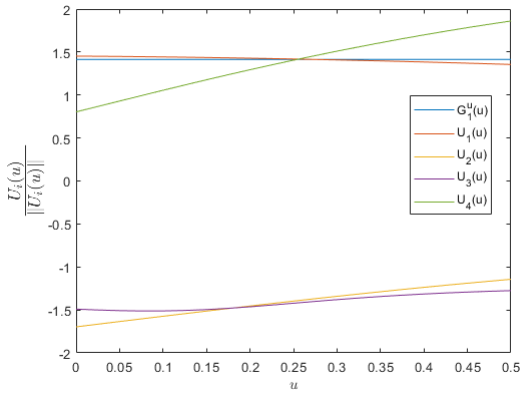

Figure 14: *Normalized functions $U_i(u_1)$ for $i = 1, \cdots, 5$ produced by the PGD solution of the Advection-Diffusion problem.*

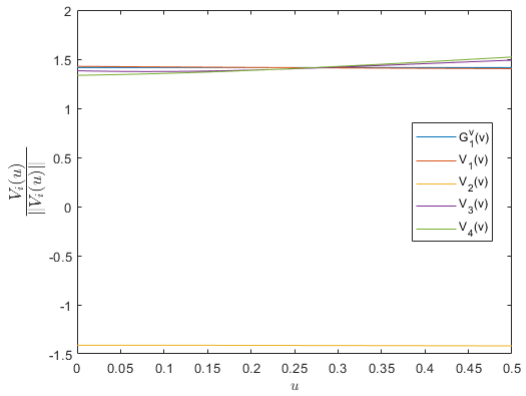

Figure 15: *Normalized functions $V_i(u_2)$ for $i = 1, \cdots, 5$ produced by the PGD solution of the Advection-Diffusion problem.*

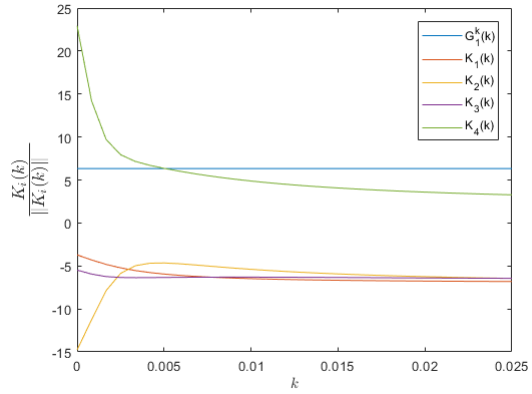

Figure 16: *Normalized functions $K_i(\kappa)$ for $i = 1, \cdots, 5$ produced by the PGD solution of the Advection-Diffusion problem.*