A new approximation framework for PGD-based nonlinear solvers

Matteo Capaldo

NNT : 2015SACLN011

THESE DE DOCTORAT
DE L'UNIVERSITE PARIS-SACLAY,
préparée à l'École Normale Supérieure de Cachan

ÉCOLE DOCTORALE N° 579
Sciences mécaniques et énergétiques, matériaux et géosciences

Spécialité de doctorat : Mécanique des solides

Par

M. Matteo Capaldo

A new approximation framework for PGD-based nonlinear solvers

Thèse soutenue à Cachan, le 23/11/2015 :

Composition du Jury :

M., Cueto, Elias	Professeur Université de Saragosse	Rapporteur
M., Dureisseix, David	Professeur INSA Lyon	Président
M., Guidault, Pierre-Alain	Maître de conférences ENS Cachan	Co-encadrant
M., Ladevèze, Pierre	Professeur ENS Cachan	Directeur de thèse
M., Legoll, Frédéric	Professeur EN des Ponts et Chaussées	Examinateur
M., Néron, David	Professeur ENS Cachan	Co-directeur de thèse
M., Ryckelynck, David	Professeur Mines ParisTech	Rapporteur

To Adriana, the reason of all my efforts

It is not simple at all to resume four years of emotions related to this fantastic experience in less than one page. It is impractical to include all the people I would in few lines full of *thanks to*. Anyway I make my best, hoping to be right.
I would like to say thank you to the members of my thesis jury for the time they spent analysing this document and also for the interesting discussions we had. In particular, thanks to David Dureisseix for having kindly accepted to be the president of the jury. With his experience, he made me comfortable during the defence. Thanks to David Ryckelynck and Elias Cueto for the attentive review and the interesting comments, helpful to improve the quality of the defence. Thanks to Frédéric Legoll, for having accepted to examine this document, offering his large experience in mathematics to my defence discussion.
Thanks to Pierre Ladevèze for all the trust he accorded to me and for this compelling subject. Thanks to Pierre-Alain Guidault for all the time spent together and his wise advices. Thanks to David Néron for the time spent working together, for his kindness and his contagious enthusiasm.
Thanks to the *big family* of the LMT of Cachan. It would be risky to start the complete list of people, I would overpass the size of the rest of the document..
but I need to say a special thank you to the group of Italians (+ the French/Italian Amaury), the group of Elena and *her boys*, the group of *immigrants* (my Argentinian, Balkan, Brazilian, Chinese, German, Lebanese, Tunisian, USA and Vietnamese friends) but also my French friends (thanks to all of you for your hospitality), the football *équipe*, the CdC habitants, the ones who attend the LMT on Saturday, for working, or just for a coffee, for a football match event or for a music concert. These have been the best moments..
I would like to thank Lydia for all the help, and also Frisou, Boubou (and our Skopsko's team), Stéphanie, Danielle, Jean-Pierre, Jean-Luc and Remy(x). I want to say thank you to Olivier A., for his joviality and help and also to Ludovic C., Federica D., Yoann G., Emmanuel B., Fabrice G., for their help preparing the defence. Then, Olivier H., Christine V. and Sophie G. of the EDSP, the director Frédéric R., and all the other people have spent a period at the LMT from the beginning of the 2012 and the end of 2015. 1600 Km far from my city, at the LMT I always felt home.
I would also like to remember the 5 months spent as an engineer in Expliseat S.A.S., saying thank you to all my ex-colleagues (specially to Joffrey, Thom and Antoine).
Thanks to the the roommates of Gentilly, I won't forget this *colocation*.
Thanks to Roberto, Pippo, Stefano and Peppino (thank you so much for all our dinners full of theories about everything).
I would say an affectionate thank you to my father, mother and sister.. Thank you for pushing me further than I ever thought I could go. Thank you also to the part of family who is not any more, but helps me to reach my goals.
Finally, thank you Adriana for being the ultimate reason to every effort.

Contents

Introduction

Recently, numerical simulation has been playing an increasingly important role in science and engineering because of the need to describe realistic scenarios and derive tools to facilitate the virtual design of new structures. The aim of this simulation is to reduce the use of real prototypes and assist the design phase. The ingredients of a model are the physical features of the problem and its mathematical description. Mathematically, many systems and processes in nature, applied science and engineering are modelled by partial differential equations (PDEs) that translate into mathematical language the physical phenomena. Often, the most important is not the description of the model in each point of the considered domain, but rather certain selected quantities called *outputs* in some particular zones of the field. These quantities, in the engineering area, can be energies, forces, critical stresses or strains, pressure drops, temperature, and flux. These outputs are functions of system parameters, called *inputs*, that identify a particular configuration of the model; typically, geometry, properties, or boundary conditions and loads. The inputs-outputs relationship incorporates the description and the behaviour of the considered model. The evaluation of the outputs needs the solution of the underlying PDE. This solution requires a computational cost that, in some engineering fields, such as aeronautics, can be very high, especially when the simulations concern nonlinear analyses of complex high-fidelity models. Moreover the engineering design and optimization could require thousands of this evaluation, sometimes, in real-time.

A possible answer to this issue is to provide to engineers a *Chart* (that, in this conditions, would be a *Virtual Chart*). This one is constructed in an *off-line* stage that can be CPU intensive. Once it is built, it enables one to have all the solutions for the considered model in an *on-line* stage, in order to solve real-time or many queries simulations.

Despite of the continuing progress in computer speeds and hardware capabilities, the construction of those charts would be still unsuitable for all the classical numerical approaches e.g., the finite element method (FEM), the boundary element method (BEM) or the finite volumes (FV) despite the fact that in the off-line part the time expenditure is not the priority, these techniques involve huge approximation subspaces for the underlying PDE and that leads to systems so large that direct techniques are inappropriate for the computational efforts that could be unreasonable.

Model reduction techniques constitute one of the tools to circumvent this obstacle by seeking the solution of a problem in a reduced-order basis (ROB), whose dimension is much smaller than the original vector space. These techniques take advantage of the redundancy of information that is usually used to describe the solution. In applied mathematics, the first proposed model reduction technique has been the Proper Orthogonal Decomposition (introduced as Singular Values Decomposition in [Beltrami, 1873] and then in [Jordan, 1874] also known as Principal Components Analysis [Wu et al., 2003] and Karhunen-Loeve Decomposition [Karhunen, 1943]). It is vastly used in different domains ([Chatterjee, 2000, Atwell and Kings, 2001, Kunish and Xie, 2005, Carlberg et al., 2010], see this review [Kerschen et al., 2005] for more examples). In order to solve a given problem, techniques based on POD involve a learning phase which consists in selecting, arbitrarily, some particular time instants and/or parameter values of the discretized domain and the full order model in space. At each of these particular values, the solution of the full order model is computed. These solutions are called snapshots and can be CPU-expensive. Then, a ROB is formed considering only the most relevant POD modes of these snapshots. The reduced-order model (ROM) is generated by projection on the ROB and solved for the entire time and/or parameter domain. The strong point of this technique is the fact that the number of the most relevant POD modes is much lower than the size of the full order model in space, but, on the other hand, it is *case sensitive* [Glüsmann and Kreuzer, 2009]: changes in the dynamic system behavior (e.g. changes in boundary conditions) can decisively affect the POD accuracy and the number of required snapshots can increase [Glüsmann and Kreuzer, 2009, Boucinha et al., 2014]. Since the 80's, another model reduction technique based on the projection has been developed for nonlinear analysis of structures: it is the Reduced Basis (RB) [Prud'homme et al., 2002, Maday and Ronquist., 2004, Veroy and Patera, 2005, Nguyen et al., 2005, Rozza, 2004, Grepl et al., 2007, Rozza and Patera, 2008, Nguyen, 2008, Buffa et al., 2012, Galvis and Kang, 2014]. The idea of this technique is to select the required snapshots by a greedy algorithm in the way that the new $(n+1)^{th}$ snapshot minimizes the residue, defined by the chosen norm, of the solution achieved solving the original problem projected onto the $n-$order reduced basis. This ensures the quality of the basis for the construction of the ROM but, on the other hand, in order to select a new snapshot it requires the resolutions of linear systems related to the reduced-order model over the whole parametric domain. This method palliates the *case-sensitivity* of the POD but, it is more CPU-intensive in the construction of the ROB.

In this work, Proper Generalized Decomposition (PGD) is considered. This technique does not need to solve the full order model because it does not need snapshots to build up the ROB. Basically, PGD consists in seeking the solution of a problem in a relevant ROB which is generated automatically and on-the-fly by a greedy algorithm. Firstly introduced in [Ladevèze, 1985] under the name *radial loading approximation*, it consisted in the separation of time and space variables. In that

period, the *radial loading approximation* was introduced as a part of the LaTIn (Large Time Increment) method [Ladevèze, 1999]. This is a non-incremental solver introduced for complex nonlinear problems which generates the approximations of the solution over the entire time-space domain by successive enrichments. Since its introduction in [Ladevèze, 1985, Ladevèze, 1989], the LaTIn method has been applied for various types of problems: elasto-plasticity [Boisse et al., 1990], elasto-viscoplastic problems [Boisse et al., 1990, Cognard and Ladevèze, 1993, Champaney et al., 1997], finite displacements [Michel-Ponnelle, 2001], contact problems [Roulet et al., 2011, Champaney et al., 1999, Giacoma et al., 2014, Giacoma et al., 2015], thermo-mechanical problems [Cognard et al., 1999, Ryckelynck, 2002], delamination and damage mechanics for composite materials [Allix et al., 1989, Allix and Ladevèze, 1992, Aubard et al., 2000], mesh adaptivity for visco-plastic problems [Pelle and Ryckelynck, 2000], post-buckling [Boucard et al., 1997], multiparametric problems [Boucard and Ladevèze, 1999] and multiphysics problems [Dureisseix et al., 2003, Néron and Dureisseix, 2008, Néron and Ladevèze, 2010]. The LaTIn method is also the numerical framework of a multiscale computational strategy for mechanical structures decomposed by a mixed domain decomposition developed at LMT Cachan [Ladevèze and Lorong, 1991, Blanze et al., 1996, Champaney et al., 1997, Dureisseix and Ladevèze, 1998, Ladevèze et al., 2001, Ladevèze and Nouy, 2003, Ladevèze et al., 2006, Guidault et al., 2008, Ladevèze et al., 2010, Cremonesi et al., 2013]. Another possibility offered by the LaTIn framework is to solve multiparametric models by exploiting the previous solution of the LaTIn method to initialize the solver for the new set of parameters (e.g. [Boucard and Ladevèze, 1999, Heyberger et al., 2011, Relun et al., 2011, Néron et al., 2015]). The LaTIn method represents a very convenient framework for the PGD. Indeed, PGD, mainly based on time-space separated representation, has been applied in many of the previous works enabling to decrease the necessary memory and the calculation time. The LaTIn-PGD computational strategy represents an efficient solver for nonlinear problems.

Subsequently, at LMT Cachan and in other research groups, a number of extensions have been proposed. The principal ones are the introduction of parameters of the model as additional coordinates in the PGD representation [Bognet et al., 2012, González et al., 2010, Chinesta et al., 2010, Ghnatios et al., 2012, Mokdad et al., 2007], developed in Chinesta's research group and real-time simulations, developed in Cueto's research group [Niroomandi et al., 2008, Monserrat et al., 2001, Niroomandi et al., 2012b]. Other extensions are: inverse problems [Gonzalez et al., 2012], separation of the time axis in a multidimensional time space [Ammar et al., 2012] structural identification problems [Allix and Vidal, 2002], separation of spatial and stochastic parameters in [Nouy, 2009], non-coercive hyperbolic equations [Barbarulo et al., 2014], elastodynamic models [Boucinha et al., 2013]. An analysis about the error and verification of the method is in [Ladevèze and Chamoin, 2011]. The interested reader is referred to [Chinesta et al., 2011] for a review of these types of techniques with more than 100 references and to the book [Chinesta

and Ladevèze, 2014] for a handbook about separated variables representations and model reduction techniques.

In the current work, at each iteration of the LaTIn method, the previously calculated ROB is used first, leading to a ROM and a new approximation of the solution. It has been proved that this step of the strategy, called here "Preliminary step" ([Boisse et al., 1990, Ryckelynck, 2002, Ladevèze and Nouy, 2003, Ladevèze et al., 2009, Ladevèze et al., 2010] or "Update step" [Bonithon and Nouy, 2012]), allows to reduce the number of PGD functions generated, since it enables one to get the best approximation in the separated variables form, minimizing the approximation given by the PGD approach to the solution of the problem. If this approximation is not enough accurate, the ROB is enriched by defining a new functional product by using a greedy power algorithm.

To deal with nonlinear problems, PGD has been coupled with other linearization techniques. One of these techniques is the PGD based asymptotic expansions that is the combination of the PGD formulation of a problem with an asymptotic expansion of the displacement. This has been introduced in [Niroomandi et al., 2010, Niroomandi et al., 2012a] to tackle hyperelasticity. The goal of this conjunction is to avoid complex consistent linearization schemes necessary in Newton-like solution strategies. The advantage of this approach is the presence of only one tangent operator, identical for every term in the series.

For all kinds of model reduction techniques based on the projection of the problem on a basis, the computational cost associated with assembling the ROM's low-dimensional operators scales with the large dimension of the original high-dimensional model. For this reason, model reduction techniques are particularly efficient when the ROM needs to be constructed only once or when this step can be performed off-line, prior to the on-line resolution of this model which can then be very fast. This is the case of parametrized time-invariant systems [Amsallem et al., 2009], linear stationary and quasi-stationary systems whose operators are affine functions of the input parameters [Rozza and Patera, 2008], and a class of nonlinear systems characterized by quadratic nonlinearities [Nguyen et al., 2005, Veroy and Patera, 2005]. On the contrary, when the projection is applied to linear dynamics or stationary systems with non-affine parameter dependence, or general nonlinear problems, the resulting ROM is costly to assemble, decreasing the efficiency of reduced-order modelling. This high cost results from the need to evaluate the high-dimensional nonlinear function (and eventually its Jacobian) and then to project it to get the low-dimensional operators at each computational step of a solution algorithm. This results to be the bottleneck of model reduction strategies in the nonlinear case.

In the literature, some techniques have already been introduced for model reduction methods based on a learning stage. A very popular technique is the Empirical Interpolation Method (EIM) developed for linear elliptic problems with non-affine parameter dependence [Barrault et al., 2004] as well as for nonlinear elliptic and

parabolic problems [Grepl et al., 2007, Nguyen, 2005]. It reduces the computational cost associated with nonlinearities by interpolating the governing nonlinear function at a few spatial locations using an empirically derived basis. This method operates directly on the governing partial differential equation and therefore at the continuous level. Its variant proposed in [Nguyen and Peraire, 2008] relies for the same purpose on *best interpolation points*. Another variant, called Discrete EIM (DEIM), includes the semi-discrete analogue to the empirical and best points interpolation methods that have been developed for parameterized nonlinear stationary problems and for nonlinear dynamics problems [Chaturentabut and Sorensen, 2010].

Other techniques address the same issue. A family of techniques that tackles nonlinear problems for reduced-order modelling is the one that belongs to the Gappy-POD application, as the A priori Hyper-Reduction (APHR), the Missing Point Estimation (MPE) and the Gauss Newton with approximated tensors (GNAT). The APHR, introduced in [Ryckelynck, 2005] and described for nonlinear mechanical problems in [Ryckelynck and Benziane, 2010, Ryckelynck, 2009], is based on Gappy-POD too but does not exploit the off-line/on-line strategy since it builds the ROB by the POD along the iterations by performing the Petrov-Galerkin projection over a restricted subset of the spatial domain. This domain is sought by a heuristic method controlled by a prediction/correction algorithm (see the articles [Miled et al., 2013] for recent applications and [Ryckelynck et al., 2012] for developments to multidimensional models). The MPE, developed in [Astrid et al., 2008], performs online computations by using POD basis computing Galerkin projections over a restricted subset of the spatial domain, similarly to what is done in the APHR. MPE operates at the semi-discrete level and proposes a quantitative criteria for selecting such a spatial subset based on an heuristic method. The GNAT explained in [Carlberg et al., 2013] operates on fully discretized computational models, the dimension reduction is achieved by a Petrov-Galerkin projection associated with residual minimization and the hyper-reduction is based on the Gappy-POD as well as the MPE and the APHR. The difference with the APHR is that the GNAT requires a reduced-order basis dedicated to the residual and a reduced-order basis dedicated to the Jacobian matrix.

The trajectory piece-wise linear (TPWL) method developed in [Rewienski and White, 2006] has been introduced in other context. It constructs a ROM as a weighted combination of linearized models, where each linearization point lies on a training trajectory. TPWL operates at the semi-discrete level, i.e. on the ordinary differential equation obtained after discretizing the PDE in space.

The aim of this work is to develop a new technique, well suited to the PGD, to efficiently deal with nonlinear problems. For that purpose, an approximation framework, called Reference Points Method (RPM), is introduced in order to decrease the computational complexity of algebraic operations when dealing with separated variables approximations in the PGD framework. The RPM approximation framework is based on the concept of reference times, points and parameters and enables

to define a *compressed* version of the data. This representation of the fields has been initially proposed in [Ladevèze, 1997] and in [Ladevèze et al., 2009]. The first academic example have been introduced in [Néron and Ladevèze, 2012]. In this Phd thesis this concept is first developed for simple PDEs and then extended to be an approximation framework adapted to the PGD representation for the solution of mechanical problems. The RPM is applied to some benchmarks existing in literature, in order to have a comparison with respect to other similar techniques. Compared to these techniques [Barrault et al., 2004, Chaturentabut and Sorensen, 2010, Astrid et al., 2008] this is not an interpolation technique but an algebraic framework allowing to give an inexpensive first approximation of all quantities in a separated variables form by explicit formulas. The space of compressed data shows interesting properties dealing with the elementary algebraic operations [Ladevèze, 1997]. Moreover until now, only a limited amount of the fields is represented in the PGD form in the LaTIn method. This leads to manage fields with different representations and it hinders the potential gain of the PGD in terms of calculation. The Reference Point Method makes possible to give a first approximation of all the fields in the PGD form without resorting to SVD-based techniques. This approximation can be, if necessary, enriched by new PGD pairs.

RPM is introduced in the LaTIn-PGD algorithm in order to approximate the preliminary step of this solver. As said before, this stage of the algorithm enables to decrease the number of necessary PGD pairs but it involves repetitive operations that represent about the $50 - 70\%$ of the entire computational cost. RPM enables to decrease the number of the necessary operations, decreasing the cost of this stage of an order of magnitude and leading it to represent less than 10% of the entire computational cost, which is very promising.

The document is organized as follows:

- The first chapter is devoted to a state-of-the-art on reduced-order modelling with a focus on the application to nonlinear problems. In this chapter the main scientific motivation behind this thesis are highlighted, showing the main issues of reduced-order modelling in the case of nonlinear problems.

- In the second chapter the RPM is presented. It represents the novelty of this thesis and efficiently allows to decrease the computational complexity of algebraic operations when dealing with separated variables representation in the PGD framework.

- Chapter three shows how the RPM is introduced in the procedure of the LaTIn-PGD technique, detailing the procedure for an elliptic parametric problem. The preliminary step is reformulated in the RPM framework. It is shown how this reduces the number of operations necessary to evaluate the search direction and assemble the ROM.

- Chapter four is dedicated to the computational framework for which RPM is dedicated: the LaTIn based domain decomposition method with PGD approximation. RPM is implemented in this computational strategy in order to decrease the number of necessary operations to construct the ROM.

Chapter 1

Reduced-Order modelling: a brief overview of the state-of-the-art

In this first chapter, a literature survey on reduced-order modelling is proposed. The chapter is organized in two parts:

- First one is dedicated to highlight the interest of the reduced-order modelling regarding the computational complexity to solve a simple elastic $3D$ problem under a load depending on a parameter. This approach is compared to a direct technique that tackles the same problem for each parameter value.
- The second one explains why the efficiency of reduced-order modelling is brought into question when addressing nonlinear problems. It is highlighted how reduced-order modelling can be even more expensive than direct techniques when the size of the discretized problem overpasses a threshold, due to the necessity of a linearization technique that implies, at each iteration, the evaluation and integration of the nonlinear term over the entire domain and its projection onto the reduced-order basis.

Numerical approximation of the solution of partial differential equations plays an important role in many areas such as engineering, mechanics, physics, chemistry, biology... for computer-aided design-analysis, computer-aided decision-making or simply better understanding. The fidelity of the simulations with respect to reality is achieved through the combined efforts to derive: (i) better models, (ii) faster numerical algorithms, (iii) more accurate discretisation methods and (iv) improved large scale computing resources. In many situations, including optimization and control, the same model, depending on a parameter that evolves, has to be simulated over and over, multiplying by a large factor the solution procedure cost of one simulation.

Model reduction methods allow to define a surrogate solution procedure enabling to speed up the computations by several orders of magnitude while maintaining a sufficient accuracy. In this section a brief state-of-the-art of these techniques is developed detailing the different approaches that depend on different computational strategies.

The aim of this analysis is to demonstrate the efficiency of reduced-order modelling for parameter-dependent linear problems, compared to direct techniques (Finite Element Method (FEM), Finite Volumes (FV) or Boundary Element Method (BEM)) which seek the solution of the problem in an algebraic sub-space that, in many cases, is oversized in terms of dimensions. Nonlinear problems and non-affine parametric problems are analyzed in the second part of this chapter. A problem with a nonlinear term is considered. Necessity of an iterative solver to linearize the problem hinders the efficiency of reduced-order modelling.

1.1 Model order reduction by separation of variables

For many years, in different application areas for simulations, experiments or design, engineers have applied what is known as *operational model order reduction*. This way to tackle engineering problems relies on the assumption that it may be not necessary to calculate all details of the problem and, nevertheless, to obtain a good understanding of the phenomena behind the problem. There may be many reasons why some details of the solution are not needed. The operational model order reduction translates the use of physical (or other) insight to reduce the complexity of models.

In particular for simulations, there may be physical reasons that can be formulated beforehand, and that can be incorporated into the model before starting calculations. Let us consider an example belonging to the civil engineering field to make the concept clearer: for years, buildings have been designed and sized using truss structures, avoiding $3D$ expensive models. This is possible since $1D$ models give enough informations to describe this kind of structures. Another example is

the hypothesis of quasi-static evolutions, exploited in different areas, as mechanics, in order to avoid to integrate dynamical equations, or in electromagnetism in order to avoid the full Maxwell equations. For some phenomena, this assumption does not affect the accuracy of the solution but drastically reduces the complexity of its computation.

In this thesis the analyzed reduced-order techniques are based on the separated variables assumption. This hypothesis relies on the observation that many natural phenomena and, consequently, their numerical models have an evolution over the different coordinates that can be, often, considered independent. Hence, it is admissible to seek the description of this evolution in a separated variables form.

In order to analyse the features of these techniques, let us consider a bounded regular domain Ω included in $\mathbb{R}^d$ ($d = 1, 2, 3$). Introducing a parameter domain $\mathcal{D} = [0, D]$, let us consider the two-coordinates domain $\Omega \times \mathcal{D}$.

For a given space $\mathcal{I}$ on Ω, let us denote

$$L^2(\mathcal{D}, \mathcal{I}) \equiv \{\ u\ : \mathcal{D} \to \mathcal{I},\ \int_{\mathcal{D}} \|u\|_{\mathcal{I}}^2 \,\mathrm{d}\mu\ <\ \infty\ \}$$

and

$$\| \bullet \|_{\mathcal{I}} = \left(\int_{\Omega} \bullet^2 \ \mathrm{d}\Omega \right)^{1/2}$$

a L^2 norm on $\mathcal{I}$. Then, let us define $\mathcal{U} \equiv H^1(\Omega)$, $\mathcal{P} = L^2(\mathcal{D}, \mathbb{R})$ and $\mathcal{V} = L^2(\mathcal{D}, \mathcal{U})$. $\forall x \in \Omega$ and $\forall \mu \in \mathcal{D}$, let us denote $u(x, \mu) \in \mathcal{V}$ a scalar field related to this space-parameter problem.

Reduced-order techniques by separation of variables seek $\tilde{u}_m(x, \mu) \in \tilde{\mathcal{V}}_m = \mathcal{P} \otimes \mathcal{U}$, the approximation of $u(x, \mu)$ in separated variables form:

$$u(x, \mu)\ \approx\ \tilde{u}_m(x, \mu)\ =\ \sum_{i=1}^{m} \lambda_i(\mu)\ \Phi_i(x), \qquad (\lambda_i, \Phi_i)\ \in\ \mathcal{P} \times \mathcal{U}. \tag{1.1}$$

Approximation of rank one Let us note $\tilde{u}$ the first order approximation of u in the form:

$$\tilde{u}(x, \mu) =\ \lambda(\mu)\ \Phi(x) \tag{1.2}$$

whit $\lambda \in \mathcal{P}$ and $\Phi \in \mathcal{U}$. The searching for the *best* separated variables form of the function $u(x, \mu)$ leads to the following problem.

Problem 1 (first order separated variable representation) *Find* $(\lambda,\ \Phi) \in\ \mathcal{P} \times \mathcal{U}$ *that minimizes the error:*

$$e(u\ -\ \lambda\ \Phi) = \|\ u\ -\ \lambda\ \Phi\ \|_{\mathcal{P} \times \mathcal{U}}, \tag{1.3}$$

with the norm $\| \bullet \|_{\mathcal{P} \times \mathcal{U}}$ *defined as follow:*

$$\| \bullet \|_{\mathcal{P} \times \mathcal{U}} = \left(\int_{\mathcal{D}} \int_{\mathcal{U}} \bullet^2 \ \mathrm{d}\Omega\ \mathrm{d}\mu \right)^{1/2} \tag{1.4}$$

In the variational form, (1.3) becomes:

$$\begin{aligned} &\forall \lambda^\star \in \mathcal{P},\ \forall \Phi^\star \in \mathcal{U},\ \int_{\Omega\times\mathcal{D}} (u\ -\ \lambda\ \Phi)\ (\lambda^\star\Phi + \lambda\Phi^\star)\mathrm{d}\Omega\,\mathrm{d}\mu = 0 \\ &\iff \int_{\mathcal{D}} \lambda^\star \int_\Omega\ \Phi(u\ -\ \lambda\ \Phi)\ \mathrm{d}\Omega\,\mathrm{d}\mu\ +\ \int_\Omega\ \Phi^\star \int_{\mathcal{D}} \lambda(u\ -\ \lambda\ \Phi)\ \mathrm{d}\mu\ \mathrm{d}\Omega\ = 0 \end{aligned} \tag{1.5}$$

which gives:

$$\left\{ \begin{array}{l} \lambda \int_\Omega\ \Phi^2 d\Omega = \int_\Omega\ \Phi\ u\ d\Omega \iff \lambda = \dfrac{\int_\Omega\ \Phi\ u\ d\Omega}{\int_\Omega\ \Phi^2\mathrm{d}\Omega} \\ \Phi \int_{\mathcal{D}} \lambda^2\mathrm{d}\mu\ = \int_{\mathcal{D}} \lambda\ u\ \mathrm{d}\mu \iff \Phi = \dfrac{\int_{\mathcal{D}} \lambda\ u\ \mathrm{d}\mu}{\int_{\mathcal{D}} \lambda^2\mathrm{d}\mu}. \end{array} \right. \tag{1.6}$$

Functions λ and Φ are defined to within about a given multiplicative factor. One of the two functions has to be normalized. For instance one can set: $\int_{\mathcal{D}} \lambda^2 d\mu = 1$. The procedure leading to system (1.6) from Problem 1 can be seen as a power algorithm. Replacing Φ in the first equation of the system (1.6) leads to:

$$\lambda \int_\Omega\ \Phi^2 d\Omega = \frac{1}{\int_{\mathcal{D}} \lambda^2\mathrm{d}\mu} \int_\Omega\ \left(\int_{\mathcal{D}} \lambda\ u\ \mathrm{d}\mu\ \right) u\ \mathrm{d}\Omega, \tag{1.7}$$

$$\lambda = \frac{1}{\int_{\mathcal{D}} \lambda^2\mathrm{d}\mu \int_\Omega\ \Phi^2 d\Omega} \int_\Omega \left(\int_{\mathcal{D}} \lambda\ u\ \mathrm{d}\mu\ \right)\ u\ \mathrm{d}\Omega, \tag{1.8}$$

$$\lambda = \frac{1}{\|\lambda\|^2_{\mathcal{D}} \|\Phi\|^2_{\mathcal{U}}} (u, (u,\lambda)_{\mathcal{D}})_{\mathcal{U}} \tag{1.9}$$

By defining the operator $\mathbf{T} : \lambda \rightarrow \mathbf{T}(\lambda) = (u, (u,\lambda)_{\mathcal{D}})_{\mathcal{U}}$, system (1.6) can be expressed as an eigen-decomposition [Ladevèze, 1999, Nouy, 2010]. Find (λ, ω) such that:

$$\mathbf{T}(\lambda) = \omega\lambda, \qquad \text{with } \omega \in \mathbb{R}^+. \tag{1.10}$$

Solution (λ, Φ) of Problem 1, has the following properties:

- λ minimizes the Rayleigh quotient:

$$\boldsymbol{R}(\lambda) = \frac{(\ (u,\lambda)_{\mathcal{D}},\ (u,\lambda)_{\mathcal{D}}\ )_{\mathcal{U}}}{(\lambda,\lambda)_{\mathcal{D}}} \tag{1.11}$$

- $\Phi = \frac{(u,\lambda)_{\mathcal{D}}}{\|\lambda\|_{\mathcal{D}}}$
- $e(u\ -\ \lambda\ \Phi) = e(u) - \omega$, where ω is the biggest eigenvalue of the Rayleigh quotient $\boldsymbol{R}(\lambda)$.

In the case of an approximation of order $m > 1$ the best approximation, reads:

$$\tilde{u}_m(x,\mu) = \sum_{j=1}^{m}\ \lambda_j(\mu)\ \Phi_j(x) \tag{1.12}$$

where λ_i are the m first eigenfunctions of the eigen-decomposition of $\mathbf{T}$. The complementary functions are evaluated by $\Phi_i = \int_{\mathcal{D}}\ u(x,\mu)\ \lambda_i\ \mathrm{d}\mu$.

Solution of a linear system: the reduction by projection The field $u(x,\mu)$, function in the Hilbert space $\mathcal{V}$, may eventually be the solution of a problem written in the weak form:

$$\forall\ \mu \in \mathcal{D}, \quad a(u,v) = \ell(v,\mu) \qquad \forall v \in \mathcal{V}, \tag{1.13}$$

where $\ell(\cdot,\mu)$ is a linear form on $\mathcal{V}$ and $a(.,.)$ is a continuous symmetric bilinear form on $\mathcal{V}\times\mathcal{V}$.

Once the spatial semi-discretisation is operated, the field $u(x,\mu)$ is represented by a vector of values $[u(\mu)]$ and it is sought in the finite vectorial sub-space $\mathcal{V}_h \subset \mathcal{V}$ of size N. One has to solve the following discretised system for each value of parameter $\mu \in \mathcal{D}$:

$$[v]^T\,\mathbb{A}\,[u] = [v]^T\,[L(\mu)], \qquad \forall v \in \mathcal{V}_h, \quad \forall \mu \in \mathcal{D} \tag{1.14}$$

Thus, it reads:

$$\mathbb{A}\,[u] = [L(\mu)], \qquad \forall\ \mu \in \mathcal{D} \tag{1.15}$$

The usual way to proceed is to choose a sampling of size p for the parameter domain $\mathcal{D}_p \subset \mathcal{D}$. Hence, for each value $\mu_i \in \mathcal{D}_p$, one has to solve the $N \times N$ linear system (1.15) at the computational cost of $\mathcal{O}(N^3)$ (or $\mathcal{O}(N^2)$ if $\mathbb{A}$ is sparse).

The reduced-order modelling by projection is strongly connected to the separated variable representation. In fact, in order to solve this parametrised linear system, the idea is: (i) to extract somehow the "relevant" space modes from the linear system (1.15); (ii) to construct a reduced-order basis (ROB) of order $k < N$ from these relevant modes; (iii) to express the sought field as a linear combination of this ROB and, (iv) finally, to solve the Reduced-Order Model (ROM), obtained by projecting the linear system on the ROB.

In practice, the field $u(x,\mu)$ is approximated by:

$$u(x,\mu) \approx \tilde{u}_k(x,\mu) = \sum_{j=1}^{k}\ \lambda_j(\mu)\ \Phi_j(x), \tag{1.16}$$

The set of k functions $\Phi_j(x)$ makes a reduced-order basis $\mathbf{V}_k$. Let us define the discretized version of $\mathbf{V}_k$ as $\mathbb{V}_k = \{[\Phi_j]\}_{1\le j\le k}$ and $\Lambda_k = \{[\lambda_j]\}_{1\le j\le k}$. Choosing Φ_i as a test function for v, system (1.15) reads:

$$\mathbb{V}_k^T\,\mathbb{A}\,\mathbb{V}_k\ \Lambda_k(\mu) = \mathbb{V}_k^T\,[L(\mu)], \qquad \forall\ \mu \in \mathcal{D}. \tag{1.17}$$

Once $[\Phi_j]_{1\le j\le k}$ has been chosen, the ROM on the left-hand-side can be solved for every value $\mu_i \in \mathcal{D}_p$. In order to obtain the k functions $[\lambda_j]_{1\le j\le k}$ the computational cost is in the order of $\mathcal{O}(k^3)$, instead of $\mathcal{O}(N^3)$ for (1.15). Once the $[\lambda_j]_{1\le j\le k}$ are obtained, the solution u represented over the full parameter-space domain is obtained by the approximation (1.16).

Let us remark that the separation of variables in two functions, one depending on the space and one depending on the parameter enables one to compute the set of space function $\mathbb{V}_k = \{[\Phi_j]\}_{1\le j\le k}$ during a learning stage. However, the computational cost related to the construction of the reduced-order basis $\mathbb{V}_k = \{[\Phi_j]\}_{1\le j\le k}$

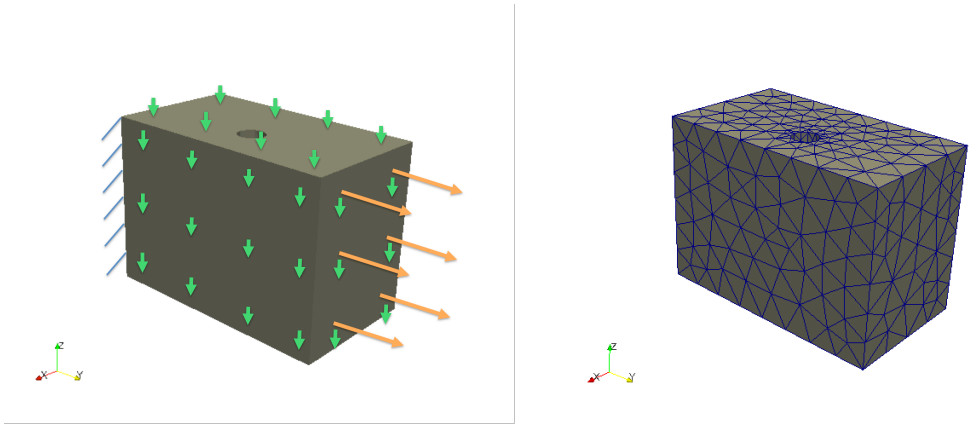
has also to be taken into account. This represents the most expensive part of the solution.

In literature, model reduction techniques based on the projection differ by the way in which the ROB is constructed. A first family of techniques involves a *learning phase*, called *off-line* stage. The Proper Orthogonal Decomposition (POD) is probably the most popular technique to generate the ROB. POD is widely used through two typical approaches. First one consists in considering during the learning stage a coarse discretization of the parameter domain. The other approach consists in solving the original model over the entire parameter interval during the learning phase, and then using the corresponding ROB to efficiently solve similar problems with, for instance, slight variations in material parameters or boundary conditions (this is, for instance, widely used in the CFD field [Zimmermann, 2012, Epureanuand et al., 2001, Zhao and Ling, 2003]). For this kind of problems the stability of the method can be improved by a Petrov-Galerkin formulation as proposed in [Carlberg et al., 2010, Bergmann et al., 2009].

The strong point of this kind of techniques is the fact that the number of the most relevant modes in the ROB is much lower than the size of the full order model in space, but, on the other hand, the relevance of the ROB is crucial regarding the quality of the approximation. A consequence of this feature is that the accuracy of the approximation is *case sensitive* [Glüsmann and Kreuzer, 2009]. That is to say that, changes in the system behaviour (e.g. changes in boundary conditions) can decisively affect the POD approximation accuracy and the number of required snapshots can increase [Glüsmann and Kreuzer, 2009, Boucinha et al., 2014].

In order to palliate this issue, another technique of the same family, the reduced-basis approximation [Rheinboldt, 1993, Rozza and Patera, 2008, Rozza, 2006, Prud'homme et al., 2002] improves the procedure for the selection of the *appropriate* snapshots providing an error indicator that gives a measure of the quality of the ROB. That palliates the case-sensitivity of arbitrary choices of snapshots but the construction of the ROB is generally more CPU-intensive. In fact, a greedy algorithm involving the resolution of reduced-order models over the whole parametric domain is required to enrich the ROB. However, this enables one to ensure the accuracy of the solution for the online part of the analysis.

Another path consists in seeking the solution of the targeted problem in the span of a consistent ROB, progressively built by dedicated algorithm. In this family there are the Proper Generalized Decomposition (PGD) ([Ladevèze, 1985, Ladevèze, 1999, Chinesta and Ladevèze, 2014, Chinesta et al., 2011]) and the A Priori Hyper Reduction (APHR) ([Ryckelynck, 2005, Ryckelynck, 2009, Ryckelynck et al., 2012, Ammar et al., 2006]). In the following, some of these techniques are described and illustrated on a linear elasticity problem.



(a) *boundary conditions.* (b) *finite elements discretisation.*

Figure 1.1: Boundary conditions and mesh for the reference problem.

1.1.1 Linear elasticity problem

Let us consider an elastic structure. The volume of this structure is the open domain $\Omega \in \mathbb{R}^3$ with boundary $\partial\Omega$. This structure is submitted to the following boundary conditions:

- a body force $\underline{f}_d$ in Ω;
- a given displacement field $\underline{u}_d$ (clamped structure for this case, i.e. $\underline{u}_d = 0$) prescribed on a part of the boundaries $\partial_u\Omega \neq \emptyset$ (Dirichlet boundary condition);
- a given surface force $\underline{F}_d(\mu)$ on the part of the boundaries $\partial_f\Omega$ complementary to $\partial_u\Omega$ ($\partial_u\Omega \cup \partial_f\Omega = \partial\Omega$, $\partial_u\Omega \cap \partial_f\Omega = \emptyset$) (Neumann boundary condition). Force $\underline{F}_d(\mu)$ depends on a parameter $\mu \in \mathcal{D}$.

The Hilbert space of functions, whose squared value is integrable on Ω, is denoted by $L^2(\Omega)$. The Sobolev space of functions and first derivatives, whose the square is integrable on Ω, is defined as follows:

$$H^1(\Omega) \equiv \{u \in L^2(\Omega) : \ \boldsymbol{\nabla} u \in L^2(\Omega)\}.$$

Let us introduce $H^1_0(\Omega)$ designating the subspace of functions vanishing on $\partial_u\Omega$:

$$H^1_0(\Omega) \equiv \{u \in H^1(\Omega) : \ u = 0 \text{ on } \partial_u\Omega\}.$$

Let us recall $\mathcal{U}$ extended to this $3-D$ problem. For simplicity $\mathcal{U} = [H^1_0(\Omega)]^3$. Let us remember spaces $\mathcal{P} = L^2(\mathcal{D}, \mathbb{R})$ and $\mathcal{V} = L^2(\mathcal{D}, \mathcal{U})$. Let us denote $\mathcal{S}$ the space of distributions of the symmetric tensor defined over Ω:

$$\mathcal{S} \equiv \{\boldsymbol{\sigma} \in \ [L^2(\Omega)]^6 \ : \ \boldsymbol{\sigma} = \boldsymbol{\sigma}^T\}. \tag{1.18}$$

Hence, the reference problem can be formulated as follows:

Problem 2 (strong form) *Under the hypothesis of small perturbations, find* $\underline{u}(\underline{x},\mu)$ *and* $\boldsymbol{\sigma}(\underline{x},\mu)$ *with* $\underline{x}=(x,y,z)\in\Omega$ *and* $\mu\in\mathcal{D}$ *that verifies,* $\forall\mu\in\mathcal{D}$*:*

- *the kinematic assumptions (or compatibility equations):*

$$\underline{u}\in\mathcal{V} \qquad \textit{and} \qquad \boldsymbol{\epsilon}(\underline{u})=\frac{1}{2}(\underline{\nabla}\,\underline{u}+\underline{\nabla}^T\underline{u}) \;\; \textit{in}\;\Omega \tag{1.19}$$

- *the equilibrium equations:*

$$\boldsymbol{\sigma}\in\mathcal{S}; \qquad \underline{\nabla}\cdot\boldsymbol{\sigma}+\underline{f}_d=\underline{0}, \;\; \textit{in}\;\Omega; \qquad \boldsymbol{\sigma}\cdot\underline{n}=\underline{F}_d(\mu) \quad \textit{on}\;\;\partial_f\Omega \tag{1.20}$$

- *the constitutive behaviour:*

$$\boldsymbol{\sigma}=\mathbf{K}\,\boldsymbol{\epsilon}(\underline{u})\;\textit{in}\;\Omega \tag{1.21}$$

$\mathbf{K}$ represents the Hooke tensor of the material, $\boldsymbol{\sigma}(\underline{x})$ the stress tensor and $\boldsymbol{\epsilon}(\underline{u})$ the linearized strain tensor associated to the displacement field $\underline{u}$, i.e. the symmetric part of the gradient of $\underline{u}$. The Hooke tensor $\mathbf{K}$ is a linear, symmetric and definite positive operator. It enables one to introduce the following energetic norms with the associated scalar products:

$$\langle\bullet,\bullet\rangle_{\underline{u},\Omega}=\int_\Omega \boldsymbol{\epsilon}(\bullet):\mathbf{K}:\boldsymbol{\epsilon}(\bullet)\mathrm{d}\Omega \qquad \|\cdot\|_{\underline{u},\Omega}=\Big[\langle\bullet,\bullet\rangle_{\underline{u},\Omega}\Big]^{1/2}, \tag{1.22}$$

$$\langle\bullet,\bullet\rangle_{\sigma,\Omega}=\int_\Omega \bullet\;:\;\mathbf{K}^{-1}\;:\;\bullet\;\mathrm{d}\Omega \qquad \|\cdot\|_{\sigma,\Omega}=\Big[\langle\bullet,\bullet\rangle_{\sigma,\Omega}\Big]^{1/2}, \tag{1.23}$$

The existence and uniqueness of a solution for Problem 2 is guaranteed under the conditions of $\partial_u\Omega\neq\emptyset$ and $mes(\partial_u\Omega)>0$.

Problem 3 (weak form) *Find* $\underline{u}(\underline{x},\mu)$ *and* $\boldsymbol{\sigma}(\underline{x},\mu)$ *with* $\underline{x}=(x,y,z)\in\Omega$ *and* $\mu\in\mathcal{D}$ *that verifies,* $\forall\mu\in\mathcal{D}$*:*

- *the kinematic assumptions (or compatibility equations):*

$$\underline{u}\in\mathcal{V}; \qquad \boldsymbol{\epsilon}(\underline{u})=\frac{1}{2}(\underline{\nabla}\,\underline{u}+\underline{\nabla}^T\underline{u})\;\textit{in}\;\Omega \tag{1.24}$$

- *the equilibrium equations:*

$$\begin{aligned}&\boldsymbol{\sigma}\in\mathcal{S};\\ &\int_\Omega\boldsymbol{\sigma}:\boldsymbol{\epsilon}(\underline{v})\mathrm{d}\Omega=\int_\Omega\underline{f}_d\cdot\underline{v}\mathrm{d}\Omega+\int_{\partial_f\Omega}\underline{F}_d(\mu)\cdot\underline{v}\,\mathrm{d}\,\Sigma, \quad \forall\mu\in\mathcal{D},\;\forall\underline{v}\in\mathcal{U}\end{aligned} \tag{1.25}$$

- *the constitutive behaviour:*

$$\boldsymbol{\sigma} = \mathbf{K}\, \boldsymbol{\epsilon}(\underline{u}) \; in \; \Omega \tag{1.26}$$

In order to simplify the notation Problem 3 is rewritten as follows:

Problem 4 *Given $\mu \in \mathcal{D}$, find $\underline{u}(\underline{x}, \mu) \in \mathcal{V}$ such that:*

$$a(\underline{u}, \underline{v}) - \ell(\underline{v}, \mu) = 0, \qquad \forall \underline{v} \in \mathcal{V}, \tag{1.27}$$

$$a(\underline{u}, \underline{v}) = \int_\Omega \boldsymbol{\epsilon}(\underline{u}) : \mathbf{K} : \boldsymbol{\epsilon}(\underline{v}) \mathrm{d}\Omega \tag{1.28}$$

$$\ell(\underline{v}, \mu) = \int_\Omega \underline{f}_d \cdot \underline{v} \, \mathrm{d}\Omega + \int_{\partial_f \Omega} \underline{F}_d(\mu) \cdot \underline{v} \, \mathrm{d}\Sigma \tag{1.29}$$

$a(\underline{u}, \underline{v})$ is a continuous bilinear form on the Hilbert's space $\mathcal{V}$ and $\ell(\underline{v}, \mu)$ is a linear form on $\mathcal{V}$.

The reference problem consists in a $3D$ beam, represented with its boundary conditions in **Fig. 1.1**. The reference solution, $(\underline{u}_{ref}, \boldsymbol{\sigma}_{ref})$, is the one obtained with a piecewise-linear finite element method. In this model the space is discretized by 4146 tetrahedron elements, resulting in $N = 2637$ dofs and the parameter domain is divided in $p = 100$ intervals. The following surface force is prescribed:

$$\underline{F}_d(\underline{x}, \mu) = 10^6 \; [(x/\mu) \; + \; z \; \cos(100 \; \mu \; x) \; x^{(1/100 \; \mu)} + z \; x)] \; \underline{e}_y \quad [Pa]. \tag{1.30}$$

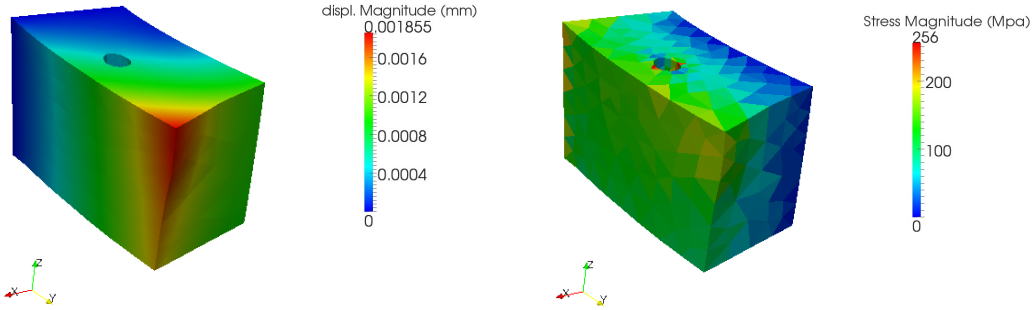
This force is chosen as it couples space and parameter domains. The structure is subjected to bending and torsion. The parametric domain is chosen as $\mathcal{D} = [0.1, 1]$. The considered material is an Aluminium alloy $2024 - T3$ with a Young's modulus $E = 71300 MPa$, and a Poisson's ratio $\nu = 0.33$. In **Fig. 1.2** the solution fields for $\mu = 0.1$, corresponding to the highest norm for $\underline{F}_d(\underline{x}, \mu)$ are shown.

In the following, this solution is compared with the ones obtained applying model reduction methods. The aim of this comparison is to highlight the benefits of applying reduced-order modelling to linear problems. The following error measure, according to the solution $\underline{u}_{ref}$ obtained by the classical FE method, is introduced:

$$\eta = \|\underline{u}_{ROM} - \underline{u}_{ref}\|_{\Omega,\mathcal{D}} / \|\underline{u}_{ref}\|_{\Omega,\mathcal{D}}, \qquad \| \circ \|^2_{\Omega,\mathcal{D}} = \int_{\mathcal{D}} \int_\Omega \circ \; : \; \mathbf{K} \; : \; \circ \, \mathrm{d}\Omega \mathrm{d}\mu \tag{1.31}$$

1.1.2 Proper Orthogonal Decomposition and Principal Components Analysis

The first type of reduced-order modelling is the one based on Proper Orthogonal Decomposition (POD). It has been introduced as Singular Values Decomposition



(a) *displacement field magnitude.* (b) *Von Mises stress field.*

Figure 1.2: Solution for $\mu = 0.1$. The deformed shape of the structure is amplified by a factor 10.

in [Beltrami, 1873], then in [Sylvester, 1889] and, finally, extended to integral operators in [Schmidt, 1907]. In the case of finite dimensional space the POD is equivalent to Principal Components Analysis [Wu et al., 2003] and Karhunen-Loeve Decomposition [Karhunen, 1943]. These are well-known in the field of data analysis. This technique is vastly used in different domains such as, stochastic analysis of experimental or numerical data, acoustics, image treatment, bio-medics engineering, fluid-structure interaction, etc. ([Chatterjee, 2000, Atwell and Kings, 2001, Kunish and Xie, 2005, Carlberg et al., 2010], see this review [Kerschen et al., 2005] for more examples).

The POD of the output field is founded on the awareness that, in many instances, the set of all solutions depending on the parameters and/or time can be approximated very well by its projection on a finite and low dimensional vector space. This approximation reads:

$$\underline{u}(\underline{x},\mu) \;\approx\; \underline{\tilde{u}}_k(\underline{x},\mu) \;=\; \sum_{j=1}^{k} \underline{\Phi}_j(\underline{x})\;\lambda_j(\mu), \tag{1.32}$$

Expression (1.32) is common to every model reduction technique involving separation of variables. For the POD the set of of function $\{\underline{\Phi}_j(\underline{x})\}_{j=1,..,k}$ forms an orthogonal basis. Thus, during the online part of the strategy, knowing the set $\{\underline{\Phi}_j(\underline{x})\}_{j=1,..,k}$ it is sufficient to evaluate the set of interpolating functions $\lambda_j(\mu)$ to get the best approximation of the function $\underline{u}(\underline{x},\mu)$. This is computed by:

$$\lambda_j(\mu) = \int_\Omega \underline{u}(\underline{x},\mu)\;\underline{\Phi}_j(\underline{x})\;d\Omega. \tag{1.33}$$

In the next paragraphs, the classical procedures to extract the most relevant POD modes are presented. The article [Wu et al., 2003] shows the equivalence,

in the discrete case, between the three methods regarding the way the set of functions of the basis are constructed: Karhunen-Loeve decomposition (KLD), Principal Component Analysis (PCA) and Singular Value Decomposition (SVD).

Let us consider the discrete framework in the following. An amount of state variables n is analyzed (e.g. experimental data as well as numerical data, like unknowns of a numerical model). The evolutions of these state variables are supposed to be known at p time instants or given parameter values. In our case, these are the snapshots obtained by solving the problem for the p parameter values of $\mu_i \in \mathcal{D}_p$. The standard POD techniques consist in arranging these informations in a matrix A of size $n \times p$, as shown in **Fig. 1.3**.

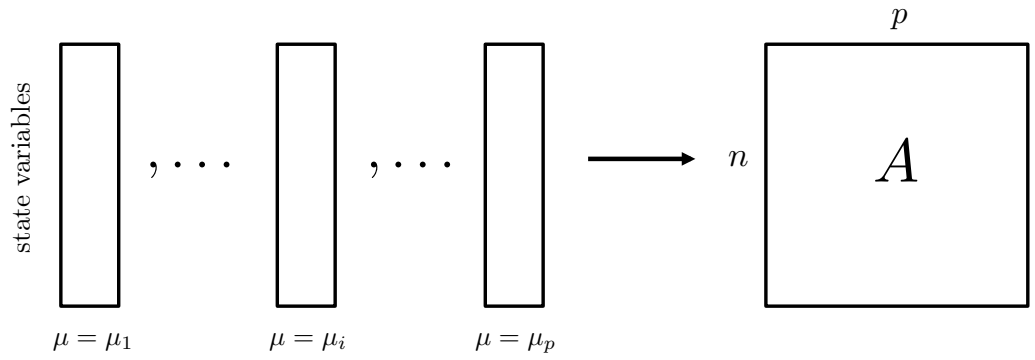


Figure 1.3: Data arrangement under a matrix A of size $n \times p$.

KLD and PCA The Karhunen-Loeve Decomposition [Karhunen, 1943, Loeve, 1955] has been used extensively in the fields such as digital communication, image processing and many others. The KLD consists in seeking the eigenvalues and eigenvectors of the covariance matrix C:

$$C\ \underline{v}_i = s_i\ \underline{v}_i \quad \text{with} \quad C = AA^T \qquad \text{or} \qquad C\ \underline{u}_i = s_i\ \underline{u}_i \quad \text{with} \quad C = A^TA \quad (1.34)$$

where $(s_i, \underline{v}_i)$ are respectively the eigenvalues and the eigenvectors of $C = AA^T$ and $(s_i, \underline{u}_i)$ are respectively the eigenvalues and the eigenvectors of $C = A^TA$. The first or the second form is chosen whether the basis is a set of space functions or time (parameter) functions. Generally the one which has the biggest number of unknowns is chosen (the max between n and p). The set of complementary functions is evaluated as shown in (1.33).

SVD Another technique, to get a POD basis, is to operate a singular value decomposition of the matrix A. This is a factorization of a real or complex rectangular matrix, introduced first in [Beltrami, 1873] for real matrix, extended to integral operators in [Schmidt, 1907] and later to complex matrix in [Eckart and Young, 1939]. This factorization has the form:

$$A = V \Sigma U^T \tag{1.35}$$

where V is the $n \times n$ matrix of the orthonormal space functions, U is the $p \times p$ matrix of the orthonormal parametric functions and Σ the $n \times p$ diagonal matrix where the non-zero entries (the $min(n, p)$ diagonal) are the singular values of the matrix. This decomposition is closely related to the eigen-decomposition. In fact, the singular values are the square roots of the eigenvalues of the operator $A^T A$, where A^T is the complex conjugate matrix of A. Consequently the left-singular vectors V are eigenvectors of AA^T and the right-singular vectors U are eigenvectors of $A^T A$

Looking at the expression of the KLD and PCA decomposition of the previous paragraph, one can note that the singular values are the square roots of the eigenvalues s_i of the covariance matrix C.

Let us consider the problem shown in **Fig. 1.1**. The magnitude of the singular values obtained by the SVD of the solution over the entire parameter domain $\mathcal{D}$ is shown in **Fig. 1.4**. It is clear that the singular values drop swiftly. Thus, the functions associated to the smallest singular values, with respect to the first ones, are negligible. The relevance of this method resides in the fact that, to describe the parametric evolution of the given structure over the parametric domain $\mathcal{D}$, a few POD modes are required. It can be seen a change of slope after the first 15 singular values. POD functions associated to the remaining singular values are negligible.

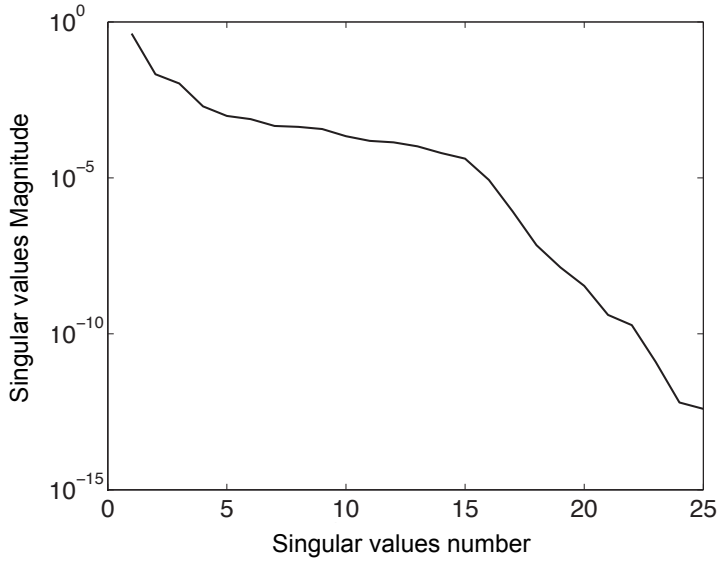


Figure 1.4: The singular values of Problem depicted in **Fig. 1.1**.

Galerkin-POD During the off-line stage the full order model is first solved for a coarse discretisation of parameter values $\mathcal{D}_p = \{\mu_i \in \mathcal{D}\}_{1\leq i\leq p}$ with $p \ll N$. These are the snapshots of the solution. A Galerkin projection is classically used to construct a ROM thanks to a ROB of dimension $k \ll N$. From the snapshots a ROB of dimension $k \leq p$ is obtained by SVD, selecting the first k orthogonal modes $\mathbf{V}_k = \{\underline{\Phi}_i\}_{1\leq i\leq k}$.

Let define as $\mathbf{\Phi}_k$ the linear span of the reduced-order basis $\mathbf{V}_k$. Following the classical Galerkin-POD approximation, the solution is sought in the linear span $\mathbf{\Phi}_k$ of the orthogonal ROB $\mathbf{V}_k$, by a standard Galerkin projection. Problem 5 reads:

Problem 5 (Galerkin-POD problem) *Given $\mu \in \mathcal{D}$, find $\underline{\tilde{u}} \in \mathbf{\Phi}_k$ such that:*

$$\forall \underline{v} \in \mathbf{\Phi}_k, \quad a(\underline{\tilde{u}}, \underline{v}) - \ell(\underline{v}, \mu) = 0 \tag{1.36}$$

Thus, by denoting the approximation of the solution $\underline{\tilde{u}} = \sum_{i=1}^{k} \lambda_i \underline{\Phi}_i$, and by choosing $\underline{\Phi}_i$ as a test function for $\underline{v}$ in Problem 5, the approximation $\underline{\tilde{u}}$ is obtained by the $k \times k$ linear algebraic system:

$$\forall\, 1 \leq j \leq k, \quad \sum_{i=1}^{k} a(\underline{\Phi}_i, \underline{\Phi}_j)\ \lambda_i(\mu) - \ell(\underline{\Phi}_j, \mu) = 0 \tag{1.37}$$

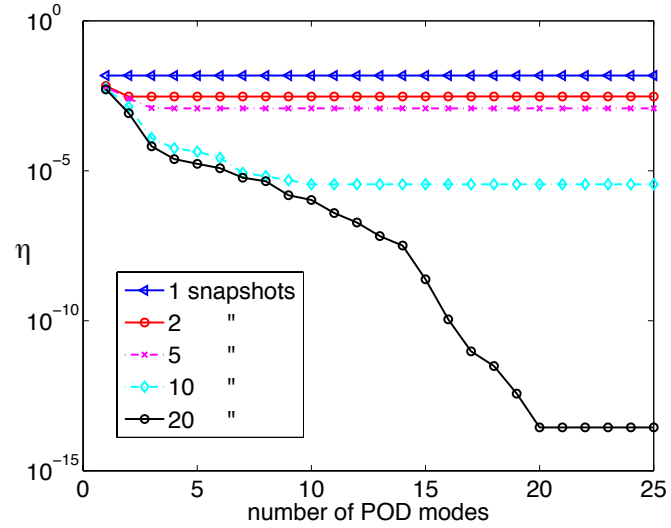
Let us note that first term on left-hand side member, $a(\underline{\Phi}_i, \underline{\Phi}_j)$, is parameter-independent and can be precomputed off-line. Equation (1.37) is valid for every value of $\mu \in \mathcal{D}$. Hence, for every parameter value it is possible to solve a reduced system involving a number of operations in the order of $\mathcal{O}(k^3)$ instead of $\mathcal{O}(N^3)$.

Fig. 1.5 shows the evolution of error (1.31) according to reference solution for different numbers p of snapshots used to built the ROB and a varying number k of ROB modes used to compute the solution approximation $\tilde{u}$. Note that once k is equal to p (all the ROB modes are used for the approximation), the error remains constant. It can be seen that adding more snapshots greatly improves the solution accuracy.

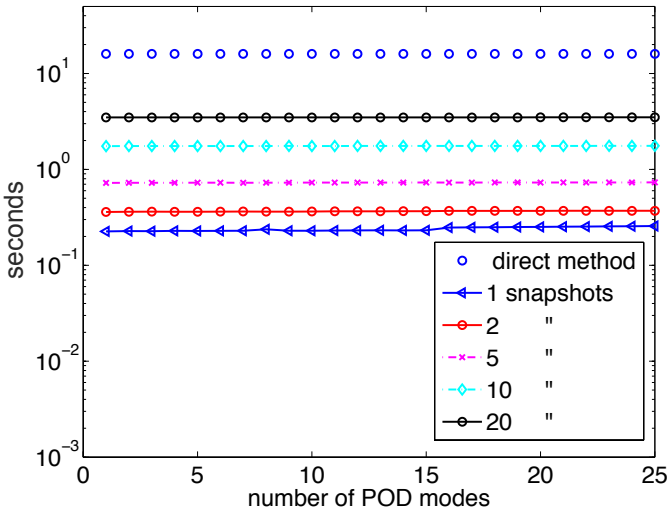
The gain in terms of CPU time compared with a direct simulation can reach up to two orders of magnitude, but it decreases with the number of snapshots. Note that the time spent to collect the snapshots in the off-line stage is taken into account **Fig. 1.5(b)**. It corresponds to the major part of the total CPU time.

1.1.3 The Reduced Basis method

For the Galerkin-POD, the quality of the final solution is affected by the pertinence of the formed ROB. In particular, for problems which have strong variations in boundary conditions or strong dynamic evolutions the number of necessary snapshots to have a suitable basis can increase so much that the technique becomes unsuitable [Glüsmann and Kreuzer, 2009, Boucinha et al., 2014]. For this issue, the



(a) *quality of the POD approximation.*



(b) *wall clock time.*

Figure 1.5: The error given by the POD approximation of Problem of **Fig. 1.1** and the CPU wall clock time compared to the direct method.

reduced-basis improves the procedure for the selection of the *appropriate* snapshots (see [Maday and Ronquist., 2004, Veroy and Patera, 2005, Nguyen et al., 2005, Rozza and Patera, 2008, Nguyen, 2008, Galvis and Kang, 2014]).

The reduced-basis method has been introduced in the beginning of the 1980s for nonlinear analysis of structures in dynamics [Noor and Peters, 1980], using a Rayleigh-Ritz decomposition in order to get the ROB. Later it has been extended to a larger class of parametrized PDEs [Fink and Rheinboldt, 1983, Rheinboldt, 1993]. The reduced-basis belongs to the separated variables model reduction techniques. Hence, the idea behind this technique is the same one than the POD. The reduced-basis consists in approximating the solution $\underline{u}(\underline{x},\mu)$ of a parameter dependent problem by a linear combination of, preliminary computed, solutions $\underline{u}(\underline{x},\mu_i)$ corresponding to properly chosen parameters μ_i, with $i=1,..,k$. This can be assumed on the condition that the set of precomputed solutions $S_k = \underline{u}(\underline{x},\mu_i)$ for $(i=1,...,k)$ is appropriate in the sense that its *Kolmogorov n-width* converges to zero for k that goes to ∞.

Definition 1 *Kolmogorov n-width. Let $\mathcal{Z}$ be an Hilbert linear space, X be a subset of $\mathcal{Z}$ and Z_n a generic n-dimensional subspace of $\mathcal{Z}$. The deviation of X from Z_n is*

$$E(X;Z_n) = \sup_{x\in X}\inf_{y\in Z_n} \parallel x-y \parallel_{\mathcal{Z}}. \tag{1.38}$$

The Kolmogorov n-width of X in $\mathcal{Z}$ is given by:

$$d_n(X,\mathcal{Z}) = \inf\ E(X;Z_n) = \inf_{Z_n}\sup_{x\in X}\inf_{y\in Z_n} \parallel x-y \parallel_{\mathcal{Z}}. \tag{1.39}$$

In practice the n-width measures the extent to which X may be approximated by a n-dimensional subspace of $\mathcal{Z}$. For the sake of simplicity, one can say that assuming a regularity of the solution $\underline{u}(\underline{x},\mu)$ with respect to the parameter μ this n-width goes rapidly to zero as n goes to infinity. More details can be found in [Grepl et al., 2007, Maday, 2006].

Hence, in the light of this condition, the main difference of the reduced-basis from the POD is in the construction of the ROB during the learning stage. These snapshots are constructed by a greedy algorithm such as [Maday, 2006]:

$$\mu_1 = \arg\sup_{\mu\in\mathcal{D}} \|\underline{u}(\underline{x},\mu)\|_{\mathcal{Z}}, \tag{1.40}$$

$$\mu_{i+1} = \arg\sup_{\mu\in\mathcal{D}} \|\underline{u}(\underline{x},\mu) - P_i\underline{u}(\underline{x},\mu)\|_{\mathcal{Z}}, \tag{1.41}$$

with the norm $\|\circ\|_{\mathcal{Z}}$:

$$\|\circ\|_{\mathcal{Z}} = \left(\int_{\mathcal{Z}} (\circ,\circ)\ \mathrm{d}z\right)^{1/2}. \tag{1.42}$$

P_i is an orthogonal projection, onto

$$\mathbf{\Phi}_i = \text{ span }\{\underline{u}(\underline{x},\mu_1),...,\underline{u}(\underline{x},\mu_i)\}. \tag{1.43}$$

In this way the new $(i+1)^{th}$ snapshot is selected minimizing the residue, defined by the chosen norm, of the solution achieved by solving the original problem projected onto this basis.

This is generally more CPU-intensive than the learning stage described for the POD. Indeed, it needs the resolutions of linear systems related to the reduced-order model over the whole parametric domain. However, it ensures the quality of the basis for the online part of the analysis. Much current effort is devoted to development of (i) a posteriori error estimation procedures and in particular rigorous error bounds for outputs of interest [Prud'homme et al., 2002, Barrault et al., 2004, Rozza and Patera, 2008, Rozza, 2011], and (ii) effective sampling strategies in particular for higher (than one) dimensional parameter domains [Rozza, 2004, Nguyen et al., 2005]. A posteriori error bounds are ensured for certification of the technique for output prediction (for the online stage). The error estimators can also play an important role in efficient and effective (greedy) sampling procedures: the inexpensive error bounds allow: (i) to explore much larger subsets of the parameter domain in order to find the most representative or best snapshots, (ii) to determine when there are just enough basis functions [Rozza, 2006, Rozza and Patera, 2008]. In [Rozza and Patera, 2008] a large amount of references related to this kind of strategy is presented.

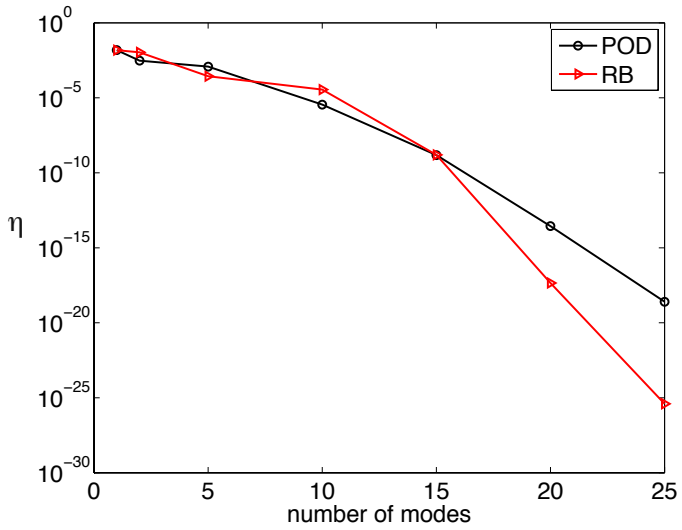
Galerkin reduced-basis The procedure to solve Problem of **Fig. 1.1** by the Galerkin reduced-basis is similar to the Galerkin-POD one. During the off-line stage the full order model is first solved for some *relevant* parameter values $\mathcal{D}_p = \{\mu_i \in \mathcal{D}\}_{1\leq i\leq p}$ with $p \ll N$. These are the snapshots of the solution. Hence a ROB is obtained simply by collecting the k relevant snapshot solutions $\mathbf{V}_k = \{\underline{\Phi}_i = \underline{u}(\underline{x}, \mu_i)\}_{1\leq i\leq k}$. In this case, the $\underline{\Phi}_i$ are also orthonormalized in practice. Then Problem 5 is solved. In **Fig. 1.6** is reported error (1.31) of the reduced-basis approximation for the problem shown in **Fig. 1.1**. It is compared with respect to classical POD-Galerkin approach for the same number of snapshots, chosen with a uniform coarse sampling.

It is clear that in the beginning the error levels for the two techniques are comparable, but after 15 snapshots, the reduced-basis is much more convenient thanks to the greedy algorithm in the selection of the snapshots **Fig. 1.6(b)**. The gain with respect to the direct method in terms of CPU time is comparable with the one obtained by the POD approximation.

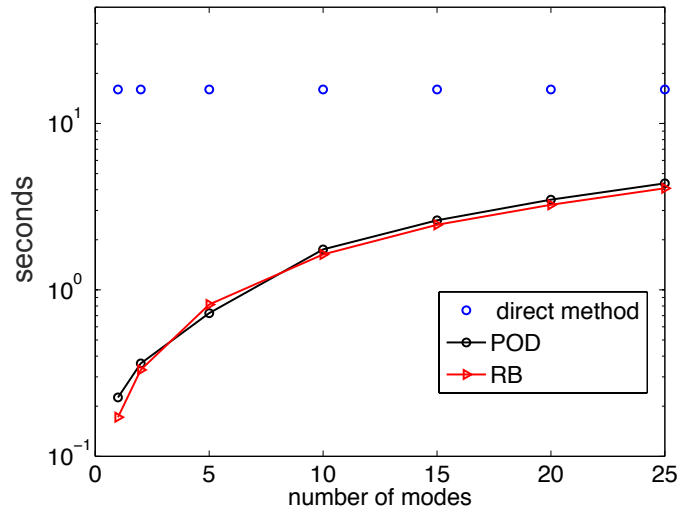
1.1.4 The Proper Generalized Decomposition

Other techniques do not involve preliminary high-fidelity simulations in order to construct a reduced-order basis. They progressively build, by online dedicated algorithms, a pertinent ROB, during the resolution of the same problem.

During the period of the increasing interest in the snapshot POD, a technique following a different strategy was developed and introduced as *radial approximation* [Ladevèze, 1985] in the framework of the LaTIn method (LArge Time INcrements).



(a) *accuracy of the reduced-basis approximation.*



(b) *wall clock time.*

Figure 1.6: The error given by the reduced-basis approximation of Problem of **Fig. 1.1** and the CPU wall clock time compared to the POD method and the direct method.

This is a non-incremental solver for nonlinear problems which generates the approximations of the solution over the entire time-space (parameter-space) domain by successive enrichments. Since its introduction in [Ladevèze, 1985, Ladevèze, 1989], the LaTIn method has been applied for various type of high-complexity problems: elasto-plasticity [Boisse et al., 1990], elasto-viscoplastic problems [Boisse et al., 1990, Cognard and Ladevèze, 1993, Champaney et al., 1997], finite displace-

ments [Michel-Ponnelle, 2001], contact problems [Roulet et al., 2011, Champaney et al., 1999, Giacoma et al., 2014, Giacoma et al., 2015], thermo-mechanical problems [Cognard et al., 1999, Ryckelynck, 2002], delamination and damage mechanics for composite materials [Allix et al., 1989, Allix and Ladevèze, 1992, Aubard et al., 2000], mesh adaptivity for visco-plastic problems [Pelle and Ryckelynck, 2000], post-buckling [Boucard et al., 1997], multiparametric problems [Boucard and Ladevèze, 1999] and multiphysics problems [Dureisseix et al., 2003, Néron and Dureisseix, 2008, Néron and Ladevèze, 2010]. The LaTIn method is also the numerical framework of a multiscale computational strategy for mechanical structures decomposed by a mixed domain decomposition developed at LMT Cachan [Ladevèze and Lorong, 1991, Blanze et al., 1996, Champaney et al., 1997, Dureisseix and Ladevèze, 1998, Ladevèze et al., 2001, Ladevèze and Nouy, 2003, Ladevèze et al., 2006, Guidault et al., 2008, Ladevèze et al., 2010, Cremonesi et al., 2013].

The LaTIn method represents a very convenient framework for the PGD. Indeed, PGD, mainly based on time-space separated representation, has been applied in many of the previous works enabling to decrease the necessary memory and the calculation time. In that time, LaTIn-PGD computational strategy represented a precursor of model reduction techniques applied to high-complexity problems.

Subsequently, at LMT Cachan and in other research groups, a number of extensions were proposed. The principal ones are the introduction of parameters of the model as additional coordinates in the PGD representation [Bognet et al., 2012, González et al., 2010, Chinesta et al., 2010, Ghnatios et al., 2012, Mokdad et al., 2007], developed in Chinesta's research group and real-time simulations, developed in Cueto's research group [Niroomandi et al., 2008, Monserrat et al., 2001, Niroomandi et al., 2012b]. Other extensions are: inverse problems [Gonzalez et al., 2012], separation of the time axis in a multidimensional time space [Ammar et al., 2012] structural identification problems [Allix and Vidal, 2002], separation of spatial and stochastic parameters in [Nouy, 2009], non-coercive hyperbolic equations [Barbarulo et al., 2014], elastodynamic models [Boucinha et al., 2013]. An analysis about the error and verification of the method is in [Ladevèze and Chamoin, 2011].

Another possibility offered by the LaTIn-PGD framework is to solve multiparametric models by exploiting the previous solution of the LaTIn method to initialize the solver for the new set of parameters (e.g. [Boucard and Ladevèze, 1999, Heyberger et al., 2011, Relun et al., 2011, Néron et al., 2015]). The interested reader is referred to [Chinesta et al., 2011] for a review of these types of techniques with more than 100 references and to the book [Chinesta and Ladevèze, 2014] for a handbook about separated variables representations and model reduction techniques.

PGD is a method that allows one to find the approximation of a solution by its best decomposition of order m. It does not require to know a basis to compute the solution of a problem, since in this method the two functions $\lambda_j(\mu)$ and $\underline{\Phi}_j(\underline{x})$ of the (1.32) are unknowns and they are originated by an iterative method that alternatively produce one of the two, knowing the other one. This can be consid-

ered as an algorithm of type Gauss-Seidel. An a posteriori error indicator can be evaluated and the quality of the approximation can be enriched adding a new pair of functions. Hence, PGD can be viewed as a real a priori resolution technique and a generalization of the POD since it does not need any information on the solution. This decomposition is not orthogonal, but in many cases the number of terms in the finite sum is very close to the optimal decomposition obtained by applying the POD on the model solution [Nouy, 2010].

In order to illustrate the method, let us consider again the parametric problem of **Fig. 1.1**. This problem can be easily extended to the case of more parameter coordinates. However, for multiparametric problems the convergence of the PGD depends on the choice of the norm [Bonithon and Nouy, 2012, Falco and Nouy, 2010].

The PGD approximation of this problem consists in seeking an approximation of the solution in the form:

$$\underline{u}(\underline{x},\mu) \approx \underline{\tilde{u}}_k(\underline{x},\mu) = \sum_{j=1}^{k} \lambda_j(\mu)\ \underline{\Phi}_j(\underline{x}), \qquad \lambda_j \in \mathcal{P},\ \ \underline{\Phi}_j \in \mathcal{U} \tag{1.44}$$

Galerkin-PGD definition The classic definition of the Proper Generalized Decomposition consists in building the test field, expressed as:

$$\underline{v}(\underline{x},\mu) = \sum_{i=1}^{k} (\lambda_i^\star(\mu)\ \underline{\Phi}_i(\underline{x})\ +\ \lambda_i(\mu)\ \underline{\Phi}_i^\star(\underline{x})) \tag{1.45}$$

and inject it in Problem 4.

Let us start with the first order approximation. The best first order approximation is defined as the optimal pair which verifies the Galerkin orthogonality criterium. The problem resides in seeking $(\lambda, \underline{\Phi})$ such as $\forall(\lambda^\star, \underline{\Phi}^\star)\ \in\ \mathcal{P}\times\mathcal{U}$,

$$a(\lambda\ \underline{\Phi},\ \lambda\ \underline{\Phi}^\star) = \ell(\lambda\ \underline{\Phi}^\star, \mu) \tag{1.46}$$

$$a(\lambda\ \underline{\Phi},\ \lambda^\star\ \underline{\Phi}) = \ell(\lambda^\star\ \underline{\Phi}, \mu) \tag{1.47}$$

(1.46) is a spatial problem independent of the parametric problem, but that involves the integration on the parameter domain in order to set up the operators. It is an application $\boldsymbol{S} : \mathcal{P} \to \mathcal{U}$ which maps a parametric function $\lambda \in \mathcal{P}$ into a space function $\underline{\Phi} = \boldsymbol{S}(\lambda) \in \mathcal{U}$. The second equation (1.47) is a simple differential ordinary problem (ODE), but it involves integrals over the space.

The resolution technique of this problem, detailed in [Ladevèze, 1999, Nouy, 2010], is an iterative process that alternatively solves the spatial problem to get $\underline{\Phi}$, assuming known λ (from the previous iteration). The same procedure is, then, run for the parametric problem to obtain $\lambda(\mu)$ assuming known the other function. This

is an algorithm of type Gauss-Seidel since the computation of one of the unknowns, at each iteration, uses only the knowledge of the other unknowns.

- initialization $\lambda^0(\mu)$. One could choice to initialize function $\underline{\Phi}^0$. $\lambda^0(\mu)$ is simpler than $\underline{\Phi}^0$ since it is a scalar function.
for $l = 1, ..., l_{max}$ **do**
  - knowing $\lambda^{l-1}(\mu)$, find $\underline{\Phi}^l(\underline{x})$ solution of Problem (1.46);
  - knowing $\underline{\Phi}^l(\underline{x})$, find $\lambda^l(\mu)$ solution of Problem (1.47);
  - normalization of $\underline{\Phi}^l(\underline{x})$.
end

Algorithm 1: Algorithm to generate the rank one approximation for a parametric problem.

The algorithm to achieve the approximation of first order involves l_{max} resolutions of D-dimensional spatial problems ($D = 2, 3$) for functions $\underline{\Phi}$ and l_{max} resolutions of 1-dimensional ODE for function $\lambda(\mu)$. The computational effort relies almost completely in the l_{max} D-dimensional spatial problems (the cost of the ODE is negligible compared to the spatial problem). The number of iterations l_{max} necessary to obtain a sufficiently accurate approximation ($\eta = 10^{-2}$ in practice) is quite low in practice (3 or 4 generally).

The next step of the progressive construction of the PGD consists in seeking a new PGD set of functions as the optimal one, verifying the Galerkin orthogonality criterium. Given a known decomposition $\underline{\tilde{u}}_{m-1}$ of order $(m-1)$, a new set of PGD functions $(\lambda, \underline{\Phi}) \in \mathcal{P} \times \mathcal{U}$ is defined as follows: seek $(\lambda, \underline{\Phi})$ such that $(\lambda^\star, \underline{\Phi}^\star) \in \mathcal{P} \times \mathcal{U}$,

$$a(\underline{\tilde{u}}_{m-1} + \lambda\ \underline{\Phi},\ \lambda\ \underline{\Phi}^\star) = \ell(\lambda\ \underline{\Phi}^\star, \mu) \tag{1.48}$$

$$a(\underline{\tilde{u}}_{m-1} + \lambda\ \underline{\Phi},\ \lambda^\star\ \underline{\Phi}) = \ell(\lambda^\star\ \underline{\Phi}, \mu) \tag{1.49}$$

In order to build an approximation of order $m \geq 2$ there are several methods. Contrary to the progressive construction of the PGD approximation described previously, another approach consists in finding simultaneously the m spatial functions and subsequently the m parameter functions. However, it leads to algebraic system m times bigger than the first order approximation and to prohibitive calculation costs, as soon as the number of PGD functions overpasses a given number. There are other methods, based on the Arnoldi's technique enabling one, by Krylov's subspace construction, to generate the m spatial functions [Nouy, 2010]. This approach is not taken into account in this context.

In practice, the approximation of order $m \geq 2$ is generated using the same iterative algorithm seen before but by updating the approximation at each new generated PGD set (progressive construction):

for $m = 2$ *to* m_{max} **do**
  - initialization $\lambda_m^0(\mu)$;
  for $l = 1, ..., l_{max}$ **do**
    - knowing $\lambda_m^{l-1}(\mu)$, find $\underline{\Phi}_m^l(\underline{x})$ solution of the problem 1.48;
    - knowing $\underline{\Phi}_m^l(\underline{x})$, find λ_m^l solution of the problem 1.49;
    - normalization of $\underline{\Phi}_m^l(\underline{x})$.
    - Set $\underline{\tilde{u}}_m = \underline{\tilde{u}}_{m-1} + \lambda_m \underline{\Phi}_m$.
    - Update of the right-hand member: $\ell \leftarrow \ell - a(\underline{\tilde{u}}_m, \underline{v}_m)$
    - Check convergence
  end
end

Algorithm 2: Algorithm to generate the $m-th$ order PGD-Galerkin. The rank one approximation is already available from Algorithm 1.

The Galerkin-PGD formulation shows some problems in the convergence for some kind of applications, such as, multiscale problems or acoustics problems. This issue has led to the introduction of a more general and more *robust* procedure to generate the PGD form of the solution of PDEs [Ladevèze and Nouy, 2003, Nouy, 2010, Relun et al., 2011, Allier et al., 2015].

PGD defined by the minimization of a residue This procedure has been proposed as a palliative for convergence problems of Galerkin-PGD formulation view in certain situations. Indeed, it is more *robust* than the Galerkin-PGD in the sense that monotonic convergence of the decomposition in the residual norm can be proved, even though convergence can be slow. It consists in defining the PGD approximation by a problem of minimization of the residue of Problem 4.

Problem 4 can be reformulated as:

Problem 6 *Find* $\underline{u}(\underline{x}, \mu) \in \mathcal{V}$ *such that:*

$$A(\underline{u}, \underline{v}) = L(\underline{v}), \quad \forall \underline{v} \in \mathcal{V} \tag{1.50}$$

with:

$$A(\underline{u}, \underline{v}) = \int_{\mathcal{D}} a(\underline{u}, \underline{v}) \, \mathrm{d}\mu \tag{1.51}$$

$$L(\underline{v}) = \int_{\mathcal{D}} \ell(\underline{v}, \mu) \, \mathrm{d}\mu \tag{1.52}$$

with $A(\underline{u}, \underline{v})$ *a bilinear form on* $\mathcal{V}$ *and* $L(\underline{v}, \mu)$ *a linear form on* $\mathcal{V}$ *defined in Problem 4.*

In order to define the progressive minimal residual PGD formulation of Problem 6, let us define $\langle \cdot, \cdot \rangle$ as an inner product on $\tilde{\mathcal{V}} \equiv \mathcal{P} \otimes \mathcal{U}$ and $\| \cdot \|_{\tilde{\mathcal{V}}}$ the associated norm. The residual $\mathcal{R}(\underline{u})$ of Problem 6 is define as follows:

$$\langle\ \underline{v}, \mathcal{R}\ \rangle\ =\ L(\underline{v})\ -\ A(\underline{u}, \underline{v}) = \langle\ \underline{v},\ \mathcal{L} - \mathcal{A}\ \rangle\,, \quad \forall \underline{v} \in \tilde{\mathcal{V}} \tag{1.53}$$

where $\mathcal{L} \in \tilde{\mathcal{V}}$ and operator $\mathcal{A} : \tilde{\mathcal{V}}\ \rightarrow\ \tilde{\mathcal{V}}$ are obtained thanks to Riesz representation theorem in Hilbert space $\mathcal{P} \otimes \mathcal{U}$. For instance, for Problem 6 and L^2-norm, one can choose:

$$\mathcal{R}(\underline{u})\ =\ div\ (\mathbf{K}\ \boldsymbol{\epsilon}(\underline{u}))\ +\ \underline{f}_d, \qquad \mathcal{A}(\underline{u})\ =\ -\mathbf{K}\ \Delta\underline{u}, \qquad \mathcal{L}\ =\ \underline{f}_d \tag{1.54}$$

since $\mathbf{K}$ is a constant symmetric operator.

Definition 2 (Progressive minimal residual PGD) *Find* $(\lambda_m, \underline{\Phi}_m) \in \mathcal{P} \times \mathcal{U}$ *which minimize the residual norm:*

$$(\lambda_m,\ \underline{\Phi}_m)\ =\ \arg \min_{(\lambda^\star,\ \underline{\Phi}^\star)\ \in\ \mathcal{P}\times\mathcal{U}} \|\mathcal{R}(\sum_{i=1}^{m-1} \lambda_i\ \underline{\Phi}_i + \lambda\ \underline{\Phi})\| \tag{1.55}$$

that is to say:

$$(\lambda_m,\ \underline{\Phi}_m)\ =\ \arg \min_{(\lambda^\star,\ \underline{\Phi}^\star)\ \in\ \mathcal{P}\times\mathcal{U}} \frac{1}{2}\ \langle\ \mathcal{A}\ (\lambda,\ \underline{\Phi}),\ \mathcal{A}\ (\lambda,\ \underline{\Phi})\ \rangle - \left\langle\ \mathcal{R}(\underline{u}_{m-1}),\ \mathcal{A}\ (\lambda,\ \underline{\Phi})\ \right\rangle \tag{1.56}$$

Remark 1 *By introducing the adjoint operator* $\mathcal{A}^\star$ *of* $\mathcal{A}$, (1.56) *is equivalent to the Galerkin-PGD applied to a least-square formulation of the problem (symmetrized problem) [Nouy, 2010]: find* $\underline{u}(\underline{x}, \mu) \in \mathcal{V}$ *such that:*

$$\bar{A}(\underline{u}, \underline{v})\ =\ \bar{L}(\underline{v}), \quad \forall \underline{v} \in \mathcal{V} \tag{1.57}$$

with:

$$\bar{A}(\underline{u}, \underline{v}) \quad = \quad \langle\ \mathcal{A}(\underline{v}),\ \mathcal{A}(\underline{u})\ \rangle = \langle\ \underline{v},\ \mathcal{A}^\star\ \mathcal{A}(\underline{u})\ \rangle \tag{1.58}$$

$$\bar{L}(\underline{v}) \quad = \quad \langle\ \mathcal{A}(\underline{v}),\ \mathcal{L}\ \rangle = \langle\ \underline{v},\ \mathcal{A}^\star\ \mathcal{L}\ \rangle \tag{1.59}$$

and, in this case, progressive minimal residual PGD is defined as follows:

$$(\lambda_m,\ \underline{\Phi}_m)\ =\ \arg \min_{(\lambda^\star,\ \underline{\Phi}^\star)\ \in\ \mathcal{P}\ \times\mathcal{U}} \frac{1}{2}\ \langle\ \lambda\ \underline{\Phi},\ \mathcal{A}^\star\ \ \mathcal{A}\ (\lambda,\ \underline{\Phi})\ \rangle - \left\langle\ \lambda\ \underline{\Phi},\ \mathcal{A}^\star\ \ \mathcal{R}(\tilde{\underline{u}}_{m-1})\ \right\rangle \tag{1.60}$$

Consequently, algorithms presented previously for Galerkin-PGD (see Algorithm 1 and 2) can be easily transposed to this formulation by considering $\bar{\mathcal{A}}$ *(resp.* $\bar{\mathcal{L}}$*) instead of* $\mathcal{A}$ *(resp.* $\mathcal{L}$*). Note that, for Problem 1.56 and natural* L^2*-norm operator* $\mathcal{A}$ *is self-adjoint:* $\mathcal{A}^\star\ =\ \mathcal{A}$.

Stationarity conditions associated with (1.56) reads:

$$\forall(\underline{\Phi}^\star,\lambda^\star)\in\mathcal{P}\times\mathcal{U},\quad \langle\ \mathcal{A}(\lambda\ \underline{\Phi}),\ \mathcal{A}\ (\lambda^\star\ \underline{\Phi}+\lambda\ \underline{\Phi}^\star)\ \rangle\ =\ \left\langle\ \mathcal{R}(\underline{\tilde{u}}_{m-1}),\ \mathcal{A}\ (\lambda^\star\ \underline{\Phi}+\lambda\ \underline{\Phi}^\star)\ \right\rangle \tag{1.61}$$

Applications $\boldsymbol{S}_m:\mathcal{P}\to\mathcal{U}$ (resp. $\boldsymbol{P}_m:\mathcal{U}\to\mathcal{P}$) which maps a parametric function λ (resp. space function $\underline{\Phi}$) into a space (resp. parameter) function are defined by the following equations:

- $\underline{\Phi}=S_m(\lambda)\in\mathcal{U}$ is defined by:

$$\forall\underline{\Phi}^\star\in\mathcal{U},\quad \langle\ \mathcal{A}(\lambda\ \underline{\Phi}),\ \mathcal{A}\ (\lambda\ \underline{\Phi}^\star)\ \rangle\ =\ \left\langle\ \mathcal{R}(\underline{\tilde{u}}_{m-1}),\ \mathcal{A}\ (\lambda\ \underline{\Phi}^\star)\ \right\rangle \tag{1.62}$$

- $\lambda=P_m(\underline{\Phi})\in\mathcal{P}$ is defined by:

$$\forall\underline{\lambda}^\star\in\mathcal{P},\quad \langle\ \mathcal{A}(\lambda\ \underline{\Phi}),\ \mathcal{A}\ (\lambda^\star\ \underline{\Phi})\ \rangle\ =\ \left\langle\ \mathcal{R}(\underline{\tilde{u}}_{m-1}),\ \mathcal{A}\ (\lambda^\star\ \underline{\Phi})\ \right\rangle \tag{1.63}$$

The verification of these two equations can be seen as a pseudo-eigenproblem on operator $\mathcal{G}_m=S_m\ \circ\ P_m$ [Nouy, 2010]. The (pseudo) eigenvalue associated with an eigenfunction $\underline{\Phi}=\mathcal{G}(\underline{\Phi})$ is defined by $\boldsymbol{\sigma}_m(\underline{\Phi})=\langle\ \mathcal{A}(\underline{\Phi}\ P_m(\underline{\Phi})),\ \mathcal{A}(\underline{\Phi}\ P_m(\underline{\Phi}))\ \rangle$. The optimal function $\underline{\Phi}_m$ which maximizes $\boldsymbol{\sigma}_m(\underline{\Phi})$ is the dominant eigenfunction of $\mathcal{G}_m$. The couple $(\underline{\Phi},\ P_m(\underline{\Phi})$ is optimal in the sense that it minimizes the residual norm. Indeed, it can be shown that:

$$\|\mathcal{R}(\underline{\tilde{u}}_m)\|^2\ =\ \|\mathcal{R}(\underline{\tilde{u}}_{m-1})\|^2-\boldsymbol{\sigma}_m(\underline{\Phi}_m)\ =\ \mathcal{L}^2\ -\ \sum_{i=1}^{m}\boldsymbol{\sigma}_i(\underline{\Phi}_i) \tag{1.64}$$

This shows that the residual norm monotically decreases, which provides a convenient convergence indicator of the order m approximation $\underline{u}_m$ of the solution. In that sense, this is a robust construction of the separated representation and this formulation can be used when Galerkin-based PGD fails.

Remark 2 *Convergence rate depends on the choice of the residual norm $\|\cdot\|$. Using classical norm in $L^2(\mathcal{U})\otimes L^2(\mathcal{D})$, i.e.:*

$$\|\mathcal{R}(\underline{\tilde{u}}_m)\|=\langle\mathcal{R}(\underline{\tilde{u}}_m),\mathcal{R}(\underline{\tilde{u}}_m)\rangle=(\mathcal{R}(\underline{\tilde{u}}_m),\mathcal{R}(\underline{\tilde{u}}_m))$$

may lead to a slow convergence. Choosing a residual norm based on the operator $\mathcal{A}$ of the problem as close as possible to $\|\underline{u}-\underline{\tilde{u}}_m\|$ generally improves the convergence rate. For instance, by introducing $\mathcal{M}$ a suitable symmetric bounded coercive operator as close as possible to the inverse of the symmetric part of $\mathcal{A}$, one can choose:

$$\|\mathcal{R}(\underline{\tilde{u}}_m)\|_{\mathcal{M}}=\langle\mathcal{R}(\underline{\tilde{u}}_m),\mathcal{R}(\underline{\tilde{u}}_m)\rangle=(\mathcal{R}(\underline{\tilde{u}}_m),\ \mathcal{M}\ \mathcal{R}(\underline{\tilde{u}}_m))\,.$$

In this case, one has:

$$\begin{aligned}\|\mathcal{R}(\underline{\tilde{u}}_m)\|_{\mathcal{M}}&=\ (\mathcal{R}(\underline{\tilde{u}}_m),\ \mathcal{M}\ \mathcal{R}(\underline{\tilde{u}}_m))\\&=\ (\mathcal{L}-\mathcal{A}(\underline{\tilde{u}}_m),\ \mathcal{M}\ \mathcal{L}-\mathcal{A}(\underline{\tilde{u}}_m))\\&=\ (\mathcal{A}(\underline{u}-\underline{\tilde{u}}_m),\ \mathcal{M}\ \mathcal{A}(\underline{u}-\underline{\tilde{u}}_m))\\&=\ (\underline{u}-\underline{\tilde{u}}_m,\mathcal{A}^\star\ \mathcal{M}\ \mathcal{A}(\underline{u}-\underline{\tilde{u}}_m))\\&=\ \|\underline{u}-\underline{\tilde{u}}_m\|_{\mathcal{A}^\star\mathcal{M}\mathcal{A}}\end{aligned}$$

The operator $\mathcal{M}$ can be seen as a preconditioner for the problem. Its choice is crucial in order that residual norm $\|\mathcal{R}(\underline{\tilde{u}}_m)\|$ gives a measure of the error close to $\|\underline{u} - \underline{\tilde{u}}_m\|$. The construction of such a norm/preconditionner is not straightforward and may lead to additional computational issues. Note that, if $\mathcal{A}$ is self-adjoint (hermitian or symmetric) and $\mathcal{M}$ is the inverse of $\mathcal{A}$, one has:

$$\|\mathcal{R}(\underline{\tilde{u}}_m)\| \;=\; \|\underline{u} - \underline{\tilde{u}}_m\|_{\mathcal{A}^\star} \;=\; \|\underline{u} - \underline{\tilde{u}}_m\|_{\mathcal{A}}$$

In **Fig. 1.7**, convergence of error (1.31), for the problem described in **Fig. 1.1** is given for the three approximations: POD, Reduced-basis and progressive Galerkin-PGD. Error levels and CPU time gains for POD, PGD approximation and reduced-basis approach are similar. Finally the three levels of possible gain, in terms of CPU time, with respect to the direct method, are comparable.

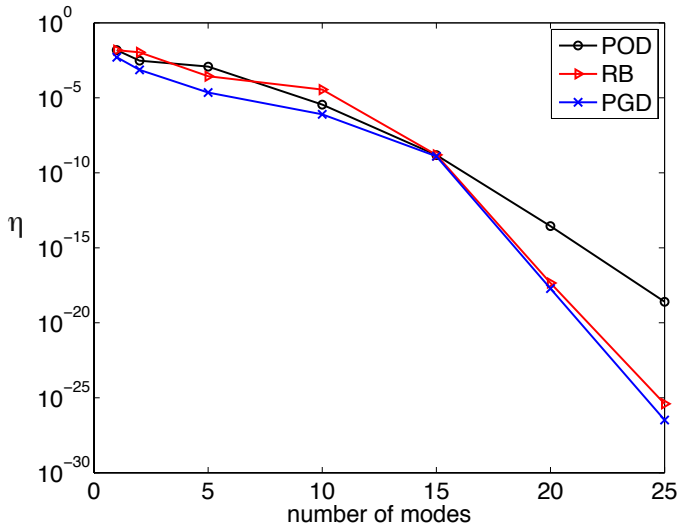
For linear problems, this can be considered as a general result. In fact the modes generated by the three different techniques are closed and, in general, the reduction of the computational complexity is in the same order of magnitude.

1.2 Model Reduction methods for nonlinear problems

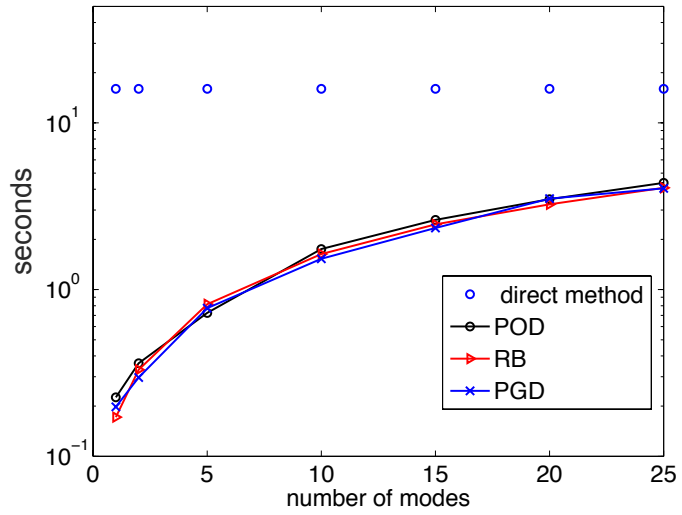
This section considers the application of model reduction techniques to nonlinear problems. For nonlinear problems, model reduction techniques need to be coupled with linearization techniques. Each model reduction technique has been coupled to specific iterative methods that are considered well-suited to one model reduction technique rather than another one. The most known linearization techniques employed in model reduction are:

- the Newton-Raphson method, mainly coupled with POD and reduced-basis for strategies based on a learning stage, but also with the APHR, a technique that does not involve any learning stage to construct the ROB.
- LaTIn method, well-suited for the PGD approximation because it is a non-incremental method that generates approximations of the solution over the entire time-parametric-space domain by successive enrichments;
- Asymptotic Numerical Method (ANM), coupled both with PGD and techniques constructing a ROB in a learning stage. The ANM coupled with PGD has been introduced in [Niroomandi et al., 2010, Niroomandi et al., 2012a] to tackle hyperelasticity. The advantage of this approach is the presence of only one tangent operator, identical for every iteration, but it needs high order of interpolation of the nonlinear terms to obtain suitable results.

The aim of this section is to highlight a drawback of applying model reduction techniques based on projection in the framework of nonlinear problems. This issue



(a) *quality of the PGD approximation.*



(b) *wall clock time.*

Figure 1.7: The error given by the PGD approximation of Problem 2 and the CPU wall clock time compared to the POD (offline and online), reduced-basis methods (offline and online) and the direct method.

derives from the necessity of a linearization scheme. Linearization schemes generate a series of different linear problems. For each linear problem it is necessary to construct a new reduced-order model since, in each linear problem of the series, the nonlinear terms are different from the previous one. This implies (i) the evaluation of the nonlinear terms, (ii) the integration of those terms over the entire domain and (iii) their projections on the ROB.

The complexity of those operations scales with the size of the original discretized problem, i.e., for problem of **Fig. 1.1** the $N = 2637$ dofs and the $p = 100$ parameter steps. Thus, the computational cost associated with assembling the ROM low-dimensional operators, scales with the large dimension of the original high-dimensional models. For this reason, model reduction techniques are more efficient when the ROM needs to be constructed only once or when this step can be performed off-line, prior to the online resolution of this model which can then be very fast. This is the case of parametrized time-invariant systems, linear stationary and quasi-stationary systems whose operators are affine functions of the input parameters. On the contrary, when projection is applied to linear dynamic systems, linear stationary systems with nonaffine parameter dependence, or general nonlinear problems, the resulting ROM is costly to assemble. Often, this results to be the bottleneck of nonlinear model reduction strategies based on Galerkin projection.

Let us consider the reference Problem 4 but with a nonlinear term. This problem has been introduced as a benchmark in [Grepl et al., 2007] for numerical tests with the EIM and then considered again in [Chaturentabut and Sorensen, 2010] in the framework of the DEIM:

Problem 7 (reference nonlinear problem) *Given $\mu \in \mathcal{D}$, find $\underline{u}(\underline{x}, \mu) \in \mathcal{U}$ such that:*

$$\forall \underline{v} \in \mathcal{U}, \quad a(\underline{u}, \underline{v}) + \int_\Omega g(\underline{u}; \mu)\ \underline{v}\ \mathrm{d}\Omega - \ell(\underline{v}, \mu) = 0 \tag{1.65}$$

where $a(\underline{u}, \underline{v})$ is a continuous bilinear operator on $\mathcal{U}$, $\int_\Omega g(\underline{u}; \mu)\ \underline{v}\ \mathrm{d}\Omega$ is a continuous nonlinear operator on $\mathcal{U}$ depending on the solution $\underline{u}$ and parameter μ and $\ell(\underline{v}, \mu) = \int_\Omega\ f(\underline{x}, \mu)\ \underline{v}\ \mathrm{d}\Omega$ is a linear form on $\mathcal{U}$.

Here, Problem 7 is solved considering the reduced-basis technique, described in section 1.1.3, in order to highlight the computational complexity related to model reduction techniques by projection, in the framework of nonlinear problems.

The solution is sought in the linear span $\mathbf{\Phi}_k$ of the orthogonal ROB, $\mathbf{V}_k = \{\underline{\Phi}_1, \ldots, \underline{\Phi}_k\}$, by a standard Galerkin projection. Problem 7 reads:

Problem 8 (reduced-basis problem) *Given $\mu \in \mathcal{D}$, find $\underline{\tilde{u}}_k(\underline{x}, \mu) \in \mathbf{\Phi}_k$ such that:*

$$\forall \underline{v} \in \mathbf{\Phi}_k, \quad a(\underline{\tilde{u}}_k, \underline{v}) + \int_\Omega\ g(\underline{\tilde{u}}_k; \mu)\ \underline{v}\ \mathrm{d}\Omega = \ \ell(\underline{v}, \mu). \tag{1.66}$$

Thus, by denoting the approximation of the solution $\underline{\tilde{u}}(\underline{x}, \mu) = \sum_{i=1}^k \lambda_i(\mu)\underline{\Phi}_i$, and by choosing $\underline{\Phi}_i$ as a test function for $\underline{v}$ in (1.66), the approximation $\underline{\tilde{u}}$ is obtained by the $k \times k$ nonlinear algebraic system, $\forall\, 1 \leq j \leq k$, $\forall\, \mu \in \mathcal{D}$:

$$\sum_{i=1}^{k}\ a(\underline{\Phi}_i, \underline{\Phi}_j)\ \lambda_i(\mu) + \int_\Omega\ g\left(\sum_{i=1}^{k} \lambda_i(\mu)\underline{\Phi}_i; \mu\right) \underline{\Phi}_j\ \mathrm{d}\Omega = \ \ell(\underline{\Phi}_j, \mu). \tag{1.67}$$

Let us notice that first term in left-hand side member, $a(\underline{\Phi}_i, \underline{\Phi}_j)$, is parameter-independent and can be precomputed off-line. However, the second term depends on $g(.;\mu)$ and, consequently, it is a nonlinear parameter-dependent term. It has to be evaluated online for each new parameter value μ. Next section shows that the complexity of the online stage scales with $\mathcal{O}(N\ k^2)$ with N the dimension of the underlying finite element approximation space. Following [Chaturentabut and Sorensen, 2010], computational complexity is evaluated in terms of FLOPS counting both the multiplications and additions as FLOPS.

1.2.1 Computational complexity analysis

Nonlinear algebraic system (1.67) is classically solved by iterative methods, such as, the Newton-Raphson method for each value of parameter μ. Each tangent problem of the iterative scheme is, then, projected on a precomputed ROB, $\mathbf{V}_k$. This section aims at detailing the N-dependence computational complexity of classical model reduction techniques for nonlinear problems.

Newton method consists in solving a linearized formulation of (1.67). Knowing iterate $\underline{\tilde{u}}^{(n)}$, one looks for $\underline{\tilde{u}}^{(n+1)} = \underline{\tilde{u}}^{(n)} + \delta\underline{\tilde{u}}^{(n+1)}$ by solving:

Problem 9 (tangent problem) *Given $\mu \in \mathcal{D}$, find $\delta\underline{\tilde{u}}^{(n+1)} \in \mathcal{U}$*

$$\forall \underline{v} \in \mathcal{U}, \quad \mathcal{R}(\underline{\tilde{u}}^{(n)}, \underline{v}; \mu) + \left\langle \mathcal{R}'(\underline{\tilde{u}}^{(n)}, \underline{v}; \mu), \delta\underline{\tilde{u}}^{(n+1))} \right\rangle = 0 \tag{1.68}$$

with $\mathcal{R}(\underline{w}, \underline{v}; \mu) = a(\underline{w}, \underline{v}) + \int_\Omega\ g(\underline{w}; \mu)\ \underline{v}\ \mathrm{d}\Omega\ -\ \ell(\underline{v}, \mu)$.

Linear tangent application $\mathcal{R}'$ is defined by:

$$\mathcal{R}(\underline{w} + \underline{z}, \underline{v}; \mu) - \mathcal{R}(\underline{w}, \underline{v}; \mu)\ \approx\ \langle \mathcal{R}'(\underline{w}, \underline{v}; \mu), z \rangle \tag{1.69}$$

that is to say:

$$\langle \mathcal{R}'(\underline{w}, \underline{v}; \mu), \underline{z} \rangle = a(\underline{z}, \underline{v}) + \int_\Omega\ g'(\underline{w}; \mu)\ z \cdot \underline{v}\ \mathrm{d}\Omega \tag{1.70}$$

with g' the derivative according to the first argument. By denoting the reduced-basis approximation of the solution $\underline{\tilde{u}}_k(\underline{x}, \mu) = \sum_{i=1}^k \lambda_i \underline{\Phi}_i$, and by choosing $\underline{\Phi}_i$ as a test function for $\underline{v}$, it leads to the following $k \times k$ linear algebraic system:

$$\forall\, 1 \leq j \leq k, \quad \sum_{i=1}^{k} \left[a(\underline{\Phi}_i, \underline{\Phi}_j) + \int_\Omega\ \underline{\Phi}_i\ g'\left(\underline{\tilde{u}}^{(n)}; \mu\right) \underline{\Phi}_j\ \mathrm{d}\Omega \right]\ \lambda_i = -\mathcal{R}(\underline{\tilde{u}}^{(n)}, \underline{\Phi}_j; \mu) \tag{1.71}$$

By taking into consideration the underlying finite element approximation ($[\underline{\Phi}_i] = \sum_{j=1}^N [\Phi_i^j] \underline{\varphi}_j(\underline{x})$), the discretized version reduced-order basis is $\mathbb{V}_k$. The discrete approximation of λ leads to the $\Lambda_k = \{\ [\lambda_j]\ \}_{1 \leq j \leq k}$:

$$\mathbb{V}_k^T \left(\mathbb{A} + \mathbb{G}'(\underline{\tilde{u}}^{(n)}; \mu) \right) \mathbb{V}_k\ \Lambda_k = -\mathbb{V}_k^T [R(\underline{\tilde{u}}^{(n)}; \mu)]. \tag{1.72}$$

where $\mathbb{A}$ and $\mathbb{G}'$ are $N \times N$ matrices such that

$$\mathbb{A}_{ij} = a(\varphi_i, \varphi_j), \tag{1.73}$$

$$\mathbb{G}'_{ij} = \int_\Omega \varphi_i \; g'\left(\tilde{\underline{u}}^{(n)}; \mu\right) \varphi_j \; \mathrm{d}\Omega. \tag{1.74}$$

Residue $\{R(\tilde{\underline{u}}^{(n)}; \mu)\}$ is a N-length vector with $\{R\}_i = -\mathcal{R}(\tilde{\underline{u}}^{(n)}, \varphi_i; \mu)$. The discretized ROB is a $N \times k$ matrix with $\mathbb{V}_k = \{[\Phi_1], [\Phi_2], \ldots, [\Phi_k]\}$. The unknowns Λ_k has a size of k.

Both Jacobian matrice $\mathbb{G}'(\tilde{\underline{u}}^{(n)}; \mu)$ and residue $\{R(\tilde{\underline{u}}^{(n)}; \mu)\}$ depend on parameter μ. For each new parameter value, one has (i) to evaluate Jacobian and residue, (ii) to project it onto the discretized ROB $\mathbb{V}_k$, and, finally, (iii) to solve system (1.72) to obtain Λ_k. Projection of Jacobian (resp. residue) onto ROB has a computational complexity that depends on $\mathcal{O}(N\,k^2)$ since $\mathbb{G}'$ is sparse (resp. $\mathcal{O}(N\,k)$) [Grepl et al., 2007, Chaturentabut and Sorensen, 2010]. Solving system (1.72) has a complexity in the order of $\mathcal{O}(k^3)$ (for a Cholesky factorization of a full matrice). The total computational complexity of a Newton iteration is consequently in the order of $\mathcal{O}(N\,k^2 + N\,k + k^3)$ and, hence, it depends on N. This complexity number can even overpass the number of elementary operations to solve the original full order problem 8, that is in the order of $\mathcal{O}(N^3)$. It is shown in [Chaturentabut and Sorensen, 2010] that, for Problem 8, the CPU time for solving the POD reduced model (POD-Galerkin strategy) for each parameter value exceeds the CPU time of the original full order problem as soon as the dimension of the finite element discretisation space reaches around $N = 80$. The analysis of complexity illustrates the well-known *bottleneck* issue of model reduction methods in the framework of nonlinear problems.

In order to overcome this obstacle, several approaches have been proposed. In literature, some of such techniques are indicated as hyper-reduction methods [Carlberg et al., 2013, Ryckelynck, 2005]. In the following section the following techniques are briefly presented:

- Empirical Interpolation Method and its different versions combined with the reduced-basis [Barrault et al., 2004, Rozza, 2006, Nguyen et al., 2005, Maday et al., 2009, Rozza and Patera, 2008] and POD techniques [Chaturentabut and Sorensen, 2010, Galbally et al., 2010].

- Techniques based on the Gappy POD [Everson and Sirovich, 1995]: Missing Points Estimation (MPE) [Astrid et al., 2008], Gauss Newton with approximated tensors (GNAT) [Carlberg et al., 2013], A priori hyper reduction (APHR) [Ryckelynck, 2005, Ryckelynck et al., 2012].

1.2.2 Empirical Interpolation Techniques

The EIM, firstly introduced in [Barrault et al., 2004], has been proposed in an empirically derived finite-dimensional functional space. This method deals with linear

elliptic and parabolic problems with nonaffine parameter dependence as well as nonlinear elliptic and parabolic problems. It operates directly on the governing partial differential equation and therefore at the continuous level. Its variant proposed in [Nguyen and Peraire, 2008] relies for the same purpose on *best (interpolation) points* and a POD basis. In [Chaturentabut and Sorensen, 2010] the Discrete-EIM (DEIM) is proposed. It represents the semi-discrete analogue, without *greedy algorithm*, to the empirical interpolation method that have been developed for parameterized nonlinear stationary problems and for nonlinear dynamic problems. The DEIM can be applied to arbitrary systems of ODEs. This method reduces the computational cost associated with nonlinearities by combining interpolation with projection.

EIM approximates the nonlinear term and the Jacobian of the problem by interpolating a pre-computed basis (built by snapshots computed in off-line stage) related to the nonlinear term evaluated for the snapshots. In order to get the interpolating coefficients of the basis, the method selects some points (called *Magic Points*) by a greedy algorithm. The nonlinear term is, then, evaluated only at these points and combined with the precomputed basis, opportunely masked in order to extract only the lines related to the Magic Points. Finally, the complexity of the evaluation of the nonlinear term, of the Jacobian and their orthogonal projection, is proportional to the number of these Magic Points instead of the finest space dimension. A posteriori error bounds for the EIM can be found in [Grepl et al., 2007, Barrault et al., 2004, Nguyen et al., 2005, Maday et al., 2009], also for its discrete variant [Chaturentabut and Sorensen, 2010]. An a priori error estimation, based on the Lebesgue constant has also been developed. This constant gives an idea of how accurate the interpolant of a function (at the given points) is in comparison with the best polynomial approximation of the function (the degree of the polynomials are obviously fixed). In the case of equidistant nodes, for polynomial interpolations the Lebesgue constant grows exponentially. Using the EIM, for the kind of problem tackled in literature by this technique, this constant tends to the unity as the number of Magic Points increases. The EIM is briefly detailed and applied to a simple elliptic problem in section 3.4.2.

This technique has been coupled with all the principal model reduction techniques. Developed initially for reduced-basis [Barrault et al., 2004], extended then to POD [Chaturentabut and Sorensen, 2010] and recently to PGD [Aguado et al., 2013]. However, for this latter the development is still in progress and leaves some open questions. Indeed, in [Aguado et al., 2013] the EIM has been coupled with the PGD searching for the new points related to a new ROB, generated at each iteration. In this way it reduces the complexity to evaluate and integrate the nonlinear term but the Galerkin projection still scales with the size of the original discretisation because with the PGD the ROB is progressively enriched by new functions and it is not possible to pre-compute the Galerkin projection operators in an off-line stage.

1.2.3 Techniques based on the Gappy POD

Some techniques, dealing with nonlinear model reduction, exploit some results originally found in the field of optics and images. Those studies concerned the problem of using the Karhunen-Loeve transform with partial data. Such problem is explained in this article [Everson and Sirovich, 1995]. The idea originates from the fact that a given amount of data, such as an image or a mechanical evolution field, can be accurately described by few functions (eigenfunctions). This number is much lower than the number of the degrees of freedom in standard discretisations (as the number of pixels for images or the number of nodes for a finite element description of mechanical fields). In [Everson and Sirovich, 1995] marred images are reconstructed by the unmarred ones using a mean square minimization procedure. This method, called Gappy-POD, recovers modal coefficients from an incomplete set of data. In [Everson and Sirovich, 1995], a scheme is proposed for finding empirical eigenfunctions from gappy data. It is shown numerically that this procedure obtains spectra and eigenfunctions that are close to those obtained from the unmarred data.

The Gappy POD has been applied to different data reconstruction problems, such as reconstruction of facial images [Everson and Sirovich, 1995] as well as CFD problems [Willcox, 2006].

Missing Points Estimation The Missing Point Estimation (MPE) developed in [Astrid et al., 2008] performs online computation by POD basis computing Galerkin projections over a restricted subset of the spatial domain. It proposes a quantitative criteria for selecting such a spatial subset based on an heuristic method. The MPE operates at the semi-discrete level and can be coupled to several projection-based model reduction techniques, such as balanced truncation, Krylov subspace methods and POD.
Summarizing the main features of the MPE, we can say that as the Gappy-POD recovers modal coefficients from an incomplete set of data, POD-MPE enables to recover reduced state variables by considering an incomplete set of equations related to the discretisation of PDEs.

The Gauss-Newton with approximated tensors method The Gauss-Newton with approximated tensors (GNAT) method is a nonlinear model reduction method that operates on fully discretized computational models. It achieves dimension reduction by a Petrov-Galerkin projection associated with residual minimization and delivers computational efficiency by a hyper-reduction procedure based on the gappy-POD technique. Originally presented in [Carlberg et al., 2010], where it has been applied to implicit nonlinear structural-dynamics model, this method is further developed in [Carlberg et al., 2013] and applied to the solution of a benchmark turbulent viscous flow problem.

This technique operates on the system of nonlinear equations arising at each time step, which are obtained after discretizing the PDE in both space and time.

GNAT main features are (i) consistency: when GNAT is implemented without snapshots compression, it introduces no additional error in the solution at the training inputs. (ii) discrete-optimality: the error in the discrete approximation decreases monotonically as the approximation spaces expand. Error bounds are provided for this method and they are developed in [Carlberg et al., 2013].

The A Priori Hyper-Reduction method The A priori Hyper-Reduction method (APHR) is an incremental adaptive approach based on a classic step-by-step time-integration scheme. Introduced in [Ryckelynck, 2005] it operates on the functional space of the test-functions used in the weak form of the differential equations. It is based on a specific Petrov-Galerkin formulation, the computational speed-up is achieved by a procedure based on the Gappy-POD.

This method relies on an *a priori* approach because it does not need, in order to build reduced-order models, to solve the full problem by incremental approach in a learning stage. It is an incremental method, then, a classical implicit time integration scheme is used to forecast the state evolution of the space domain. The APHR is a predictor-corrector algorithm. This algorithm provides an approximate solution of the PDEs and an hyper-reduced model (HRM) formed by the reduced basis and the Reduced Integration Domain (RID). At each iteration, the prediction is provided by the current HRM. A truncated error indicator is computed. If this error indicator is too high, then a correction of this prediction is performed by using the solution of the finite elements equations. The computation of the finite elements correction can be performed using any classical incremental algorithm. A POD of this solution is operated to adapt the reduced-bases [Ryckelynck et al., 2012]. A specific spatial integration scheme is introduced when computing the reduced-state variables related to the ROM prediction. The constitutive equations are integrated over the RID, provided that this equations are local. The number of operations to integrate and project the operators scales with the size of the RID. If the reduced-bases have been adapted, the reduced integration domain and the truncated test-functions are therefore updated.

Initially, the APHR method has been used with Krylov subspaces in order to extend the subspace spanned by the ROM basis (see, for instance, [Ammar et al., 2006]). The recent advance on Hyper-Reduction methods proposed in [Ryckelynck, 2009] makes possible the extension of the APHR method to the complex nonlinear mechanical models involving internal variables. But, the expansion using Krylov subspace was not conserved because the Krylov approach increases the number of balance residual evaluations. Therefore the computational time devoted to residuals becomes too expensive in case of complex constitutive laws.

1.2.4 Other techniques

There are some others techniques proposed to palliate the problem of the computational cost induced by nonlinear terms in the reduced-order modelling.

The trajectory piecewise-linear approximations The trajectory piecewise-linear approximations (TPWL), introduced in [Rewienski and White, 2006], constructs a ROM as weighted combination of linear models, where each model lays on the tangent of the original problem trajectory. The TPWL operates at the semi-discrete level, as for instance, on the ODEs obtained after discretizing the PDEs in space. It can be seen as an actual first order Taylor expansion of the nonlinear terms around some chosen state values.

Nonuniform Transformation Field Analysis For solid mechanical problems with idealized interfaces (i.e. without displacement jumps) but (possibly) nonlinear bulk behaviour, a promising approach has been presented in terms of the Nonuniform Transformation Field Analysis (NTFA) [Michel and Suquet, 2003]. The NTFA belongs to the class of reduced basis methods. More precisely, dealing with nonlinear micromechanically problems, the plastic strain tensor is approximated by nonuniform global basis functions while the hardening variables are assumed to be phase-wise constant. Recently, the method has been extended for the modelling of the cohesive interfaces with hyperelastic cohesive laws [Fritzen and Leuschner, 2015]. The method aims at to reduce parameterization of the internal variables in the material thanks to a previously developed a reduced order model for nonlinear solid materials that is based on a low-dimensional parameterization of the internal variables of the dissipative phases.

The expanded space approach In [Hernandez et al., 2014] is addressed the problem of model reduction for the solution of the fine-scale equilibrium problem appearing in computational homogenization. In this work it is shown that, for the problems coming from homogenization, the standard approach of replacing the nonaffine term by an interpolant constructed using only POD modes (the EIM) leads to ill-posed formulations. Such problem can be avoided by enriching the approximation space with the span of the gradient of the empirical shape functions. Furthermore, interpolation points are chosen, not only by accuracy requirements, but also by stability considerations. In some numerical results is shown that computational complexity is independent of the size and geometrical complexity of the Representative Volume Element.

1.3 Conclusions

In this section it is shown how, for linear problems, model reduction techniques enable one to decrease the computational complexity related to systems of linear equations obtained by discretizing PDEs. The detailed techniques seek the solution of a problem in a reduced-order basis (ROB), whose dimension is much smaller than the original vectorial space. This ROB can be acquired either by solving the original system for some *appropriate* values of the parameter domain (snapshots of

the solution) in a learning stage or it can be constructed *on-the-fly* by a greedy algorithm, during the solution of the PDE projected on the ROB. The comparison with respect to the direct solution techniques on a simple $3D$ problem shows an important gain in the computational cost.

On the other hand, when the projection is applied to linear dynamic systems, linear stationary systems with non-affine parameter dependence, or general nonlinear problems, the resulting ROM is expensive to assemble. This high cost results from the need to evaluate the high-dimensional nonlinear function (and eventually its Jacobian) and, then, to project it to get the low-dimensional operators at each computational step of a solution algorithm. This results to be the bottleneck of nonlinear model reduction strategies.

Some solutions have already been introduced in literature. The most common one is the Empirical Interpolation Method (EIM) [Barrault et al., 2004, Grepl et al., 2007, Nguyen, 2005] and its semi-discrete version (DEIM) [Chaturentabut and Sorensen, 2010]. These techniques have been introduced for model reduction techniques that resort to a learning stage, i.e., POD and reduced-basis. Extensions to PGD approximation are in progress, i.e., in [Aguado et al., 2013]. However the use of the EIM coupled with the PGD leaves some opened questions. Indeed, in [Aguado et al., 2013] the EIM has been coupled with the PGD searching for the new points related to a new ROB, generated at each iteration. In this way it reduces the complexity to evaluate and integrate the nonlinear term, but the Galerkin projection still scales with the size of the original discretisation because with the PGD the ROB is progressively enriched by new functions and it is not possible to pre-compute the Galerkin projection operators in an off-line stage.

The aim of this work is to propose an approximation framework, called Reference Points Method (RPM), in order to decrease the computational complexity of algebraic operations when dealing with separated variable approximations and in the case of evolving ROB.

Chapter 2

The Reference Points Method

In this chapter a new approximation framework, well-suited for the PGD technique, is developed. This framework is based on the concept of reference times, points and parameters and enables one to define a reduced version of the data which allows to decrease the computational complexity and the number of algebraic operations, to construct a reduced order model. Herein, starting from the *compressed* version of data, a method to reconstruct the data in a separated-variables form is developed, analysed and demonstrated by numerical examples.

In chapter 1, it has been shown how reduced order modelling, by separation of variables, represents a way to reduce the complexity related to the solution of parametric linear problems. On the other hand, in section 1.2 it has been explained that, dealing with nonlinear problems, a linearization technique is necessary and, thus, a new ROM has to be constructed at each new iteration of the solution method. The complexity related to the necessary operations scales with the underlying space and parameter discretisation and it hints the potential gain of the reduced order modelling.

Let us consider again Problem 7, introduced previously in chapter 1, which is an μ-parametrized elliptic nonlinear problem and let us assume that the problem has been linearized by a Newton scheme in Problem 9. By denoting the separated-variables approximation of the solution $\underline{\tilde{u}} = \sum_{i=1}^{k} \lambda_i \, \underline{\Phi}_i$, and by choosing $\underline{\Phi}_i$ as a test function for $\underline{v}$, the following $k \times k$ linear algebraic system has been obtained:

$$\forall\, 1 \leq j \leq k, \quad \sum_{i=1}^{k} \left[a(\underline{\Phi}_i, \underline{\Phi}_j) + \int_\Omega \underline{\Phi}_i \; g'\left(\tilde{\underline{u}}^{(n)}, \mu\right) \underline{\Phi}_j \; \mathrm{d}\Omega \right] \; \lambda_i = -\mathcal{R}(\tilde{\underline{u}}^{(n)}, \underline{\Phi}_j; \mu) \tag{2.1}$$

where $a(\underline{\Phi}_i, \underline{\Phi}_j)$ is a bilinear form and $\mathcal{R}(\tilde{\underline{u}}^{(n)}, \underline{\Phi}_j; \mu)$ a linear form. $g'\left(\tilde{\underline{u}}^{(n)}, \mu\right)$ is a continuous nonlinear operator depending on the solution and the parameter μ. The reduced model is obtained from the operation on the left-hand side:

$$a(\underline{\Phi}_i, \underline{\Phi}_j) + \int_\Omega \underline{\Phi}_i \; g'\left(\tilde{\underline{u}}^{(n)}, \mu\right) \underline{\Phi}_j \; \mathrm{d}\Omega. \tag{2.2}$$

First term in left-hand side member, $a(\underline{\Phi}_i, \underline{\Phi}_j)$, is parameter-independent and can thus be pre-computed offline. However, the second term:

$$\int_\Omega \underline{\Phi}_i \; g'\left(\tilde{\underline{u}}^{(n)}, \mu\right) \underline{\Phi}_j \; \mathrm{d}\Omega \tag{2.3}$$

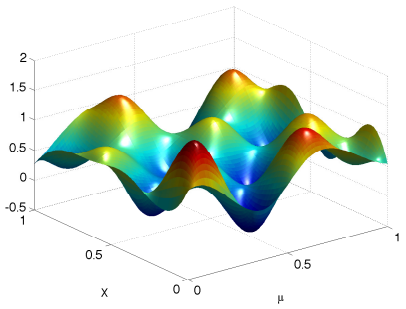
depends on $g'\left(\tilde{\underline{u}}^{(n)}, \mu\right)$ and, consequently, is parameter-dependent and has to be evaluated online for each new parameter value $\mu \in \mathcal{D}$. It was shown that the projection of the Jacobian onto the reduced-order basis has a computational complexity that scales with $\mathcal{O}(N\ k^2)$. This point, described in the previous chapter, represents the *bottleneck* of reduced order modelling in the framework of nonlinear problems and it is even more important when the number of space functions in the ROB increases. For that reason, the aim of this work is to propose a new technique, called Reference Points Method (RPM), which consists in:

- providing a compressed version of quantities based on the concept of *reference times*, *points* and *parameters* [Ladevèze, 1997]. This approximation framework enables one to reduce the complexity of algebraic operations between quantities in separated-variables form.
- reconstructing the quantities to give a first approximation of them in a separated-variables form by explicit formulas.
- avoiding the artificial increasing of the PGD modes representing the result of the algebraic operation between quantities in separated-variables form.

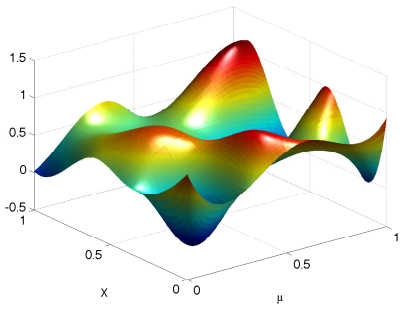
In this chapter, for the sake of clarity, let us focus on the evaluation of the integrand of expression (2.3), considered as the product of two functions:

$$F(\mu, x) = f(\mu, x)\, f'(\mu, x). \tag{2.4}$$

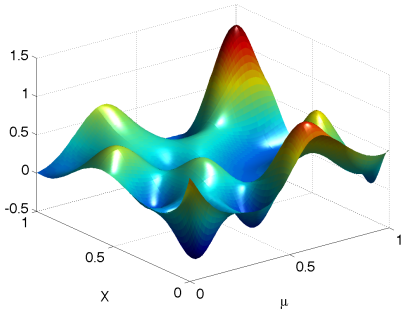
$f(\mu, x)$ and $f'(\mu, x)$ are two scalar functions of two variables, $\mu \in \mathcal{D} = [0, 1]$ and $x \in \Omega = [0, 1]$. An example for functions f and f' and their product F is given in **Fig. 2.1**. The extension to function defined in $3D$ domains or depending on the time will be presented in the following.



(a) *scalar function* $f(\mu, x)$.



(b) *scalar function* $f'(\mu, x)$.



(c) *the resulting product function* $F(\mu, x) = f(\mu, x)\, f'(\mu, x)$.

Figure 2.1: Example of two scalar functions and their product.

2.1 Compressed format and generalized components

Let us assume that functions f and f' are described under separated-variables representations:

$$f(\mu, x) = \sum_{i=1}^{k} \lambda_i(\mu)\ \Lambda_i(x) \qquad \text{and} \qquad f'(\mu, x) = \sum_{i=1}^{k'} \theta_i(\mu)\ \Theta_i(x). \tag{2.5}$$

The evaluation of product F, under its separated variable form, involves the sum of $M{=}k \times k'$ term by term products, which results into a M-term representation of product F. This M-term representation is likely to be non-optimal and M may increase swiftly if terms are added to f and f' representations (k and k' increase). Performing a singular value decomposition of F may be necessary to achieve a separated-variables representation of F with a reasonable number of terms. RPM follows a different path to avoid the artificial increasing of the PGD modes.

The RPM approximation framework is based on the concept of *reference times*, *points* and *parameters* and enables one to define a reduced version of the data [Ladevèze, 1997]. The parameter domain is split in m_μ sub-intervals $\mathcal{D}_i$ of the size $\Delta\mu_i$. The center μ_i of the sub-interval $\mathcal{D}_i$ are called *reference parameter points*. For the space domain m_x points x_j are introduced and the domain Ω is divided into m_x sub-domains Ω_j. The x_j are called *reference spatial points* and the size of the patch is indicated $\Delta\omega_j$.

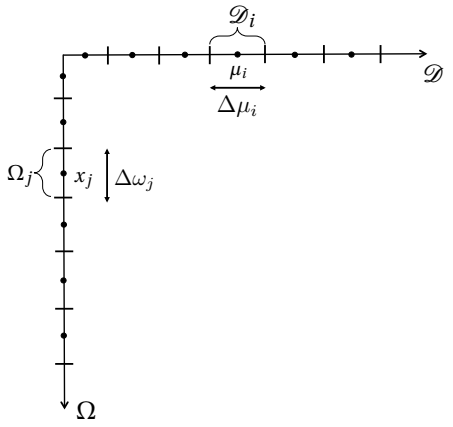


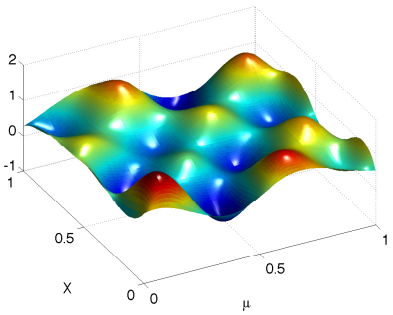
Figure 2.2: Reference parameters and reference points when $\mathcal{D} = [0, 1]$ and $\Omega = [0, 1]$.

An influence zone is defined around each reference point given by the x_j space point and the μ_i parameter point. This part of the domain, $\mathcal{D}_i \times \Omega_j$, is called reference patch (i, j). Then, the entire domain $\mathcal{D} \times \Omega$ is divided in $m_\mu \times m_x$ patches.

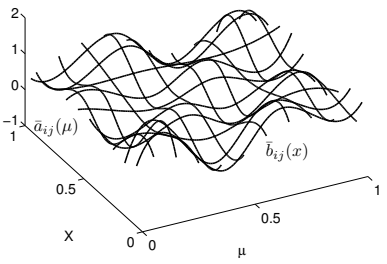
A function f defined on the domain $\mathcal{D} \times \Omega$ is represented by its *generalized*

components $\bar{f} = \{(\bar{a}_{ij}, \bar{b}_{ij})\}$, defined as follows. For $i = 1, .., m_\mu$ and $j = 1, .., m_x$:

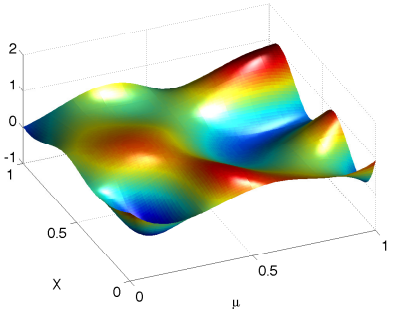
$$\bar{f} \equiv \left\{ \begin{array}{rcccl|rcccl} \bar{a}_{ij}(\mu) & = & f(\mu, x_j) & \text{if}\, \mu & \in \mathcal{D}_i & \bar{b}_{ij}(x) & = & f(\mu_i, x) & \text{if}\, x & \in \Omega_j \\ \bar{a}_{ij}(\mu) & = & 0 & \text{otherwise} & & \bar{b}_{ij}(x) & = & 0 & \text{otherwise} & \end{array} \right\} \quad (2.6)$$



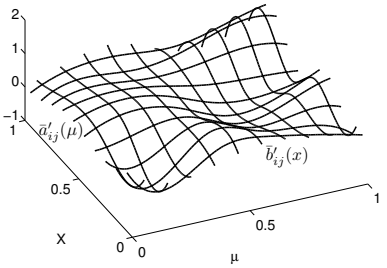
(a) *function* $f(\mu, x)$.



(b) $\bar{f}$, *the generalized components of function* $f(\mu, x)$.



(c) *function* $f'(\mu, x)$.



(d) $\bar{f}'$, *the generalized components of function* $f'(\mu, x)$.

Figure 2.3: The generalized components of functions f, f' and their product F, for a choice of $m_\mu = 10$ and $m_x = 10$.

The generalized components $\{\bar{a}_{ij}(\mu)\}_{i=1,...,m_\mu}$ related to spatial point x_j gives the description of the function f at spatial point x_j over the entire parameter domain $\mathcal{D}$. Inversely the generalized components $\{\bar{b}_{ij}(x)\}_{i=1,...,m_x}$ related to parameter value μ_i gives the description of the function f at parameter value μ_i over the entire spatial domain Ω.

Remark 3 *Whenever the function f is expressed in a separated-variables form, each pair of functions is described by the generalized components. Hence, for*

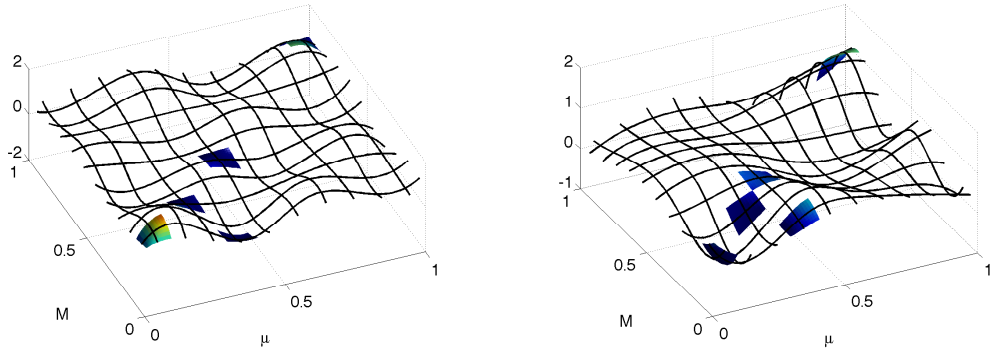
$$f(\mu, x) = \sum_{i=1}^{k} \lambda_i(\mu)\ \Lambda_i(x),$$

it yields: for $i = 1, .., m_\mu$ *and* $j = 1, .., m_x$

$$\bar{f} \equiv \left\{ \begin{array}{ll|ll} \bar{a}_{ij}(\mu) = \sum_{i=1}^{k} \lambda_i(\mu)\Lambda_i(x_j) & \textit{if } \mu \in \mathcal{D}_i & \bar{b}_{ij}(x) = \sum_{i=1}^{k} \lambda_i(\mu_i)\Lambda_i(x) & \textit{if } x \in \Omega_j \\ \bar{a}_{ij}(\mu) = 0 & \textit{other.} & \bar{b}_{ij}(x) = 0 & \textit{other.} \end{array} \right\} \tag{2.7}$$

Fig. 2.3 depicts surfaces defined by functions f and f'. Their generalized components are shown also when $m_\mu = 10$ and $m_x = 10$ reference points are considered. Let us remark that for a $2D$ defined function, a generalized components associated to a reference point is a quantity defined over a line. For instance, for a fixed reference space point it contains the description of the quantity over the entire parameter domain.

In **Fig. 2.4**, some patches for functions f and f' are depicted. These patches can easily be imagined, since it is a parametric function defined over a two-dimensional coordinate.



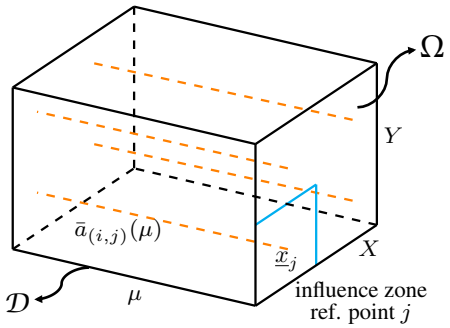
(a) *some of the patches for function* $f(\mu, x)$.

(b) *some of the patches for function* $f'(\mu, x)$.

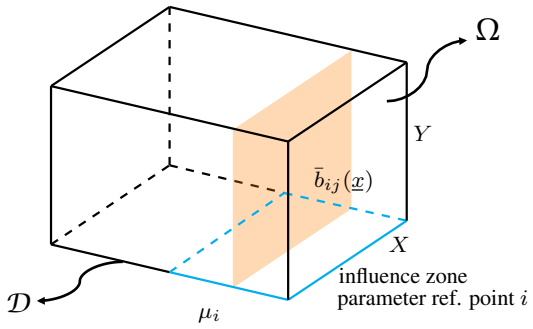
Figure 2.4: Some of the parameter-space patches $\mathcal{D}_i \times \Omega_j$ for function f and f'.

For a single parameter problem defined on a two-dimensional space domain, patches become volumes. For a rectangular $2D$ space domain ($F(\underline{x}, \mu)$ with $\underline{x} = (X, Y)$), a patch is a parallepiped (see **Fig. 2.5**). Reference and parameter points x_j and μ_i can be chosen on a regular grid or arbitrary provided that patches are easily defined. Generalized components in this case are highlighted in **Fig. 2.5**. For $3D$ space domain, patch dimension is higher than three and, consequently, is more complicated to represent.

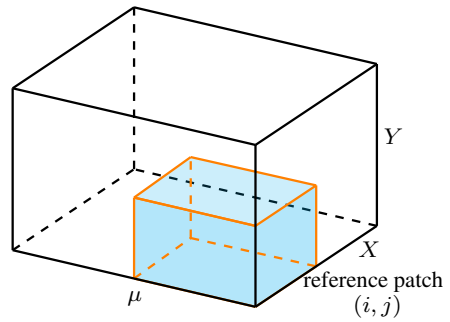
The algorithm to define the patches, for a chosen number of reference points on each coordinate, takes into account the eventually irregular geometry of the domain. If a point takes place outside of the domain where the given field $f(x, \mu)$ or $f'(x, \mu)$ is undefined, e.g. in an hole of the space domain, the reference point is moved to the closest node inside the domain. **Fig. 2.6** shows how the points move accordingly to the problem in the case of the presence of a hole in the domain.



(a) $\bar{a}_{ij}(\mu)$ generalized component related to the spatial point x_j



(b) $\bar{b}_{ij}(x)$, generalized component related to the parameter value μ_i



(c) patch ij for a $2D$ parametric space field

Figure 2.5: Generalized components and patch (i, j) for a $2D$ parametrized function.

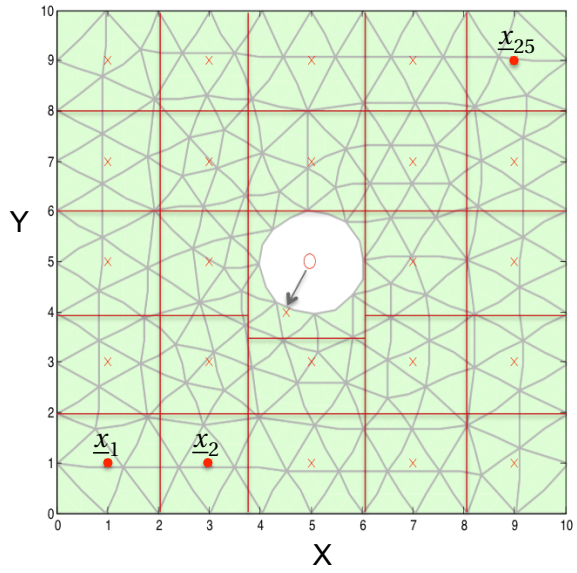


Figure 2.6: The construction of the reference patches in the space domain around the reference points. These will be extruded in the third dimension to take into account the reference parameter evolution.

2.2 Algebra in the compressed framework

It is straightforward to show that the space of compressed fields shows interesting properties regarding elementary operations (**Tab. 2.1**).

Addition	$\overline{f + f'} = \bar{f} + \bar{f}'$
Multiplication	$\overline{f\, f'} = \bar{f}\, \bar{f}'$
Derivation	$\overline{\partial f/\partial\mu} = \partial \bar{f}/\partial\mu$
Operator	$\overline{H\, f} = \bar{H}\, \bar{f}$

Table 2.1: Elementary operations in the compressed framework.

For instance, the evaluation of the product of two functions f and f' (see (2.4)) is obtained as follows:

1. Determination of the compressed formats of f and f' (see (2.6)): $\bar{f} = \{(\bar{a}_{ij}, \bar{b}_{ij})\}$ and $\bar{f}' = \{(\bar{a}'_{ij}, \bar{b}'_{ij})\}$.

2. $\bar{F} = \{(\bar{A}_{ij}, \bar{B}_{ij})\}$, with:

$$\bar{A}_{ij} = \bar{a}_{ij}\ \bar{a}'_{ij} \qquad \textit{and} \qquad \bar{B}_{ij} = \bar{b}_{ij}\ \bar{b}'_{ij}. \tag{2.8}$$

This stage is illustrated in **Fig. 2.7**.
In the case when f and f' are in separated variables form (see (2.5)). Following (2.7), it leads to:

$$\bar{A}_{ij} = \sum_{i=1}^{k} \lambda_i(\mu)\Lambda_i(x_j) \sum_{i=1}^{k} \theta_i(\mu)\Theta_i(x_j) \quad \text{and} \quad \bar{B}_{ij} = \sum_{i=1}^{k} \lambda_i(\mu_i)\Lambda_i(x) \sum_{i=1}^{k} \theta_i(\mu_i)\Theta_i(x) \tag{2.9}$$

Remark 4 *Let us remark that in* (2.9) *the reconstruction of the quantities in the "full" format is made only on the generalised components. This involves that the complexity related to the* (2.9) *is decreased with respect to the one related to the classic tensor product expressed in* (2.5)*. The RPM enables one to avoid the complete reconstruction of the quantities in the "full" format, considering the only generalised components of them.*

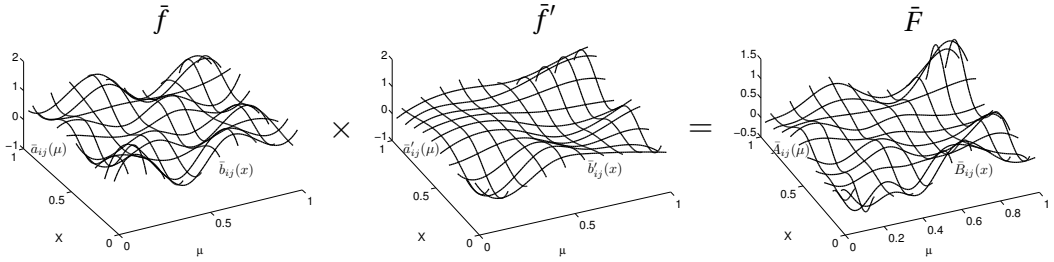


Figure 2.7: The product $F = f\,f'$ in the RPM algebra is operated on the generalized components $\bar{F} = \bar{f}\,\bar{f}'$

2.3 Reconstruction from the RPM format

2.3.1 First approximation

The generation of a first approximation of F in a separated-variables format, denoted $\bar{\bar{F}}$, is obtained from the compressed format $\bar{F}$ by generating one product of functions per parameter-space patch $\mathcal{D}_i \times \Omega_j$ (**Fig. 2.8**):

$$\forall\,(i,j) \qquad \forall(\mu,x) \in \mathcal{D}_i \times \Omega_j \qquad F(\mu,x) \approx \bar{\bar{F}}(\mu,x) = a_{ij}(\mu)b_{ij}(x) \tag{2.10}$$

Products of functions $a_{ij}(\mu)b_{ij}(x)$ for all patches (i,j) are determined from the generalized components of $\bar{F}$, $\{\bar{A}_{ij}(\mu)\bar{B}_{ij}(x)\}$ thanks to the solution of a minimization problem as described in the following.

The norms with respect to the parameter domain and the space are introduced as follows:

$$\|f\|^2_{\mathcal{D}_i} = \int_{\mathcal{D}_i} f^2 d\mu \qquad and \qquad \|f\|^2_{\Omega_j} = \int_{\Omega_j} f^2 d\Omega_j \tag{2.11}$$

Let us introduce the functional (2.12) on patch $\mathcal{D}_i \times \Omega_j$:

$$J(a_{ij}, b_{ij}) = \sum_{k=1}^{m_x} \left[\Delta\omega_k \|(\bar{A}_{ik}(\mu) - a_{ik}(\mu) b_{ik}(x_k))\ \lambda_{ik}\|^2_{\mathcal{D}_i} + \Delta\mu_i \|\bar{B}_{ik}(x) - a_{ik}(\mu_i) b_{ik}(x)\|^2_{\Omega_k}\right]. \tag{2.12}$$

This functional imposes to the reconstructed patch to minimize its distance from the generalized components. For the reconstruction of a given patch, the influence of the neighbouring patches over the space coordinates is taken into the account by the sum over the space coordinates. For that λ_{ik} is an influence function which gives more importance to the patches next to the considered patch. Let us write the minimization problem:

$$\min_{(a_{ij}(\mu), b_{ij}(x)) \ \in \ \mathcal{P}\times\mathcal{U}} \sum_{k=1}^{m_x} \left[\Delta\omega_k \int_{\mathcal{D}_i} (\bar{A}_{ik}(\mu) - a_{ik}(\mu) b_{ik}(x_k))^2\ \lambda^2_{ik}\ d\mu + \ldots \right.$$
$$\left. \ldots\ \Delta\mu_i \int_{\Omega_j} (\bar{B}_{ik}(x) - a_{ik}(\mu_i) b_{ik}(x))^2\ d\Omega_j \right] \tag{2.13}$$

Minimization of functional $J(a_{ij}, b_{ij})$ leads to a variational problem:

$$\delta(a_{ik}(\mu)\ b_{ik}(x)) \ = \ a^\star_{ik}(\mu)\ b_{ik}(x) \ + \ \ a_{ik}(\mu)\ b^\star_{ik}(x) \tag{2.14}$$

$$\sum_{k=1}^{m_x} \left[\Delta\omega_k \int_{\mathcal{D}_i} 2\ (a^\star_{ik}(\mu)\ b_{ik}(x_k) \ + \ a_{ik}(\mu)\ b^\star_{ik}(x_k))\ (\bar{A}_{ik}(\mu) - a_{ik}(\mu) b_{ik}(x_k))\ \lambda^2_{ik}\ d\mu \ + \ldots \right.$$
$$\left. \ldots\ \Delta\mu_i \int_{\Omega_j} 2\ (a^\star_{ik}(\mu_i)\ b_{ik}(x) \ + \ a_{ik}(\mu_i)\ b^\star_{ik}(x))\ (\bar{B}_{ik}(x) - a_{ik}(\mu_i) b_{ik}(x))\ d\Omega_j \right] = 0 \tag{2.15}$$

Let us choose, $\forall\ (i,k)$:

$$b^\star_{ik}(x) \ = \ 0 \tag{2.16}$$
$$a^\star_{ik}(\mu_i) \ = \ 0. \tag{2.17}$$

Hence, $\forall\ a^\star_{ik}(\mu)$ and in particular for a given one such as $a^\star_{ik}(\mu_i) = 0$:

$$\sum_{k=1}^{m_x} 2\ \Delta\ \omega_k \ \int_{\mathcal{D}_i} (a^\star_{ik}(\mu)\ b_{ik}(x_k))\ (\bar{A}_{ik}(\mu) - a_{ik}(\mu)\ b_{ik}(x_k))\ \lambda^2_{ik}\ d\mu \ = \ 0 \tag{2.18}$$

This leads to:

$$2 \int_{\mathcal{D}_i} a^\star_{ik}(\mu) \, (\sum_{k=1}^{m_x} \Delta\omega_k \;\; b_{ik}(x_k) \, (\bar{A}_{ik}(\mu) - a_{ik}(\mu) \, b_{ik}(x_k) \, \lambda^2_{ik}) \, d\mu \;=\; 0 \tag{2.19}$$

Equation (2.19) is verified $\forall \; a^\star_{ik}(\mu)$ and in particular for a given one such as $a^\star_{ik}(\mu_i) = 0$. It yields:

$$\sum_{k=1}^{m_x} \Delta\omega_k \;\; b_{ik}(x_k) \; \bar{A}_{ik}(\mu) \; \lambda^2_{ik} = \sum_{k=1}^{m_x} \Delta\omega_k \; a_{ik}(\mu) \; b^2_{ik}(x_k) \; \lambda^2_{ik} \tag{2.20}$$

$$a_{ik}(\mu) = \frac{\sum_{k=1}^{m_x} \Delta\omega_k \; \bar{A}_{ik}(\mu) \; \lambda^2_{ik}}{\sum_{k=1}^{m_x} \; \Delta\omega_k \; b^2_{ik}(x_k) \; \lambda^2_{ik}}, \quad \forall \; \mathcal{D}_i \tag{2.21}$$

Now, in order to seek an expression for $b_{ik}(x)$, let us come back to the (2.15). Let us impose, $\forall \; i$:

$$b^\star_{ik}(x_j) \;=\; 0 \tag{2.22}$$

$$a^\star_{ik}(\mu) \;=\; 0. \tag{2.23}$$

This choice leads to:

$$\sum_{k=1}^{m_x} 2 \; \Delta\mu_i \; \int_{\Omega_j} b^\star_{ik}(x) \; a_{ik}(\mu_i) \; (\bar{B}_{ik}(x) - a_{ik}(\mu_i) b_{ik}(x_k) \;) \; d\Omega \;=\; 0 \tag{2.24}$$

This latter is verified $\forall \; b^\star_{ik}(\mu)$ and in particular for a given one such as $b^\star_{ik}(x_j) = 0$. It yields:

$$b_{ij}(x) = \frac{\bar{B}_{ij}(x)}{a_{ij}(\mu_i)}, \quad \forall x \; \in \Omega_j \tag{2.25}$$

Now, replacing (2.25) in (2.21) one obtains:

$$a_{ij}(\mu) = \left(\frac{\sum_{k=1}^{m_x} \Delta\omega_k \; \bar{A}_{ik}(\mu) \; \bar{B}_{ik}(\mu_i) \; \lambda^2_{ik}}{\sum_{k=1}^{m_x} \; \Delta\omega_k \; \bar{B}_{ik}(x_k)^2 \; \lambda^2_{ik}} \right) \; a_{ik}(\mu_i), \quad \forall \; \mathcal{D}_i \tag{2.26}$$

Considering that by definition of the generalized components:

$$\bar{B}_{ik}(x_k) \;=\; \bar{A}_{ik}(\mu_i) \tag{2.27}$$

It follows:

$$a_{ij}(\mu) = \left(\frac{\sum_{k=1}^{m_x} \Delta\omega_k \; \bar{A}_{ik}(\mu) \; \bar{A}_{ik}(\mu_i) \; \lambda^2_{ik}}{\sum_{k=1}^{m_x} \; \Delta\omega_k \; \bar{A}_{ik}(\mu_i)^2 \; \lambda^2_{ik}} \right) \; a_{ij}(\mu_i), \quad \forall \; \mu \in \mathcal{D}_i \tag{2.28}$$

Let us consider the function F product of the two functions f and f'. The approximation of F by the RPM is given by:

$$F(x, \mu) \;\approx\; \bar{\bar{F}} \;=\; a_{ij}(\mu) b_{ij}(x) \tag{2.29}$$

Replacing (2.28) and (2.25) in (2.29), one obtains:

$$F(x,\mu) \approx \bar{\bar{F}} = \left(\frac{\sum_{k=1}^{m_x} \Delta\omega_k \ \bar{A}_{ik}(\mu) \ \bar{A}_{ik}(\mu_i) \ \lambda_{ik}^2}{\sum_{k=1}^{m_x} \Delta\omega_k \ \bar{A}_{ik}(\mu_i) \ \bar{A}_{ik}(\mu_i)_{ik} \ \lambda_{ik}^2} \right) \bar{B}_{ij}(x) \tag{2.30}$$

Observing (2.30), it is possible to simplify expressions (2.28) and (2.25) for functions a_{ij} and b_{ij} just by scaling a_{ij} and multiplying b_{ij} by $a_{ik}(\mu_i)$. The (2.30) shows that this choice does not affect the reconstruction, since their product does not change, and enables one to obtain the following explicit formulas:

$$a_{ij}(\mu) = \frac{\sum_{k=1}^{m_x} \Delta\omega_k \ \bar{A}_{ik}(\mu) \ \bar{A}_{ik}(\mu_i) \ \lambda_{ik}^2}{\sum_{k=1}^{m_x} \Delta\omega_k \ \bar{A}_{ik}(\mu_i) \ \bar{A}_{ik}(\mu_i) \ \lambda_{ik}^2} \qquad b_{ij}(x) = \bar{B}_{ij}(x) \tag{2.31}$$

Hence, space domain is favored holding all the information arising from the spatial generalized component $\bar{B}_{ij}(x)$. This result is suitable for structural mechanics where the spatial gradients of quantities are usually stronger than their variations in parameters/time.

Let us notice that, if one chooses:

$$\lambda_{ik} = 1 \quad \forall k \in m_x,$$

it follows:

$$\bar{\bar{F}}(x,\mu_i) = \bar{B}_{ij}(x), \quad \text{for } \mu = \mu_i$$

Hence, if influence function λ is not taken into account ($\lambda_{ik} = 1 \quad \forall k \in m_x$), the continuity of the approximated function $\bar{\bar{F}}$ over the space domain is ensured for all $\mathcal{D}_i$.

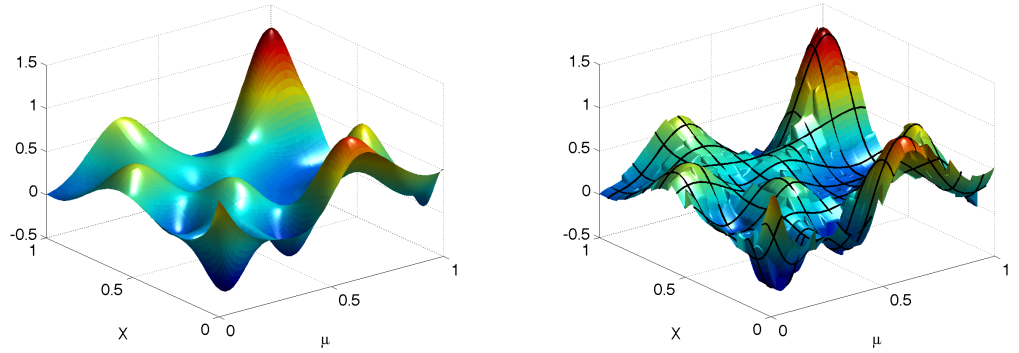
Using $m_\mu = 10$ reference times and $m_x = 10$ reference points, selected as a regular grid, the reconstruction from the set of generalized components $\{\bar{A}_{ij}, \bar{B}_{ij}\}$ (right-hand plot in **Fig. 2.8**) leads to an error $e = 5\%$ with respect to the exact solution (see 2.32). This error is defined as follows:

$$e = \frac{\|F - \bar{\bar{F}}\|_{\mathcal{D}\times\Omega}}{\|\bar{\bar{F}}\|_{\mathcal{D}\times\Omega}}, \quad \|F\|^2_{\mathcal{D}\times\Omega} = \int_{\mathcal{D}\times\Omega} F^2 d\Omega d\mu = \sum_{i=1}^{m_\mu} \sum_{j=1}^{m_x} \int_{\mathcal{D}_i\times\Omega_j} F^2 d\Omega d\mu. \tag{2.32}$$

Whenever the quality of the approximation is poor, one can choose to enrich this first approximation in two ways:

- the approximation is improved by adding new PGD pairs;
- the approximation is improved by adding more reference points. This latter choice leads to the redefinition of the influence patches and the issue of the selection of the new points arises.

These procedures for the enrichment of the first approximation are analysed later, in section 2.3.2.



(a) F *exact.* (b) $\bar{\bar{F}}$ *reconstructed by RPM.*

Figure 2.8: The exact function F and its approximation $\bar{\bar{F}}$ by $m_\mu = 10$ reference parameters and $m_x = 10$ reference points.

Cost of the reconstruction By the explicit formulas (2.31), one obtains the reconstruction of $\bar{F}$ over the entire parameter-space domain:

$$\bar{\bar{F}}(\mu, x) = \sum_{i=1}^{m_\mu} \sum_{j=1}^{m_x} a_{ij}(\mu) b_{ij}(x). \tag{2.33}$$

The cost of reconstruction, in terms of FLOPs, is simply in the order of $8(m_x^2\ m_t)+2$. This computational cost is moderate thanks to the explicit form (2.31).

On the choice of the function λ By definition of functional (2.12), function λ controls the influence of the next patches on the considered one. λ is defined as a discrete function of the space domain:

$$\lambda \ = \ \lambda(|\ x_j \ - \ x\ |) \tag{2.34}$$

In order to illustrate the analysis about λ, let us consider a simple function that enables one to depict clear images:

$$F \ = \ e^{-|(x-0.5)\ \ (\mu-1)|} \ + \ \sin(x\ \mu) \tag{2.35}$$

In this case function F is a given function and it is not a product of functions. F is displayed in **Fig. 2.9**.

The limit case of the independent patches In this case, $\lambda = 1$ when $k = j$ and slopes to 0 otherwise. In such a way the reconstructed patch is not affected by other patches in the neighbourhood (see **Fig. 2.10**). Replacing λ in (2.31), it yields:

$$a_{ij}(\mu) = \frac{\Delta\omega_j \bar{A}_{ij}(\mu)\bar{B}_{ij}(x_j)}{\Delta\omega_j \bar{B}_{ij}(x_j)\bar{B}_{ij}(x_j)} \qquad \text{and} \qquad b_{ij}(x) = \bar{B}_{ij}(x) \tag{2.36}$$

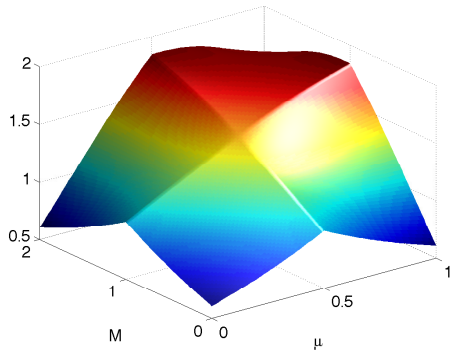
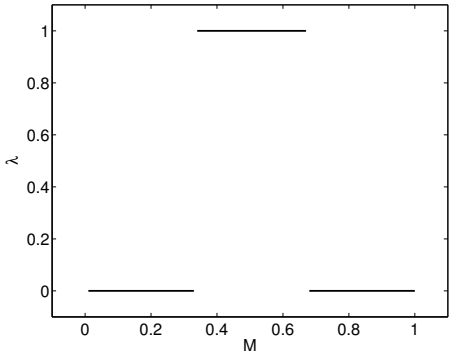


Figure 2.9: $F = e^{-|(x-0.5)\ (\mu-1)|} + \sin(x\ \mu)$.

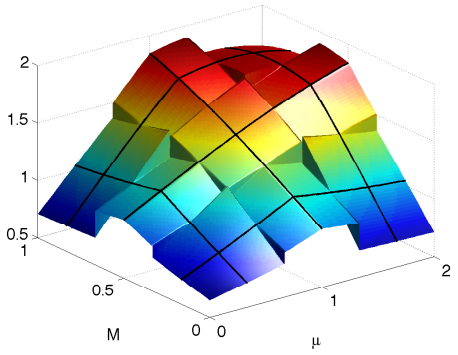
Patches are reconstructed completely independently of the others. $\bar{\bar{F}}$ is written as follows:

$$\forall\, i,j \quad \forall (\mu,x) \in \mathcal{D}_i \times \Omega_j \quad \bar{\bar{F}}(\mu,x) = \sum_{i=1}^{m_\mu}\sum_{j=1}^{m_x} a_{ij}(\mu)b_{ij}(x) = \sum_{i=1}^{m_\mu}\sum_{j=1}^{m_x} \frac{\Delta\omega_j \bar{A}_{ij}(\mu)\bar{B}_{ij}(x)}{\Delta\omega_j \bar{B}_{ij}(x_j)} \tag{2.37}$$

Fig. 2.10 presents the reconstruction given by (2.36) considering 3×3 reference points in the case when patches are reconstructed completely independently to the others.



(a) *shape of the influence function lambda for the central patch of the function* $\bar{\bar{F}}$.



(b) $\bar{\bar{F}}$, *reconstructed by the* (2.36) *by using* 3×3 *reference points.*

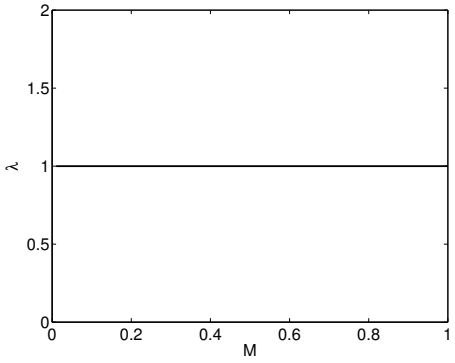
Figure 2.10: F and $\bar{\bar{F}}$ reconstructed by (2.36) when $\lambda = 1$ for $k = j$ and 0 otherwise.

The limit case of completely influenced patches In this case, at the opposite to the previous one, $\lambda = 1$, $\forall k \in m_x$. It is that, the reconstruction of a patch k

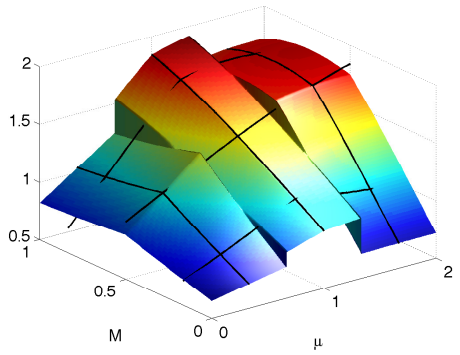
is influenced by every patch of the entire domain, with the same weight (see **Fig. 2.11**). Replacing λ in (2.31), it yields:

$$a_{ij}(\mu) = \frac{\sum_{k=1}^{m_x} \Delta\omega_k \bar{A}_{ik}(\mu)\bar{B}_{ik}(\underline{x}_k)}{\sum_{k=1}^{m_x} \Delta\omega_k \bar{B}_{ik}(x_k)\bar{B}_{ik}(x_k)} \qquad \text{and} \qquad b_{ij}(x) = \bar{B}_{ij}(x) \tag{2.38}$$

Fig. 2.11 depicts the reconstruction given considering 3×3 reference points in the case when every patch influences the reconstructed patch with the same weight. This images illustrates that, in such a way case, the function $\bar{\bar{F}}$ shows the continuity over the space coordinate. Indeed, considering (2.31) without the influence function λ ($\lambda = 1$), continuity is ensured over the entire space coordinate, for all $\mathcal{D}_i$.



(a) *shape of the influence function lambda for the central patch of the function* $\bar{\bar{F}}$.



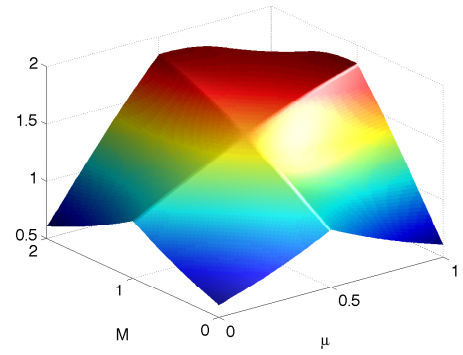
(b) $\bar{\bar{F}}$, *reconstructed for the completely influenced patches by using* 3×3 *reference points.*

Figure 2.11: F and $\bar{\bar{F}}$ reconstructed by (2.38) when $\lambda = 1$ for $\forall k \in m_x$.

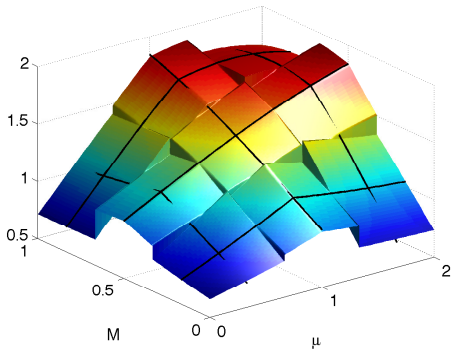
The optimization of the function λ An optimization study has found the optimal shape for function λ. Several functions have been tested to arrive to the optimum. It consists in a function that assumes value 1 when $k = i$ and slopes to 0.1 when $|k-i| = 1$ and 0 otherwise. In practice, only the first neighbouring patches give a contribution to the reconstruction of the considered patch. For most of the tested functions, this choice of function λ represents the optimal one. Function F is presented in **Fig. 2.12** with its approximation $\bar{\bar{F}}$ given by the (2.31) by using 3×3 reference points in the case when function λ is optimized.

For the reconstructed function $\bar{\bar{F}}$, **Fig. 2.13** presents the optimal shape for the influence function λ related to the central patch $i = 2, j = 2$.

In **Tab. 2.2**, the different level of approximation, according to the different choices of λ, are reported. All the reconstruction $\bar{\bar{F}}$ are given by (2.31) by using 3×3 reference points. The error is defined in (2.32). The optimized shape of λ enables one to give an approximation $\bar{\bar{F}}$ of F showing the lowest level for error (2.32). Hence, this shape for λ is considered in the following.

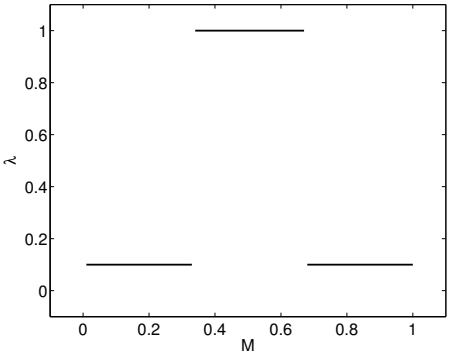


(a) $F = e^{-|(x-0.5)\ (\mu-1)|} + \sin(x\ \mu)$.

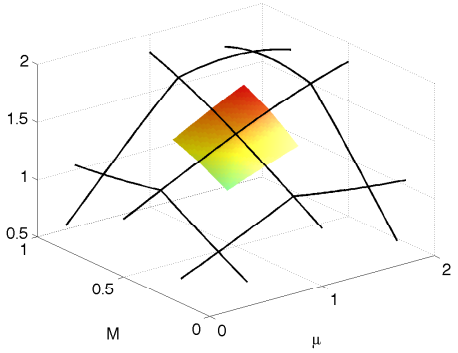


(b) $\bar{\bar{F}}$, *reconstructed by using* 3×3 *reference points.*

Figure 2.12: Initial function and its first approximation when λ is optimized.



(a) *optimized shape of the influence function* λ *for the central patch of the function* $\bar{\bar{F}}$.



(b) *central patch of the function* $\bar{\bar{F}}$, *reconstructed by using* 3×3 *reference points.*

Figure 2.13: Optimized shape of the influence function λ for the central patch of the function $\bar{\bar{F}}$.

Table 2.2: error e (see (2.32)) related to the different choices for λ.

influence function λ	error e
$\lambda = 1$ for $k = i$; $\lambda = 0.1$ for $\|k - i\| = 1$; $\lambda = 0$ otherwise	1.9%
$\lambda = 1$ for $k = i$; $\lambda = 0$ otherwise	2.2%
$\lambda = 1\ \forall k \in m_x$	3.2%

2.3.2 Approximation improvement

To quantify the quality of this first approximation given by generating one pair per each patch, let consider again the error defined in (2.32):

$$e = \frac{\|F - \bar{\bar{F}}\|_{\mathcal{D}\times\Omega}}{\|\bar{\bar{F}}\|_{\mathcal{D}\times\Omega}}, \quad \|F\|^2_{\mathcal{D}\times\Omega} = \int_{\mathcal{D}\times\Omega} F^2 d\Omega d\mu = \sum_{i=1}^{m_\mu}\sum_{j=1}^{m_x}\int_{\mathcal{D}_i\times\Omega_j} F^2 d\Omega d\mu. \quad (2.39)$$

This evaluation can be computed cheaply exploiting the fact that F is the product of two functions described in separated representation by PGD. If the PGD modes of these two functions are assumed to be sorted in descending order according to the L^2 norm of the parameter function, it is sufficient to take into account the first PGD modes to have a good prediction of the difference $(F-\bar{\bar{F}})$. Generally, for mechanical problems, mostly of the energetic information of the evolution of a time/parameter-space system is kept by the first PGD pairs. This is not a general result but an empiric observation. However, it is not straightforward to *a priori* know how many PGD functions are sufficient to get an accurate forecast of the function F.

In the following, three way to enrich the first approximation obtained by the RPM are analysed:

(a) first procedure consists in enriching the approximation by generating some new PGD pairs.

(b) Second way consists in enriching the first approximation by the reduced-basis technique.

(c) The third one resides in adding more reference points.

(a) First approximation enrichment by generation of PGD pairs. Approximation $\bar{\bar{F}}$ obtained by the RPM can be sufficient for classical mechanical problems. This approximation can be improved if needed according to three different procedures.

The first procedure consists in enriching the approximation by generating some new PGD pairs as follows:

1. Firstly, the prediction of F, denoted as $\tilde{F}$, is constructed. According to the case, this step is computed in different ways. In the case where F is a known function given by the product of two functions f and f' in separated variables form (see (2.4)), $\tilde{F}$ is the prediction given by considering only the first PGD pairs of f and f'. Let us construct $\tilde{F}$, a cheap forecast of F, by considering first 3 PGD pairs of

$$\tilde{F} = f_{k=3}(\mu, x)\; f'_{k'=3}(\mu, x) = \left(\sum_{i=1}^{3} \lambda_i(\mu)\; \Lambda_i(x)\right)\left(\sum_{i=1}^{3} \theta_i(\mu)\; \Theta_i(x)\right). \quad (2.40)$$

First are the most important modes since they are sorted in descending order according to the L^2 norm of the parameter function. Once obtained $\tilde{F}$, a residue of the first approximation can be constructed as follows:

$$\mathcal{R} \;=\; \tilde{F} \;-\; \bar{\bar{F}} \tag{2.41}$$

In the case where F is defined implicitly as solution of a algebraic problem, $\tilde{F}$ would be the residue between the right-hand side of the equation and the first approximation given by $\bar{\bar{F}}$. For instance, let us consider a fully discretized linear elliptic problem. It is governed by the discretized equation:

$$\mathbf{A}\,[F] \;=\; [B] \tag{2.42}$$

where $\mathbf{A}$ is a matrix, $[F]$ and $[B]$ are vectors of scalar values depending on two scalar variables, space and parameter. Let us suppose that, once F is expressed in a separated variables form, an approximation of F is given by RPM by choosing a number of reference points, by defining, then, its generalized components by the (2.7) and by reconstructing it by (2.31), obtaining $\bar{\bar{F}}$. In order to enrich this approximation, one can set:

$$\mathcal{R} \;=\; [B] \;-\; \mathbf{A}\,[\bar{\bar{F}}]. \tag{2.43}$$

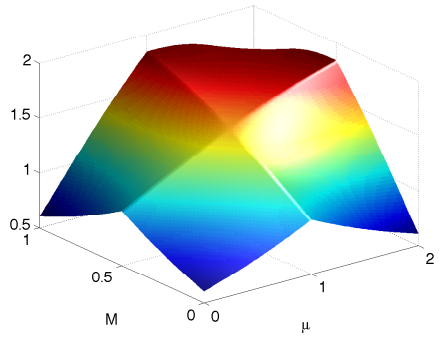
2. This residue, implicitly or explicitly expressed, enables one to generate some PGD pairs by the Algorithm 2 shown in chapter 1. Note that, in case that the field $\mathcal{R}$ represents an explicitly defined function, Algorithm 2 gives a separated variables representation of $\mathcal{R}$ (as already seen in (1.6)).

   Remark 5 *In chapter 1, the correlation between system* (1.6) *and an eigen-decomposition, such as SVD, is shown. The details of this analogy can be found in [Ladevèze, 1999, Nouy, 2010]. Hence, whenever the residual field $\mathcal{R}$ represents an explicitly defined function, the enrichment by PGD Algorithm 2 leads to an eigen-decomposition of $\mathcal{R}$.*

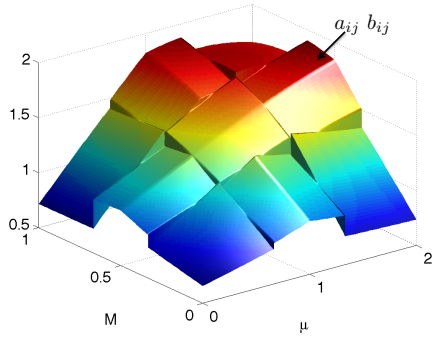
3. The generated PGD pairs are added to the first approximation $\bar{\bar{F}}$ enabling one to have a more accurate representation of F.

Let us consider, firstly, the simple function defined in (2.35). Let us take into account the separated variables description of this function, assuming that the first approximation $\bar{\bar{F}}$ is the one given by the RPM by using 3×3 reference points. The prediction of F, denoted as $\tilde{F}$, is constructed considering the first 3 PGD pairs of this function, as described in (2.40). From this prediction, a residue $\mathcal{R}$ has been constructed as defined in (2.41).

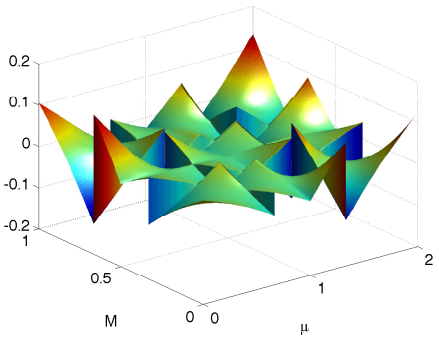
Fig 2.15 shows how the approximation of the function (2.35) evolves by adding some PGD pairs as correction to the first RPM approximation. It can be seen that, for the considered function (2.35), 3 PGD pairs are necessary to enrich the solution up to a quality corresponding to an error e lower than 0.1%. Let us remark that these PGD pairs are generated by the residue $\mathcal{R} \;=\; \tilde{F} - \bar{\bar{F}}$, where $\tilde{F}$ is the prediction of F considering only the first 3 PGD pairs.



(a) $\tilde{F}$, *prediction of* F, *constructed considering the first* 3 *PGD pairs of* F.



(b) $\bar{\bar{F}}$ *given by considering* 3×3 *reference points.*



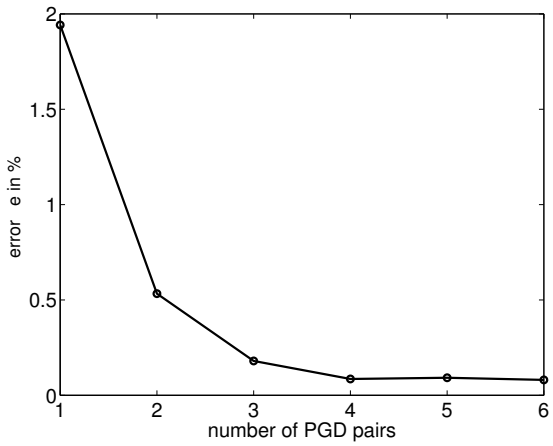
(c) *Residue of the first approximation with respect to a prediction of* F, $\mathcal{R} = \tilde{F} - \bar{\bar{F}}$.

Figure 2.14: First approximation given by RPM $\bar{\bar{F}}$ and its residue $\mathcal{R}$ with respect to a prediction of the function F.

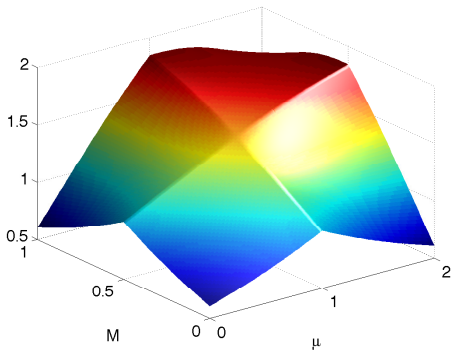
(b) First approximation enrichment by generation of reduced-basis modes. The first procedure to enrich the RPM approximation consisting in generating some new PGD pairs, can be revisited in order to enrich the first approximation by the reduced-basis technique:

1. This first step remains the same either by generating some PGD pairs or by generating a reduced-basis approximation. Hence, in this step, the prediction of F, denoted by $\tilde{F}$, is constructed in order to obtain $\mathcal{R} = \tilde{F} - \bar{\bar{F}}$, the residue of the first approximation given by the RPM.

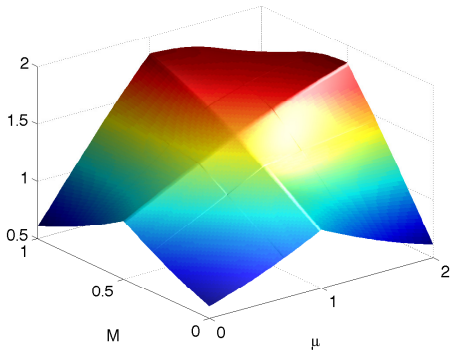
2. This residue, implicitly or explicitly expressed, is approximated by the reduced-



(a) error e (see (2.39)) by adding PGD pairs



(b) *Surface of function* (2.35).



(c) *first approximation of function* (2.35) *obtained by RPM corrected by few PGD pairs.*

Figure 2.15: Evolution of the error e (see (2.39)) by adding PGD pairs to the RPM approximation of function (2.35). 3 PGD pairs are necessary to attain an error e lower than 0.1%.

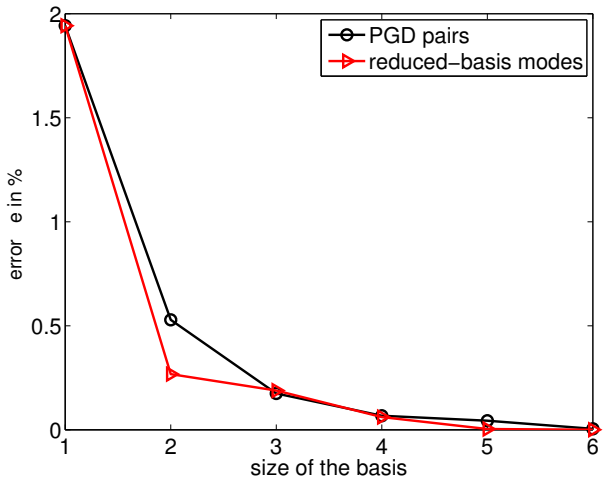


Figure 2.16: Comparison of the error e (see (2.39)) between the enrichment given by adding PGD pairs to the RPM approximation of function (2.35) and the enrichment given by using reduced-basis technique.

basis technique following the greedy algorithm:

$$\mu_1 = \arg \sup_{\mu \in \mathcal{D}} \|\mathcal{R}\|_{L^2}, \tag{2.44}$$

$$\mu_{i+1} = \arg \sup_{\mu \in \mathcal{D}} \|\mathcal{R} - P_i\, \mathcal{R}\|_{L^2}, \tag{2.45}$$

P_i is the orthogonal projection onto

$$\mathbf{\Phi}_i = \text{span}\, \{\mathcal{R}(x, \mu_1), ..., \mathcal{R}(x, \mu_i)\}. \tag{2.46}$$

3. The generated reduced-basis approximation of $\mathcal{R}$ is added to the first approximation $\bar{\bar{F}}$ enabling one to have a more accurate representation of F (or $\mathcal{R}$ in the implicit case).

For the considered example (2.35), **Fig 2.16** depicts the error e, defined in (2.39), comparing two different enrichments for the first RPM approximation. First one is given by adding some PGD pairs, obtained constructing the residue $\mathcal{R}$ and the second one is the enrichment obtained by considering the reduced-basis technique to approximate the residue $\mathcal{R}$. The level of the error is comparable for these two techniques.

Let us consider, now, a more complex function:

$$s(x, \mu) = (1 - x)\ \cos(3\pi\mu(x + 1))\ \exp(-(1 + x)\mu) \tag{2.47}$$

where $x \in \Omega = [0, 1]$ and $\mu \in \mathcal{D} = [0, 1]$ taken from an example in [Chaturentabut and Sorensen, 2010]. Surface $s(x, \mu)$ is shown in **Fig. 2.17**.

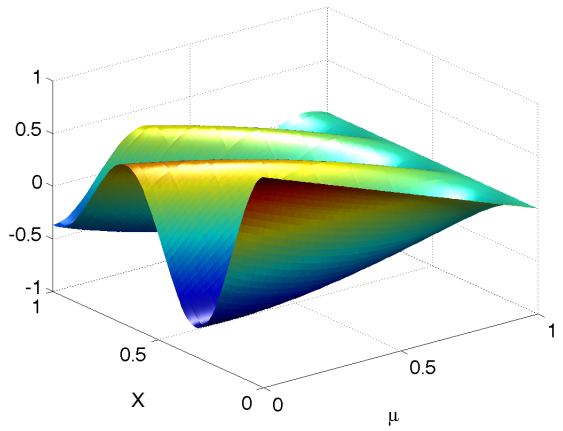


Figure 2.17: The function $s(\mu, x)$.

Let us assume that the first approximation $\bar{\bar{F}}$ is the one given by the RPM by using 3×3 reference points. **Fig 2.18** shows how the approximation of the function (2.47) evolves by adding some PGD pairs as correction to the first RPM approximation. It can be seen that, in this case, 4 PGD pairs are necessary to enrich the solution up to a quality corresponding to an error $e \approx 15\%$. Moreover, by adding one more PGD pair, error does not decrease any more. This highlights the lack of the prediction $\tilde{F}$. Indeed, by the first 4 PGD pairs the entire information, given by the residue $\mathcal{R} = \tilde{F} - \bar{\bar{F}}$, is added to the first approximation $\bar{\bar{F}}$.

Fig 2.18 depicts also how the approximation of the function (2.47) evolves by enriching the RPM approximation by the reduced-basis technique. Enriching the first approximation by considering the reduced-basis technique with a basis of 5 functions leads to the error given by adding the same number of modes PGD.

(c) First approximation enrichment by the adaptive selection of the reference points Previous paragraph has shown that it is not always possible to have a good prediction of the quantity of the necessary PGD pairs to improve the first approximation given by the RPM. If a few reference points are considered, the precision of the approximation can be unsuitable. Hence, the third procedure to enrich first approximation consists in adding more reference points. The enrichment by adding more reference points is motivated by the following consideration. The minimization of functional (2.12) represents, patch by patch, a weighted least-square regression. Considering the approximation generated on a single patch, some convergence properties detailed in [Bissantz et al., 2009, Liu et al., 2010], that can be summarized under the definition of optimality, can be extended to the RPM. Then, these properties can be applied to the entire set of patches. The RPM is optimal in the sense that a given error measure monotonically decreases as the number of reference points is hierarchically increased. Hence, given a first approximation and

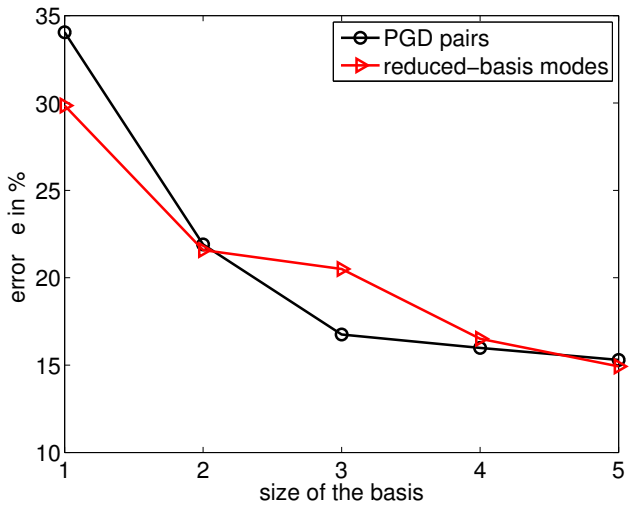


Figure 2.18: Comparison of the error e (see (2.39)) between the enrichment given by adding PGD pairs to the RPM approximation of function (2.47) and the enrichment given by using reduced-basis technique.

by adding further reference points among of the already existing patches, the norm of a chosen error decreases. Let us consider again the function (2.47) and the error defined by (2.39). **Fig 2.19** depicts how the error evolves by adding more reference points. Investigating the link between functional (2.12) and least-square regression may help to address this issue. Further studies should be done in this direction in the future.

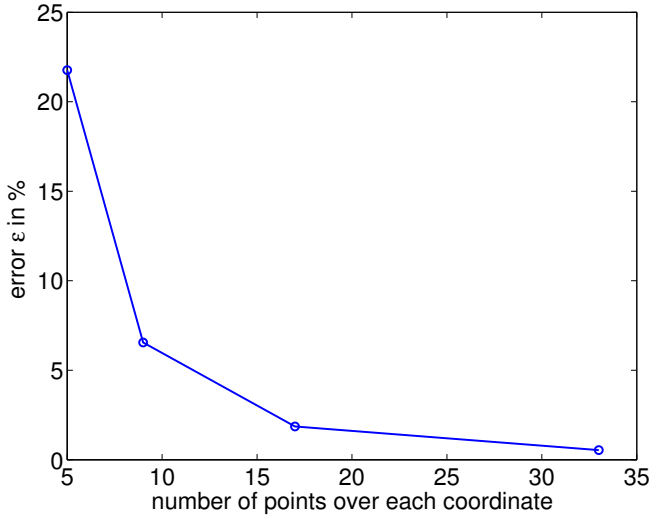


Figure 2.19: The error evolution over the number of reference points, m_x and m_μ.

2.4 Conclusions

Managing fields with different representations hinders the potential gain of the PGD in terms of calculation. Indeed, operations to pass from a full representation to the separated-variables one are, often, necessary but costly.

In this chapter a new technique, called Reference Points Method (RPM), has been developed. The RPM enables one to give a first approximation of all fields in separated-variables format. This approximation can be enriched if necessary by three procedures. One procedure consists in adding more reference points. In this case, it was shown that the error monotonically decreases as the number of reference points - hierarchically introduced - increases. However, definition of patches changes when reference points are added, which may be cumbersome. Another procedure does not need to modify the number of reference points nor the definition of patched. It consists in enriching the first approximation by adding some PGD pairs or by reduced-basis technique. This solution is very convenient since only few pairs are necessary to attain a good quality of approximation.

The development of the RPM aims at reducing the complexity related to the standard algebraic operation (products, additions, integration, ..) for the fields defined under the separated-variables form. In the separated-variables framework these operations lead to artificially increase the number of product terms to represent the resulting quantities. RPM enables one to avoid this increasing number of product terms by generating a patch-by-patch rank one PGD. In the next chapter the RPM is introduced in the LaTIn-PGD computational strategy, to solve a parameter nonlinear-dependent elliptic problem. More precisely, RPM is introduced at the preliminary step. This latter will be reformulated in the RPM framework, showing how the number of necessary operations to evaluate the search direction and assemble the ROM can be drastically reduced.

Chapter 3

A non-incremental nonlinear solver: the LaTIn-PGD with RPM approximation

This chapter is dedicated to the LaTIn framework for which RPM has been developed. In the first part, the LaTIn-PGD method is presented and described for a parametric elliptic nonlinear problem. In this description the Preliminary step of the LaTIn-PGD algorithm is detailed. This part of the algorithm is important to generate only the most relevant PGD modes, enabling one to decrease the CPU time and memory consuming. However, it represents a big amount of the remaining CPU time, as it consists in building a new reduced-order model by in integrating and projecting the updated nonlinear terms onto an evolving reduced basis. As the nonlinear term and the reduced-order basis change throughout the iterations, this model cannot be precomputed prior to the algorithm. This issue is one of the main motivations behind the development of the RPM. Hence, for the considered nonlinear problem, it is described how the Preliminary step is approximated by the RPM.

In the second part of the chapter the same problem is solved by another reduced order model computational strategy: Newton method is combined with reduced basis approximation. In this approach, the nonlinear terms are interpolating by the EIM. This part aims at highlighting similarities and differences between the RPM

and the EIM.

The reference problem considered in this section is a two-dimensional elliptic and nonlinear parametrized problem. It has been introduced as a benchmark in [Grepl et al., 2007] for numerical tests with the EIM and then considered again in [Chaturentabut and Sorensen, 2010] in the framework of the DEIM. It involves an elliptic PDE containing a linear term and a nonlinear term depending on two parameters. This problem can be viewed as a two-dimensional thermal diffusion evolution over two parameters with a nonlinear source term. The study proposed hereafter could be easily extended to general parabolic PDEs.

Given a regular domain $\Omega =]0,1[^2$ included in $\mathbb{R}^2$ and introducing a parameter domain $\mathcal{D} = [0.01, 10]^2$, let us consider the two-coordinates domain $\Omega \times \mathcal{D}$. The Hilbert space of functions, whose squared value is integrable on Ω, is denoted by $L^2(\Omega)$. The Sobolev space of functions and first derivatives, whose the square is integrable on Ω, is defined as follows:

$$H^1(\Omega) \equiv \{u \in L^2(\Omega) : \ \underline{\boldsymbol{\nabla}} u \in L^2(\Omega)\}.$$

Let us introduce $H^1_0(\Omega)$ designating the subspace of functions vanishing on $\partial_u\Omega$:

$$H^1_0(\Omega) \equiv \{u \in H^1(\Omega) : \ u = 0 \text{ on } \partial_u\Omega\}.$$

Let us define Problem 10:

Problem 10 (strong form) *Find $u(\underline{x}, \underline{\mu})$ with $\underline{x} = (x_1, x_2) \in \Omega =]0,1[^2$ and $\underline{\mu} = (\mu_1, \mu_2) \in \mathcal{D} = [0.01, 10]^2$, such that:*

$$-\nabla^2 u \ + \ \frac{\mu_1}{\mu_2} \ (e^{\mu_2 u} - 1) = 100 \ \sin(2\pi x_1) \ \sin(2\pi x_2) \tag{3.1}$$

with homogeneous Dirichlet boundary condition on $\partial\Omega$.

Let us recall $\mathcal{U}$, introduced in chapter 1 extended to Problem 10. For simplicity $\mathcal{U} = H^1_0(\Omega)$. Let us remember spaces $\mathcal{P} = L^2(\mathcal{D}, \mathbb{R})$ and $\mathcal{V} = L^2(\mathcal{D}, \mathcal{U})$. Problem 10 can be stated under the weak form:

Problem 11 (weak form) *Given $\underline{\mu} \in \mathcal{D}$, find $u(\underline{x}, \underline{\mu}) \in \mathcal{V}$ such that:*

$$\forall v \in \mathcal{V}, \quad a(u, v) + \int_\Omega \ g(u; \underline{\mu}) \ v \ \mathrm{d}\Omega = \int_\Omega \ f(\underline{x}) \ v \ \mathrm{d}\Omega \tag{3.2}$$

where:

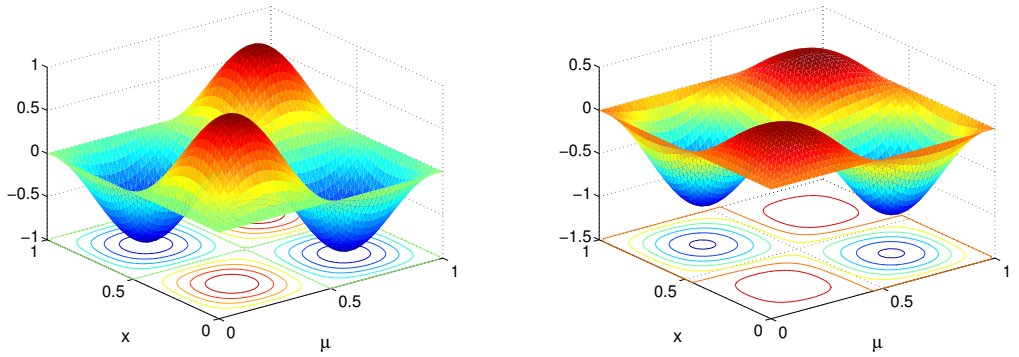
$$a(u,v) = \int_\Omega \underline{\nabla} u \cdot \underline{\nabla} v \, \mathrm{d}\Omega \tag{3.3}$$

$$g(u;\underline{\mu}) = \frac{\mu_1}{\mu_2} \left(e^{\mu_2 u} - 1\right) \tag{3.4}$$

$$f(\underline{x}) = 100 \ \sin(2\pi x_1) \sin(2\pi x_2) \tag{3.5}$$

Here $g(u;\underline{\mu})$ is a nonaffine nonlinear function of the parameter $\underline{\mu}$ and field variable $u(\underline{x},\underline{\mu})$. Bilinear term $a(u,v) = \int_\Omega \nabla u \cdot \nabla v \, \mathrm{d}\Omega$ and linear term $\int_\Omega f(\underline{x}) \, v \, \mathrm{d}\Omega$ are $\mathcal{V}$-continuous bounded functionals and they are parameter-independent. It is shown in [Grepl et al., 2007] that Problem (11) is well-posed and that it admits a unique solution $u \in \mathcal{V}$.

In this chapter, the reference solution is the one obtained with a piecewise-linear finite element for approximation space $\mathcal{V}_h \subset \mathcal{V}$. The solution u for two sets of parameters ($\underline{\mu} = (0.01, 0.01)$ and $\underline{\mu} = (10, 10)$) is given in **Fig.3**. In this case, a mesh with 50×50 bilinear quadrilateral elements was used ($N = 2601$). Note that the parameter μ_1 controls the strength of the source term whereas μ_2 changes the strength of the non-linearity.



(a) $\underline{\mu} = (0.01; 0.01)$ (b) $\underline{\mu} = (10; 10)$

Figure 3.1: FE numerical solution of Problem (11) for the extreme values of the parameter.

3.1 The LaTIn method

For Problem 11, the LaTIn-PGD computational strategy is hereafter used over the parameter-space domain.

A two-stages iterative scheme is proposed to obtain the solution of Problem 11 by introducing two search directions. The key point is to introduce a new variable,

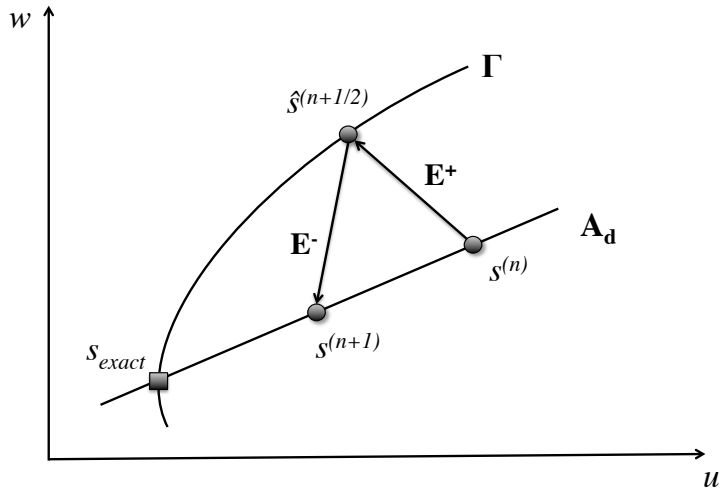


Figure 3.2: The LaTIn alternative scheme. The two manifolds are coupled by the search direction $\mathbf{E}^+$ and $\mathbf{E}^-$

$w \in W = \{\, v \in H^1(\Omega) \,\}$ in order to satisfy the nonlinear local equation $w = g(u;\underline{\mu})$ associated with manifold $\mathbf{\Gamma}$ at the *local stage*. This new variable is, then, coupled during the *global stage* with the equilibrium equation, associated with space $\mathbf{A}_d$, thanks to a search direction. The problem solution is found by solving alternatively the *local stage* and the *global stage*. This alternative scheme is represented in **Fig.3.2**.

Local stage at iteration $n+1$ Local stage consists in building solution $\hat{s}^{(n+1/2)} = (\hat{u}^{(n+1/2)}, \hat{w}^{(n+1/2)})$ in $\mathbf{\Gamma}$, knowing solution $s^{(n)} = (u^{(n)}, w^{(n)})$ coming from the previous global stage thanks to search direction $\mathbf{E}^+$, verified by $\hat{s}^{(n+1/2)} - s^{(n)}$:

$$\mathbf{E}^+ \;:\; (\hat{w}^{(n+1/2)} - w^{(n)}) + \mathbf{H}^+(\hat{u}^{(n+1/2)} - u^{(n)}) \;=\; 0, \tag{3.6}$$

$$\hat{w}^{(n+1/2)} \;=\; g(\hat{u}^{(n+1/2)};\underline{\mu}) \tag{3.7}$$

where $\mathbf{H}^+$ is a symmetric definite operator which is a parameter of the method. At this stage, the problem is nonlinear but local in space variable. A simple choice is to take a "stiff" ascending search direction ($\mathbf{H}^+ \to \infty$), which is equivalent to setting $\hat{u}^{(n+1/2)} = u^{(n)}$. In this case, solution is found explicitly without resorting to a local nonlinear solver. Solution $\hat{s}^{(n+1/2)} = (\hat{u}^{(n+1/2)}, \hat{w}^{(n+1/2)})$ is defined by:

$$\forall\, \underline{\mu} \in \mathcal{D},\, \forall\, \underline{x} \in \Omega, \quad \left\{ \begin{array}{rcl} \hat{u}^{(n+1/2)} & = & u^{(n)} \\ \hat{w}^{(n+1/2)} & = & g(\hat{u}^{(n+1/2)};\underline{\mu}) = g(u^n;\underline{\mu}) \end{array} \right. \tag{3.8}$$

Global stage at iteration $n+1$ Global stage consists in building solution $s^{(n+1)} = (u^{(n+1)}, w^{(n+1)})$ in $\mathbf{A}_d$, knowing solution $\hat{s}^{(n+1/2)} = (\hat{u}^{(n+1/2)}, \hat{w}^{(n+1/2)})$ coming from the previous local stage thanks to search direction $\mathbf{E}^-$ verified by $s^{(n+1)} - \hat{s}^{(n+1/2)}$:

$$\mathbf{E}^- : (w^{(n+1)} - \hat{w}^{(n+1/2)}) - \mathbf{H}^-(u^{(n+1)} - \hat{u}^{(n+1/2)}) \;=\; 0 \tag{3.9}$$

$\mathbf{H}^-$ is the descending search direction and a parameter of the method, as the previous one, $\mathbf{H}^+$. In [Ladevèze, 1999], it is shown that a well-suited choice is to use the tangent operator :

$$\mathbf{H}^- = \left.\frac{\partial g(u;\underline{\mu})}{\partial u}\right|_{u=\hat{u}^{(n+1/2)}} . \tag{3.10}$$

Global stage problem reads as follows:

Problem 12 (global stage) *Given* $\hat{s}^{(n+1/2)} = (\hat{u}^{(n+1/2)}, \hat{w}^{(n+1/2)})$, *find* $s^{(n+1)} = (u^{(n+1)}, w^{(n+1)}) \in \mathcal{V} \times W$ *in* $\mathbf{A}_d$ *such that:*

$$\forall v \in \mathcal{V}, \quad \int_{\mathcal{D}} a(u^{(n+1)}, v)\,\mathrm{d}\mu + \int_{\mathcal{D}\times\Omega} w^{(n+1)}\ v\ \mathrm{d}\Omega\,\mathrm{d}\mu = \int_{\mathcal{D}\times\Omega} f(\underline{x})\ v\ \mathrm{d}\Omega\,\mathrm{d}\mu \tag{3.11}$$

with: $\forall\, \underline{x} \in \Omega, \quad (w^{(n+1)} - \hat{w}^{(n+1/2)}) - \mathbf{H}^-(u^{(n+1)} - \hat{u}^{(n+1/2)}) \ = \ 0$

By introducing search direction (3.9) in (3.11), one obtains:

$$\begin{aligned}\forall v \in \mathcal{V}, \quad & \int_{\mathcal{D}} a(u^{(n+1)}, v)\,\mathrm{d}\mu + \int_{\mathcal{D}\times\Omega} \mathbf{H}^-\ u^{(n+1)}\ v\ \mathrm{d}\Omega\,\mathrm{d}\mu = \dots \\ & \dots \int_{\mathcal{D}\times\Omega} f(\underline{x})\ v\ \mathrm{d}\Omega\,\mathrm{d}\mu - \int_{\mathcal{D}\times\Omega} \left(\hat{w}^{(n+1/2)} - \mathbf{H}^-\hat{u}^{(n+1/2)}\right)\ v\ \mathrm{d}\Omega\,\mathrm{d}\mu\end{aligned} \tag{3.12}$$

The only unknown in this latter equation is the unknown $u^{(n+1)}$. This equation is global over the parameter-space domain but linear. Due to the specific choice done for search directions $\mathbf{E}^-$ (see (3.8)) and $\mathbf{E}^+$ (see (3.10)) and by introducing the correction $\delta u^{(n+1)} = u^{(n+1)} - u^{(n)}$ between two consecutive global stages, it yields:

Problem 13 (global stage) *Find* $u^{(n+1)} = \delta u^{(n+1)} + u^{(n)} \in \mathcal{V}$ *and* $w^{(n+1)} \in W$ *such that:*

$$\forall v \in \mathcal{V}, \quad \int_{\mathcal{D}} a(\delta u^{(n+1)}, v)\,\mathrm{d}\mu \int_{\mathcal{D}\times\Omega} g'(u^{(n)};\underline{\mu})\ \delta u^{(n+1)}\ v\ \mathrm{d}\Omega\,\mathrm{d}\mu = \int_{\mathcal{D}} \mathcal{R}(u^{(n)}, v;\underline{\mu})\ \mathrm{d}\mu \tag{3.13}$$

where

$$\mathcal{R}(u^{(n)}, v;\underline{\mu}) = a(u^{(n)}, v) \ - \ \int_{\Omega} f(\underline{x})\ v\ \mathrm{d}\Omega \ + \ \int_{\Omega} g(u^{(n)};\underline{\mu})\ v\ \mathrm{d}\Omega,$$

$$g'(u^{(n)};\underline{\mu}) = \left.\frac{\partial g(u;\underline{\mu})}{\partial u}\right|_{u=\hat{u}^{(n+1/2)}=u^{(n)}} .$$

The third principle of the LaTIn method is to exploit the framework supplied by the global stage to introduce a well-suited representation of the unknowns, in order to reduce the number of operations and the cost related to the memory. Hence, the solution $u^{(n+1)}$ of the *global stage* (Problem 13) is sought in $\tilde{\mathcal{V}} = \mathcal{P} \otimes \mathcal{U}$ thanks to

a parameter-space PGD. One introduces the following approximation of order k of the correction $\delta u^{(n+1)}$:

$$\delta u^{(n+1)}(\underline{x},\underline{\mu}) = \sum_{i=1}^{k} \Phi_i(\underline{x})\ \lambda_i(\underline{\mu}) \tag{3.14}$$

Each PGD pair $(\lambda_i, \Phi_i) \in \mathcal{P} \times \mathcal{V}$ is unknown and determined throughout the computation by a *greedy algorithm* (see algorithm (2) in chapter 1).

3.2 The PGD approximation with Preliminary step

The construction of a new space function Φ_i is by far the most expensive step of this process. Thus, at a given iteration $n+1$ of the nonlinear solver, it is advantageous to first reuse the reduced basis $\mathbf{W}_k$ generated up to iteration n by updating the parameter functions $\{\lambda_i\}_{1\leq i\leq k}$ [Ladevèze et al., 2010]. One proceeds with the *global stage* at the iteration $n+1$ of the nonlinear solver as follows:

1. *Preliminary step*: reuse of the reduced basis. This step consists in building an approximation of the solution, denoted $\breve{s}^{(n+1)}$, thanks to the ROB generated at the previous iteration n of the nonlinear iterative scheme. Here, the only unknowns are the functions $\{\lambda_i\}_{1\leq i\leq k}$ depending on the parameters. Given a ROB of space functions $\mathbf{W}_k = \{\Phi_i\}_{1\leq i\leq k}$, one seeks the best linear combination of this ROB which solves Problem 13.

   The *Preliminary step* problem reads:

   $$1 \leq j \leq k,$$
   $$\sum_{i=1}^{k} \int_{\mathcal{D}} \lambda_i^\star \left(a(\Phi_i,\Phi_j)\mathrm{d}\mu \ + \int_{\Omega\times\mathcal{D}} \Phi_i\ g'(u^{(n)};\underline{\mu})\Phi_j \right) \lambda_j\ \mathrm{d}\Omega\ \mathrm{d}\mu = \ldots$$
   $$\ldots - \int_{\mathcal{D}} \lambda_i^\star\ \mathcal{R}(u^{(n)},\Phi_j;\underline{\mu})\ \mathrm{d}\mu, \tag{3.15}$$

   where:

   $$\mathcal{R}(u^{(n)},\Phi_j;\underline{\mu}) = a(u^{(n)},\Phi_j)\,\mathrm{d}\mu - \int_\Omega f(\underline{x})\ \Phi_j\ \mathrm{d}\Omega + \int_\Omega g(u^{(n)};\underline{\mu})\ \Phi_j\ \mathrm{d}\Omega \tag{3.16}$$

   By taking into consideration the underlying finite element approximation ($[\Phi_i] = \sum_{j=1}^{N}[\Phi_i^j]\varphi_j(\underline{x})$) and ($[\lambda_i] = \sum_{j=1}^{p}[\lambda_i^j]\xi_j(\mu)$), this leads to a $k\times k$ linear algebraic system:

   $$\int_{\mathcal{D}} \{[\lambda^\star]\}_k\ \xi_k\ \mathbb{W}_k^T\left[\mathbb{A} + \mathbb{G}'(u^{(n)};\underline{\mu})\right]\mathbb{W}_k\ \{[\lambda]\}_k\ \xi_k\ \ \mathrm{d}\mu\ = \ldots$$
   $$-\ldots\int_{\mathcal{D}} \{[\lambda^\star]\}_k\ \xi_k\ \mathbb{W}_k^T[R(u^{(n)};\underline{\mu})]\ \mathrm{d}\mu \tag{3.17}$$

$\mathbb{A}$ and $\mathbb{G}'$ are $N \times N$ matrices such that:

$$\mathbb{A}_{ij} = a(\varphi_i, \varphi_j), \tag{3.18}$$

$$\mathbb{G}'_{ij} = \int_\Omega \varphi_i \; g'\left(u^{(n)}; \underline{\mu}\right) \varphi_j \; \mathrm{d}\Omega, \tag{3.19}$$

and right-hand side $[R(u^{(n)}; \underline{\mu})]$ is a N-length vector with:

$$[R]_i = -\mathcal{R}(u^{(n)}, \varphi_i; \underline{\mu}). \tag{3.20}$$

The discretized reduced basis is a $N \times k$ matrice with $\mathbb{W}_k = [[\Phi]_1, [\Phi]_2, \ldots, [\Phi]_k]$. The vector of unknowns $\{\lambda\}_k$ has a size of k.

This system is solved for each parameter value $\underline{\mu} \in \mathcal{D}$. Thus, considering for the parameter domain a discretisation of p parameter values $\underline{\mu}_i$, this leads to a computational complexity in the order of $\mathcal{O}(p\ N\ k^2\ + p\ N\ k + \ p\ k^3)$ as for model reduction techniques building a ROB during a learning stage (see section 3.4).

2. *Preliminary step performance indicator*: The Preliminary step produces a first approximation of Problem 13 at the iteration $n+1$ by seeking the solution u in the span of the already existing ROB, generated at the previous iteration n. An error indicator, based on the error indicator developed for the LaTIn method in [Ladevèze, 1999], is then computed to quantify the accuracy of this first prediction. The performance indicator of the preliminary stage compares the distance between the two spaces $\mathbf{A}_d$ and $\mathbf{\Gamma}$ for two consecutive iterations (see [Heyberger et al., 2011] for more details):

$$\eta_0 = \frac{e_1 - e_2}{e_1} \tag{3.21}$$

$$e_1 = \frac{\|u^{(n)} - \hat{u}^{(n-1/2)}\|}{1/2\ \|u^{(n)} + \hat{u}^{(n-1/2)}\|} \qquad e_2 = \frac{\|\breve{u}^{(n+1)} - \hat{u}^{(n+1/2)}\|}{1/2\ \|\breve{u}^{(n+1)} + \hat{u}^{(n+1/2)}\|} \tag{3.22}$$

or, due to (3.8):

$$e_1 = \frac{\|u^{(n)} - \hat{u}^{(n-1)}\|}{1/2\ \|u^{(n)} + \hat{u}^{(n-1)}\|} \qquad e_2 = \frac{\|\breve{u}^{(n+1)} - \hat{u}^{(n)}\|}{1/2\ \|\breve{u}^{(n+1)} + \hat{u}^{(n)}\|} \tag{3.23}$$

If its value is higher than a critical threshold, then the global / linear stage at Iteration $n+1$ is considered to be solved. One can proceed to next iteration. Otherwise, one proceeds to the generation of a new PGD pair.

3. *Generation of a new PGD pair*: The prediction previously computed is considered to be known and the performance indicator (3.21) is lower than the given threshold. A new PGD pair is sought to enrich the previous approximation. It is generated by solving Problem 13 by the Galerkin-PGD procedure, described in (1.1.4). Let us remember that the new PGD pair is generated to approximate the correction $\delta u^{(n+1)} = u^{(n+1)} - u^{(n)}$ of the LaTIn iteration $n+1$.

   The generation of the new PGD pair (λ, Φ) leads to the definition of the following problems:

Problem 14 (Generation of the spatial function) *Knowing* λ *from* (3.25)*, find* Φ *such that:*

$$\left(\int_{\mathcal{D}} [\lambda]^T \; \xi^T \; \left[\mathbb{A} + \mathbb{G}'(u^{(n)}, \underline{\mu}) \right] \; \xi \; [\lambda] \right) \; [\Phi] \; \mathrm{d}\mu = \int_{\mathcal{D}} \lambda \; [R(u^{(n)}; \underline{\mu})] \; \mathrm{d}\mu \tag{3.24}$$

Problem 15 (Generation of the parametric function) *Knowing* Φ *from* (3.24)*, find* λ *such that:*

$$\int_{\mathcal{D}} \xi^\star \; [\Phi]^T \left[\mathbb{A} + \mathbb{G}'(u^{(n)}, \underline{\mu}) \right] \; [\Phi] \; \xi \; [\lambda] \; \mathrm{d}\mu = \int_{\mathcal{D}} [\Phi]^T [R(u^{(n+1)}; \underline{\mu})] \mathrm{d}\mu. \tag{3.25}$$

Equation (3.24) is a spatial problem that involves the integration on the parameter domain in order to set up the operators. It is an application $\mathbf{S} : \mathcal{P} \to \mathcal{U}$ which maps a parametric function $\lambda \in L^2(\mathcal{D})$ into a space function $\Phi = \mathbf{S}(\lambda) \in \mathcal{U}$. The second equation (3.25) involves space integrals. It defines an application $\mathbf{P} : \; \mathcal{U} \to \mathcal{P}$. An algorithm that alternatively and iteratively generate parameter function λ and space function Φ is used [Nouy, 2010, Ladevèze, 1999] (see Algorithm 3).

- Initialization $\lambda^0(\mu)$. Initializing Φ^0 leads to the same result.
for $l = 1, ..., l_{max}$ **do**
  - knowing $\lambda^{(l-1)}(\mu)$, find $\Phi^{(l)}(\underline{x})$ solution of Problem 3.24;
  - knowing $\Phi^{(l)}(\underline{x})$, find $\lambda^{(l)}(\mu)$ solution of Problem 3.25;
  - normalization of $\Phi^{(l)}(\underline{x})$.
end

Algorithm 3: Algorithm to generate a new PGD pairs.

Once this new pair of parameter and space functions is calculated, the $(k+1)th$ space function is orthogonalized and added to the reduced-basis to form $\mathbf{W}_{k+1}$.

Convergence test of the nonlinear iterative solver In order to check the convergence of the iterative scheme, one constructs the error defined in (3.21):

$$\frac{\|u^{(n+1)} - u^{(n)}\|}{\frac{1}{2}\|u^{(n+1)} + u^{(n)}\|} = \frac{\|\delta u^{(n+1)}\|}{\frac{1}{2}\|u^{(n+1)} + u^{(n)}\|} \tag{3.26}$$

Under the hypothesis of monotonous operator $g(u;\underline{\mu})$, the algorithm above converges toward the reference solution $\mathbf{s}_{ref}$ (see [Ladevèze, 1999] for proof of the convergence), moreover the convergence is ensured for any choice of $\mathbf{H}^-$. Finally, solution of the *global stage* at the iteration $n+1$ of the LaTIn method is obtained by:

- Preliminary step: given a ROB of space functions $\mathbf{W}_k$, updates the time functions $\{\lambda_i^{(n+1)}\}_{1\leq i\leq k}$ by solving (3.15).
- Check preliminary step performance indicator η_0, defined in (3.21).
- **if** $\eta_0 >$ *threshold* **then**
  | *global stage* at iteration $n+1$ is considered to be solved.
else
  | generation of a new PGD pair summarized in Algorithm 3 ;
end
- LaTIn convergence indicator defined in (3.26).

Algorithm 4: *global stage* at iteration $n+1$.

Remark 6 *The search directions are parameters of the method. In [Ladevèze, 1999], it is shown that they do not affect the solution but only the convergence rate.* ***Fig. 3.3*** *depicts the convergence curves for solution of Problem 10. First curve is the convergence toward the solution using as search direction operator the tangent operator:*

$$\mathbf{H}^- = \left.\frac{\partial g(u;\underline{\mu})}{\partial u}\right|_{u=\hat{u}^{(n+1/2)}}. \tag{3.27}$$

The second curve is the convergence toward the solution using as search direction operator a constant operator. This is the tangent operator computed at first iteration of the LaTIn method:

$$\mathbf{H}^- = \mathbf{H}_\mathbf{0}^- = \left.\frac{\partial g(u;\mu)}{\partial u}\right|_{u=u^{(0)}} \tag{3.28}$$

In this case, the algorithm bears some similarities to a quasi-Newton scheme. These two different choices lead to the same solution. however, choice (3.27) *drastically improves the convergence rate.*

Remark 7 *The new PGD pairs are generated by the correction of the solution at the iteration* $n+1$, $\delta u^{(n+1)} = u^{(n+1)} - u^{(n)}$. *It means that the correction* δu *converges to zero as the number of PGD pairs increases. The above conclusion motivates the definition of the threshold defined in* (3.21). *This threshold checks the necessity of a*

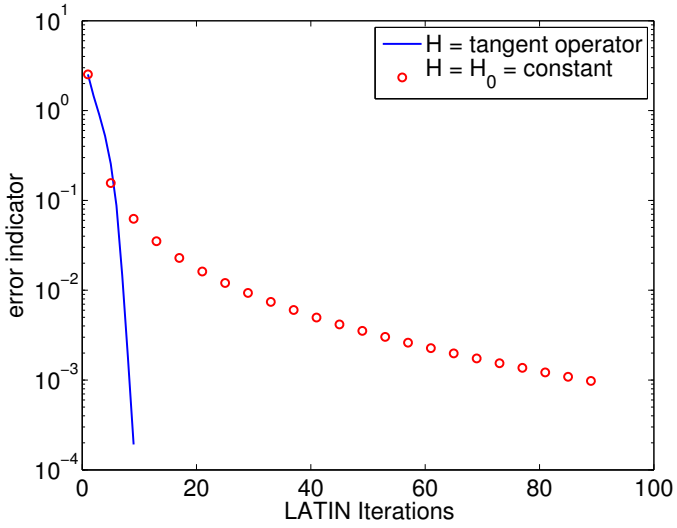


Figure 3.3: Comparison between the choice of the tangent operator and a constant operator.

new PGD pair by evaluating the quantity $\|\delta u^{(n+1)}\| = \|u^{(n+1)} - u^{(n)}\|$, *where* $u^{(n+1)}$ *is the solution of the Preliminary step. It implies that the most important PGD pairs are generated at the first iterations.*

Remark 8 ***Fig.3.4*** *shows the comparison between the convergences of the LaTIn-PGD method with the Preliminary step and the classic LaTIn-PGD without the Preliminary step. This latter consists in generating a new PGD pair for each new LaTIn iteration. The Preliminary step enables a reduction of the number of generated PGD pairs. Indeed, in order to reach an error* (3.26) *of* 10^{-4}, *the order of the ROB by using the LaTIn-PGD with Preliminary step, is two times lower than the ROB related to the classic LaTIn-PGD. Hence, this corresponds to a gain in terms of CPU time.*

3.3 Preliminary step approximated by Reference Points Method

In order to describe this procedure, let us consider equation (3.15). Let us denote as:

$$\alpha_{ij} = \int_{\Omega\times\mathcal{D}} \Phi_i(\underline{x})\; g'\left(u^{(n)};\underline{\mu}\right)\; \Phi_j(\underline{x})\; \mathrm{d}\Omega\; \mathrm{d}\mu. \tag{3.29}$$

The term $g'\left(u^{(n)};\underline{\mu}\right)$ is denoted as $G(\underline{x},\underline{\mu})$ to alleviate the notations. It yields:

$$\alpha_{ij} = \int_{\Omega\times\mathcal{D}} \Phi_i(\underline{x})\; G(\underline{x},\underline{\mu})\; \Phi_j(\underline{x})\; \mathrm{d}\Omega\; \mathrm{d}\mu. \tag{3.30}$$

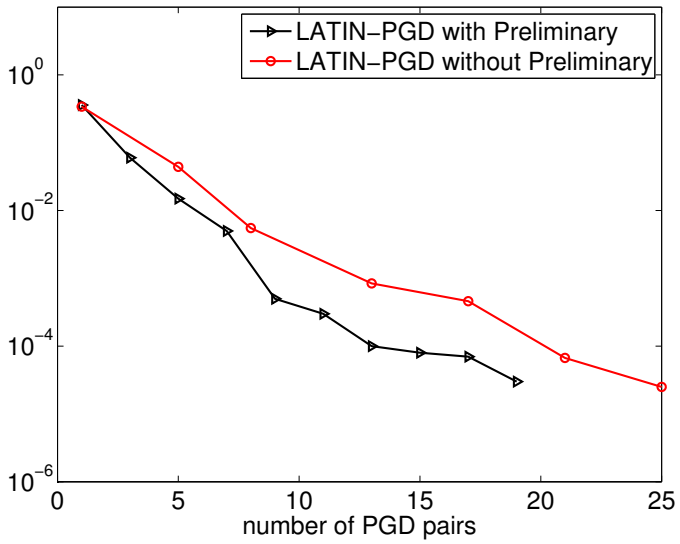


Figure 3.4: Comparison between the LaTIn-PGD method with and without the Preliminary step.

Let us define α considering the contributions of the k modes composing the reduced-order basis $\mathbf{W}_k = \{\Phi_1, \Phi_2, \ldots, \Phi_k\}$. Considering the classic discretizations, already seen before for the underlying finite element approximation ($[\Phi_i(\underline{x})] = \sum_{j=1}^{N} [\Phi_i^j] \varphi_j(\underline{x})$), α is defined as follows:

$$\alpha = \int_{\mathcal{D}} \mathbb{W}_k^T \mathbb{G}(\underline{x}, \underline{\mu}) \mathbb{W}_k \, \mathrm{d}\mu. \tag{3.31}$$

and the residue:

$$\int_{\mathcal{D}} \mathbb{W}_k^T [R(\tilde{u}^{(n)}; \underline{\mu})] \, \mathrm{d}\mu \tag{3.32}$$

where $\mathbb{G}$ is a $N \times N$ matrix such that $\mathbb{G}_{ij} = \int_\Omega \varphi_i \, G(\underline{x}, \underline{\mu}) \, \varphi_j \, \mathrm{d}\Omega$. Residue $[R(\tilde{u}^{(n)}; \underline{\mu})]$ is a N-length vector with $[R]_i = -\mathcal{R}(u^{(n)}, \varphi_i; \underline{\mu})$.

The Preliminary step, detailed in the previous section, involves, at each iteration of the LaTIn-PGD method, the construction of a new reduced-order model in order to seek the best linear combination of the already existing ROB which solves Problem 13. For nonlinear problems, the construction of a reduced-order model is performed after the evaluation of the new search direction operator (3.10), whenever it is necessary. Hence, this latter has to be integrated over the whole domain of the definition of the problem. In order to quantify the CPU cost of this operation, computational complexity is evaluated in terms of FLOPS, following [Chaturentabut and Sorensen, 2010]. One denotes by β the number of FLOPS to evaluate the search direction operator at a single integration point. Let us remember that p is the number of parameter values in the discretized parameter domain and N the number of degrees of freedom of the underlying space discretization. If the number of integration points of our model is assumed to be equal to N for the sake of simplicity, the

number of FLOPS involved to integrate the search direction operator is of the order of $\mathcal{O}(p\ N\ \beta)$.

For each new parameter value, one has (i) to evaluate Jacobian and residue, (ii) to project onto the discretized ROB $\mathbb{W}_k$. Projection of Jacobian (resp. residue) onto ROB has a computational complexity that depends on $\mathcal{O}(N\ k^2)$ since $\mathbb{G}$ is sparse (resp. $\mathcal{O}(N\ k)$) [Grepl et al., 2007, Chaturentabut and Sorensen, 2010]. The total computational complexity is consequently in the order of $\mathcal{O}(N\ k^2 + N\ k)$ and is N-dependent (see **Tab.3.1**).

Table 3.1: Computational complexity of Jacobian and residue projections onto ROB

Operation type	Complexity
Integration point evaluation	$\mathcal{O}(pN\ \beta)$
Projection of Jacobian onto ROB	$\mathcal{O}(pN\ k^2)$
Projection of residue onto ROB	$\mathcal{O}(pN\ k)$

Both the integration and the Galerkin projection of the operators involve a number of operations that scales with the dimension N of the underlying finite element approximation space and the dimension p of the underlying discretisation of the parameter domain.

At the preliminary step of LaTIn-PGD computational strategy, operations related to the construction of the reduced-order basis (3.31) and residue (3.32) are the same than for the (1.71) related to the reduced-basis technique. The difference concerns the choice of the ROB. For the reduced-basis technique, the ROB results from a learning stage and during the resolution of the problem (the online stage) does not evolve. The EIM technique makes use of this feature by pre-computing a basis for the nonlinear term and enabling one, during the online stage, to compute the nonlinear term at some particular space points (*Magic Points*) and reduce the projection of this term to these particular points. This procedure is detailed in the next section 3.4. On the other hand, EIM is not appropriate for the LaTIn-PGD technique because the construction of the ROB is operated during the resolution of the problem and the basis evolves throughout the iterations. A possible use of the EIM in the LaTIn-PGD would lead, for each new generated PGD pair, to compute a number of operations related to the research of the new *Magic Point* and the associated basis scaling with the size p and N of the underlying discretizations. This would hint all the potential of the EIM technique.

In this section, the Preliminary step is approximated by the RPM. This enables one to reduce the number of elementary operations to construct the reduced order model.

Construction of the reduced-order model by the Reference Points Method

All the operations of projection and integration, denoted as α_{ij} in (3.30), are approximated by the RPM. Let us split the parameter domain in m_μ sub-intervals

$\mathcal{D}_a$. The center $\underline{\mu}_a$ of the sub-interval $\mathcal{D}_a$ is the reference parameter point. For the space domain m_x points $\underline{x}_b$ are introduced and the domain Ω is divided into m_x sub-domains Ω_b. The $\underline{x}_b$ are the reference spatial points. Denote by ω the integrand of α_{ij}:

$$\omega \;=\; \Phi_i(\underline{x})\; G(\underline{x},\underline{\mu})\; \Phi_j(\underline{x}). \tag{3.33}$$

let us define $\bar{\omega} = (\bar{\omega}^{\mu}, \bar{\omega}^{x})$, the generalized components of ω: $\bar{\omega}^{\mu}$ is associated to the reference parameter points and $\bar{\omega}^{x}$ to the reference space points:

$$\bar{\omega}^{x} := \left\{ \begin{array}{rcll} \bar{\omega}^{x}_{ab}(\underline{x}) & = & \Phi_i(\underline{x})\; G(\underline{x},\underline{\mu}_a)\; \Phi_j(\underline{x}) & \text{if } \underline{x} \;\in \Omega_b \\ \bar{\omega}^{x}_{ab}(\underline{x}) & = & 0 & \text{otherwise} \end{array} \right\}_{a=1,..,m_\mu;\; b=1,..,m_x} \tag{3.34}$$

$$\bar{\omega}^{\mu} := \left\{ \begin{array}{rcll} \bar{\omega}^{\mu}_{ab}(\underline{\mu}) & = & \Phi_i(\underline{x}_b)\; G(\underline{x}_b,\underline{\mu})\; \Phi_j(\underline{x}_b) & \text{if } \underline{\mu} \;\in \mathcal{D}_a \\ \bar{\omega}^{\mu}_{ab}(\underline{\mu}) & = & 0 & \text{otherwise} \end{array} \right\}_{a=1,..,m_\mu;\; b=1,..,m_x} \tag{3.35}$$

The approximation of ω given by RPM, noted $\bar{\bar{\omega}} = \sum_{a=1}^{m_\mu} \sum_{b=1}^{m_x} \psi_{ab}(\underline{\mu})\; \chi_{ab}(\underline{x})$ is obtained by the explicit formulas (2.31). Following formulas (2.31), $\chi_{ab}(\underline{x})$ and $\psi_{ab}(\underline{\mu})$ are obtained from (3.34) and (3.35) as follows:

$$\psi_{ab}(\mu) = \frac{\sum_{c=1}^{m_x} \Delta\Omega_c\; \bar{\omega}^{\mu}_{ac}(\mu)\bar{\omega}^{x}_{ac}(\underline{x}_c)\lambda^2_{ac}}{\sum_{c=1}^{m_x} \Delta\Omega_c\; \bar{\omega}^{x}_{ac}(\underline{x}_c)\bar{\omega}^{x}_{ac}(\underline{x}_c)\lambda^2_{ac}} \qquad \text{and} \qquad \chi_{ab}(\underline{x}) = \bar{\omega}^{x}_{ab}(\underline{x}) \tag{3.36}$$

Analysis of computational complexity Replacing ω in (3.30) one obtains:

$$\alpha_{ij} \;=\; \int_{\Omega\times\mathcal{D}} \omega(\underline{x},\underline{\mu})\; \mathrm{d}\Omega\; \mathrm{d}\mu \;\approx\; \int_{\Omega\times\mathcal{D}} \bar{\bar{\omega}}(\underline{x},\underline{\mu})\; \mathrm{d}\Omega\; \mathrm{d}\mu \tag{3.37}$$

Replacing $\bar{\bar{\omega}}$ in (3.37) one can separate the integrals:

$$\alpha_{ij} \;\approx\; \int_{\Omega\times\mathcal{D}} \bar{\bar{\omega}}(\underline{x},\underline{\mu})\; \mathrm{d}\Omega\; \mathrm{d}\mu = \sum_{a=1}^{m_\mu} \sum_{b=1}^{m_x} \int_{\Omega_b} \chi_{ab}(\underline{x})\; \mathrm{d}\Omega_b \int_{\mathcal{D}_a} \psi_{ab}(\underline{\mu})\; \mathrm{d}\mu_a \tag{3.38}$$

Replacing (3.36) in (3.38) one obtains:

$$\alpha_{ij} \;\approx\; \sum_{a=1}^{m_\mu} \sum_{b=1}^{m_x} \int_{\Omega_b} \bar{\omega}^{x}_{ab}(\underline{x})\; \mathrm{d}\Omega_b \int_{\mathcal{D}_a} \frac{\sum_{c=1}^{m_x} \Delta\Omega_c\; \bar{\omega}^{\mu}_{ac}(\mu)\; \bar{\omega}^{x}_{ac}(\underline{x}_c)\; \lambda^2_{ac}}{\sum_{c=1}^{m_x} \Delta\Omega_c\; \bar{\omega}^{x}_{ac}(\underline{x}_c)\; \bar{\omega}^{x}_{ac}(\underline{x}_c)\; \lambda^2_{ac}}\; \mathrm{d}\mu_a \tag{3.39}$$

For the integral over $\mathcal{D}_a$ the only term depending on μ is $\bar{\omega}^{\mu}_{ac}(\mu)$. Equation (3.39) is equivalent to:

$$\alpha_{ij} \;\approx\; \sum_{a=1}^{m_\mu} \sum_{b=1}^{m_x} \int_{\Omega_b} \bar{\omega}^{x}_{ab}(\underline{x})\; \mathrm{d}\Omega_b\; \frac{\sum_{c=1}^{m_x} \Delta\Omega_c\; \left(\int_{\mathcal{D}_a} \bar{\omega}^{\mu}_{ac}(\mu)\; \mathrm{d}\mu_a\right)\; \bar{\omega}^{x}_{ac}(\underline{x}_c)\, \lambda^2_{ac}}{\sum_{c=1}^{m_x} \Delta\Omega_c\; \bar{\omega}^{x}_{ac}(\underline{x}_c)\, \bar{\omega}^{x}_{ac}(\underline{x}_c)\, \lambda^2_{ac}} \tag{3.40}$$

Considering the classic discretizations, already seen before for the underlying finite element approximation ($[\Phi_i] = \sum_{j=1}^{N} [\Phi_i^j] \varphi_j(\underline{x})$), the two integrals become as follows:

$$\int_{\Omega_b} \bar{\omega}^x_{ab}(\underline{x}) \, \mathrm{d}\Omega_b = [\Phi_i]_b^T \, \mathbb{G}_{ab} \, [\Phi_j]_b \tag{3.41}$$

$$\int_{\mathcal{D}_a} \bar{\omega}^\mu_{ac}(\mu) \, \mathrm{d}\mu_a = [\Phi_i]_{x_b}^T \int_{\mathcal{D}_a} G(\underline{x}_b, \underline{\mu}) \, \mathrm{d}\mu \, [\Phi_j]_{x_b} \tag{3.42}$$

where

$$\mathbb{G}_{ab} = \left(\sum_{el.\ in\ \Omega_b} \int_{elem.} \varphi_i(\underline{x}) \, G(\underline{x}, \underline{\mu}_a) \, \varphi_j(\underline{x}) \, \mathrm{d}\Omega \right) \tag{3.43}$$

$$[\Phi_i]_b = [\Phi_i] \qquad \forall \ \underline{x} \in \Omega_b \tag{3.44}$$

$$[\Phi_i]_{x_b} = [\Phi_i] \qquad \text{for } \underline{x} = \underline{x}_b. \tag{3.45}$$

Let us consider, now, the contribution of every PGD space mode of the reduced-order basis, $\mathbb{W}_k = ([\Phi_1], [\Phi_2], \ldots, [\Phi_k])$. It yields:

$$\mathbb{W}_b = \mathbb{W}_k \qquad \forall \ \underline{x} \in \Omega_b \tag{3.46}$$

$$\mathbb{W}_{x_b} = \mathbb{W}_k \qquad \text{for } \underline{x} = \underline{x}_b \tag{3.47}$$

$\mathbb{W}_{k\ b}$ represents the restriction of the PGD reduced-order basis of order k to the patch b. Let us suppose that the reduced-order basis is composed by three PGD modes $\mathbb{W}_3 = ([\Phi_1], [\Phi_2], [\Phi_3])$ and the reference points are a grid of 3×3 points. **Fig. 3.5** depicts PGD space modes composing $\mathbb{W}_3$ and their restriction to a generic patch b.

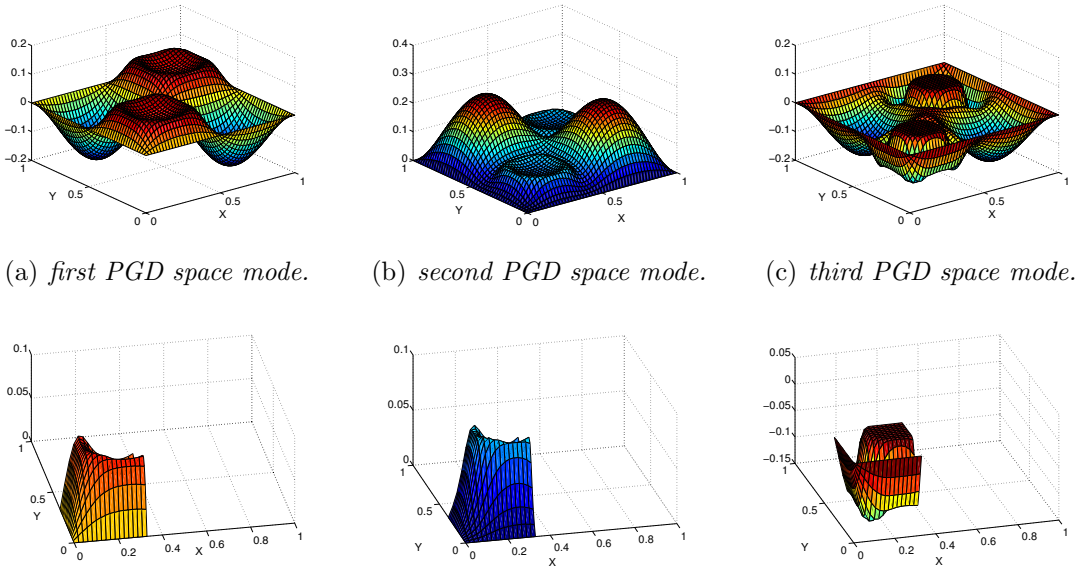
Hence, considering the contribution of the entire reduced-order basis $\mathbb{W}_k$ (denoted $\mathbb{W}$ in the following to alleviate the notations), (3.41) and (3.42) are written as follows:

$$\int_{\Omega_b} \bar{\omega}^x_{ab}(\underline{x}) \, \mathrm{d}\Omega_b = \mathbb{W}_b^T \, \mathbb{G}_{ab}(\underline{x}, \underline{\mu}_a) \, \mathbb{W}_b \tag{3.48}$$

$$\int_{\mathcal{D}_a} \bar{\omega}^\mu_{ac}(\mu) \, \mathrm{d}\mu_a = \mathbb{W}_{x_b}^T \int_{\mathcal{D}_a} G(\underline{x}_b, \underline{\mu}) \, \mathrm{d}\mu \, \mathbb{W}_{x_b} \tag{3.49}$$

The Galerkin projection onto the reduced-order basis (3.48) and (3.49) is computed patch by patch. For the patch (a, b), terms in (3.48) and (3.49) have the following size:

$$\bar{\omega}^x_{ab} = \underbrace{\mathbb{W}_b^T}_{k \times \frac{N}{m_x}} \ \underbrace{\mathbb{G}_{ab}}_{\frac{N}{m_x} \times \frac{N}{m_x}} \ \underbrace{\mathbb{W}_b}_{\frac{N}{m_x} \times k} \qquad \bar{\omega}^\mu_{ab} = \left(\underbrace{\mathbb{W}_{x_b}^T}_{k \times 1} \int_{\mathcal{D}_a} \underbrace{G(\underline{x}_b, \underline{\mu})}_{p \times 1} \, \mathrm{d}\mu \ \underbrace{\mathbb{W}_{x_b}}_{1 \times k} \right) \tag{3.50}$$



(a) *first PGD space mode.* (b) *second PGD space mode.* (c) *third PGD space mode.*

(d) *restriction of first PGD space mode to* Ω_b. (e) *restriction of second PGD space mode to* Ω_b. (f) *restriction of third PGD space mode to* Ω_b.

Figure 3.5: Reduced-order basis composed by three PGD modes $\mathbb{W}_3$ and its restriction to Ω_b.

Consequently, for a single patch the number of operations is in the order of:

$$\frac{N}{m_x}\,\beta \;+\; \frac{N}{m_\mu}\,k^2 \;+\; \frac{p}{m_\mu}\,\beta \;+\; k^2. \tag{3.51}$$

$\frac{N}{m_x}\ \beta$ represents the number of operations to compute the integral $\mathbb{G}_{ab}$ (3.43). $\frac{N}{m_\mu}\ k^2$ represents the number of operations to perform the Galerkin projection $\mathbb{W}_b^T\ \mathbb{G}_{ab}(\underline{x},\underline{\mu}_a)\ \mathbb{W}_b$ (see (3.50)). $\frac{p}{m_\mu}\ \beta$ represents the number of operations to compute the integral $\int_{\mathcal{D}_a} G(\underline{x}_b,\underline{\mu})\ \mathrm{d}\mu$ and k^2 the Galerkin projection in (3.50).

Computing the sum over the $m_\mu\, m_x$ patches, it yields:

$$\left(\underbrace{\frac{N}{m_x}\,\beta \;+\; \frac{N}{m_x}\,k^2}_{\bar{\omega}^x_{ab}} \;+\; \underbrace{\frac{p}{m_\mu}\,\beta \;+\; k^2}_{\bar{\omega}^\mu_{ab}}\right)\ m_\mu\ m_x. \tag{3.52}$$

The same procedure for the residue in (3.32) leads to:

$$\left(\frac{N}{m_x}\,k \;+\; k^2\right)\ m_\mu\ m_x. \tag{3.53}$$

The results of this analysis are summarized in **Tab.3.2**. Order of complexity for numerical simulation of Problem 10 is also given as an example. In this case, one has $N = 2500;\ p = 225;\ k_{max} = 20;\ m_x = m_\mu \approx 10;\ \beta \approx 1$.

Table 3.2: Computational complexity of integration and projection onto ROB for Jacobian and residue, in the RPM format

	Operation type	Complexity	Problem 10
$\bar{\omega}_\mu$	Integration point evaluation	$\mathcal{O}(m_\mu\ N\ \beta)$	$\approx 10^5$
	Projection of Jacobian onto ROB	$\mathcal{O}(m_\mu\ N\ k^2)$	$\approx 10^7$
	Projection of residue onto ROB	$\mathcal{O}(m_\mu\ N\ k)$	$\approx 10^6$
$\bar{\omega}_x$	Integration point evaluation	$\mathcal{O}(m_x\ p\ \beta)$	$\approx 10^3$
	Projection of Jacobian onto ROB	$\mathcal{O}(m_\mu\ m_x\ k^2)$	$\approx 10^4$
	Projection of residue onto ROB	$\mathcal{O}(m_\mu\ m_x\ k)$	$\approx 10^3$

The number of operations corresponding to $\bar{\omega}^\mu$ is almost three orders of magnitude bigger than the number of operations related to $\bar{\omega}^x$. Hence, this latter can be neglected. Finally, whenever the ratio $\frac{N}{m_x}$ is higher than three order of magnitude, the ideal gain is in the order of the ratio:

$$\frac{p\ N\ (\beta + k^2)}{m_\mu\ N\ (\beta + k^2)} = \frac{p}{m_\mu}. \tag{3.54}$$

This gain, for typical mechanical problems, can be around one order of magnitude.

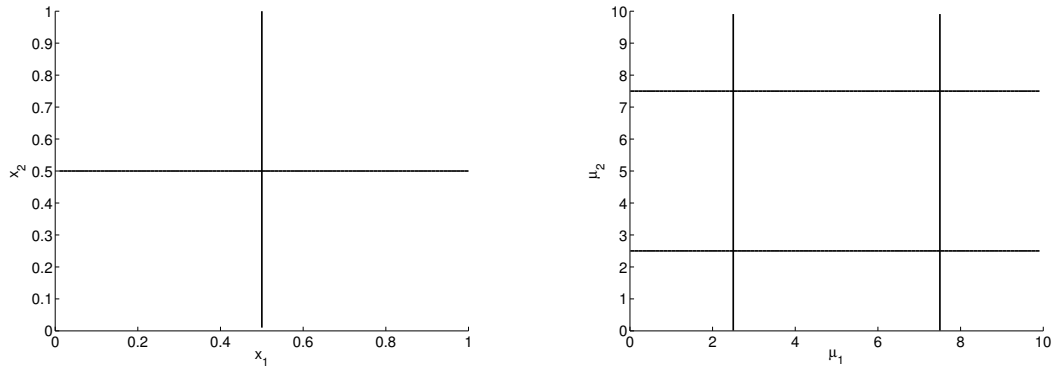
Cost of the reconstruction As introduced in chapter 2.3 the generation of a first approximation of the quantity α in a separated-variables format is obtained from the compressed format $\bar{\omega}$ by generating one product of functions per parameter-space patch $\mathcal{D}_a \times \Omega_b$. This leads to explicit formulas (3.36):

$$\psi_{ab}(\mu) = \frac{\sum_{c=1}^{m_x} \Delta\Omega_c \bar{\omega}^\mu_{ac}(\mu)\bar{\omega}^x_{ac}(\underline{x}_c)\lambda^2_{ac}}{\sum_{c=1}^{m_x} \Delta\Omega_c \bar{\omega}^x_{ac}(\underline{x}_c)\bar{\omega}^x_{ac}(\underline{x}_c)\lambda^2_{ac}} \qquad \text{and} \qquad \chi_{ab}(\underline{x}) = \bar{\omega}^x_{ab}(\underline{x}) \tag{3.55}$$

The cost of reconstruction given by the (3.55) is simply in the order of $8(m_x^2\ m_\mu)+2$. This computational cost has to be added to the computational complexity of the operation, analysed in **Tab. 3.2**.

Numerical results Problem 11 is solved by the LaTIn-PGD computational strategy with the preliminary step approximated by the RPM (see previous section). Different combinations of reference points are investigated. Space and parameter domains have two components, $\underline{x} = (x_1, x_2)$ and $\underline{\mu} = (\mu_1, \mu_2)$. Hence, selection of reference points leads to the definition of two grids, one over the space and one over the parameter domain. For instance, in **Fig. 3.6** are shown the two grids when 1×1 points over the space and 2×2 over the parameter domain are chosen.

Fig. 3.7 shows the relative error ϵ with respect to the reference solution, when the Preliminary step is approximated by the RPM. The relative error ϵ is defined



(a) 1×1 *points over the space.* (b) 2×2 *over the parameter domain.*

Figure 3.6: Example of selection for 1×1 reference point over the space and 2×2 reference points over the parameter domain.

by:

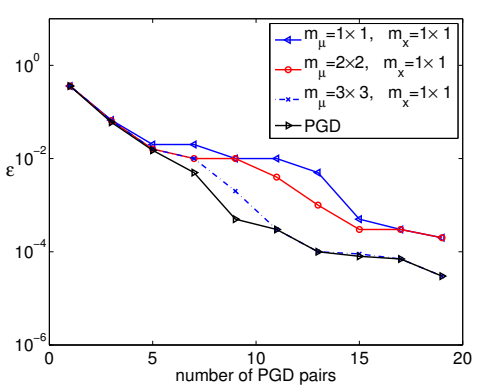
$$\epsilon = 1/p \sum_{j=1}^{p} \frac{\| u_{ref}\ (\underline{x}, \underline{\mu}_j) - u_{RPM}\ (\underline{x}, \underline{\mu}_j)\|_{L_2}}{\| u_{ref}\ (\underline{x}, \underline{\mu}_j)\|_{L_2}} \tag{3.56}$$

where $u_{ref}(\underline{x}, \underline{\mu}_j)$ is the solution for the parameter value $\underline{\mu}_j$ obtained by solving Problem 11 with a direct incremental method (i.e. a classic finite element analysis for the same number N of d.o.f. with Newton solution method). Error between the classic PGD technique and the reference solution is also presented. **Fig. 3.7** also depicts, the wall-clock time for the RPM approximation compared to the direct incremental method, the POD and the standard PGD techniques.

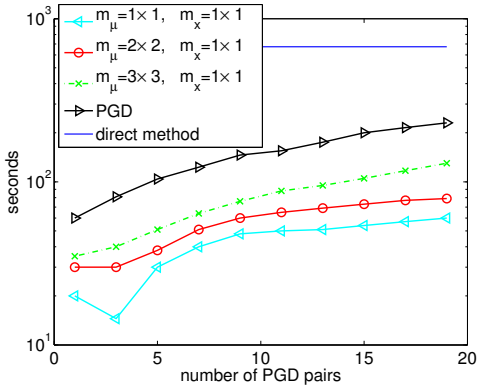
One can see that by increasing the number of reference points, the error curve tends to the one obtained with the classic PGD technique (i.e. without RPM approximation of the Preliminary step).

Let us consider a single curve, for instance the blue one in **Fig. 3.7** *(c)*. The error ϵ of the RPM approximation decreases by adding new PGD pairs along the LaTIn iterations. At each LaTIn iteration, the first approximation obtained by approximating the preliminary step by the RPM can be enriched by generating a new PGD pair (see Algorithm 4). The blue line corresponds to the case where only one reference point is used for both space and parameter domains. This curve shows that even by using only one reference point an error ϵ of 10^{-3} can be reached.

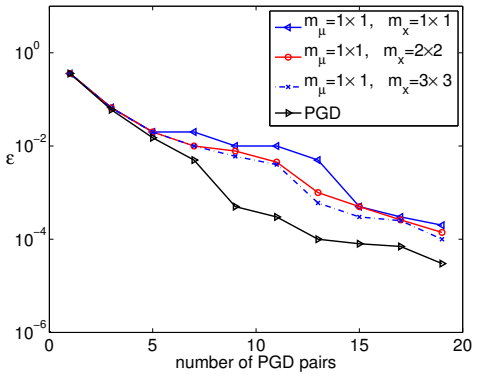
Fig. 3.7 *(c)* shows, globally, an error higher than **Fig. 3.7** *(a)*. This is because in **Fig. 3.7** *(a)* more parameter reference points are used. For each reference parameter point a complete integration of the operators over the space domain is computed. The dependence to the space variable plays a more important role than the dependence to the parameter variable. In fact, generally, for parametrized elliptical problems (as it is for many mechanical problems), gradients in space variable are stronger than gradients in time/parameter variable. Finally, the error ϵ decreases



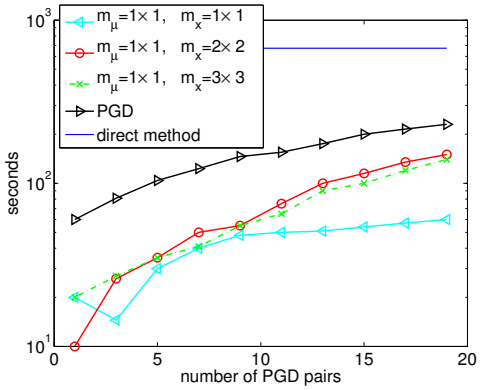
(a) *error* ϵ *for* $m_\mu = \{1 \times 1;\ 2 \times 2;\ 3 \times 3\}$ *and* $m_x = 1 \times 1$.



(b) *CPU time for* $m_\mu = \{1 \times 1;\ 2 \times 2;\ 3 \times 3\}$ *and* $m_x = 1 \times 1$.



(c) *error* ϵ *for* $m_\mu = 1 \times 1$ *and* $m_x = \{1 \times 1;\ 2 \times 2;\ 3 \times 3\}$.



(d) *CPU time for* $m_\mu = 1 \times 1$ *and* $m_x = \{1 \times 1;\ 2 \times 2;\ 3 \times 3\}$.

Figure 3.7: The error ϵ, defined in (3.56) and the wall clock curves of the RPM approximation for the Preliminary step (Problem 10) compared to the error ϵ of the standard LaTIn-PGD without approximation of the Preliminary step.

either by adding more RPM point or by adding more PGD pairs. The choice between these two options can be adapted to the considered problem.

Remark 9 ***Fig. 3.7*** *shows that increasing the number of reference points, the error curve tends to the one obtained with the classic PGD technique (i.e. without RPM approximation of the Preliminary step). Let us remark that the error obtained with the classic PGD technique represents the limit for the approximation given by the RPM. Indeed, at the LaTIn iteration* $n+1$*, the level of accuracy of the approximation is defined by the order of the reduced-order basis. Hence, the first approximation of the Preliminary step given by the RPM can be enriched either by adding more RPM point (up to reach the same approximation given by the classic PGD technique) or by adding more PGD pairs.*

Tab. 3.3 presents the comparison between the standard LaTIn-PGD and the LaTIn-PGD using the RPM framework to perform the Preliminary step, in terms of speed up of CPU time. For a given level of error $\epsilon = 10^{-2}$ the gain with respect to the direct incremental method is in the order of 15 and it is three times bigger than the gain of the classic LaTIn-PGD.

Table 3.3: Complete simulation: CPU time gain with respect to the classic finite element simulation for a given level of error ($\epsilon = 10^{-2}$)

	PGD	PGD-RPM 1 Parameter point 1 Space points	PGD-RPM 4 Parameter point 1 Space points	PGD-RPM 9 Parameter point 1 Space points
num. of PGD pairs	7	9	7	7
Total gain	6	18	14.2	11.5

Let us analyse the preliminary step, where the RPM approximation takes place. **Tab. 3.4** analyses the computational cost of the preliminary step. It shows the ideal and the real gain reachable by applying the RPM to approximate the Preliminary step. The ideal gain is in the order of p/m_μ (see the analysis of computational complexity in section 4.4), where p is the number of values in the parameter discretization and m_μ is the number of reference points along the parameter coordinate. The ideal gain and the real gain are equals when 9 parameter points are used. This gain is in the order of 25. Using less reference points the real gain is higher than 25, but the ideal gain is not reached. For Problem 10 the preliminary step of the classic LaTIn-PGD is computed in 166 seconds. Using the RPM method it can be computed in 1.84 seconds.

Table 3.4: Preliminary step: CPU time gain with respect to the classic LaTIn-PGD for a given level of error ($\epsilon = 10^{-2}$)

	PGD-RPM 1 Parameter point 1 Space points	PGD-RPM 4 Parameter point 1 Space points	PGD-RPM 9 Parameter point 1 Space points
Ideal gain p/m_μ	225	56	25
Real gain	90	39	25

3.4 Comparison with respect to the Empirical Interpolation Method

The aim of this section is to compare the quality and the efficicency of the approximation given by the RPM-PGD approach, previously presented, with the one achieved by the popular EIM-Reduced Basis approach. This technique belongs to the family of reduced-order modelling that constructs the ROB during a learning stage. The full order model is first solved for some "appropriate" parameter values $S^n = \{\underline{\mu}_i \in \mathcal{D}\}_{1\leq i\leq n}$ with $n \ll N$. From these snapshots a ROB of dimension $k \leq n$ is obtained either by a POD, selecting the first k orthogonal modes $\mathbf{V}_k = \{\Phi_i\}_{1\leq i\leq k}$ (POD-Galerkin strategy) or simply by collecting the k "relevant" snapshot solutions $\mathbf{V}_k = \{\Phi_i = u(\underline{x}, \underline{\mu}_i)\}_{1\leq i\leq k}$ (reduced-basis approach). In the latter case, the Φ_i are also orthonormalized in practice. Galerkin projection is typically used to construct a ROM thanks to the obtained ROB of dimension $k \ll N$.

Following the classic reduced-basis approximation, the solution is sought in $\mathbf{\Phi}_k$, the linear span of the orthogonal ROB, $\mathbf{V}_k$, by Galerkin projection. Problem 11 reads:

Problem 16 (reduced basis problem) *Given $\underline{\mu} \in \mathcal{D}$, find $u(\underline{x}, \underline{\mu}) \in \mathbf{\Phi}_k$ such that:*

$$\forall v \in \mathbf{\Phi}_k, \quad a(u, v) + \int_\Omega g(u; \underline{\mu})\ v\ \mathrm{d}\Omega = \int_\Omega f(\underline{x})\ v\ \mathrm{d}\Omega \tag{3.57}$$

Thus, by denoting the approximation of the solution $\tilde{u}(\underline{x}, \underline{\mu}) = \sum_{i=1}^{k} \lambda_i(\underline{\mu})\ \Phi_i(\underline{x})$, and by choosing Φ_i as a test function for v in Problem 16, the approximation $\tilde{u}$ is obtained by the $k \times k$ nonlinear algebraic system:

$$\forall\, 1 \leq j \leq k, \quad \sum_{i=1}^{k} a(\Phi_i, \Phi_j)\ \lambda_i(\underline{\mu}) + \int_\Omega g\left(\sum_{i=1}^{k} \lambda_i(\underline{\mu})\underline{\Phi}_i; \underline{\mu}\right) \Phi_j\ \mathrm{d}\Omega = \int_\Omega f(\underline{x})\ \Phi_j\ \mathrm{d}\Omega \tag{3.58}$$

Let us note that first term in left-hand side member, $a(\Phi_i, \Phi_j)$, is parameter-independent and can be precomputed just once. However, the second term depends on $g(u; \underline{\mu})$ and, consequently, is parameter-dependent and must be evaluated online for each new parameter value $\underline{\mu}$. One can show that the complexity of the online stage scales with $\mathcal{O}(Nk^2)$ with N the dimension of the underlying finite element approximation space. Empirical interpolation method [Grepl et al., 2007] replaces $g(.; \underline{\mu})$ in (3.58) by an affine approximation $g(.; \underline{\mu}) \approx \sum_{i=1}^{M} \Psi_i(\underline{\mu}) q_i(\underline{x})$, in order to recover N-independence. Contrary to the RPM, the EIM is not an approximation method but rather an interpolation method in a sense that it aims at interpolating the nonlinear term from values at M *"Magic Points"* points.

3.4.1 Computational complexity analysis of reduced-basis method

Nonlinear algebraic system (3.58) is classically solved by iterative methods such as the Newton-Raphson method for each value of parameter $\underline{\mu}$. Each tangent problem of the iterative scheme is, then, projected on the precomputed ROB, $\mathbf{V}_k$. This section aims at detailing the N-dependence computational complexity of classic model reduction techniques for nonlinear problems.

Newton method consists in solving a linearized formulation of Problem (16). Knowing iterate $\tilde{u}^{(n)}$, one looks for $\tilde{u}^{(n+1)} = \tilde{u}^{(n)} + \delta\tilde{u}^{(n)}$ by solving:

Problem 17 (tangent problem) *Given $\underline{\mu} \in \mathcal{D}$, find $\delta\tilde{u}^{(n)} \in \mathbf{\Phi}_k$ such that:*

$$\forall v \in \mathbf{\Phi}_k, \quad \mathcal{R}(\tilde{u}^{(n)}, v; \underline{\mu}) + \left\langle \mathcal{R}'(\tilde{u}^{(n)}, v; \underline{\mu}), \delta\tilde{u}^{(n))} \right\rangle = 0 \tag{3.59}$$

with $\mathcal{R}(w, v; \underline{\mu}) = a(w, v) + \int_\Omega \ g(w; \underline{\mu}) \ v \ \mathrm{d}\Omega - \int_\Omega \ f(\underline{x}) \ v \ \mathrm{d}\Omega$.

Linear tangent application $\mathcal{R}'$ is defined by:

$$\mathcal{R}(w + z, v; \underline{\mu}) - \mathcal{R}(w, v; \underline{\mu}) = \left\langle \mathcal{R}'(w, v; \underline{\mu}), z \right\rangle + o(z)$$

that is to say:

$$\left\langle \mathcal{R}'(w, v; \underline{\mu}), z \right\rangle = a(z, v) + \int_\Omega \ g'(w; \underline{\mu}) \ z \ v \ \mathrm{d}\Omega$$

with g' the derivative according to the first argument. By denoting the reduced-basis approximation of the solution $\delta\tilde{u}_i^{(n)} = \sum_{i=1}^k \lambda_i \ \Phi_i$, and by choosing Φ_i as a test function for v, it leads to the following $k \times k$ linear algebraic system:

$$\forall\, 1 \leq j \leq k, \quad \sum_{i=1}^{k} \left[a(\Phi_i, \Phi_j) + \int_\Omega \ \Phi_i \ g' \left(\tilde{u}^{(n)}; \underline{\mu} \right) \Phi_j \, \mathrm{d}\Omega \right] \ \lambda_i = -\mathcal{R}(\tilde{u}^{(n)}, \Phi_j; \underline{\mu}) \tag{3.60}$$

By taking into consideration the underlying finite element approximation ($[\Phi_i] = \sum_{j=1}^N [\Phi]_i^j \varphi_j(\underline{x})$), this reads:

$$\mathbb{V}_k^T \left[\mathbb{A} + \mathbb{G}(\tilde{u}^{(n)}; \underline{\mu}) \right] \mathbb{V}_k \ \{\lambda\}_k = -\mathbb{V}_k^T \ [R(\tilde{u}^{(n)}; \underline{\mu})] \tag{3.61}$$

$\mathbb{A}$ and $\mathbb{G}$ are $N \times N$ matrices such that $\mathbb{A}_{ij} = a(\varphi_i, \varphi_j)$, $\mathbb{G}_{ij} = \int_\Omega \ \varphi_i \ g' \left(\tilde{u}^{(n)}; \underline{\mu} \right) \varphi_j \, \mathrm{d}\Omega$. Residue $[R(\tilde{u}^{(n)}; \underline{\mu})]$ is a N-length vector with $[R]_i = -\mathcal{R}(\tilde{u}^{(n)}, \varphi_i; \underline{\mu})$. The discretized reduced basis is a $N \times k$ matrix with $\mathbb{V}_k = \{[\Phi_1], [\Phi_2], \ldots, [\Phi_k]\}$. The vector of unknowns $\{[\lambda]\}_k$ has a size of k.

Note that (3.61) is the same as (3.17) where the ROB is not given by $\mathbf{W}_k = \{\Phi_i\}_{1 \leq i \leq k}$ but by $\mathbf{V}_k = \{\Phi_i\}_{1 \leq i \leq k}$. These two ROBs are not the same since PGD generates modes throughout the process and $\mathbf{W}_k = \{\Phi_i\}_{1 \leq i \leq k}$ may change at each iteration.

Both Jacobian matrices $\mathbb{G}(\tilde{u}^{(n)};\underline{\mu})$ and residue $[R(\tilde{u}^{(n)};\underline{\mu})]$ depends on parameter $\underline{\mu}$. For each new parameter value, one has (i) to evaluate Jacobian and residue, (ii) to project onto the discretized ROB $\mathbb{V}_k$, and, finally, (iii) to solve system (3.61) to obtain $\{\lambda\}$. Projection of Jacobian (resp. residue) onto ROB has a computational complexity that depends on $\mathcal{O}(N\ k^2)$ since $\mathbb{G}$ is sparse (resp. $\mathcal{O}(N\ k)$) [Grepl et al., 2007, Chaturentabut and Sorensen, 2010]. Solving system (3.60) has a complexity in the order of $\mathcal{O}(k^3)$ (for a Cholesky factorization of a full matrice). The total computational complexity of a Newton iteration is consequently in the order of $\mathcal{O}(N\ k^2+N\ k+k^3)$ and is N-dependent. This complexity number can even overpass the number of elementary operations to solve the original full order Problem 11, that is in the order of $\mathcal{O}(N^3)$. It is shown in [Chaturentabut and Sorensen, 2010] that, for Problem 10, the CPU time for solving the POD reduced model (POD-Galerkin strategy) for each parameter value exceeds the CPU time for solving the original full order problem as soon as the dimension of the finite element discretisation space reaches around $N = 80$.

Learning stage of reduced-basis method The procedure explained in this section exploits a reduced-basis constructed during a learning stage. The process to collect the k *relevant* snapshot solutions $\mathbf{V}_k = \{\Phi_i = u(\underline{x};\underline{\mu}_i)\}_{1\leq i\leq k}$ is based on the following greedy algorithm:

1. First snapshot is computed for first parameter. It can be chosen randomly in the discretized parameter domain $\mathcal{D}_p \subset \mathcal{D}$:

$$\underline{\mu}_1 \ \in \ \mathcal{D}_p \tag{3.62}$$

2. Given $i-1$ samples in the parameter set, $\underline{\mu}_1,\ldots,\underline{\mu}_i$, one constructs the space

$$\mathbf{\Phi}_i = \ \text{span}\ \{u(\underline{x},\underline{\mu}_1),...,u(\underline{x},\underline{\mu}_i)\}.$$

   In actual practice $u(\underline{x},\underline{\mu})$ is approximated by the underlying finite element approximation. One denotes by P_i the Galerkin projection onto the space $\mathbf{\Phi}_i$:

$$a(P_iu(\underline{x},\underline{\mu}),v)\ +\ \int_\Omega g(P_iu(\underline{x},\underline{\mu});\underline{\mu}) = a(w,\underline{\mu},v)\ +\ \int_\Omega g(w;\underline{\mu}),\quad \forall v\in\mathbf{\Phi}_i. \tag{3.63}$$

3. The next snapshots are related to the parameters chosen as follows

$$\underline{\mu}_{i+1} = \ \arg\ \sup_{\underline{\mu}\in\mathcal{D}} \|u(\underline{x},\underline{\mu})\ -\ P_iu(\underline{x},\underline{\mu})\|_{L^2} \tag{3.64}$$

4. Iterate until $\|u(\underline{x},\underline{\mu})\ -\ P_iu(\underline{x},\underline{\mu})\|_{L^2}\ <\ tol$.

Then, $\mathbf{\Phi}_i$ is orthogonalize with respect to the scalar product related to the L^2 norm. One denotes by $\mathbf{V}_k$ the obtained basis.

3.4.2 Empirical Interpolation method

The EIM is briefly developed in this section. See the article [Grepl et al., 2007] for more details. Under the hypothesis of sufficient regularity ($C^0(\Omega)$ is sufficient) of $g(u;\underline{\mu}) = g(u;\underline{x},\underline{\mu})$ in this approach, a secondary reduced-basis expansion $g_M(u;\underline{x},\underline{\mu})$ for the nonlinear term $g(u;\underline{x},\underline{\mu})$ is developed and replaced in (3.58) with some necessarily affine approximation, defined as: $g_M(u;\underline{x},\underline{\mu}) = \sum_{m=1}^{M} \phi_m^M(\underline{\mu})\, q_m(\underline{x})$. Hence, function $g(u;\underline{x},\underline{\mu})$ is approximated in a separated variables form and a ROB for this term has to be generated. Note that, since there is already a basis for the field $u(\underline{x},\mu)$, the term "secondary" is used to distinguish the two reduced-basis expansions. To this end, let introduce the nested sample sets $S_M^g = \{\underline{\mu}_1^g, .., \underline{\mu}_M^g \in \mathcal{D}\}$ and the associated nested reduced-basis

$$\mathbf{W}_M^g = \text{ span } \{\xi_m(\underline{x}) \equiv g(u;\underline{x},\underline{\mu}_m^g),\ 1 \leqslant m \leqslant M\}$$

in which g_M shall reside. In order to construct this sets let introduce the best approximation

$$g^\star(u;\underline{x},\underline{\mu}) \equiv \arg \min_{z\in\mathbf{W}_M^g} \|g(u;\underline{x},\underline{\mu}) - z\|_{L^2(\Omega)} \tag{3.65}$$

and the associated error

$$\epsilon^\star(\underline{\mu}) = \|g(u;\underline{x},\underline{\mu}) - g^\star(u;\underline{x},\underline{\mu})\|_{L^2(\Omega)}. \tag{3.66}$$

The construction of S_M^g and $\mathbf{W}_M^g$ is based on a greedy selection process (as already seen before this is a feature of the RB). The first sample point can be chosen randomly in the parameter domain $\mu_1^g \in \mathcal{D}_p$, and define:

$$S_1^g = \underline{\mu}_1^g, \qquad \xi_1 = g(u;\underline{x},\underline{\mu}_1^g), \qquad \mathbf{W}_1^g = \text{span}\{\xi_1\} \tag{3.67}$$

For $M \geq 2$:

$$\underline{\mu}_M^g = \arg \max_{\underline{\mu}\in\mathcal{D}_p} \epsilon_{M-1}^\star(\underline{\mu}), \qquad S_M^g = S_{M-1}^g \cup \underline{\mu}_M^g, \qquad \xi_M = g(u;\underline{x},\underline{\mu}_M^g). \tag{3.68}$$

and

$$\mathbf{W}_M^g = \text{span}\{\xi_m, 1 \leqslant m \leqslant M\}. \tag{3.69}$$

Then $\mathbf{W}_M^g$ consists of a basis of functions from the parametrically induced manifold $\mathcal{M}^g \equiv g(u;\underline{x},\underline{\mu})|\underline{\mu} \in \mathcal{D}_p$. The computational cost to evaluate the new μ_M^g is $\mathcal{O}(M\ N) + \mathcal{O}(M^3)$, where N is the dimension of the underlying finite element space. At this stage, the "secondary" ROB related to the nonlinear term $g(u;\underline{x},\underline{\mu})$ is obtained.

Let us construct, now, the nested sets of interpolation points (the *"Magic Points"*) $T_M = \{\underline{x}_1, .., \underline{x}_M\}_{1\leqslant M\leqslant M_{max}}$. Set firstly:

$$\underline{x}_1 = \arg \sup_{\underline{x}\in\Omega} |\xi_1(\underline{x})|, \qquad q_1 = \xi_1(\underline{x})/\xi_1(x_1), \qquad B_{11}^1 = q_1(x_1) = 1, \tag{3.70}$$

where $\xi_i(\underline{x}) \equiv g(u;\underline{x},\underline{\mu}_i^g)$.

The first interpolation point $\underline{x}_1$ is the point where the maximum of the absolute value of $g(\underline{x},u_1^g)$ is reached. Then for $M = 2,...,M_{max}$, one solves the following linear system for the coefficients σ:

$$\sum_{j=1}^{M-1} \sigma_j \; B_{ij} \;=\; \xi_M(\underline{x}_i) \;=\; 1, \qquad 1 \leqslant i \leqslant M-1; \tag{3.71}$$

then, the residue is calculated by:

$$r_M(\underline{x}) \;=\; \xi_M(\underline{x}) \;-\; \sum_{j=1}^{M-1} \sigma_j \; q_j(\underline{x}_i), \tag{3.72}$$

Thus, one can set:

$$\underline{x}_M = \arg \sup_{\underline{x}\in\Omega} |r_M(\underline{x})|, \qquad q_M(\underline{x}) = r_M(\underline{x})/r_M(\underline{x}_M), \qquad B_{ij} = q_j(\underline{x}_i),\; 1 \leqslant i,j \leqslant M. \tag{3.73}$$

Remark 10 *At this stage, it is not straightforward to understand the meaning of matrix B. Looking at* (3.74) *and* (3.75)*, this is the matrix that enables one to obtain the coefficients interpolating the secondary reduced-order basis $\mathbf{W}_M^g = \{\xi_m, 1 \leqslant m \leqslant M\}$ evaluating the nonlinear function $g(u;\underline{x},\underline{\mu})$ only at the* Magic Points*: $g(u;\underline{x}_i,\underline{\mu})$, $1 \leqslant i \leqslant M$. Hence, the linear system* (3.71)*, relying on the solutions of the nonlinear function $g(u;\underline{x},\underline{\mu})$ for the nested sample sets S_M^g, enables one to locate the space points (*Magic Points*) where to evaluate the nonlinear function $g(u;\underline{x},\underline{\mu})$ in order to obtain the right interpolating coefficients associated to the secondary reduced-order basis $\mathbf{W}_M^g$ (see* (3.75)*).*

Now, whether B_{ij} is invertible the construction of the interpolation points is well-defined and the functions $\{\, q_1,...,q_M \,\}$ form a basis for $\mathbf{W}^g$. To prove that it is sufficient to observe that from the construction procedure $|B_{ij}| = |r_j(\underline{x}_i)/r_j(\underline{x}_j)| = 0$ for $1 < j$; that $|B_{ij}| = |r_j(\underline{x}_i)/r_j(\underline{x}_j)| = 1$ for $i = j$ and that $|B_{ij}| = |r_j(\underline{x}_i)/r_j(\underline{x}_j)| \leqslant 1$ for $i > j$. Consequently, B is lower triangular with unity diagonal, and thus invertible.

At this stage "secondary" ROB for $g_M(u;\underline{x},\underline{\mu})$ is built and also the way to approximate $g_M(u;\underline{x},\underline{\mu})$ in the linear span of this ROB is found. Hence, let us express the reduced-basis expansion for $g_M(u;\underline{x},\underline{\mu})$. Based on the approximation space $\mathbf{W}_M^g$ and set of the interpolation points T_M, the approximation of $g(u;\underline{x},\underline{\mu})$ can be written as

$$g_M(u;\underline{x},\underline{\mu}) = \sum_{m=1}^{M} \phi_m^M(\underline{\mu}) \; q_m(\underline{x}), \tag{3.74}$$

where the interpolating functions $\phi^M(\underline{\mu})$, defined in $L(\mathcal{D})$ with values in $\mathbb{R}^M$, are given by:

$$\sum_{j=1}^{M} B_{ij}\ \phi_j(\underline{\mu}) = g(u;\underline{x}_i,\underline{\mu}),\ 1 \leqslant i \leqslant M; \tag{3.75}$$

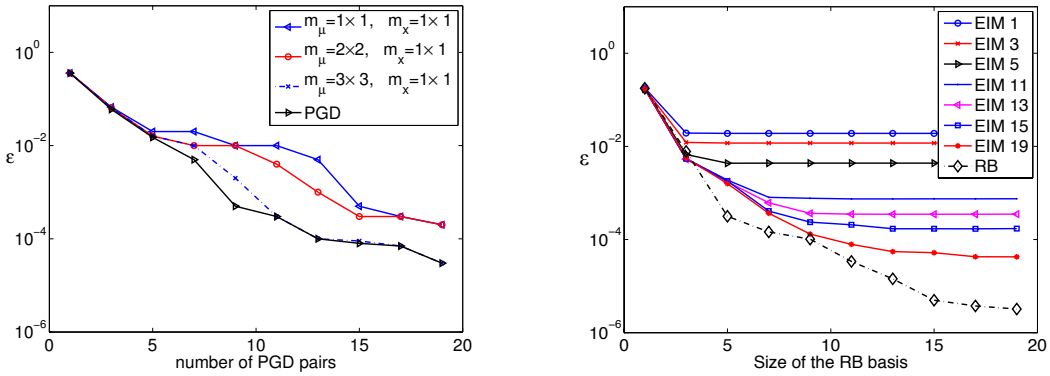
Note that $g_M(u;\underline{x}_i,\underline{\mu}) = g(u;\underline{x}_i,\underline{\mu}),\ 1 \leqslant i \leqslant M$. Finally the nonlinear function is evaluated just on the M points defined in T_M. These values, thanks to the matrix B allow one to generate $g_M(u;\underline{x},\underline{\mu})$, a reduced-basis expansion for $g(u;\underline{x};\underline{\mu})$.

Numerical results Problem 11 is solved, here, by the Reduced Basis technique coupled with the Empirical Interpolation Method following [Grepl et al., 2007]. Let us consider the discretized parameter domain $\mathcal{D}_p \subset \mathcal{D}$. $\mathcal{D}_p$ is denoted by $\mathcal{D}_c$ to define a regular parameter coarse grid $\mathcal{D}_c = \{\underline{\mu}_i\}_{i=1..n_c}$ with $n_c = 12$ over $\mathcal{D}$ is considered. During an off-line stage the solution is calculated by an incremental direct method for every parameter set $\underline{\mu} \in \mathcal{D}_c$. For nonlinear problems the solution of the PDE is necessary in order to construct the secondary basis $\mathbf{W}_M^g$ for the nonlinear function. Then, the reduced basis is constructed by an adaptive sampling construction (see the previous section). At this point the nested sets of $1 \leq M \leq M_{max}$ interpolation points are constructed using a greedy algorithm based on a L_2 norm. Once the learning stage is accomplished, the discretized parameter domain $\mathcal{D}_p$ is denoted by $\mathcal{D}_f$ to define a finer set of parameters $\mathcal{D}_f = \{\underline{\mu}_i\}_{i=1..n_f}$ with $n_f = 15$. This leads to $p = 225$ values for parameter $\underline{\mu}$.

Let us consider the error ϵ in (3.56). In **Fig. 3.8** *(b)* the error obtained by applying the EIM to the Problem 10 is presented. It is compared with the error found with the RPM, already seen in **3.7** and reported in **Fig. 3.8** *(a)*. The RPM solutions obtained with only one space reference point and a various number of parameter points have been considered for this comparison. Indeed, as previsously seen for this example, adding more parameter points is better than adding more reference spatial points in order to get a solution with a good level of accuracy.

The quality of the approximation is comparable between the two techniques. However, some differences must be pointed out. In fact, the approximation given by the EIM shows the plateau in the curves when M is higher than the reduced basis dimension approximately. It reflects the trade-off between the reduced basis approximation and the EIM approximation contribution to the error: for fixed M the error in the nonlinear function approximation $g(u;\mu)$ will ultimately dominate for large N. Increasing M renders the EIM approximation of the nonlinear term more accurate. To sum up, the error can only be reduced by increasing both N and M.

On the other hand, the approximation given by the RPM shows the monotonic reduction in the error by adding more PGD pairs. Equivalently, the approximation can be enriched by using more reference points avoiding the generation of new PGD pairs. Hence, the error can be reduced by increasing either the number of reference points or the PGD pairs.



(a) *error ϵ of RPM-PGD using 1 point over the space and more points over the parameters domain.*

(b) *error ϵ of EIM-RB using different numbers of interpolation points.*

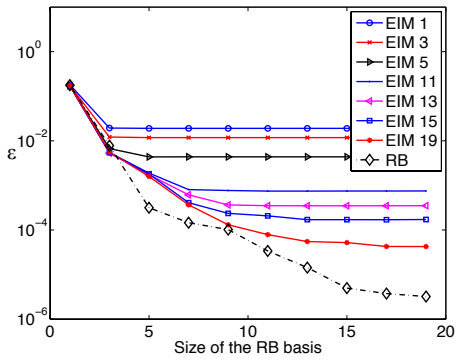
Figure 3.8: The error ϵ (3.56) introduced by the RPM approximation (left) for the solution of the problem (10) compared to the one introduced by the EIM.

As said before, the EIM enables one to decrease the number of operations to integrate and project the problem from $\mathcal{O}(p\ N\ (\beta + k^2))$ to $\mathcal{O}(p\ M\ (\beta + k^2))$, where $M \ll N$. However, the learning stage to construct a ROB can be very expensive, especially for nonlinear problems, for which it is necessary to produce more snapshots to obtain a pertinent ROB for the problem. For Problem 10 defined on a discretized parametric domain $\mathcal{D}_f = \{\underline{\mu}_i\}_{i=1..n_f}$ with $n_f = 15$ over $\mathcal{D}$, the EIM technique imposes to compute and store, in the learning stage, the solutions by a direct solver u_{ref} for a coarse grid $\mathcal{D}_c = 12 \times 12$ over $\mathcal{D}$. It means that the learning stage to construct the ROB represents 64% of the computational cost given by a direct classic analysis.

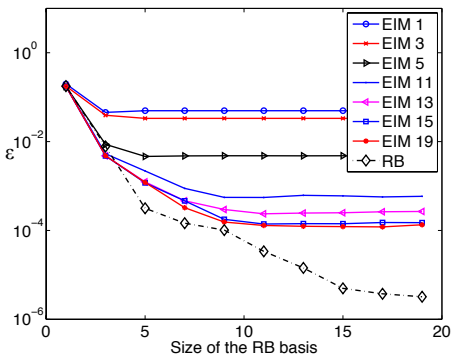
Fig. 3.9 reports the relative error ϵ (3.56) for different coarse grids Σ_c of snapshots with $n_c = 2$, $n_c = 5$ et $n_c = 10$. They are compared with the error obtained considering a $\Sigma_c = 12 \times 12$ coarse grid of snapshots over $\mathcal{D}$.

The lack of snapshots produces a lost in accuracy. For the graph of **Fig. 3.9** *(c)* related to the grid with $n_c = 5$, the lack of snapshots affects the curves related to a number of interpolation points from 13 to 19. In this case, error level reaches a plateau one order of magnitude higher than convergence curve for the grid with $n_c = 12$. **Fig. 3.9** *(d)* shows that a very few number of snapshots leads to level of error higher than $\epsilon = 10^{-2}$, whatever the number M of magic points is.

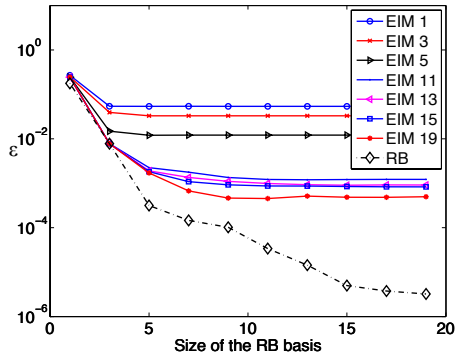
In order to give an idea of the efficiency of the EIM let us consider a level of error ϵ (3.56) equal to 10^{-2}. A 5×5 grid of snapshots is required for the EIM to reach this error level. Then, a nested set of 5 interpolation points ($M = 5$) is determined from this snapshots thanks to a greedy algorithm based on a L_2 norm. Once the learning stage is done, the finer parameters sample is introduced. It is defined by a 15×15 grid over $\mathcal{D}$. Speed up in CPU time compared with a classic FE analysis



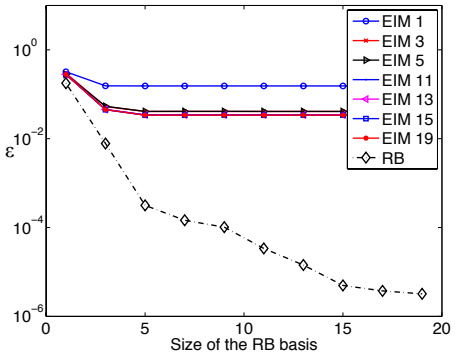
(a) *error ϵ of EIM-RB for a grid $n_c = 12$ of snapshots.*



(b) *error ϵ of EIM-RB for a grid $n_c = 10$ of snapshots.*



(c) *error ϵ of EIM-RB for a grid $n_c = 5$ of snapshots.*



(d) *error ϵ of EIM-RB for a grid $n_c = 2$ of snapshots.*

Figure 3.9: The quality of the EIM approximation depends on the pertinence of the precomputed ROB. Decreasing the number of snapshots makes the error level increase.

is given for the PGD-RPM and the RB-EIM. Results are given for the considered error level ($\epsilon = 10^{-2}$) in **Tab. 3.5**.

Table 3.5: Wall clock to attain an error $\epsilon = 10^{-2}$ for Problem 10. The Gain is relative to the standard Finite Element Analysis

	PGD-RPM 4 parameter points 1 space points	RB-EIM 5 EIM points
size of reduced basis	7	5
Gain	14.2	18

Note that, CPU time for the RB-EIM includes the time spent for the ROB construction during the learning stage. This time represents actually the largest

part of CPU time, the on-line reduced basis generation taking only few seconds for the entire parametric domain $\mathcal{D}$. Thereby, the definition of a ROB that does not evolve during the iterative method represents a strong point of the EIM. In fact, this feature enables one to pre-compute, during the learning stage, the required reduced operators. Thus, during the online stage the independence of the reduced model by the size of the underlying space discretisation is recovered. This strong point may also represent, for some applications, a weak point. Indeed, for multiparametric problems, the EIM needs to solve during the learning stage, the full size problem for some appropriate snapshots. In order to ensure pertinent ROB, the amount of necessary resolutions can drastically increase due to the amount of necessary snapshots for each parameter coordinate.

3.5 Conclusions

The computational framework in which the RPM has been developed is the LaTIn-PGD nonlinear solver. In this chapter, this approach is presented and described on a two-dimensional elliptic and nonlinear parametrized problem (Problem 10). It is a benchmark introduced in [Grepl et al., 2007] as a numerical test for the EIM and then considered again in [Chaturentabut and Sorensen, 2010] in the framework of the DEIM.

The Preliminary step of the LaTIn-PGD algorithm has been detailed. This step is important to generate only the most relevant PGD modes, enabling one to decrease the CPU time and memory consuming. Nevertheless, this step involves the construction of a new reduced-order model. It leads to compute repetitive operations that scale with the underling discretization of the domain and already existing techniques (as EIM) are not well-suited for the LaTIn-PGD computational strategy. This issue is one of the main motivations behind the development of the Reference Points Method.

RPM enables one to decrease the number of operations of the preliminary step by a factor p/m_μ where p is the number of discretized values over the parameter domain and m_μ is the chosen number of reference points along the parameter coordinate. For the considered problem the approximation gives an error of 10^{-3} using just a few reference points. This error decreases either by adding more reference points or by adding more PGD pairs to the ROB.

The second part of the chapter has been dedicated to a comparison between the RPM and the reduced-order model obtained by applying the reduced-basis technique on the same problem. In fact, also for techniques that consider a learning stage to construct a reduced basis, the construction of the reduced-order model necessitates, for each new iteration of the method, the update of the tangent operator and its projection onto the precomputed reduced basis. The number of these operations scales with the underling discretization of the computational domain. For the considered problem (Problem 10), in this chapter, it has been shown that the

reduced-order problems obtained for the two different methods (Preliminary stage for the LaTIn-PGD in (3.17) and reduced-basis in (3.61)) are similar. They are affected by the same computational "bottleneck". The EIM enables one, during the on-line stage, to reduce the number of operations for constructing the ROM by a factor of N/M, where M is the number of interpolating points and N is the size of the underlying FE approximation space. Hence, for the on-line stage the problem is independent of the underlying space discretization.

The quality of the approximations obtained by the two techniques has been compared in **Fig. 3.8** showing a comparable level of error with some differences. In fact the quality of the approximation given by the EIM can be increased only by increasing both the number of interpolation points and the size of the precomputed ROB. On the contrary, the error obtained with the RPM combined with the LaTIn-PGD solver can be not only decreased by adding more reference points but also by adding new PGD pairs if needed in order to automatically enrich the ROB.

Chapter 4

The multiscale nonlinear LaTIn-PGD with RPM approximation

This last chapter is dedicated to the implementation of the RPM in the numerical analysis of structural mechanics. For that purpose, the multiscale LaTIn-based domain decomposition method is considered.

First part of this chapter describes this computational strategy. The coupling of the PGD solver with the LaTIn method is detailed. The second part shows how the RPM is introduced in the LaTIn-PGD computational strategy to approximate the Preliminary step, enabling one to reduce the number of operations to construct the ROM. The chapter ends with a numerical example to assess the efficiency of the RPM within the LaTIn-PGD nonlinear solver.

In structural mechanics, there is a growing interest in a class of techniques called *multiscale computational approaches*, capable to analyse structures in which two or more very different scales can be identified. This is the case of material models described on a scale smaller than that of the macroscopic structural level or when one is interested in the microscale phenomena, such as the comprehension of dislocations propagation. Often, the analysis at the scale of the material is not sufficient because large scale or bulk effects have to be accounted for. In that cases, two models are coupled together: one at the scale of the material (microscale) and one at the scale

of the structure (the macroscale). In order to have a precise description at the microscale and to avoid huge calculations, a natural idea is to try to take advantage of both models. One of the first proposed approaches to address this problem is the theory of homogenization of periodic media (introduced in [Sanchez-Palencia, 1974]) which leads to the resolution of a coarse homogenized problem whose the model is associated with the behaviour of a representative elementary volume. Local properties can be deduced from the homogenised quantities. This theory is the basis of many strategies [Devries et al., 1989, Fish et al., 1997, Oden et al., 1999, Feyel, 2003, Efendiev et al., 2013]. Other strategies have been proposed and one can distinguish a first family of techniques based on a coarse description enriched by the local solution. The microscale at the local solution is described by a finer model [Ibrahimbegovic and Melnyk, 2007], by a different model (e.g. discrete model for atomistic description [Blanc et al., 2005, Legoll, 2009]) or analytically [Hughes et al., 1998].

For a second family of techniques, the strategy starts from the problem at the fine scale, and uses the coarse scale to accelerate the convergence. This is the case, for instance, of domain decomposition methods [Mandel, 1993, Farhat and Roux, 1991, Ladevèze and Dureisseix, 2000] or multigrid methods [Briggs et al., 2000, Fish and Belsky, 1995]. In this work, in order to deal with structural mechanics, the LaTIn method is considered. This iterative method represents the engine of a multiscale computational strategy for structural mechanics based on a mixed domain decomposition method. In the following this is briefly described. The complete details are available in [Ladevèze, 1999, Dureisseix and Ladevèze, 1998, Ladevèze and Nouy, 2003, Ladevèze et al., 2009, Ladevèze et al., 2010]. The RPM is implemented in this computational strategy to deal with structural mechanics.

4.1 The reference problem

For the sake of simplicity, let us consider the quasi-static evolution of a structure defined over the time-space domain $I \times \Omega$ with $\Omega = \mathbb{R}^3$ and $I = (0, T)$, under the hypothesis of small perturbations (for a complete range of behaviour models see [Ladevèze, 1999]). The dependence on parameters is not taken into account in this part, but this computational strategy has already been successfully applied to problems depending on parameters [Heyberger et al., 2011, Relun et al., 2011, Néron et al., 2015]. The volume of this structure is the open domain $\Omega \in \mathbb{R}^3$ with boundary $\partial\Omega$. The structure is submitted to the following boundary conditions (**Fig.4.1**):

- a body force $\underline{f}_d$ in Ω;
- a displacement field $\underline{u}_d$ imposed on a part of the boundaries $\partial_u\Omega \neq \emptyset$ (Dirichlet boundary condition);
- a given quasi-static pressure field $\underline{F}_d(t)$ on the part of the boundaries $\partial_f\Omega$ complementary to $\partial_u\Omega$ ($\partial_u\Omega \cup \partial_f\Omega = \partial\Omega,\ \partial_u\Omega \cap \partial_f\Omega = \emptyset$) (Neumann boundary

condition).

The Hilbert space of functions defined on I, the square of which is integrable on Ω, is denoted by $L^2(I,\Omega)$:

$$L^2(I,\Omega) = \{\underline{u} :\ I \to \Omega,\ \int_I \|\underline{u}(t)\|^2_\Omega \mathrm{d}t\ < \infty\}. \tag{4.1}$$

The Sobolev space of functions defined on I, whose first derivatives have square that is integrable on Ω, is denoted by $H^1(I,\Omega)$:

$$H^1(I,\Omega) = \{\underline{u} \in L^2(I,\Omega) :\ \boldsymbol{\nabla}\underline{u} \in [L^2(I,\Omega)]^3\}. \tag{4.2}$$

$\mathcal{U} = [H^1_0(\Omega)]^3$ is the space of finite energy distributions of displacement defined over Ω:

$$\mathcal{U} \equiv \{\underline{u} \in \left[H^1_0(\Omega)\right]^3\ :\ \underline{u} = 0 \text{ on } \partial_u\Omega\}. \tag{4.3}$$

Let us remember spaces $\mathcal{P} = L^2(\mathcal{D},\mathbb{R})$ and $\mathcal{V} = L^2(\mathcal{D},\mathcal{U})$.

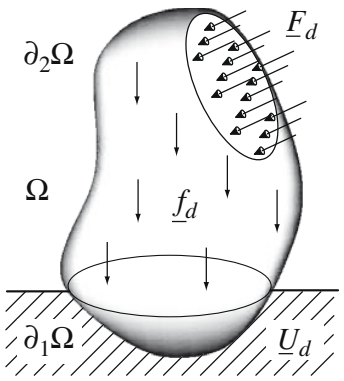


Figure 4.1: The reference problem over the domain Ω

The reference problem can be formulated as below:

Problem 18 (reference problem continuous form) *Find* $\underline{u}(\underline{x},t)$ *and* $\boldsymbol{\sigma}(\underline{x},t)$ *with* $\underline{x} = (x,y,z) \in \Omega$ *and* $t \in I$ *that verifies:*

- *the kinematic assumptions (or compatibility equations):*

$$\underline{u} \in \mathcal{V} \qquad \textit{and} \qquad \boldsymbol{\varepsilon}(\underline{u}) = \frac{1}{2}(\nabla\underline{u} + \nabla^T\underline{u}) \textit{ in } \Omega \tag{4.4}$$

- *the equilibrium equations:*

$$\boldsymbol{\sigma} \in [L^2(I,\Omega)]^6; \qquad \underline{\nabla} \cdot \boldsymbol{\sigma} + \underline{f_d} = \underline{0} \textit{ in } \Omega; \qquad \boldsymbol{\sigma} \cdot \underline{n} = \underline{F_d} \textit{ on } \partial_f\Omega \tag{4.5}$$

- *Let $\boldsymbol{\varepsilon}(\underline{u})$ be the linearized strain tensor associated to the displacement field $\underline{u}$ and $\boldsymbol{\sigma}$ the Cauchy stress tensor. The strain can be divided into an elastic strain $\boldsymbol{\varepsilon}_e$ and an inelastic strain $\boldsymbol{\varepsilon}_p$, so that $\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}_e + \boldsymbol{\varepsilon}_p$. The elastic strain $\boldsymbol{\varepsilon}_e$ is related to the stress through the state law:*

$$\boldsymbol{\sigma} = \mathbf{K}\ \boldsymbol{\varepsilon}_e. \tag{4.6}$$

The inelastic strain $\boldsymbol{\varepsilon}_p$ is related to the stress through the evolution law:

$$\dot{\boldsymbol{\varepsilon}}_p = \mathbf{B}(\boldsymbol{\sigma}). \tag{4.7}$$

- *The homogeneous initial conditions.*

$\mathbf{K}$ represents the Hooke's tensor of the material while $\mathbf{B}$ is a differential operator which can be time-dependent and nonlinear. In the numerical example (section 4.5) a nonlinear time-dependent behaviour is considered:

$$\dot{\boldsymbol{\varepsilon}}_p = \frac{\mathbf{K^{-1}}}{E\ \eta(t)} Tr(\boldsymbol{\sigma})\boldsymbol{\sigma}.$$

η is the viscosity coefficient. To fix the ideas, one can imagine that this viscosity depends on the temperature whose evolution along the time is assumed to be known, which leads to the knowledge of $\eta(t)$. E is the Young modulus of the material. Such a material behaviour (see 4.5) is typical of a viscoelastic constitutive law. Other types of constitutive laws, such as elasto-viscoplastic behaviour, which requires the introduction of internal variables, can be found in [Relun et al., 2011] or [Ladevèze, 1999] expressed by using what is called a "normal formulation". Dealing with such laws does not require any change in the strategy which is developed herein. The Hooke's tensor $\mathbf{K}$ is a linear, symmetric and defined positive operator. It enables one to introduce the following energetic norms with the associated scalar products:

$$\|\cdot\|_{\underline{u},\Omega} = \left[\langle \bullet, \bullet \rangle_{\underline{u},\Omega}\right]^{1/2} \qquad \langle \bullet, \bullet \rangle_{\underline{u},\Omega} = \int_\Omega \boldsymbol{\varepsilon}(\bullet) : \mathbf{K} : \boldsymbol{\varepsilon}(\bullet) \mathrm{d}\Omega, \tag{4.8}$$

$$\|\cdot\|_{\boldsymbol{\sigma},\Omega} = \left[\langle \bullet, \bullet \rangle_{\boldsymbol{\sigma},\Omega}\right]^{1/2} \qquad \langle \bullet, \bullet \rangle_{\boldsymbol{\sigma},\Omega} = \int_\Omega \bullet : \mathbf{K}^{-1} : \bullet \ \mathrm{d}\Omega, \tag{4.9}$$

4.2 The LaTIn multiscale method

In this section, the main characteristics of the LaTIn multiscale method are reviewed. The interested reader can refer to [Ladevèze et al., 2010] for additional details.

4.2.1 Decomposition of the domain

The domain Ω which is occupied by the structure is divided into subdomains and interfaces. Each subdomain Ω_E has its own variables ($\dot{\boldsymbol{\varepsilon}}_{pE}$ and $\boldsymbol{\sigma}_E$) and is subjected,

over its boundary $\partial\Omega_E$, to the action of its environment (the neighboring interfaces) defined by a displacement distribution $\underline{W}_E$ and a force distribution $\underline{F}_E$. Thus, the state of subdomain Ω_E is described by the set $\mathbf{s}_E = (\dot{\boldsymbol{\varepsilon}}_{pE}, \underline{\dot{W}}_E, \boldsymbol{\sigma}_E, \underline{F}_E)$. In order to formulate the reference problem, we need to introduce some functional subspaces. Hereafter, $\square^\star$ will denote the vector space associated with affine space $\square$.

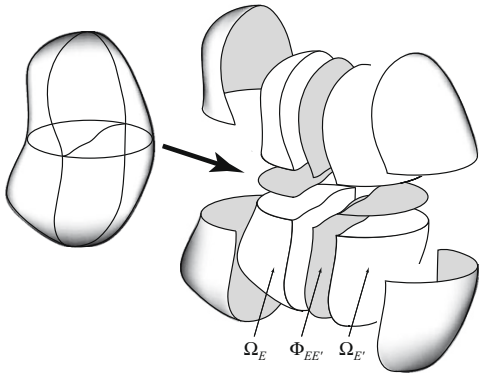


Figure 4.2: Decomposition of the domain

Over boundary $\partial\Omega_E$, displacement $\underline{u}_E$ must be equal to the interface displacement $\underline{W}_E$ and must satisfy the initial condition at $t = 0$:

$$\underline{u}_{E|\partial\Omega_E} = \underline{W}_E \quad \text{and} \quad \underline{u}_{E|t=0} = \underline{u}_{E0} \tag{4.10}$$

The corresponding space of the kinematically admissible displacement fields $(\underline{u}_E, \underline{W}_E)$ is denoted by $\mathcal{U}_E$. Standard techniques (i.e. Lagrange multipliers or penalization) can be considered to apply this boundary conditions for domain Ω_E. Stress $\boldsymbol{\sigma}_E$ must balance interface forces $\underline{F}_E$ over $\partial\Omega_E$ and body forces $\underline{f}_d$ in Ω_E: $\forall\, (\underline{u}^*_E, \underline{W}^*_E) \in \mathcal{U}^*_E$,

$$\int_{\Omega_E \times I} \boldsymbol{\sigma}_E : \boldsymbol{\varepsilon}^*(\underline{\dot{u}}_E)\, d\Omega\, dt = \int_{\Omega_E \times I} \underline{f}_d \cdot \underline{\dot{u}}^*_E\, d\Omega\, dt + \int_{\partial\Omega_E \times I} \underline{F}_E \cdot \underline{\dot{W}}^*_E\, dS\, dt \tag{4.11}$$

Condition (4.11) defines the space $\mathcal{S}_E$ of the statically admissible fields $(\boldsymbol{\sigma}_E, \underline{F}_E)$. Let $\mathcal{S}$ denote the extension of $\mathcal{S}_E$ to the entire problem: $\mathcal{S} = \bigotimes_{\Omega_E \subset \Omega} \mathcal{S}_E$.

Strain $\boldsymbol{\varepsilon}_E$ must derive directly from admissible displacements, $(\underline{u}_E, \underline{W}_E) \in \mathcal{U}_E$:

$$\forall\, (\boldsymbol{\sigma}^*_E, \underline{F}^*_E) \in \mathcal{S}^*_E, \qquad \int_{\Omega_E \times I} \boldsymbol{\sigma}^*_E : \dot{\boldsymbol{\varepsilon}}_E\, d\Omega\, dt = \int_{\partial\Omega_E \times I} \underline{F}^*_E \cdot \underline{\dot{W}}_E\, dS\, dt \tag{4.12}$$

Condition (4.12) defines the space $\mathcal{E}_E$ of the kinematically admissible fields $(\dot{\boldsymbol{\varepsilon}}_E, \underline{\dot{W}}_E)$.

Finally, one introduces the space $\mathbf{A}_{dE}$ of the $\mathbf{s}_E$ in which $(\boldsymbol{\sigma}_E, \underline{F}_E)$ is statically admissible, $\boldsymbol{\sigma}_E$ and $\dot{\boldsymbol{\varepsilon}}_{eE}$ satisfy the state law (4.6) and $(\dot{\boldsymbol{\varepsilon}}_E, \underline{\dot{W}}_E)$ is kinematically admissible. These conditions read:

$$(\boldsymbol{\sigma}_E, \underline{F}_E) \in \mathcal{S}_E, \quad \left(\mathbf{K}^{-1}\dot{\boldsymbol{\sigma}}_E + \dot{\boldsymbol{\varepsilon}}_{Ep}, \underline{\dot{W}}_E\right) \in \mathcal{E}_E \tag{4.13}$$

Concerning the interfaces, in the set $\mathcal{N}_E$ of the neighboring subdomains of Ω_E, let $\Phi_{EE'}$ denote the interface between Ω_E and subdomain $\Omega_{E'}$. This interface is characterized by the restrictions $(\underline{W}_{EE'}, \underline{W}_{E'E})$ of displacement field $(\underline{W}_E, \underline{W}_{E'})$ and $(\underline{F}_{EE'}, \underline{F}_{E'E})$ of force field $(\underline{F}_E, \underline{F}_{E'})$ to $\Phi_{EE'}$. A constitutive relation for every interface $\Phi_{EE'}$ can also be introduced under the form:

$$\mathbf{b}_{EE'}(\underline{\dot{W}}_{EE'}, \underline{\dot{W}}_{E'E}, \underline{F}_{EE'}, \underline{F}_{E'E}) = 0 \tag{4.14}$$

For instance, a perfect interface can be described by $\underline{\dot{W}}_{EE'} = \underline{\dot{W}}_{E'E}$ and $\underline{F}_{EE'} + \underline{F}_{E'E} = \underline{0}$. Other possible types of behaviour (contact, contact with friction ...) can be found in [Ladevèze et al., 2010].

4.2.2 Multiscale description of the interface unknowns

Two different descriptions of the interface unknowns (displacements and forces) are introduced: a macroscale description ($\square^M$) and a microcomplement ($\square^m$). Over an interface $\Phi_{EE'}$ between subdomains Ω_E and $\Omega_{E'}$, one has:

$$\underline{\dot{W}}_{EE'} = \underline{\dot{W}}^M_{EE'} + \underline{\dot{W}}^m_{EE'} \qquad \underline{F}_{EE'} = \underline{F}^M_{EE'} + \underline{F}^m_{EE'} \tag{4.15}$$

The choice of the spaces of the macrodisplacements and macroforces, denoted respectively by $\mathcal{W}^M_{EE'}$ and $\mathcal{F}^M_{EE'}$, is arbitrary, but has a strong influence on the scalability of the method. In practice, in the spatial domain, the macropart is chosen to be the linear part of the forces and displacements, which guarantees the scalability of the domain decomposition method in space, but other possible choices are discussed in [Guidault et al., 2008]. Scalability in time is more difficult to achieve and requires an adaptive generation of the temporal macrobasis. A detailed analysis of this question was presented in [Passieux et al., 2010]. Here, the separation of the macroscale from the microscale is carried out only in space, which means that the temporal microscale and macroscale are identical.

Once spaces $\mathcal{W}^M_{EE'}$ and $\mathcal{F}^M_{EE'}$ have been defined, the macropart of the displacement field can be defined as:

$$\forall \underline{F}^* \in \mathcal{F}^M_{EE'}, \qquad \int_{\Phi_{EE'} \times I} \left(\underline{\dot{W}}^M_{EE'} - \underline{\dot{W}}_{EE'} \right) \cdot \underline{F}^* dS dt = 0 \tag{4.16}$$

and the macropart of the force field can be defined as:

$$\forall \underline{\dot{W}}^* \in \mathcal{W}^M_{EE'}, \qquad \int_{\Phi_{EE'} \times I} \left(\underline{F}^M_{EE'} - \underline{F}_{EE'} \right) \cdot \underline{\dot{W}}^* dS dt = 0 \tag{4.17}$$

The spaces of the interface quantities can be extended to the interfaces $\mathcal{N}_E$ of subdomain Ω_E as: $\mathcal{W}^M_E = \bigotimes_{E' \in \mathcal{N}_E} \mathcal{W}^M_{EE'}$. $\mathcal{F}^M_E = \bigotimes_{E' \in \mathcal{N}_E} \mathcal{F}^M_{EE'}$. The extension to the whole set of the subdomains of Ω leads to $\mathcal{W}^M$ and $\mathcal{F}^M$. The micropart of the displacements and forces can be derived from (4.15).

Over the interfaces, the macroforces are assumed to verify the transmission conditions:

$$\underline{F}^M_{EE'} + \underline{F}^M_{E'E} = \underline{0} \quad \text{over } \Phi_{EE'} \tag{4.18}$$

$$\underline{F}^M_{E} + \underline{F}^M_{d} = \underline{0} \quad \text{over } \partial\Omega_E \cap \partial_f\Omega \tag{4.19}$$

This assumption is called the "admissibility of the macroquantities".

The corresponding subspace is defined as $\mathcal{F}^M_{ad}$. Now let us introduce the space $\mathcal{W}^M_{ad}$ of the displacements which are continuous at the interfaces and equal to the macropart of prescribed displacement $\underline{u}_d$ over $\partial_u\Omega$. The spaces whose elements have their macroparts in $\mathcal{W}^M_{ad}$ and $\mathcal{F}^M_{ad}$ are denoted $\mathcal{W}_{ad}$ and $\mathcal{F}_{ad}$.

4.2.3 The LaTIn solver

The governing equations of the problem are divided into two groups: the linear equations and the local equations. First, let $\mathbf{A}_d$ be the space of the solutions of the linear equations (defined globally in the spatial domain) whose elements $\mathbf{s} = (\mathbf{s}_E)_{\Omega_E \subset \Omega}$ verify:

- the admissibility condition (4.13) for every $\Omega_E \subset \Omega$: $\mathbf{s}_E \in \mathbf{A}_{dE}$
- the admissibility of the macroforces (4.18,4.19)

Then, let $\mathbf{\Gamma}$ be the space of the solutions $\hat{\mathbf{s}} = (\mathbf{s}_E)_{\Omega_E \subset \Omega}$ of the local (possibly nonlinear) equations whose elements satisfy:

- the evolution law (4.7) for every $\Omega_E \subset \Omega$: $\forall (t, \underline{x}) \in I \times \Omega_E,\ \dot{\boldsymbol{\varepsilon}}_{pE} = \mathbf{B}(\boldsymbol{\sigma}_E)$
- the interface behaviour (4.14) over every interface $\Phi_{EE'}$

Solutions of both the spaces verify the state law ((4.6)) and the initial condition. The solution of the problem $\mathbf{s}_{ref}$ is defined as:

$$\mathbf{s}_{ref} \in \mathbf{A}_d \cap \mathbf{\Gamma} \tag{4.20}$$

The LaTIn method used for the resolution of Problem (4.20) is an iterative scheme between two stages called the "local stage" and the "linear stage". In the local stage, one seeks a solution of the local nonlinear equations in space $\mathbf{\Gamma}$. In the linear stage, one seeks a solution of the global linear equations in space $\mathbf{A}_d$. The linear stage and the local stage are repeated alternatively until convergence. More precisely, two approximations are calculated at iteration $n+1$: $\hat{\mathbf{s}}^{(n+1/2)} \in \mathbf{\Gamma}$ and $\mathbf{s}^{(n+1)} \in \mathbf{A}_d$. The introduction of search directions enables one to switch back and forth between space $\mathbf{\Gamma}$ and space $\mathbf{A}_d$.

4.2.4 The local stage

At iteration $n+1$, given $\mathbf{s}^{(n)} \in \mathbf{A}_d$, the local stage leads to an approximation of the solution $\hat{\mathbf{s}}^{(n+1/2)}$ in space $\boldsymbol{\Gamma}$ such that $(\hat{\mathbf{s}}^{(n+1/2)} - \mathbf{s}^{(n)})$ belongs to search direction $\mathbf{E}^+$. Skipping the subscripts for the sake of clarity, in each subdomain $\Omega_E \subset \Omega$, $\mathbf{E}^+$ is defined by:

$$\mathbf{H}(\hat{\boldsymbol{\sigma}}_E - \boldsymbol{\sigma}_E) + (\hat{\dot{\boldsymbol{\varepsilon}}}_{pE} - \dot{\boldsymbol{\varepsilon}}_{pE}) = 0 \tag{4.21}$$

$$\mathbf{h}(\underline{\hat{F}}_E - \underline{F}_E) - (\underline{\hat{\dot{W}}}_E - \underline{\dot{W}}_E) = 0 \tag{4.22}$$

where $\mathbf{H}$ and $\mathbf{h}$ are symmetric, positive definite operators which have an influence on the convergence rate of the algorithm, but do not affect the solution. The solution $\hat{\mathbf{s}}$ of the local stage is the "intersection" of spaces $\mathbf{E}^+$ and $\boldsymbol{\Gamma}$. This leads to a set of equations which are defined locally in space.

4.2.5 The linear stage

The linear stage consists, given the solution $\hat{\mathbf{s}}^{(n+1/2)} \in \boldsymbol{\Gamma}$ of the local stage, in finding a solution $\mathbf{s}^{(n+1)} \in \mathbf{A}_d$ such that $(\mathbf{s}^{(n+1)} - \hat{\mathbf{s}}^{(n+1/2)})$ belongs to search direction $\mathbf{E}^-$. Skipping the subscripts, $\mathbf{E}^-$ is defined by:

$$\forall \boldsymbol{\sigma}^* \in \mathcal{S}, \quad \sum_{\Omega_E \subset \Omega} \int_{\Omega_E \times I} \Big(\mathbf{H}\,(\boldsymbol{\sigma}_E - \hat{\boldsymbol{\sigma}}_E) - (\dot{\boldsymbol{\varepsilon}}_{pE} - \hat{\dot{\boldsymbol{\varepsilon}}}_{pE}) \Big) : \boldsymbol{\sigma}^*\, d\Omega\, dt = 0 \tag{4.23}$$

$$\forall \underline{F}^* \in \mathcal{F}_{ad}, \quad \sum_{\Omega_E \subset \Omega} \int_{\partial\Omega_E \times I} \Big(\mathbf{h}(\underline{F}_E - \underline{\hat{F}}_E) + (\underline{\dot{W}}_E - \underline{\hat{\dot{W}}}_E) \Big) \cdot \underline{F}^*\, dS\, dt = 0 \tag{4.24}$$

which is written in a weak sense in order to take into account the admissibility of the macroforces (4.18,4.19).

In [Ladevèze, 1999], it was shown that an optimal choice of $\mathbf{H}$ in terms of the convergence rate consists in defining the manifold $\mathbf{E}^-$ as the vector space which is tangent to $\boldsymbol{\Gamma}$ in $\hat{\mathbf{s}}$. This requires the calculation of $\mathbf{H} = \partial\mathbf{B}/\partial\sigma|_{\hat{\sigma}}$ at each iteration (or, at least, every few iterations) as in a classical Newton algorithm. $\mathbf{h}$ can be viewed as a "viscosity" effect at the interface. The choice of this parameter is discussed in [Ladevèze et al., 2010].

The admissibility of the macroforces $\underline{F}^* \in \mathcal{F}_{ad}$ in (4.24), which expresses that macropart $\underline{F}^{M*}$ belongs to $\mathcal{F}^M_{ad}$, is enforced by introducing a Lagrange multiplier $\underline{\dot{\hat{W}}}^M_E \in \mathcal{W}^M_{ad}$. Then, (4.24) is defined locally in each subdomain Ω_E as:

$$\forall \underline{F}^* \in \mathcal{F} \quad \int_{\partial\Omega_E \times I} \Big(\mathbf{h}(\underline{F}_E - \underline{\hat{F}}_E) + (\underline{\dot{W}}_E - \underline{\hat{\dot{W}}}_E) - \underline{\dot{\hat{W}}}^M_E \Big) \cdot \underline{F}^*\, dS\, dt = 0 \tag{4.25}$$

Conversely, the problem of the admissibility of the macroquantities is defined on the

global interface level by:

$$\forall \dot{\underline{\tilde{W}}}_E^{M*} \in \mathcal{W}_{ad}^{M*} \quad \sum_{\Omega_E \subset \Omega} \int_{\partial\Omega_E \times I} \dot{\underline{\tilde{W}}}_E^{M*} \cdot \underline{F}_E \, dS \, dt - \int_{\partial\Omega_E \cap \partial_f\Omega \times I} \dot{\underline{\tilde{W}}}_E^{M*} \cdot \underline{F}_d \, dS \, dt = 0 \tag{4.26}$$

Moreover, kinematic admissibility and state law lead to the following condition for each subdomain:

$$\forall \, (\boldsymbol{\sigma}_E^*, \underline{F}_E^*) \in \mathcal{S}_E^* \quad \int_{\Omega_E \times I} \left(\mathbf{K}^{-1} \dot{\boldsymbol{\sigma}}_E + \dot{\boldsymbol{\varepsilon}}_{pE} \right) : \boldsymbol{\sigma}_E^* \, d\Omega \, dt = \int_{\partial\Omega_E \times I} \dot{\underline{W}}_E \cdot \underline{F}_E^* \, dS \, dt \tag{4.27}$$

The microproblem Introducing the weak form of search direction (4.23,4.25) into Equation (4.27), one gets the following microproblem:

Problem 19 (microproblem) *Find* $(\boldsymbol{\sigma}_E, \underline{F}_E) \in \mathcal{S}_E$ *such that:*

$$\begin{aligned} \forall \, (\boldsymbol{\sigma}_E^*, \underline{F}_E^*) \in \mathcal{S}_E^*, \; & \int_{\Omega_E \times I} \left(\mathbf{K}^{-1} \dot{\boldsymbol{\sigma}}_E + \mathbf{H} \boldsymbol{\sigma}_E \right) : \boldsymbol{\sigma}_E^* \, d\Omega \, dt + \int_{\partial\Omega_E \times I} \mathbf{h} \underline{F}_E \cdot \underline{F}_E^* \, dS \, dt = \\ & \int_{\Omega_E \times I} \left(\mathbf{H} \hat{\boldsymbol{\sigma}}_E - \hat{\dot{\boldsymbol{\varepsilon}}}_{pE} \right) : \boldsymbol{\sigma}_E^* \, d\Omega \, dt + \int_{\partial\Omega_E \times I} \left(\mathbf{h} \hat{\underline{F}}_E + \hat{\underline{\dot{W}}}_E + \dot{\underline{\tilde{W}}}_E^M \right) \cdot \underline{F}_E^* \, dS \, dt \end{aligned} \tag{4.28}$$

Problem 19 is linear and depends on the hat quantities $(\hat{\mathbf{s}}_E)$, which are known from the previous local stage, and on Lagrange multiplier $\dot{\underline{\tilde{W}}}_E^M$, which is a new unknown of the problem. Therefore, the solution can be divided into two parts:

$$\mathbf{s}_E = \hat{\mathbf{s}}_E + \bar{\bar{\mathbf{s}}}_E(\dot{\underline{\tilde{W}}}_E^M) \tag{4.29}$$

where $\hat{\mathbf{s}}_E$ is the solution of Problem 19 when $\dot{\underline{\tilde{W}}}_E^M$ is equal to zero, and $\bar{\bar{\mathbf{s}}}_E(\dot{\underline{\tilde{W}}}_E^M)$ depends linearly on Lagrange multiplier $\dot{\underline{\tilde{W}}}_E^M$ through a linear operator. Relation (4.29) can also be written in terms of the macroforces:

$$\underline{F}_E^M = \hat{\underline{F}}_E^M + \bar{\bar{\underline{F}}}_E^M \qquad \text{with} \qquad \bar{\bar{\underline{F}}}_E^M = \mathbf{L}_E^F(\dot{\underline{\tilde{W}}}_E^M) \tag{4.30}$$

where $\hat{\underline{F}}_E^M$ represents the macroforces corresponding to $\hat{\mathbf{s}}_E$, and $\mathbf{L}_E^F$ is a homogenized operator defined over the space-time domain $\Omega_E \times I$. The details of the calculation of the homogenized operator in both the linear case and the nonlinear case will be given in section 4.2.7.

The macroproblem In order to obtain the value of Lagrange multiplier $\underline{\dot{\tilde{W}}}_E^M \in \mathcal{W}_{ad}^M$, homogenized operator (4.30) is introduced into the admissibility condition of the macroquantities (4.26), leading to a problem defined on the macroscale: $\forall \underline{\dot{\tilde{W}}}_E^{M*} \in \mathcal{W}_{ad}^{M*}$,

$$\sum_{\Omega_E \subset \Omega} \int_{\partial\Omega_E \times I} \underline{\dot{\tilde{W}}}_E^{M*} \cdot \left(\mathbf{L}_E^F(\underline{\dot{\tilde{W}}}_E^M) + \underline{\hat{\tilde{F}}}_E^M \right) dS\, dt = \sum_{\Omega_E \subset \Omega} \int_{\partial\Omega_E \cap \partial_f\Omega} \underline{\dot{\tilde{W}}}_E^{M*} \cdot \underline{F}_d \, d\mathbf{\Gamma}\, dt \tag{4.31}$$

This macroproblem boils down to a linear homogenized problem in space and time defined over the whole set of interfaces and the entire time interval.

Resolution of the linear stage The linear stage consists in solving a series of microproblems (Problem 19) defined in each time-space subdomain $I \times \Omega_E$ along with macroproblem (4.31) defined over the entire time interval I and for the whole set of interfaces, leading to Lagrange multiplier $\underline{\dot{\tilde{W}}}_E^M$. One can note that parallel resolution of these microproblems is possible, which reduces the computation cost.

4.2.6 Convergence of the algorithm

Under the assumption of perfect interfaces and a monotonous operator $\mathbf{B}$ for the description of the constitutive law, the algorithm presented above converges toward the reference solution $\mathbf{s}_{ref}$ (see [Ladevèze, 1999] for proof of the convergence). In order to check the convergence of the iterative scheme, one introduces the following error indicator:

$$\eta = \frac{\|\mathbf{s}^{(n+1)} - \hat{\mathbf{s}}^{(n+1/2)}\|}{\frac{1}{2}\|\mathbf{s}^{(n+1)} + \hat{\mathbf{s}}^{(n+1/2)}\|} \tag{4.32}$$

where the norm is defined as:

$$\begin{aligned} \|\mathbf{s}\|^2 = \sum_{\Omega_E \subset \Omega} \int_{\Omega_E \times I} & (\boldsymbol{\sigma}_E : \mathbf{H}\boldsymbol{\sigma}_E + \dot{\boldsymbol{\varepsilon}}_{pE} : \mathbf{H}^{-1}\dot{\boldsymbol{\varepsilon}}_{pE})\, d\Omega\, dt \\ & + \int_{\partial\Omega_E \times I} (\underline{F}_E \cdot \mathbf{h}\underline{F}_E + \underline{\dot{W}}_E \cdot \mathbf{h}^{-1}\underline{\dot{W}}_E)\, dS\, dt \end{aligned} \tag{4.33}$$

Let us note that if the reference solution $\mathbf{s}_{ref}$ is available one can also introduce the actual relative error η_{ref}:

$$\eta_{ref} = \frac{\|\mathbf{s}^{(n+1)} - \mathbf{s}_{ref}\|}{\|\mathbf{s}_{ref}\|} \tag{4.34}$$

4.2.7 Construction of the homogenized operator

In (4.30), the homogenized operator $\mathbf{L}_E$ of the subdomain Ω_E maps $\underline{\dot{\hat{W}}}_E^M$ to $\underline{F}_E^M$ through the relation:

$$\underline{F}_E^M = \mathbf{L}_E(\underline{\dot{\hat{W}}}_E) + \hat{\hat{F}}_E, \tag{4.35}$$

Classically in the multiscale LaTIn method [Ladevèze and Nouy, 2003, Ladevèze et al., 2010], $\mathbf{L}_E$ is obtained by solving the following microproblem:

$$\begin{aligned}&\forall(\boldsymbol{\sigma}^\star, \underline{F}^\star) \in \mathcal{F}^\star,\\ &\int_{I\times\Omega_E} \boldsymbol{\sigma}^\star : (\mathbf{K^{-1}}\dot{\hat{\boldsymbol{\sigma}}} + \mathbf{H}^-\boldsymbol{\sigma})\,\mathrm{d}\Omega\,\mathrm{d}t + \int_{I\times\partial\Omega_E} \underline{F}^\star\cdot\mathbf{h}^-\underline{F}\,\mathrm{d}S\,\mathrm{d}t = \int_{I\times\partial\Omega_E} \underline{F}^\star\cdot\underline{\dot{\hat{W}}}_E^M\,\mathrm{d}S\,\mathrm{d}t.\end{aligned} \tag{4.36}$$

which corresponds to Problem 19 in which the hat quantities ($\hat{\mathbf{s}}_E$) have been zeroed. In this thesis the separation of the macroscale from the microscale is carried out only in space, but a coarse discretisation of $I = [0, T]$ has already been introduced in [Passieux et al., 2010]. The Lagrange multiplier $\underline{\dot{\hat{W}}}_E^M$ becomes:

$$\underline{\dot{\hat{W}}}_E^M(\underline{x}, t) = \sum_{i=1}^{d} \dot{\alpha}_i(t) e_i^M(\underline{x}) \tag{4.37}$$

where $\{e_i^M(\underline{x})\}_{i=1,\dots,d}$ is the spatial macrobasis for each of the $\mathcal{N}_E$ interfaces of the subdomain (**Fig.4.3**). A classical choice is a macrobasis which extracts the linear part of the quantities. Other choices can be found in [Guidault et al., 2008]. Time functions $\{\alpha_i(t)\}_{i=1,\dots,d}$ are unknowns during the construction of the homogenized operator.

Each of these time functions can be written using, for example, a finite element basis with shape functions $\{\phi^j(t)\}_{j=1,\dots,p+1}$ (where p is the number of time steps in the discretisation of I). Hence, $\alpha_i(t) = \sum_{j=1}^{p+1} \phi^j(t) c_i^j$. And (4.37) becomes:

$$\underline{\dot{\hat{W}}}_E^M(\underline{x}, t) = \sum_{i=1}^{d} \dot{\alpha}_i(t) e_i^M(\underline{x}) = \sum_{i=1}^{d}\sum_{j=1}^{p+1} c_i^j \left(\dot{\phi}^j(t) e_i^M(\underline{x})\right) \tag{4.38}$$

which can be rewritten as:

$$\underline{\dot{\hat{W}}}_E^M(\underline{x}, t) = \sum_{i=1}^{d}\sum_{j=1}^{p+1} c_i^j \underline{w}_i^j(\underline{x}, t) \tag{4.39}$$

where the $d(p+1)$ vectors $\underline{w}_i^j(\underline{x}, t) = \dot{\phi}^j(t)\underline{e}_i^M(\underline{x})$ represent the time-space macrobasis associated to each of the r interfaces of the problem. The coefficients c_i^j are unknowns

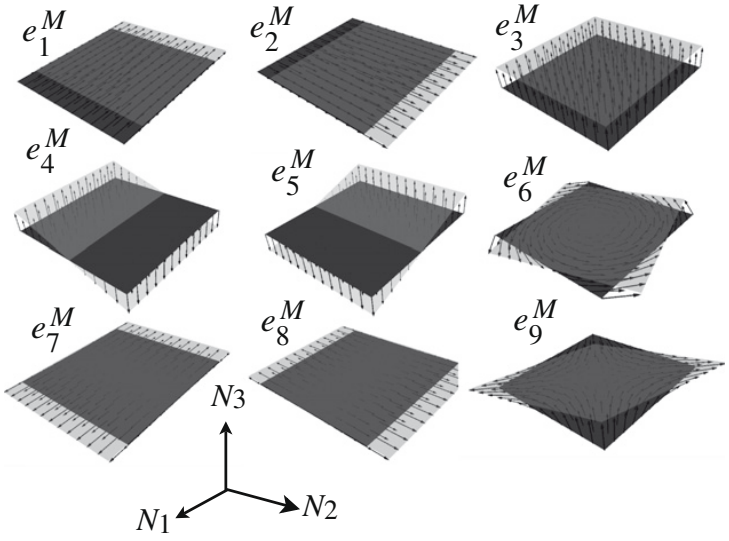


Figure 4.3: Linear macrobasis for a plane interface.

at this stage. They are determined by solving the macroproblem (4.31), once the homogenized operator has been constructed.

The homogenized operator is obtained by solving $rd(p+1)$ microproblems (see 4.36) in which $\dot{\underline{W}}_E^M$ is taken to be equal to $\underline{w}_i^j(\underline{x},t)$. The construction of the homogenized operator can represent a big amount of the entire computational cost. It is particularly evident when the number of subdomains increases or when search directions change. Recently, in [Cremonesi et al., 2013], a new technique to construct the homogenized operator has been developed. The computation cost is reduced by solving the $rd(p+1)$ microproblems (4.36) thanks to an extension of the PGD representation to the variable which defines the Lagrange multiplier. This technique has not been implemented in the current work.

If $\mathbf{H}$ is the tangent search direction (*i.e.* $\mathbf{H} = \partial\mathbf{B}/\partial\boldsymbol{\sigma}|_{\hat{\boldsymbol{\sigma}}}$), a physical interpretation can be given to the homogenized operator. Let us recall that homogenized operator $\mathbf{L}_E$ maps Lagrange multiplier $\dot{\tilde{\underline{W}}}_E^M$ to macroforce $\underline{F}_E^M = \mathbf{L}_E \dot{\tilde{\underline{W}}}_E^M$ in subdomain Ω_E, so that:

$$\underline{F}_E^M = \mathbf{L}_E(\dot{\underline{W}}_E) + \hat{\dot{F}}_E, \tag{4.40}$$

This relation can be rewritten in terms of macroforce $\underline{F}_E^M$ and macrodisplacement $\underline{W}_E^M$ at the interface, leading to a linear relation between the small perturbations of these quantities in the form:

$$\Delta\underline{F}_E^M = \mathbf{L}_E^W \Delta\dot{\underline{W}}_E \tag{4.41}$$

Once the algorithm has converged, (4.41) can be viewed as the local macrobehaviour near the calculated solution, defined over $I = [0, T]$ and in each subdomain Ω_E. This behaviour establishes a relation between a small perturbation in terms of macrodisplacements and a small perturbation in terms of macroforces. In the particular case of a linear problems, such as in viscoelasticity, the tangent search direction is independent of the solution being sought. Therefore, it is calculated only once at the beginning of the algorithm and then remains constant throughout the iterations. Thus, the previous interpretation is valid throughout the iterations.

If search direction $\mathbf{H}$ is not tangent to subspace $\mathbf{\Gamma}$, the algorithm bears some similarities to a quasi-Newton scheme and no obvious physical interpretation can be given to the homogenized operator.

4.3 Proper generalized decomposition for the solution of the microproblems

In order to perform the linear stage at the iteration $n+1$, a bunch of microproblems, on each subdomain, has to be solved. The LaTIn-PGD provides a first approximation of these problems by the Preliminary step: each microproblem is projected on the ROB that arises from the previous iteration n. Hence, the solutions of the microproblems are approximated in the linear span of the already existing ROB. If this approximation is not enough accurate, each subdomain can eventually generate a new PGD pair by the Algorithm 5. That strategy can involve the sharing of the ROB among subdomains derived by similar cells.

Reformulation of the microproblem At the iteration $n+1$ the linear stage consists in building $\mathbf{s}^{(n+1)} \in \mathbf{A}_d$ knowing $\hat{\mathbf{s}}^{(n+1/2)} \in \mathbf{\Gamma}$ and verifying search direction $\mathbf{E}^-$ (see (4.23) and (4.25)).

For the subdomain E the search direction is given by the following system of equations:

$$\left\{\begin{array}{rcl} (\dot{\boldsymbol{\varepsilon}}_p^{(n+1)} - \dot{\hat{\boldsymbol{\varepsilon}}}_p^{(n+1/2)}) - (\boldsymbol{\sigma}^{(n+1)} - \hat{\boldsymbol{\sigma}}^{(n+1/2)})\mathbf{H} & = & 0, \\ (\dot{\underline{W}}^{(n+1)} - \dot{\hat{\underline{W}}}^{(n+1/2)}) + \mathbf{h}((\underline{F}^{(n+1)} - \hat{\underline{F}}^{(n+1/2)}) - \dot{\tilde{\underline{W}}}^{M}) & = & 0. \end{array}\right. \tag{4.42}$$

Let us introduce correction of the solution:

$$\Delta\mathbf{s} = \mathbf{s}^{(n+1)} - \mathbf{s}^{(n)}. \tag{4.43}$$

The previous system of equations, given by (4.42), reads:

$$\left\{\begin{array}{rcl} (\Delta\dot{\boldsymbol{\varepsilon}}_p^{(n+1)} - \dot{\bar{\boldsymbol{\varepsilon}}}) - (\Delta\boldsymbol{\sigma}^{(n+1)} - \bar{\boldsymbol{\sigma}})\mathbf{H} & = & 0, \\ (\Delta\dot{\underline{W}}^{(n+1)} - \dot{\bar{\underline{W}}}) + \mathbf{h}((\Delta\underline{F}^{(n+1)} - \bar{\underline{F}})) & = & 0. \end{array}\right. \tag{4.44}$$

where $\bar{\mathbf{s}}$ are known quantities at this stage:

$$\bar{\mathbf{s}} = \left((\dot{\hat{\boldsymbol{\varepsilon}}}_p^{(n+1/2)} - \dot{\boldsymbol{\varepsilon}}_p^{(n)}), (\dot{\hat{\underline{W}}}^{(n+1/2)} - \dot{\underline{W}}^{(n)} + \dot{\tilde{\underline{W}}}^{M}), (\hat{\boldsymbol{\sigma}}^{(n+1/2)} - \boldsymbol{\sigma}^{(n)}), (\hat{\underline{F}}^{(n+1/2)} - \underline{F}^{(n)}) \right)$$
$$= \hat{\mathbf{s}}^{(n+1/2)} - \mathbf{s}^{(n)} + \left(0, \dot{\tilde{\underline{W}}}^{M}, 0, 0 \right). \tag{4.45}$$

The PGD approximation is sought by approximating the search directions (4.44) rewritten in terms of a constitutive relation error. Hence, for the subdomain E, the microproblem becomes:

Problem 20 *Seek $\Delta\mathbf{s} \in \mathbf{A}_d$ minimising the error in the constitutive behaviour related to the search direction:*

$$e^2_{CB}(\Delta\mathbf{s}) = \|(\Delta\boldsymbol{\sigma}^{(n+1)} - \bar{\boldsymbol{\sigma}})\mathbf{H} - (\Delta\dot{\boldsymbol{\varepsilon}}_p^{(n+1)} - \dot{\bar{\boldsymbol{\varepsilon}}})\|^2_\Omega + \|(\Delta\dot{\underline{W}}^{(n+1)} - \dot{\bar{\underline{W}}}) + \mathbf{h}((\Delta\underline{F}^{(n+1)} - \bar{\underline{F}}))\|^2_{\partial\Omega}, \tag{4.46}$$

where

$$\|\circ\|^2_\Omega = \int_{\Omega_E \times I} \circ \cdot \mathbf{M}(t) \circ \mathrm{d}\Omega\mathrm{d}t \quad \textit{and} \quad \|\circ\|^2_{\partial\Omega} = \int_{\partial\Omega_E \times I} \circ \cdot \mathbf{m}(t) \circ \mathrm{d}\Omega\mathrm{d}t. \tag{4.47}$$

The error in constitutive behaviour constructed is made of two norms: a norm related to the quantities in the subdomain $\|\circ\|_\Omega$ and a norm related to the quantities on the interfaces $\|\circ\|_{\partial\Omega}$. In practice, a good choice for $\mathbf{M}(t)$ and $\mathbf{m}(t)$ is to take the symmetric part of the search direction operators:

$$\mathbf{M}(t) = \mathbf{H}^-{}_{symm}(t) = \mathbf{H}^-(t) \tag{4.48}$$
$$\mathbf{m}(t) = \mathbf{h}^-{}_{symm}(t) = \mathbf{h}^-(t) \tag{4.49}$$

Separated variable description of the unknowns PGD is introduced in Problem 20 to approximate primal variables as follows:

$$\Delta\boldsymbol{\varepsilon}_p^{(n+1)}(\underline{x}, t) \approx \Delta\tilde{\boldsymbol{\varepsilon}}_{p,k}^{(n+1)}(\underline{x}, t) = \sum_{i=1}^{k} \alpha_i^u(t)\ \mathbf{E}_i(\underline{x}) \tag{4.50}$$

$$\Delta\underline{W}^{(n+1)} \approx \Delta\tilde{\underline{W}}_k^{(n+1)}(\underline{x}, t) = \sum_{i=1}^{k} \alpha_i^W(t)\ \underline{W}_i(\underline{x}) \tag{4.51}$$

where $\{\alpha_i(t)\}_{1\le i\le k} \in \mathcal{P} \equiv L^2(I, \mathbb{R})$, $\{\mathbf{E}_i(\underline{x})\}_{1\le i\le k} \in \mathcal{U}$, and $\{W_i(\underline{x})\}_{1\le i\le k}$ corresponds, on the bound of the subdomain, to the trace of displacement field, which is linearly related to $\boldsymbol{\varepsilon}_p$ by (4.4). Hence, the functions depending on time $\alpha_i^u(t)$ and $\alpha_i^W(t)$ are set to be equal $\alpha_i^u(t) = \alpha_i^W(t) = \alpha_i(t)$.

The dual variables $\Delta\boldsymbol{\sigma}^{(n+1)}$ and $\Delta\underline{F}^{(n+1)}$ are deduced as:

$$\Delta\boldsymbol{\sigma}^{(n+1)}(\underline{x},t) \approx \Delta\tilde{\boldsymbol{\sigma}}_k^{(n+1)}(\underline{x},t) = \sum_{i=1}^{k} \alpha_i^u(t)\ \mathbf{C}_i(\underline{x}) \tag{4.52}$$

$$\Delta\underline{F}^{(n+1)}(\underline{x},t) \approx \Delta\tilde{\underline{F}}_k^{(n+1)}(\underline{x},t) = \sum_{i=1}^{k} \alpha_i^W(t)\ \mathbf{C}_i(\underline{x})\cdot\underline{n}|_\Phi \tag{4.53}$$

with $\mathbf{C}_i(\underline{x}) = \mathbf{K}\ \mathbf{E}_i(\underline{x})$ the space function related to the dual variables.

The best PGD approximation is the one that minimize the error in Problem 20:

Problem 21 *Seek $\Delta\mathbf{s} \in \mathbf{A}_d$ minimising the error in the constitutive behaviour related to the search direction:*

$$e^2_{CB}(\Delta\tilde{\mathbf{s}}-\bar{\mathbf{s}}) = \|(\Delta\tilde{\boldsymbol{\sigma}}_k^{(n+1)}-\bar{\boldsymbol{\sigma}})\mathbf{H}-(\Delta\dot{\tilde{\boldsymbol{\varepsilon}}}_{p\,k}^{(n+1)}-\dot{\bar{\boldsymbol{\varepsilon}}})\|^2_\Omega+\|(\Delta\dot{\underline{\tilde{W}}}_k^{(n+1)}-\dot{\underline{W}})+\mathbf{h}(\Delta\tilde{\underline{F}}_k^{(n+1)}-\bar{\underline{F}})\|^2_{\partial\Omega}. \tag{4.54}$$

with

$$\Delta\tilde{\boldsymbol{\varepsilon}}_{p,k}^{(n+1)}(\underline{x},t) = \sum_{i=1}^{k} \alpha_i^u(t)\ \mathbf{E}_i(\underline{x}) \tag{4.55}$$

$$\Delta\tilde{\underline{W}}_k^{(n+1)}(\underline{x},t) = \sum_{i=1}^{k} \alpha_i^W(t)\ \underline{W}_i(\underline{x}) \tag{4.56}$$

$$\Delta\tilde{\boldsymbol{\sigma}}_k^{(n+1)}(\underline{x},t) = \sum_{i=1}^{k} \alpha_i^u(t)\ \mathbf{C}_i(\underline{x}) \tag{4.57}$$

$$\Delta\tilde{\underline{F}}_k^{(n+1)}(\underline{x},t) = \sum_{i=1}^{k} \alpha_i^W(t)\ \mathbf{C}_i(\underline{x})\cdot\underline{n}|_\Phi \tag{4.58}$$

with $\alpha_i^u = \alpha_i^W = \alpha_i$

First order approximation and k-th order approximation The best approximation of first order ($k = 1$) is defined as the solution of the minimization problem related to the error $e^2_{CB}(\Delta\mathbf{s})$ with the following description of the unknowns:

$$\Delta\tilde{\boldsymbol{\varepsilon}}_p^{(n+1)}(\underline{x},t) = \alpha(t)\ \mathbf{E}(\underline{x}) \tag{4.59}$$

$$\Delta\tilde{\underline{W}}^{(n+1)}(\underline{x},t) = \alpha(t)\ \underline{W}(\underline{x}) \tag{4.60}$$

$$\Delta\tilde{\boldsymbol{\sigma}}^{(n+1)}(\underline{x},t) = \alpha(t)\ \mathbf{C}(\underline{x}) \tag{4.61}$$

$$\Delta\tilde{\underline{F}}^{(n+1)}(\underline{x},t) = \alpha(t)\ \mathbf{C}(\underline{x})\cdot\underline{n}|_{\partial\Omega} \tag{4.62}$$

Generation of the pair of PGD functions is given by minimization of Problem 21. It reads:

$$(\alpha(t), \mathbf{E}(\underline{x}), \mathbf{C}(\underline{x}), \underline{W}(\underline{x})) = \arg \min_{\alpha\in\mathcal{P},(\mathbf{E}(\underline{x}),\mathbf{C}(\underline{x}),\underline{W}(\underline{x}))\in\mathcal{U}} \dots$$
$$\dots \|(\alpha\ \mathbf{C}\ -\ \bar{\boldsymbol{\sigma}})\mathbf{H} - (\alpha\ \mathbf{E}\ -\ \dot{\bar{\boldsymbol{\varepsilon}}})\|^2_\Omega + \|(\alpha\ \underline{W} - \underline{\dot{W}}) + \mathbf{h}(\alpha\ \mathbf{C}\cdot\underline{n}|_{\partial\Omega} - \underline{\bar{F}})\|^2_{\partial\Omega} \tag{4.63}$$

Applications $\boldsymbol{S}_m : \mathcal{P} \to \mathcal{U}$ (resp. $\boldsymbol{P}_m : \mathcal{U} \to \mathcal{P}$) which maps a time function α (resp. space function $\mathbf{E}$) into a space (resp. time) function are defined by the following equations:

- $(\mathbf{E}, \mathbf{C}, \underline{W}) = S_m(\alpha) \in \mathcal{U}$ is defined by:

$$\forall(\mathbf{E}^\star, \mathbf{C}^\star, \underline{W}^\star) \in \mathcal{U}, \dots$$
$$\int_{\Omega_E\times I} [\alpha\mathbf{C}^\star\ \mathbf{H}\ -\dot{\alpha}\mathbf{E}^\star]\ \mathbf{M}\ \left[(\alpha\mathbf{C} - \bar{\boldsymbol{\sigma}})\mathbf{H}\ -(\dot{\alpha}\mathbf{E} - \dot{\bar{\boldsymbol{\varepsilon}}})\right]\ \mathrm{d}\Omega\ \mathrm{d}t + \dots$$
$$\dots \int_{\partial\Omega_E\times I} [\alpha\underline{W}^\star\ +\ \mathbf{h}\ \alpha\mathbf{C}^\star]\ \mathbf{m}\ \left[(\alpha\underline{W} - \underline{\dot{W}})\ +\ \mathbf{h}(\alpha\mathbf{C} - \underline{\bar{F}})\right]\ \mathrm{d}\Sigma\ \mathrm{d}t = 0 \tag{4.64}$$

- $\alpha = P_m(\mathbf{E}, \mathbf{C}, \underline{W}) \in \mathcal{P}$ is defined by:

$$\forall\alpha^\star \in \mathcal{P}, \dots$$
$$\int_{\Omega_E\times I} [\alpha^\star\mathbf{C}\ \mathbf{H}\ -\dot{\alpha}^\star\mathbf{E}]\ \mathbf{M}\ \left[(\alpha\mathbf{C} - \bar{\boldsymbol{\sigma}})\mathbf{H}\ -(\dot{\alpha}\mathbf{E} - \dot{\bar{\boldsymbol{\varepsilon}}})\right]\ \mathrm{d}\Omega\ \mathrm{d}t + \dots$$
$$\dots \int_{\partial\Omega_E\times I} [\alpha^\star\underline{W}\ +\ \mathbf{h}\ \alpha^\star\mathbf{C}]\ \mathbf{m}\ \left[(\alpha\underline{W} - \underline{\dot{W}})\ +\ \mathbf{h}(\alpha\mathbf{C} - \underline{\bar{F}})\right]\ \mathrm{d}\Sigma\ \mathrm{d}t = 0 \tag{4.65}$$

The algorithm to construct the first order PGD approximation is based on an alternative minimization of (4.54) where the variables are represented by equation (4.59) to (4.62):

- Initialization of $\alpha^1(t)$;
for $l = 2, ..., l_{max}$ **do**
  - Space problem: knowing $\alpha^{l-1}(t)$, find $(\mathbf{E}(\underline{x}), \mathbf{C}(\underline{x}), \underline{W}(\underline{x}))^l$ which solves (4.64);
  - Time problem: knowing $(\mathbf{E}(\underline{x}), \mathbf{C}(\underline{x}), \underline{W}(\underline{x}))^l$, find α^l which solves (4.65);
  - normalization of $(\mathbf{E}(\underline{x}), \mathbf{C}(\underline{x}), \underline{W}(\underline{x}))^l$.
end

Algorithm 5: Algorithm to generate the first order PGD for the microproblem.

In order to build the approximation of order k, it is sufficient to update the variables by adding the already generated PGD pairs to $\left(\bar{\boldsymbol{\sigma}}, \dot{\bar{\boldsymbol{\varepsilon}}}, \underline{\bar{F}}, \underline{\dot{\bar{W}}}\right)$ in Problem 21.

Problem 21 provides an intrinsic convergence criteria related to the verification of the search direction. Let us note that even a partial verification of the search direction is sufficient to attain a good convergence of the computational strategy. This feature of the LaTIn method enables one to generate new PGD pairs only when necessary, since it represents the most expensive part of the strategy. In practice, an updating of all the time functions α_i with the current ROB at iteration $n+1$ is first performed before any generation of new PGD pairs. This stage is called *Preliminary step* in the following.

Microproblem of the linear stage for iteration $n+1$ at the LaTIn-PGD strategy

1. *Preliminary step [Boisse et al., 1990, Ryckelynck, 2002, Ladevèze and Nouy, 2003, Ladevèze et al., 2009, Ladevèze et al., 2010]*: reuse of the ROB (called *update step* in [Nouy, 2010, Bonithon and Nouy, 2012]). This step consists in building an approximation of the solution, denoted $\breve{s}^{(n+1)}$, thanks to the ROB generated at the previous iteration n of the LaTIn-PGD. Here, the only unknowns are the time functions $\{\alpha_i^{(n+1)}\}_{1\leq i\leq k}$. Given a ROB of space functions $\{\mathbf{E}_i, \mathbf{C}_i, \underline{W}_i\}_{1\leq i\leq k}$, one seeks the best linear combination of this ROB which minimizes the error (4.54). Let us regroup known quantities in:

$$\Delta_0 = \mathbf{H}\,\bar{\boldsymbol{\sigma}} - \dot{\bar{\boldsymbol{\varepsilon}}} \qquad\qquad \Delta_1 = -\underline{\dot{\bar{W}}} - \mathbf{h}\underline{\bar{F}} \tag{4.66}$$

Denoting as $\underline{\alpha}$ the set as time functions $\{\alpha_i\}_{1\leq i\leq k}$, the stationary of the error (4.54) with respect the set of time functions leads to Problem 22:

Problem 22 (preliminary step) *Find* $\{\alpha_i(t)\}_{i=1,\dots,k} \in \mathcal{P}$ *such that,* $\forall \alpha_i^\star \in \mathcal{P}$*,*

$$\begin{aligned}\forall\, \underline{\alpha}^\star,\quad & \int_I \underline{\dot{\alpha}}^\star \left(A_{00}\,\underline{\alpha} + A_{01}\,\underline{\dot{\alpha}} - Q_0\right) \mathrm{d}t + \int_I \underline{\alpha}^\star \left(A_{10}\,\underline{\alpha} + A_{11}\,\underline{\dot{\alpha}} - Q_1\right) \mathrm{d}t + \dots \\ \dots\quad & \int_I \underline{\dot{\alpha}}^\star \left(B_{00}\,\underline{\alpha} + B_{01}\,\underline{\dot{\alpha}} - Z_0\right) \mathrm{d}t + \int_I \underline{\alpha}^\star \left(B_{10}\,\underline{\alpha} + B_{11}\,\underline{\dot{\alpha}} - Z_1\right) \mathrm{d}t \;=\; 0\end{aligned} \tag{4.67}$$

with the matrices:

$$\begin{aligned}
A_{00} &= \int_\Omega \mathbf{C}\,\mathbf{H}\,\mathbf{M}\,\mathbf{H}\,\mathbf{C}^T\,\mathrm{d}\Omega & A_{01} &= -\int_\Omega \mathbf{C}\,\mathbf{H}\,\mathbf{M}\,\mathbf{E}^T\,\mathrm{d}\Omega \\
A_{10} &= -\int_\Omega \mathbf{E}\,\mathbf{M}\,\mathbf{H}\,\mathbf{C}^T\,\mathrm{d}\Omega & A_{11} &= \int_\Omega \mathbf{E}\,\mathbf{M}\,\mathbf{E}^T\,\mathrm{d}\Omega \\
Q_0 &= \int_\Omega \mathbf{C}\,\mathbf{H}\,\mathbf{M}\,\Delta_0\,\mathrm{d}\Omega & Q_1 &= -\int_\Omega \mathbf{E}\,\mathbf{M}\,\Delta_0\,\mathrm{d}\Omega \\
B_{00} &= -\int_{\partial\Omega} \underline{W}\,\mathbf{m}\,\mathbf{h}\,\mathbf{C}^T\,\mathrm{d}\Sigma & B_{01} &= \int_{\partial\Omega} \underline{W}\,\mathbf{m}\,\underline{W}^T\,\mathrm{d}\Sigma \\
B_{10} &= -\int_{\partial\Omega} \mathbf{C}\,\mathbf{h}\,\mathbf{m}\,\mathbf{h}\,\mathbf{C}^T\,\mathrm{d}\Sigma & B_{11} &= \int_{\partial\Omega} \mathbf{C}\,\mathbf{m}\,\mathbf{h}\,\underline{W}^T\,\mathrm{d}\Sigma \\
Z_0 &= \int_{\partial\Omega} \underline{W}\,\mathbf{m}\,\Delta_1\,\mathrm{d}\Sigma & Z_1 &= \int_{\partial\Omega} \mathbf{C}\,\mathbf{h}\,\mathbf{m}\,\Delta_1\,\mathrm{d}\Sigma
\end{aligned} \tag{4.68}$$

By taking into consideration the underlying finite element approximation ($\mathbf{E}_i = \sum_{j=1}^{(n)} (\Phi_i)_j \varphi_j(\underline{x})$), this leads to a $k \times k$ linear algebraic system. The linear algebraic system is assembled and solved for each time instants of the set $I_p = \{t_i \in I\}_{1 \leq i \leq p}$, where I_p represents the discretized time space I. For this discretized domain a linear interpolation and a Euler implicit integration scheme is considered. This leads to a computational complexity in the order of $\mathcal{O}(p\ N\ k^2\ + p\ N\ k + \ p\ k^3)$.

2. *Preliminary step performance indicator*: An error indicator, based on the error indicator (4.32) developed for the LaTIn method in [Ladevèze, 1999], is then computed to quantify the accuracy of this first prediction. This error indicator is used to quantify the rate of the convergence, by measuring the ratio (see [Heyberger et al., 2011] for more details):

$$\eta_0 = \frac{e_1 - e_2}{e_1} \tag{4.69}$$

with

$$e_1 = \frac{\|\mathbf{s}^{(n)} - \hat{\mathbf{s}}^{(n-1/2)}\|}{1/2\ \|\mathbf{s}^{(n)} + \hat{\mathbf{s}}^{(n-1/2)}\|} \qquad e_2 = \frac{\|\breve{\mathbf{s}}^{(n+1)} - \hat{\mathbf{s}}^{(n+1/2)}\|}{1/2\ \|\breve{\mathbf{s}}^{(n+1)} + \hat{\mathbf{s}}^{(n+1/2)}\|} \tag{4.70}$$

The difference at numerator compares the distance between the two spaces $\mathbf{A}_d$ and $\mathbf{\Gamma}$ for two consecutive iterations and gives a measure of how swiftly solution $\breve{\mathbf{s}}^{(n+1)}$ is leading to the convergence solution with respect to the previous LaTIn iteration.

If η_0 is higher than a critical threshold, then the linear stage at Iteration $n+1$ is considered to be solved. One can proceed to next iteration. Otherwise, one proceeds to the generation of a new PGD pair. The convergence indicator η_0 is computed for each subdomain, thus, it enables the computational strategy to seek the subdomains where the approximation needs to be enriched.

3. *Generation of a new PGD pair*: The prediction previously computed is considered to be known and the performance indicator is lower of the given threshold for a given subdomain. A new PGD pair is sought to enrich the previous approximation from the order $k-1$ up to order k. It is generated solving Problem 21 by the algorithm 5 where the right-hand side is updated by the already known previous solution $\tilde{\mathbf{s}}^{k-1}$ expressed in PGD form of order $k-1$. Problem 21 leads to the definition of two problems:

   Problem 23 (Generation of a spatial function) *Knowing* $\alpha(t)$ *from* (4.65)*, find* $(\mathbf{E}(\underline{x}), \mathbf{C}(\underline{x}), \underline{W}(\underline{x}))$ *such that it solves equation* (4.64)*.*

   Problem 24 (Generation of a time function) *Knowing* $(\mathbf{E}(\underline{x}), \mathbf{C}(\underline{x}), \underline{W}(\underline{x}))$ *from* (4.64)*, find* $\alpha(t)$ *such that it solves equation* (4.65)*.*

   Once this new pair of time and space functions is generated, the $(k+1)th$ space mode is orthogonalized and added to the ROB. The orthogonalization with respect to the already existing modes becomes necessary with the introduction of the Preliminary step. Indeed, to generate the best linear combination for a given ROB, this stage needs an orthogonal basis ensuring the well-conditioning of the linear system. The reader is referred to [Giacoma et al., 2015] in order to appreciate the importance of the basis orthogonalization in the context of nonlinear contact problems.

Remark 11 *For some particular domains of application, as for instance the analysis of structure made of composite materials, it is possible to discern a representative volume subdomain. For these problems, the domain decomposition method produces structures as assembly of similar subdomains. Two or more subdomains are considered to be similar when their geometries and their boundary conditions are very similar. In this case, it is possible to construct a single ROB shared by all those subdomains. Hence, each new PGD function added to this ROB is exploited to approximate the solution of all the other similar subdomains. This ROB sharing enables one to generate less space functions. Indeed, similar subdomains would generate similar PGD modes. This process of ROB sharing occurs at the preliminary step [Ladevèze et al., 2010].*

Convergence test of the LaTIn-PGD In order to check the convergence of the iterative scheme, one constructs the error defined in (4.32). To resume, microproblem of the linear stage at iteration $n+1$ is solved as follows:

- Preliminary step: given a ROB of space functions $\{\mathbf{E}_i, \mathbf{C}_i, \underline{W}_i\}_{1\leq i\leq k}$, updates the time functions $\{\alpha_i^{(n+1)}\}_{1\leq i\leq k}$ which minimizes (4.54).
- Preliminary step performance indicator η_0, defined in (4.69).
- **if** $\eta_0 >$ *threshold* **then**
  | microproblem of the linear stage at iteration $n+1$ is solved.
else
  | generation of a new PGD pair summarized in Algorithm 5 ;
end
- LaTIn convergence indicator defined in (4.32).

Algorithm 6: Microproblem of the linear stage at iteration $n+1$.

4.4 Approximation of the Preliminary step using the Reference Points Method

In this section the Preliminary step is approximated by using the RPM. That enables one to reduce the number of elementary operations to construct the reduced-order model. In order to describe this procedure, let us consider equation (4.67). In order to examplify the technique, let us focus on the term A_{00}, the treatment of A_{01}, A_{10}, A_{11} being similar. Let us denote the term (i,j) of the new reduced-order model as:

$$J_{ij} = \int_{I\times\Omega} \alpha_i^\star \, \mathbf{C}_i \, \mathbf{H} \;\; \mathbf{M} \, \mathbf{H} \, \mathbf{C}_j^T \alpha_j \, \mathrm{d}\Omega \, \mathrm{d}t, \tag{4.71}$$

such that

$$J = (J_{ij})_{i=1,..,k,\ j=1,..,k}\,. \tag{4.72}$$

In order to simplify the notation let us note

$$\mathbf{G}(\underline{x},t) = \; \mathbf{H}(\underline{x},t) \; \mathbf{M}(\underline{x},t) \; \mathbf{H}(\underline{x},t).$$

The Preliminary step, detailed in the previous section, involves, at each iteration of the LaTIn-PGD method, the construction of a new reduced-order model in order to seek the best linear combination of the already existing ROB (see (4.67)). For nonlinear problems, the construction of a reduced-order model is performed after the evaluation of the new search direction operator (*i.e.* $\mathbf{H} = \partial\mathbf{B}/\partial\boldsymbol{\sigma}|_{\hat{\boldsymbol{\sigma}}}$), whenever it is necessary. Hence, this latter has to be integrated over the entire domain of the definition of the problem. The computational complexity analysis of this part of the strategy has been already shown in (3.3) for an elliptic problem. This analysis remains the same in the present situation. The results of the computational complexity analysis to compute J are summarized in **Tab.4.1**.

Both the integration and the Galerkin projection of the operators involve a number of operations that scales with the dimension N of the underlying finite element approximation space and the dimension p of the underlying discretisation of the time

Table 4.1: Number of operations associated to J.

Standard PGD framework		
J	Integration point evaluation of $\mathbf{G}$	$\mathcal{O}(p\ N\ \beta)$
	Projection of J onto ROB	$\mathcal{O}(p\ N\ k^2)$
	Projection of residue onto ROB	$\mathcal{O}(p\ N\ k)$

domain. In [Néron et al., 2015] the authors have shown that for a parametrized $3D$ visco-plastic problem, the computational cost related to these two operations represents 75% of the entire CPU time spent for the simulation.

Two approaches are investigated:

- Method 1: separated representation of operator by RPM. It consists into approximating the evaluation and integration of the search direction by the RPM, and then into reconstructing it in a separated representation by the explicit formulas (as described in (2.31)). The approximation of $\mathbf{G}(\underline{x}, t)$ in separated variables form enables one to separate the computation of the time-space integrals. This leads to a reduction of the complexity for the operation (4.71).

- Method 2: construction of the reduced-order model by the Reference Points Method. This one consists into constructing the reduced-order model by using the RPM. In that case, all the operations are computed using the RPM approximation framework. Hence, all the quantities are represented with their generalized components. Operations are executed between generalized components of quantities enabling one to separate the time-space integrals and reduce the number of elementary operations.

In order to clarify the different approaches of this two methods let consider again (4.71):

$$J_{ij} = \int_{I\times\Omega} \alpha_i^\star\ \mathbf{C}_i\ \mathbf{G}\ \mathbf{C}_j^T \alpha_j\ \mathrm{d}\Omega\ \mathrm{d}t, \tag{4.73}$$

First method (Method 1) gives a separated variables approximation of the operator $\mathbf{G}$. This enables one to separate the integrals of the (4.73). When the integrals over the time and space are separated, the operation of projection of $\mathbf{G}$ over the reduced-order basis $\{\mathbf{C}_i\}_{i=1,\dots,k}$ does not have to be computed for every value of the time domain, but it can be computed just once. Considering $\bar{\bar{\mathbf{G}}}$ a separated variables representation of $\mathbf{G}$:

$$\bar{\bar{\mathbf{G}}}(t, \underline{x}) = g(t)\mathbf{\Gamma}(\underline{x}), \tag{4.74}$$

equations (4.74) and (4.73) lead to:

$$J_{ij} = \int_I \alpha_j^\star(t)\ g(t)\alpha_j(t)\ \mathrm{d}t \int_\Omega \mathbf{C}_i\ \mathbf{\Gamma}(\underline{x})\ \mathbf{C}_j^T\ \mathrm{d}\Omega. \tag{4.75}$$

The second method (Method 2) does not give a separated variables approximation of the operator $\mathbf{G}$. It computes the operation $\mathbf{C}_i\ \mathbf{G}\ \mathbf{C}_j^T$ of (4.73) exclusively

between the generalized components of the operands: $\bar{\mathbf{C}}_i\ \bar{\mathbf{G}}\ \bar{\mathbf{C}}_j^T$. It leads to operate the integrals only over a restricted part of the time and space domain:

$$J_{ij} = \sum_{a=1}^{m_t}\sum_{b=1}^{m_x}\int_{I_a\times\Omega_b} \bar{\mathbf{C}}_i\,\bar{\mathbf{G}}\,\bar{\mathbf{C}}_j^T\ \mathrm{d}\Omega_b\ \mathrm{d}t \tag{4.76}$$

That enables one to decrease the computational complexity related to (4.73). Thus, this method does not give a separated variables approximation of quantities but consider a restricted part of these quantities to perform the operations. Then, thanks to the explicit formulas (2.31) the result of the operation is reconstructed in a separated variables representation.

4.4.1 Method 1: approximation of the tangent operator by the Reference Points Method

Let us split the time domain in m_t sub-intervals I_a. The center $\underline{\mu}_a$ of the sub-interval I_a is the reference instance point. For the space domain m_x points $\underline{x}_b$ are introduced and the domain Ω is divided into m_x sub-domains Ω_b. The $\underline{x}_b$ are the reference spatial points. The quantities are represented by their generalized components.

In this method, the tangent operator $\mathbf{G}$ is approximated by the RPM before reconstruction in a separated variable format by the explicit formulas presented in section 2.3.1. Let us introduce the generalized component $\bar{\mathbf{G}} = (\bar{\mathbf{G}}^t, \bar{\mathbf{G}}^x)$, defined as follows:

$$\bar{\mathbf{G}}^t := \left\{ \begin{array}{rcccl} \bar{\mathbf{G}}^t_{ab}(t) & = & \mathbf{G}(\underline{x}_b, t) & \text{if } t\ \in I_a & \\ \bar{\mathbf{G}}^t_{ab}(t) & = & 0 & \text{otherwise} & \end{array} \right\}_{a=1,..,m_t;\ b=1,..,m_x} \tag{4.77}$$

$$\bar{\mathbf{G}}^x := \left\{ \begin{array}{rcccl} \bar{\mathbf{G}}^x_{ab}(\underline{x}) & = & \mathbf{G}(\underline{x}, t_a) & \text{if } \underline{x}\ \in \Omega_b & \\ \bar{\mathbf{G}}^x_{ab}(\underline{x}) & = & 0 & \text{otherwise} & \end{array} \right\}_{a=1,..,m_t;\ b=1,..,m_x} \tag{4.78}$$

For the first generalized component $\bar{\mathbf{G}}^x$, the operator $\mathbf{G}(\underline{x}, t_a)$ is evaluated over the entire space domain for m_t reference instants $\{t_a\}_{i=1,...,m_t}$. For the second generalized component $\bar{\mathbf{G}}^t$, the operator $\mathbf{G}(\underline{x}_b, t)$ is evaluated over the entire time domain for m_x reference space points $\{\underline{x}_b\}_{i=1,...,m_x}$.

Once $\bar{\mathbf{G}}^t$ and $\bar{\mathbf{G}}^x$ are known, the full format is obtained thanks to the explicit formulas (2.31). This leads to a separated representation of operator $\mathbf{G}$:

$$\bar{\bar{\mathbf{G}}}(t, \underline{x}) = g_{ab}(t)\boldsymbol{\Gamma}_{ab}(\underline{x}). \tag{4.79}$$

Analysis of computational complexity Equations (4.79) and (4.71) lead to:

$$J_{ij} = \int_{I\times\Omega} \alpha_i^\star\ \mathbf{C}_i\ \bar{\bar{\mathbf{G}}}\ \mathbf{C}_j^T\ \alpha_j\ \mathrm{d}\Omega\ \mathrm{d}t, \tag{4.80}$$

$$J_{ij} = \sum_{a=1}^{m_t} \sum_{b=1}^{m_x} \int_{I_a} \alpha_j^\star(t)\ g_{ab}(t)\alpha_j(t)\ \mathrm{d}t \quad \int_{\Omega_b} \mathbf{C}_i\ \mathbf{\Gamma}_{ab}(\underline{x})\ \mathbf{C}_j^T\ \mathrm{d}\Omega. \tag{4.81}$$

Considering the contribution of all the k space modes $\{\mathbf{C}_i\}_{i=1,..,k}$ of the PGD reduced-order basis, one obtains the Jacobian J. The approximation of $\mathbf{G}(\underline{x},t)$ in separated representation enables one to separate the integrals, as shown in (4.81). The operator $\mathbf{\Gamma}(\underline{x})$ does not depend on t. Hence, the operation of integration over the space (scaling with N) and projection of the operator on the basis, $\mathbf{C}_i\ \mathbf{\Gamma}(\underline{x})\mathbf{C}_j$, is computed just once instead of p times. The number of operations to calculate J decreases from $\mathcal{O}\left(p\ N\ (\beta + k^2)\right)$ to $\mathcal{O}\left(p\ +\ N\ (\beta + k^2)\right)$ with a gain equal to p. Let us remember that β is the number of FLOPS to evaluate the search direction operator at a single integration point, p, in this case is the number of time instants in the discretized time domain and N the number of degrees of freedom of the underlying space discretization. Finally the gain in the number of operations is equal to the ratio between the number of operations to integrate and project the Jacobian in the standard PGD and the same operations computed by the RPM:

$$\frac{p\ N\ (\beta\ +\ k^2)}{p\ +\ N\ (\beta\ +\ k^2)} \tag{4.82}$$

Scaling numerator and denominator by p, (4.82) can be written as follows:

$$\frac{N\ (\beta\ +\ k^2)}{1\ +\ \frac{N}{p}\ (\beta\ +\ k^2)} \tag{4.83}$$

Now, considering that, for mechanical problems:

$$\frac{N}{p}\ (\beta\ +\ k^2)\ >>\ 1 \tag{4.84}$$

the gain in the number of operations is expressed as follows:

$$\frac{N\ (\beta\ +\ k^2)}{\frac{N}{p}\ (\beta\ +\ k^2)}\ =\ p \tag{4.85}$$

Finally the gain in the number of operations to compute the Jacobian J (4.81) is in the order of p. **Tab. 4.2** resumes this analysis.

4.4.2 Method 2: construction of the reduced-order model by the Reference Points Method

In that case, all the operations of projection and integration to construct the reduced-order model, denoted as J_{ij} in (4.71), are approximated by the RPM. The quantities are represented by their generalized components and the operations are executed between generalized components of quantities.

Table 4.2: Comparison of the number of operations between J and its approximation by RPM.

Standard PGD framework		
J	Integration point evaluation of Jacobian	$\mathcal{O}(p\ N\ \beta)$
	Projection of Jacobian onto ROB	$\mathcal{O}(p\ N\ k^2)$
RPM framework		
J	Integration point evaluation of Jacobian	$\mathcal{O}(p\ +\ N\ \beta)$
	Projection of Jacobian onto ROB	$\mathcal{O}(N\ k^2)$

$$\bar{\omega}^t := \left\{ \begin{array}{rcccl} \bar{\omega}^t_{ab}(\underline{x}) & = & \mathbf{C}_i(\underline{x})\ \mathbf{G}(\underline{x},t_a)\ \mathbf{C}^T_j(\underline{x}) & \text{if}\, \underline{x}\ \in \Omega_b \\ \bar{\omega}^t_{ab}(\underline{x}) & = & 0 & \text{otherwise} \end{array} \right\}_{a=1,..,m_t;\ b=1,..,m_x} \tag{4.86}$$

$$\bar{\omega}^x := \left\{ \begin{array}{rcccl} \bar{\omega}^x_{ab}(t) & = & \mathbf{C}_i(\underline{x}_b)\ \mathbf{G}(\underline{x}_b,t)\ \mathbf{C}^T_j(\underline{x}_b) & \text{if}\, t\ \in I_a \\ \bar{\omega}^x_{ab}(t) & = & 0 & \text{otherwise} \end{array} \right\}_{a=1,..,m_t;\ b=1,..,m_x} \tag{4.87}$$

The approximation of ω given by RPM, noted $\bar{\bar{\omega}} = \ \psi(t)\ \chi(\underline{x})$ is obtained by the explicit formulas (2.31). Following formulas (2.31), $\chi(\underline{x})$ and $\psi(t)$ are obtained from (4.86) and (4.87) as follows:

$$\psi_{ab}(t) = \frac{\sum_{c=1}^{m_x} \Delta\Omega_c\ \bar{\omega}^t_{ac}(t)\bar{\omega}^x_{ac}(\underline{x}_c)\lambda^2_{ac}}{\sum_{c=1}^{m_x} \Delta\Omega_c\ \bar{\omega}^x_{ac}(\underline{x}_c)\bar{\omega}^x_{ac}(\underline{x}_c)\lambda^2_{ac}} \qquad \text{and} \qquad \chi_{ab}(\underline{x}) = \bar{\omega}^x_{ab}(\underline{x}) \tag{4.88}$$

Replacing ω in (4.71) one obtains:

$$J_{ij}\ =\ \int_{\Omega\times\mathcal{D}} \omega(\underline{x},t)\ \mathrm{d}\Omega\ \mathrm{d}\mu\ \approx\ \int_{\Omega\times\mathcal{D}} \bar{\bar{\omega}}(\underline{x},t)\ \mathrm{d}\Omega\ \mathrm{d}t \tag{4.89}$$

Replacing $\bar{\bar{\omega}} = \ \psi(t)\ \chi(\underline{x})$ in (3.37) one can separate the integrals:

$$J_{ij}\ \approx\ \int_{\Omega\times I} \bar{\bar{\omega}}(\underline{x},t)\ \mathrm{d}\Omega\ \mathrm{d}t\ =\ \int_{\Omega} \chi(\underline{x})\ \mathrm{d}\Omega \int_I \psi(t)\ \mathrm{d}t \tag{4.90}$$

The domain is the union of the $m_x\ m_\mu$ considered patched. Hence (3.38) lead to:

$$J_{ij}\ \approx\ \sum_{a=1}^{m_t}\sum_{b=1}^{m_x} \int_{\Omega_b} \chi_{ab}(\underline{x})\ \mathrm{d}\Omega_b \int_{I_a} \psi_{ab}(\underline{t})\ \mathrm{d}t_a \tag{4.91}$$

Replacing (3.36) in (3.38) one obtains:

$$J_{ij}\ \approx\ \sum_{a=1}^{m_t}\sum_{b=1}^{m_x} \int_{\Omega_b} \bar{\omega}^x_{ab}(\underline{x})\ \mathrm{d}\Omega_b \int_{I_a} \frac{\sum_{c=1}^{m_x} \Delta\Omega_c\ \bar{\omega}^t_{ac}(t)\ \bar{\omega}^x_{ac}(\underline{x}_c)\ \lambda^2_{ac}}{\sum_{c=1}^{m_x} \Delta\Omega_c\ \bar{\omega}^x_{ac}(\underline{x}_c)\ \bar{\omega}^x_{ac}(\underline{x}_c)\ \lambda^2_{ac}}\ \mathrm{d}t_a \tag{4.92}$$

For the integral over I_a the only term depending on t is $\bar{\omega}^t_{ac}(t)$. Equation (3.39) is equivalent to:

$$J_{ij} \approx \sum_{a=1}^{m_t} \sum_{b=1}^{m_x} \int_{\Omega_b} \bar{\omega}^x_{ab}(\underline{x})\, \mathrm{d}\Omega_b \; \frac{\sum_{c=1}^{m_x} \Delta\Omega_c \left(\int_{I_a} \bar{\omega}^t_{ac}(t)\, \mathrm{d}t_a \right) \bar{\omega}^x_{ac}(\underline{x}_c)\, \lambda^2_{ac}}{\sum_{c=1}^{m_x} \Delta\Omega_c\, \bar{\omega}^x_{ac}(\underline{x}_c)\, \bar{\omega}^x_{ac}(\underline{x}_c)\, \lambda^2_{ac}} \tag{4.93}$$

Analysis of computational complexity For this method, the analysis of computational complexity has already been detailed in section 3.3 considering the contribution of the entire set of PGD space functions in the ROB. This analysis is summarized in **Tab.4.3**.

Let us remember that β denotes the number of FLOPS to integrate the operator $\mathbf{G}$ at a single integration point. p is the number of parameter values in the discretized parameter domain and N the number of degrees of freedom of the underlying space discretisation. For the sake of simplicity, the number of integration points of our model is assumed to be equal to N.

Table 4.3: Computational complexity of integration and projection onto ROB to obtain J and $\mathcal{R}$, in the RPM format

	Operation type	Complexity
$\bar{\omega}_t$	Integration point evaluation	$\mathcal{O}(m_t\ N\ \beta)$
	Projection to obtain J	$\mathcal{O}(m_t\ N\ k^2)$
	Projection to obtain $\mathcal{R}$	$\mathcal{O}(m_t\ N\ k)$
$\bar{\omega}_x$	Integration point evaluation	$\mathcal{O}(m_x\ p\ \beta)$
	Projection of J onto ROB	$\mathcal{O}(m_t\ m_x\ k^2)$
	Projection of $\mathcal{R}$ onto ROB	$\mathcal{O}(m_t\ m_x\ k)$

Considering that, for structural mechanics, $N \gg m_x$, the cost to compute $\bar{\omega}_x$ can be neglected. Hence, the expected gain in terms of number of operations to construct the reduced-order model is equal to the ratio p/m_t. This is outlined in **Tab.4.4**. This gain, for typical mechanical problems, can be around one order of magnitude.

Table 4.4: Comparison of the number of operations between J and its approximation given by RPM.

Standard PGD framework		
J	Integration of $\mathbf{G}$ on $I \times \Omega$	$\mathcal{O}(p\ N\ \beta)$
	Projection of Jacobian onto ROB	$\mathcal{O}(p\ N\ k^2)$
RPM framework		
J	Integration related to $\bar{\omega}^t$	$\mathcal{O}(m_t\ N\ \beta)$
	Projection related to J	$\mathcal{O}(m_t\ N\ k^2)$

Cost of the reconstruction As introduced in chapter 2.3 the generation of a first approximation of the quantity J in a separated-variables format is obtained by generating one product of functions per time-space patch $I_a \times \Omega_b$.

$$\psi_{ab}(t) = \frac{\sum_{c=1}^{m_x} \Delta\Omega_c \ \bar{\omega}_{ac}^t(t)\bar{\omega}_{ac}^x(\underline{x}_c)\lambda_{ac}^2}{\sum_{c=1}^{m_x} \Delta\Omega_c \ \bar{\omega}_{ac}^x(\underline{x}_c)\bar{\omega}_{ac}^x(\underline{x}_c)\lambda_{ac}^2} \qquad \text{and} \qquad \chi_{ab}(\underline{x}) = \bar{\omega}_{ab}^x(\underline{x}) \tag{4.94}$$

The approximation is given by generating only a PGD pair locally, over each patch $I_a \times \Omega_b$. The cost of reconstruction given by the (4.94) is simply in the order of $8(m_x^2 \ m_t) + 2$. This computational cost has to be added to the computational complexity of the operation, analysed in 4.4.1 and 4.4.2.

4.5 Numerical example

In this section, the proposed strategy is illustrated through a numerical test. The aim is to show that the application of the RPM to approximate the Preliminary step of the strategy leads to the same order of convergence as the standard technique while leading to a significant reduction in the computation time.

Under the assumption of plane strain, let us consider the evolution over $I = [0, T], (T = 10s)$ of a two-dimensional L-shaped structure made of 4 subdomains (see the **Fig.4.4**). Each subdomain is a square of 10 mm $\times$ 10 mm. The first cell has three holes. The structure is clamped at the bottom and subjected, along its top side, to a prescribed pressure $\underline{F}_d(t) = F_0 \sin(t) \ \underline{e}_y$ with $F_0 = 10$ MPa (see **Fig.4.5**)). Each subdomain is discretized into quadratic triangular finite elements and the integration is computed over 3 Gauss points per each finite element, leading to 1368 dofs for the holed subdomain and 882 dofs for the others, for a total amount of 4014 spatial dofs. The time interval I is discretized into 100 time steps and an Euler implicit integration scheme is used. Since our example is relatively simple, it is possible to build a reference solution by using a direct incremental computation. Thus, the following error according to the reference solution is introduced:

$$\eta_{true} = \frac{\|\bar{\bar{U}} - U\|_{I\times\Omega}}{\|U\|_{I\times\Omega}}, \qquad \|U\|_{I\times\Omega}^2 = \int_{I\times\Omega} U^2 \, \mathrm{d}\Omega\mathrm{d}t. \tag{4.95}$$

where $U(\underline{M}, t)$ is the displacement field, solution over the entire time-space domain of the simulation using the direct incremental computation. $\bar{\bar{U}}(\underline{M}, t)$ is the solution of the simulation obtained with the non-linear LaTIn-PGD solver with the RPM approximation of the Preliminary step. The implementation is made in an inhouse Matlab code.

In order to study the new procedure in the case of a nonlinear problem, let us consider that the evolution law of the material is described as:

$$\dot{\boldsymbol{\varepsilon}}_p = \frac{\mathbf{K^{-1}}}{E \ \nu(t)} \ Tr(\boldsymbol{\sigma}) \ \boldsymbol{\sigma} \tag{4.96}$$

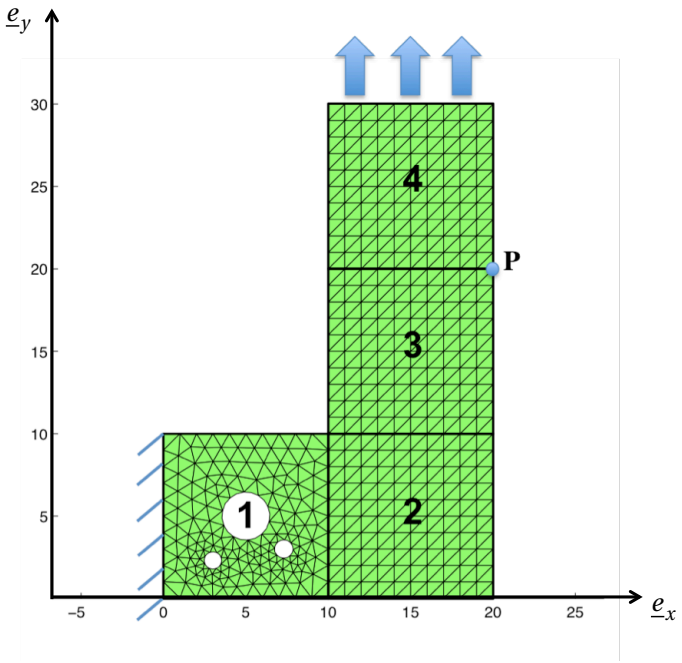


Figure 4.4: The considered structure, the partitioning into 4 subdomains and the boundary conditions. Two PGD ROB are generated: one for subdomain 1 and one shared between subdomains 2 to 4.

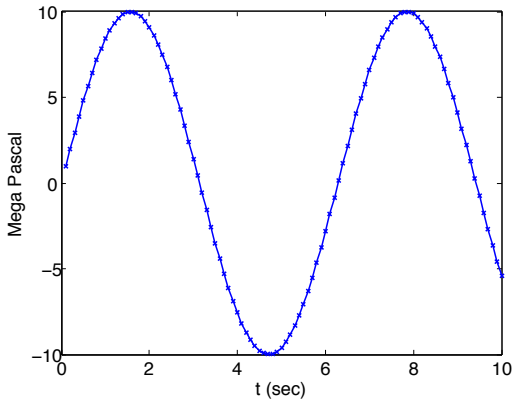


Figure 4.5: The evolution of the load over the time, $F_d(t)$.

where $\mathbf{K}$ is the Hooke's tensor of the material. Poisson's ratio is taken equal to $\nu = 0.33$ and Young's modulus is 210 GPa. The viscosity coefficient $\nu(t)$ is considered depending on the time as follows:

$$\nu(t) = 10^2 \ + \ 10 \ \sin(\frac{2\pi}{10}t) - 10 \ t \quad [\text{s}]. \tag{4.97}$$

In order to ensure rapid convergence, the search direction operators $\mathbf{H}^-$ and $\mathbf{H}^+$ are

updated at each LaTIn iteration and equal to the tangent operator:

$$\frac{d\dot{\boldsymbol{\varepsilon}}_p}{d\boldsymbol{\sigma}} = \frac{d(\frac{\mathbf{K^{-1}}}{E\ \nu(t)}\ Tr(\boldsymbol{\sigma})\ \boldsymbol{\sigma})}{d\boldsymbol{\sigma}} = \frac{\mathbf{K^{-1}}}{E\ \nu(t)}\ (\boldsymbol{\sigma}\ \mathbf{I} + Tr(\boldsymbol{\sigma})). \tag{4.98}$$

As mentioned earlier, a practical criteria could be used to set this operator to be constant, in order to avoid updating the operators when their variations are small. But this feature is not considered in this study in order to facilitate the interpretations.

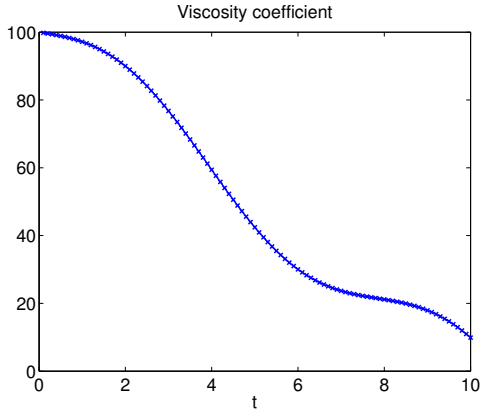
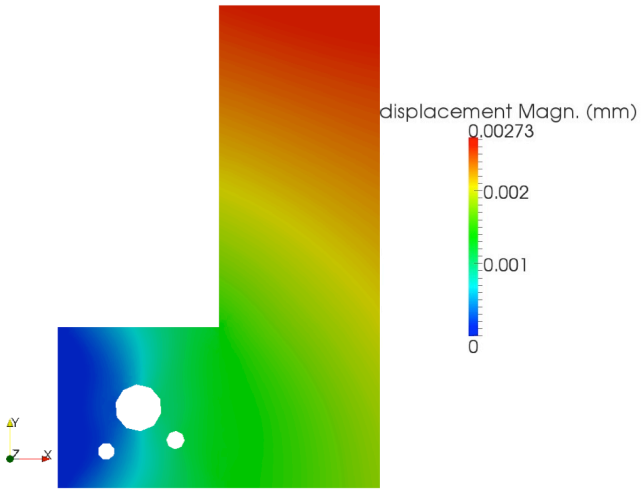


Figure 4.6: The evolution of the viscosity coefficient over the time, $\nu(t)$.

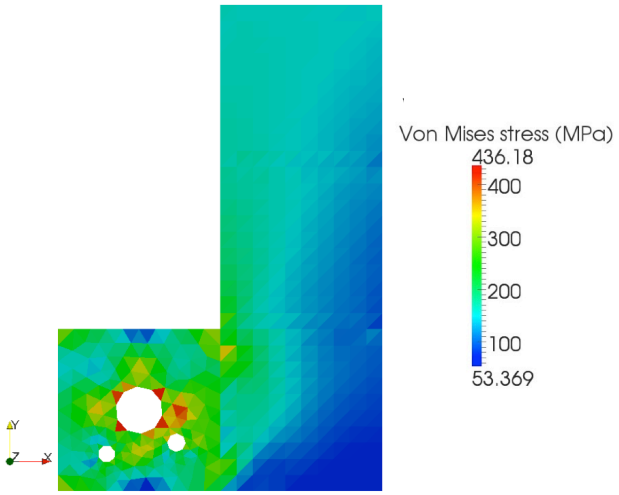
Fig.4.7 shows the displacement field and the Von Mises stress field at the final time step $T = 10$ seconds. The evolution of the vertical displacement of Point P (see **Fig.4.4**) is in perfect agreement with the result obtained with the direct incremental computation (**Fig.4.8**).

The time-space multiscale LaTIn-PGD with Preliminary step needs 48 iterations to achieve an error smaller than $5 \cdot 10^{-2}$. More precisely, the ROB is composed of 10 modes for sudomain 1 and 19 modes for subdomains 2 to 4. Indeed, here spatial functions have been shared between subdomains 2 to 4 as mentioned at the end of section 4.3.

Remark 12 *For the considered academic example, the sharing of the PGD basis within subdomains bearing similarities in terms of geometry and boundary conditions enables one to generate* 2 *different PGD basis instead of* 4 *(i.e. one for each subdomain). Dealing with only* 2 *PGD basis implies the construction of only* 2 *reduced-order models at each iteration of the LaTIn method. Considering more complicated scenarios, e.g. models for structures made of composite materials, the sharing of the basis between similar subdomains could be not sufficient to consider the construction of only one reduced-order model for all of these subdomains. In some cases, these subdomains can evolve according to different nonlinearities. This situation implies, for each subdomain, one different operator governing the evolution law (see* (4.96)*). Hence, despite of the sharing of the same PGD modes, this difference involves the*



(a) Magnitude of displacement field



(b) Von Mises stress field

Figure 4.7: The displacement field and the Von Mises stress field at the final time step $t = 10\,sec.$

construction of a reduced-order model for each one of the subdomains, increasing the necessary computational effort. In these cases, the application of the RPM can effectively decrease the necessary computational effort decreasing the computational complexity to construct each one of the reduced-order models.

Fig.4.9 shows the comparison between the convergences of the multiscale LaTIn-PGD method with and without the Preliminary step. Here, for each subdomain, a

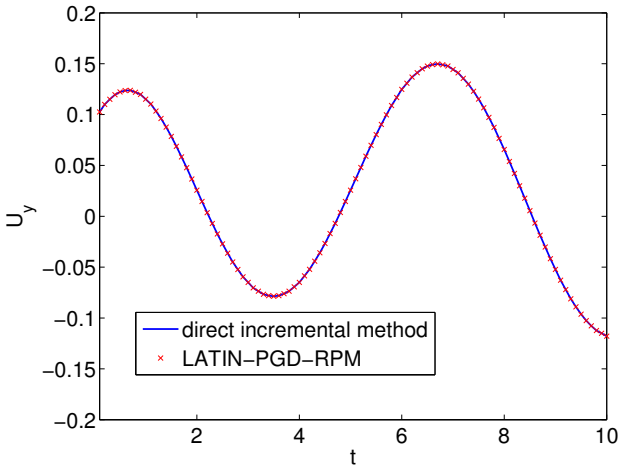


Figure 4.8: The displacement of point P over the time domain.

new PGD pair is generated at each LaTIn iteration. Not every generated PGD pair is added to the ROB. A sort, based on the norm of the time function, is considered:

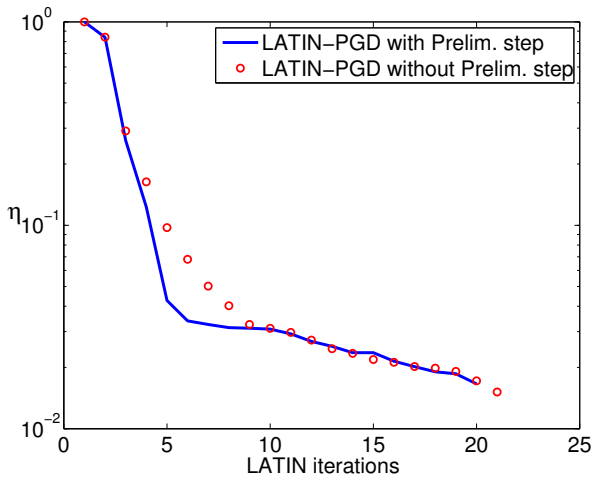
$$\phi^k = \left(\int_I (\alpha^k)^2 \mathrm{d}t \right)^{1/2} \tag{4.99}$$

That enables one to measure the relative importance of the new PGD pair $\{\mathbf{E}^k, \alpha^k\}$ with respect to the already existing PGD pairs. Only the most relevant PGD pairs are kept.

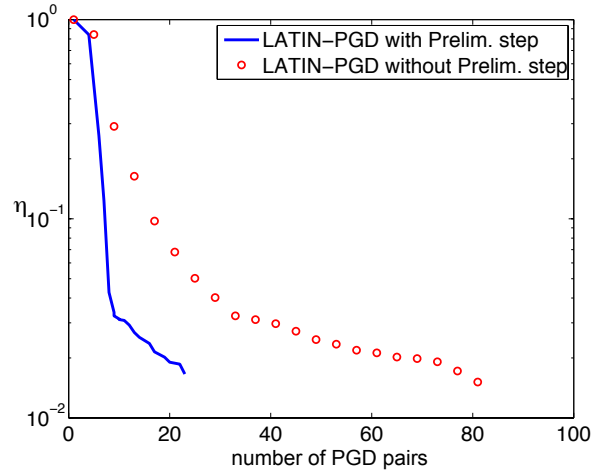
The Preliminary step enables one to drastically reduce the number of PGD pairs (**Fig. 4.9**(b)). This corresponds to an important gain in terms of CPU time spent to achieve the solution. Nevertheless, as already explained before, this represents a significant part of the remaining CPU time. Indeed, for the present numerical test considered, the CPU time spent by the Preliminary step overpasses the 50% (see the allocation of the CPU time in **Fig. 4.10**). Most of this time is spent to update the tangent operator (around 45% of the entire CPU time). For problems relatively small in terms of number of degrees of freedom, the update of the tangent operator represents the biggest part of the Preliminary step. However, when increasing the number of degrees of freedom, the computational complexity of the Galerkin projection increases.

Convergence curves of the LaTIn error indicator (see (4.32)) for the LaTIn-PGD with preliminary step with (LaTIn-PGD-RPM) and without (LaTIn-PGD) RPM are given in **Fig.4.11**(a). For the LaTIn-PGD-RPM, the two proposed approximation methods (see section 4.4) are compared:

- LaTIn-PGD-RPM1 (method 1, section 4.4.1): It consists into approximating the evaluation and integration of the search direction by the RPM, and then



(a) Error indicator vs. number of iterations



(b) Error indicator vs. number of generated PGD pairs

Figure 4.9: Comparison between the multiscale LaTIn-PGD method with the Preliminary step and the multiscale LaTIn-PGD without the Preliminary step. The RPM technique is not used.

into reconstructing it in a separated representation by the explicit formulas (as described in (2.31)).

- LaTIn-PGD-RPM2 (method 2, section 4.4.2): This one consists into constructing the reduced-order model using the RPM. In that case, all the operations to construct J_{ij} are performed using the RPM approximation framework. The quantities are represented on their generalized components. Operations are executed between generalized components of quantities enabling one to separate the time-space integrals.

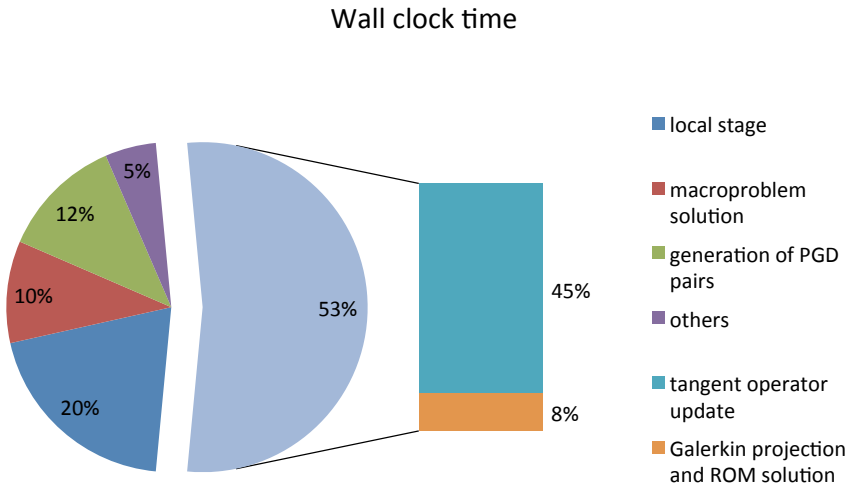
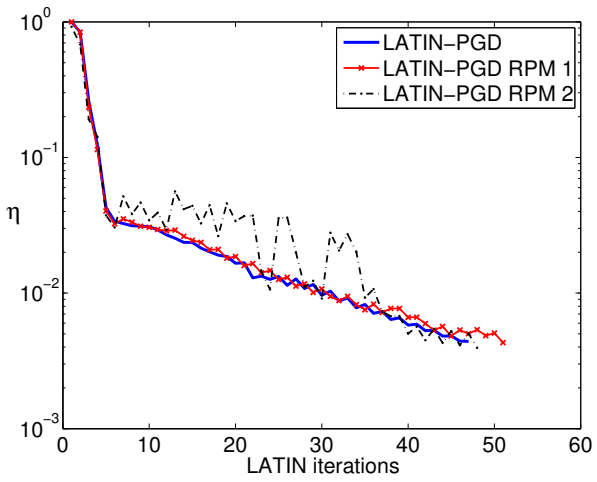


Figure 4.10: The distribution of the CPU time spent to achieve the solution.

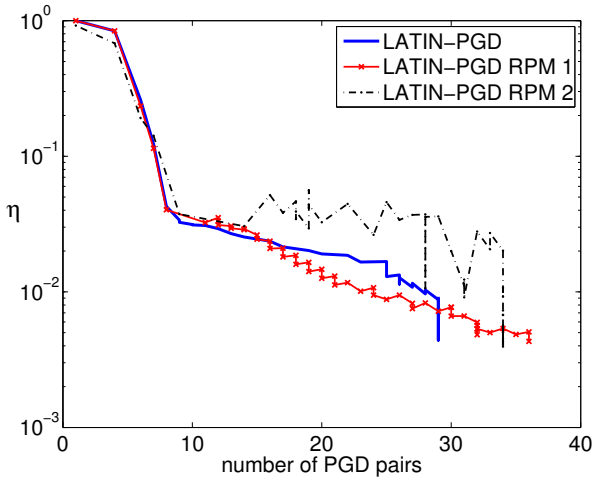
The error indicator (4.32) as a function of the number of generated PGD pairs is also plotted in **Fig.4.11**(b) for the different methods. The implementation of the RPM1, does not give satisfactory results in terms of convergence and relative error. Let us remember that RPM1 gives the separated variables representation of the search direction operator. Even if the algorithm has generated more PGD pairs to enrich the approximation, the convergence curve remains irregular. **Tab. 4.5** reports the relative error (4.95) related to each subdomain. The separated variables representation of the search direction operator seems to be not enough accurate where the gradients over the space domain are stronger (i.e. over the subdomain 1). Hence, it needs further analysis in the case of nonlinear problems.

The LaTIn-PGD-RPM2 produces better convergence curves compared with the LaTIn-PGD-RPM1. Convergence curves are smoother, even though small oscillations remain, and similar to the ones obtained with the LaTIn-PGD approach without RPM. Nevertheless, one can note that 6 additional PGD pairs are generated to reach an error level of 10^{-2}.

A posteriori error estimation. In order to appreciate the error introduced by the RPM to solve the Preliminary stage (4.67), let us consider the error (4.95). The number of reference instants is set equal to $m_t = 10$ and the number of reference spatial points varies from 1 to 25 per each subdomain. Let us recall that the ideal gain that can be reached by the RPM is in the order of $p/m_t = 10$. The computational time cost and the error η_{true} introduced by the RPM in the LaTIn-PGD strategy to solve the time problem (4.67) are given in the **Tab. 4.5**. Error η_{true} is given in percentage for each subdomain.



(a) LaTIn convergence indicator as a function of the iteration number.



(b) Error indicator 4.32 as a function of the number of generated PGD pairs.

Figure 4.11: Convergence curves

As previously said, the LaTIn-PGD-RPM1 does not give satisfactory results in terms of relative error. In fact, for the first and second subdomains, where gradients are stronger, the error remains not acceptable (higher than 15%). Adding more spatial reference points makes the error decrease but not significantly. LaTIn-PGD-RPM2, enables one to obtain a remarkable gain in terms of CPU time while maintaining a very good level of accuracy. The relative error never exceeds 4% even when only 1 spatial reference point is used for each subdomain. Let us note that the gain in terms of CPU time is close to the ideal one - i.e. $p/m_t = 10$ - when using only one spatial reference point per subdomain. This gain decreases when more spatial

Table 4.5: Wall clock time table comparing PGD standard and RPM to compute the Preliminary step.

Standard PGD to compute the Preliminary step (s)	1,040.3		
Num. of ref. points for sub-dom.	1	9	25
PGD-RPM Method 1 (s)	113	129	217
Gain	9.2	8.1	4.8
Error η_{true} subdomain 1	31%	29%	26%
Error η_{true} subdomain 2	14.4%	14.2%	13.3%
Error η_{true} subdomain 3	9.3%	8.7%	8.1%
Error η_{true} subdomain 4	6.4%	6.4%	5.8%
PGD-RPM Method 2 (s)	137	150	297
Gain	7.6	6.9	3.5
Error η_{true} subdomain 1	2.5%	2.3%	1.8%
Error η_{true} subdomain 2	4%	3.9%	3.6%
Error η_{true} subdomain 3	4.2%	4%	3.3%
Error η_{true} subdomain 4	3%	2.8%	2.3%

reference points are added. This comes from the fact that, for a small number of spatial dofs, the ratio N/m_x becomes no more negligible in the computational cost of the method 2 (see section 4.4.2). In practice, for more realistic cases, the number of degrees of freedom is much higher and the ratio N/m_x should be higher than two orders of magnitude.

4.6 Conclusions

In this chapter, the multiscale strategy based on the LaTIn-PGD has been presented. It represents a convenient framework for separated variables representation since the LaTIn method is a non-incremental iterative method that attains the solution by successive enrichments defined all over the time-space domain. At the iteration $n+1$, a first approximation of the solution is provided by the Preliminary step: it consists in minimizing the error in the search direction in the linear span of the already existing reduced-order basis, arising from the previous iteration. This approximation can be enriched by generating a new PGD pair by a greedy algorithm, whether the convergence rate is lower than a threshold. The basis of PGD functions can be shared among subdomains representing similar cells (for instance the RVE of composites materials).

The Preliminary step enables to generate only the most relevant PGD pairs with a significant time saving. However it represents a big part of the remaining computational time because it involves repetitive operations to construct the new ROM. These operations scale with the underlying time and spatial discretisations.

The RPM, developed in the chapter 2, is introduced in this part of the strategy in order to decrease the number of necessary operations to construct the new ROM

from the existing ROB. The gain in terms of number of operations is in the order of the ratio between the number p of time instants and the number m_t of reference time instants (p/m_t).

Two different ways to approximate the Preliminary step by the RPM has been developed and implemented in the computational strategy: (i) the approximation of the search direction under separated representation (denoted RPM 1) and (ii) the approximation of all the operations concerning the construction of the ROM under separated representation (denoted RPM 2). Both of them show an important gain in terms of CPU time when implemented in the considered example. Nevertheless, the approach called RPM1 needs more investigation since it appears to be less robust than the first one. In fact, it provides a low level of accuracy where gradients are stronger. On the other, RPM2 approach shows remarkable gains providing results with high level of accuracy.

Conclusion

Numerical simulation has been playing an increasingly important role in science and engineering because of the need to simulate realistic physical scenarios in order to provide reliable tools for computer-aided design-analysis. Simulations reduces the use of real prototypes and assist the design phase. In many situations, including optimization and control, the same model, depending on a parameter that is changing, has to be simulated over and over, multiplying by a large factor the solution procedure cost of one simulation.

Despite of the continuing progress in computer speeds and hardware capabilities, the construction of those solutions would be still unsuitable for all the classical numerical approaches. In fact, traditional direct techniques make use of huge approximation subspaces for the solution of the underlying PDE. This leads to systems so large that direct techniques are inappropriate for the computational efforts they involve. Model reduction techniques constitute one of the tools to circumvent this obstacle by seeking the solution of a problem in a reduced-order basis, whose dimension is much smaller than the original vector space. Chapter 1 has shown how reduced order modelling, by separation of variables, reduces the complexity related to the solution of parametric linear problems.

In literature, all the model reduction techniques based on the projection differ by the way to construct the reduced-order basis (ROB). There is a family of techniques that, in order to build the reduced-order basis, involves a learning phase, called off-line stage. In a second stage, once the reduced-order basis is built, it solves the reduced-order model (ROM) for all the parameters values. During this learning phase, these techniques perform high-fidelity simulations by direct techniques, or data acquisition by experiments, in order to obtain the main characteristics of the problem. From this phase, that can be expensive in terms of CPU, they obtain the reduced-order basis that is exploited to project the linear system and solve the reduced-order model in a smaller sub-space, with the associated computing time savings. Proper Orthogonal Decomposition (POD) and reduced-basis belong to this family of techniques.

In this work we deal with the Proper Generalized Decomposition (PGD). Basically, PGD consists in seeking the solution of a problem in a relevant reduced-order basis which is generated automatically and on-the-fly by a greedy algorithm. In the presented computational strategy, PGD is part of the LaTIn method. This is a non-incremental solver for nonlinear problems which generates the approximations of the

solution over the entire parameter/time-space domain by successive enrichments. At each iteration of the LaTIn method, the previously calculated reduced-order basis is used first, leading to a reduced-order model and a new approximation of the solution. It has been proved that this step of the strategy, called here Preliminary step ([Ladevèze and Nouy, 2003, Ladevèze et al., 2009, Ladevèze et al., 2010], enables one to reduce the number of PGD functions generated. If this approximation is not good enough, the reduced-order basis is enriched by defining a new functional product by using a greedy algorithm.

Dealing with nonlinear problems, a linearisation technique is necessary and thus, a new reduced-order model has to be constructed at each new iteration of the solution method. Section 1.2 has explained that for all kind of model reduction techniques based on the projection of the problem on a basis, the computational cost associated with assembling the ROM's low-dimensional operators scales with the large dimension of the original high-dimensional model. For this reason, model reduction techniques are particularly efficient when the reduced-order model needs to be constructed only once or when this step can be performed off-line, prior to the on-line resolution of this model which can then be very fast. This issue results to be the bottleneck of nonlinear model reduction strategies.

Some solutions have already been introduced in literature. The most known is certainly the Empirical Interpolation Method (EIM) [Barrault et al., 2004, Grepl et al., 2007, Nguyen, 2005] and its semi-discrete version (DEIM) [Chaturentabut and Sorensen, 2010]. These techniques are well-suited for model reduction techniques that resort to a learning stage, i.e., POD and reduced-basis.

However, these techniques are not suitable for the LaTIn-PGD strategy. In fact, in this strategy the reduced-order basis is progressively enriched by new functions and it is not possible to pre-compute the Galerkin projection operators in an off-line stage. It means that, even if it would be possible to start the EIM procedure at each LaTIn iteration, there would be no gains in the number of operations because the complete Galerkin projection could be not avoided. For that reason, the aim of this work has been to propose a new technique, called Reference Points Method (RPM). It consists in:

- to provide a compressed version of quantities based on the concept of *reference times*, *points* and *parameters* [Ladevèze, 1997]. This approximation framework enables one to reduce the complexity of algebraic operations between quantities in separated representation.

- to reconstruct the quantities to give a first approximation of quantities in a separated representation. The reconstruction is obtained by explicit formulas (2.31) and it arises from compressed version of quantities, generating only one product of functions locally per each patch.

- to avoid the artificial increasing of the PGD modes representing the result of the algebraic operation between quantities in separated representation.

RPM has been introduced in the LaTIn-PGD strategy in order to approximate the Preliminary step. This step involves repetitive operations between PGD quantities that scale with the underling discretisation of the computational domain. Applying the RPM to this step it is possible to decrease the number of operations by a factor p/m_μ where p is the discretisation over the parameter domain and m_μ is the number of reference parameter points chosen.

In Chapter 3 a two-dimensional elliptic and nonlinear parametrized problem has been considered to detail the technique. For that problem, the Preliminary step represents up to the 70% of the CPU time. RPM gives a gain in the order of 15 with respect to a direct technique, for an error of 10^{-3} with only a few reference points. The second part of the chapter has been dedicated to underline the similarities with respect to the reduced order model obtained by applying the reduced-basis technique on the same problem. The quality of the approximations obtained by the two techniques have been compared showing a comparable level of error with some differences. The quality of the approximation given by the EIM can be increased only by increasing both the number of interpolation points and the size of the precomputed reduced-order basis. On the contrary, the error obtained by the RPM can be reduced by increasing either the number of reference points or by adding PGD pairs throuhgout the LaTIn iterations.

Chapter 4 has been dedicated to the implementation of the RPM in the numerical analysis of structural mechanics. For that purpose, the multiscale LaTIn-based domain decomposition method has been considered. First part of this chapter has been dedicated to the description of this computational strategy. The second part has shown how the RPM has been introduced in the LaTIn-PGD strategy to approximate the Preliminary step, enabling one to reduce the number of operations to construct the reduced-order model. The chapter ends with a numerical example to assess the efficiency of the RPM within this computational strategy.

There are some perspectives related to the presented work:

- In chapter 4, the implementation of the RPM1 to give a the separated variables representation of the search direction operator is not enough accurate where the gradients over the space domain are stronger. Further investigations are required for a better implementation of the approximation of the search direction operator in separated variables format.

- Further studies are necessary in order to provide error bounds to the RPM. The way forward could be the correlation between the formulation of the functional (2.12) and the least-square regression. Indeed the functional (2.12) can be seen, patch by patch, as a least-square regression. There are already provided, for a wide family of problems, demonstrations to obtain error bounds for least-square regression and that could be the base to develop error bounds for the RPM.

- Application of the technique to more realistic 3D industrial cases.

- Implementation of the RPM in more complex nonlinear models with more physical description of materials.

- Extension of the RPM technique outside of the LaTIn framework. This would lead to the application of the RPM for techniques exploiting a learning stage to construct a reduced-order basis. In chapter 2, considering simple examples, the RPM approximation is enriched by the reduced-basis technique. Further studies are necessary, in order to have more realistic applications and extend the RPM to other reduced-order modelling techniques.

- Applied to preliminary step of the LaTIn-PGD, the RPM enables a gain in the order of the ratio p/m_t, with p the number of discrete values over the time/parameter domain and m_t the number of time/parameter reference points. This gain, for mechanical applications, can be of one or two order of magnitude. However, the independence from the finest space discretization N is not achieved since some complete integrations over the space are still necessary. In further study, a way to avoid the complete integration over the space could be figured out. For this purpose, techniques relying on gappy POD could give a contribution.

- RPM could be the starting point to develop a new PGD algorithm. This new algorithm would exploit the minimization of the functional defined in (2.12) in order to generate the new PGD pairs of functions. As in a progressive POD algorithm (see [Ryckelynck, 2005] for an example of progressive POD), at each iteration of the LaTIn method, some few solutions would be computed at the reference points. That would represent the generalized components. From these, the new PGD pairs would be generated by the explicit formulas (2.31).

Bibliography

[Aguado et al., 2013] Aguado, J. V., Chinesta, F., Leygue, A., Cueto, E., and Huerta, A. (2013). Deim-based pgd for parametric nonlinear model order reduction. In de Almeida, J. P. M., Diez, P., Tiago, C., and Parés, N., editors, *VI International Conference on Adaptive Modeling and Simulation*. ADMOS.

[Allier et al., 2015] Allier, P.-E., Chamoin, L., and Ladevèze, P. (2015). Proper generalized decomposition computational methods on a benchmark problem: introducing a new strategy based on constitutive relation error minimization. *Advanced Modeling and Simulation in Engineering Sciences*, 2(17):2–25.

[Allix and Ladevèze, 1992] Allix, O. and Ladevèze, P. (1992). Interlaminar interface modelling for the prediction of delamination. *Composite Structures*, 22(4):235 – 242.

[Allix et al., 1989] Allix, O., Ladevèze, P., Gilleta, D., and Ohayon, R. (1989). A damage prediction method for composite structures. *International Journal for Numerical Methods in Engineering*, 27(2):271–283.

[Allix and Vidal, 2002] Allix, O. and Vidal, P. (2002). A new multi-solution approach suitable for structural identification problems. *Computer Methods in Applied Mechanics and Engineering*, 191(1):2727–2758.

[Ammar et al., 2012] Ammar, A., Chinesta, F., Cueto, E., and Doblaré, M. (2012). Proper generalized decomposition of time-multiscale models. *International Journal for Numerical Methods in Engineering*, 90(5):569–596.

[Ammar et al., 2006] Ammar, A., Ryckelynck, D., Chinesta, F., and Keunings, R. (2006). On the reduction of kinetic theory models related to finitely extensible dumbbells. *Journal of Non-Newtonian Fluid Mechanics*, 134(1–3):136 – 147. 2nd Annual European Rheology Conference.

[Amsallem et al., 2009] Amsallem, D., Cortial, J., Carlberg, K., and Farhat, C. (2009). A method for interpolating on manifolds structural dynamics reduced-order models. *International Journal for Numerical Methods in Engineering*, 80(1):1241–1258.

[Astrid et al., 2008] Astrid, P., Weiland, S., Willcox, K., and Backx, T. (2008). Missing point estimation in models described by proper orthogonal decomposition. *IEEE Transactions on Automatic Control*, 53(10):2237 – 2251.

[Atwell and Kings, 2001] Atwell, J. A. and Kings, B. B. (2001). Proper orthogonal decomposition for reduced basis feedback controllers for parabolic equations. *Mathematical and computer modelling*, 33:1–19.

[Aubard et al., 2000] Aubard, X., Cluzel, C., Guitard, L., and P., L. (2000). Damage modeling at two scales for 4d carbon/carbon composites. *Computer and Structures*, 78(1-3):83–92.

[Barbarulo et al., 2014] Barbarulo, A., Ladevèze, P., Riou, H., and Kovalevsky, L. (2014). Pgd-vtcr: a reduced order model technique to solve medium frequency broad band problems on complex acoustical systems. *Journal of Mechanical Engineering*, 333:2422–2431.

[Barrault et al., 2004] Barrault, M., Maday, Y., Nguyen, N., and Patera, A. (2004). An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Académie des Sciences. Paris.*, Ser. I(339):667–672.

[Beltrami, 1873] Beltrami, E. (1873). Sulle funzioni bilineari. *Giornale di Mathematiche*, XI:98–106.

[Bergmann et al., 2009] Bergmann, M., Bruneau, C.-H., and Iollo, A. (2009). Enablers for robust pod models. *Journal of Computational Physics*, (228):516–538.

[Bissantz et al., 2009] Bissantz, N., Dümbgen, L., Munk, A., and Stratmann, B. (2009). Convergence analysis of generalized iteratively reweighted least squares algorithms on convex function spaces. *SIAM J. on Optimization*, 19(4):1828–1845.

[Blanc et al., 2005] Blanc, X., Le Bris, C., and Legoll, F. (2005). Analysis of a prototypical multiscale method coupling atomistic and continuum mechanics. *ESAIM: Mathematical Modelling and Numerical Analysis*, 39:797–826.

[Blanze et al., 1996] Blanze, C., Champaney, L., Cognard, J., and Ladevèze, P. (1996). A modular approach to structure assembly computations: application to contact problems. *Engineering Computations*, 13(1):15–32.

[Bognet et al., 2012] Bognet, B., Bordeu, F., Chinesta, F., Leygue, A., and Poitou., A. (2012). Advanced simulation of models defined in plate geometries: 3d solutions with 2d computational complexity. *Computational Methods Applied Mechanical Engineering*, (201):1–12.

[Boisse et al., 1990] Boisse, P., Bussy, P., and Ladevèze, P. (1990). A new approach in non-linear mechanics: the large time increment method. *International Journal for Numerical Methods in Engineering*, 29:647–663.

[Bonithon and Nouy, 2012] Bonithon, G. and Nouy, A. (2012). Tensor-based methods and proper generalized decompositions for the numerical solution of high dimensional problems: alternative definitions.

[Boucard and Ladevèze, 1999] Boucard, P.-A. and Ladevèze, P. (1999). A multiple solution method for non-linear structural mechanics. *Mechanical Engineering*, 50(5):317–328.

[Boucard et al., 1997] Boucard, P.-A., Ladeveze, P., Pass, M., and Rougée, P. (1997). A nonincremental approach for large displacement problems. *Computers and Structures*, 64(1):499–508.

[Boucinha et al., 2014] Boucinha, L., Ammar, A., Gravouil, A., and Nouy, A. (2014). Ideal minimal residual-based proper generalized decomposition for non-symmetric multi-field models – application to transient elastodynamics in space-time domain. *Computer Methods in Applied Mechanics and Engineering*, 273(1):56–76.

[Boucinha et al., 2013] Boucinha, L., Gravouil, A., and Ammar, A. (2013). Space–time proper generalized decompositions for the resolution of transient elastodynamic models. *Computer Methods in Applied Mechanics and Engineering*, 255:67 – 88.

[Briggs et al., 2000] Briggs, W., Henson, V., and McCormick, S. (2000). *A Multigrid Tutorial, Second Edition*. Society for Industrial and Applied Mathematics, second edition.

[Buffa et al., 2012] Buffa, A., Maday, Y., Patera, A., Prud'homme, C., and Turinici, G. (2012). A priori convergence of the greedy algorithm for the parametrized reduced basis. *Mathematical Modelling and Numerical Analysis*, 46(3):595–603.

[Carlberg et al., 2010] Carlberg, K., Bou-Mosleh, C., and Farhat, C. (2010). Efficient nonlinear model reduction via a least-squares petrov-galerkin projection and compressive tensor approximations. *International Journal for Numerical Methods in Engineering*, 86(1):155–181.

[Carlberg et al., 2013] Carlberg, K., Farhat, C., Cortial, J., and Amsallem, D. (2013). The gnat method for nonlinear model reduction : Effective implementation and application to computational fluid dynamics and turbulent flows. *Journal of Computational Physics*, (242):623–647.

[Champaney et al., 1999] Champaney, L., Cognard, J., and Ladevèze, P. (1999). Modular analysis of assemblages of three-dimensional structures with unilateral contact conditions. *Computers and Structures*, 73(1–5):249 – 266.

[Champaney et al., 1997] Champaney, L., Cognard, J. Y., Dureisseix, D., and Ladevèze, P. (1997). Large scale applications on parallel computers of a mixed domain decomposition method. *Computational Mechanics*, 19(4):253–263.

[Chatterjee, 2000] Chatterjee, A. (2000). An introduction to the proper orthogonal decomposition. *Computational Science*, special section tutorial.

[Chaturentabut and Sorensen, 2010] Chaturentabut, S. and Sorensen, D. (2010). Nonlinear model reduction via discrete empirical interpolation. *Society for Industrial and Applied Mathematics*, 32(5):2737–2764.

[Chinesta et al., 2010] Chinesta, F., Ammar, A., and Cueto, E. (2010). Proper generalized decomposition of multiscale models. *International Journal for Numerical Methods in Engineering*, 83(8-9):1114–1132.

[Chinesta and Ladevèze, 2014] Chinesta, F. and Ladevèze, P. (2014). *Separated Representations and PGD-Based Model Reduction: Fundamentals and Applications*. CISM International Centre for Mechanical Sciences. Springer Vienna.

[Chinesta et al., 2011] Chinesta, F., Ladevèze, P., and Cueto, E. (2011). A short review on model order reduction based on proper generalized decomposition. *Arch Comput Methods Eng*, (18):395–404.

[Cognard et al., 1999] Cognard, J., Ladeveze, P., and Talbot, P. (1999). A large time increment approach for thermo-mechanical problems. *Advances in engineering software*, 30:583–593.

[Cognard and Ladevèze, 1993] Cognard, J.-Y. and Ladevèze, P. (1993). A large time increment approach for cyclic viscoplasticity. *International Journal of Plasticity*, 9:141–157.

[Cremonesi et al., 2013] Cremonesi, M., Néron, D., Guidault, P.-A., and Ladevèze, P. (2013). A pgd-based homogenization technique for the resolution of nonlinear multiscale problems. *Computational Methods Applied Mechanical Engineering*, 267(1):275–292.

[Devries et al., 1989] Devries, F., Dumontet, F. Duvaut, G., and Léné, F. (1989). Homogenization and damage for composite structures. *International Journal for Numerical Methods in Engineering*, 27:285–298.

[Dureisseix and Ladevèze, 1998] Dureisseix, D. and Ladevèze, P. (1998). A 2-level and mixed domain decomposition approach for structural analysis. *Contemporary Mathematics*, 218(1):238–245.

[Dureisseix et al., 2003] Dureisseix, D., Ladevèze, P., Néron, D., and Schrefler, B. (2003). A multi-time-scale strategy for multiphysics problems: Application to poroelasticity. *International Journal for Multiscale Computational Engineering*, 1(4):387–400.

[Eckart and Young, 1939] Eckart, C. and Young, G. (1939). A principal axis transformation for non-hermitian matrices. *Bulletin of the American Mathematical Society*, 45:118–121.

[Efendiev et al., 2013] Efendiev, Y., Kronsbein, C., and Legoll, F. (2013). Multilevel monte carlo approaches for numerical homogenization.

[Epureanuand et al., 2001] Epureanuand, B. I., Hall, K., and Dowell, E. H. (2001). Reduced-order models of unsteady viscous flows in turbomachinery using viscous-inviscid coupling. *Journal of Fluids and Structures*, (15):255–273.

[Everson and Sirovich, 1995] Everson, R. and Sirovich, L. (1995). Karhunen-loève procedure for gappy data. *Journal of Optical Society of America*, 12(8):1657–1664.

[Falco and Nouy, 2010] Falco, A. and Nouy, A. (2010). A proper generalized decomposition for the solution of elliptic problems in abstract form by using a functional eckart-young approach. *Journal of Mathematical Analysis and applications*.

[Farhat and Roux, 1991] Farhat, C. and Roux, F.-X. (1991). A method of finite element tearing and interconnecting and its parallel solution algorithm. *International Journal for Numerical Methods in Engineering*, 32(6):1205–1227.

[Feyel, 2003] Feyel, F. (2003). A multilevel finite element method (fe2) to describe the response of highly non-linear structures using generalized continua. *Computer Methods in Applied Mechanics and Engineering*, 192(28–30):3233 – 3244. Multiscale Computational Mechanics for Materials and Structures.

[Fink and Rheinboldt, 1983] Fink, J. and Rheinboldt, W. (1983). On the error behavior of the reduced basis technique for nonlinear finite element approximations. *Journal of Applied Mathematics and Mechanics*, 63(1):21–28.

[Fish and Belsky, 1995] Fish, J. and Belsky, V. (1995). Multi-grid method for periodic heterogeneous media part 2: Multiscale modeling and quality control in multidimensional case. *Computer Methods in Applied Mechanics and Engineering*, 126(1–2):17 – 38.

[Fish et al., 1997] Fish, J., Shek, K., Pandheeradi, M., and Shephard, M. S. (1997). Computational plasticity for composite structures based on mathematical homogenization: Theory and practice. *Computer Methods in Applied Mechanics and Engineering*, 148(1–2):53 – 73.

[Fritzen and Leuschner, 2015] Fritzen, F. and Leuschner, M. (2015). Nonlinear reduced order homogenization of materials including cohesive interfaces. *Computational Mechanics*, 56:131–151.

[Galbally et al., 2010] Galbally, D., Fidkowski, K., Willcox, K., and Ghattas, O. (2010). Non-linear model reduction for uncertainty quantification in large-scale inverse problems. *International Journal for Numerical Methods in Engineering*, 81(1):1581–1608.

[Galvis and Kang, 2014] Galvis, J. and Kang, S. K. (2014). Spectral multiscale finite element for nonlinear flows in highly heterogeneous media: A reduced basis approach. *Journal of Computational and Applied Mathematics*, 260(1):494–508.

[Ghnatios et al., 2012] Ghnatios, C., Masson, F., Huerta, A., Leygue, A., Cueto, E., and Chinesta, F. (2012). Proper generalized decomposition based dynamic data-driven control of thermal processes. *Computational Methods in Applied Mechanical Engineering*, 213:29–41.

[Giacoma et al., 2014] Giacoma, A., Dureisseix, D., Gravouil, A., and Rochette, M. (2014). A multiscale large time increment/fas algorithm with time-space model reduction for frictional contact problems. *International Journal for Numerical Methods in Engineering*, 97(3):207–230.

[Giacoma et al., 2015] Giacoma, A., Dureisseix, D., Gravouil, A., and Rochette, M. (2015). Toward an optimal a priori reduced basis strategy for frictional contact problems with latin solver. *Computer Methods in Applied Mechanics and Engineering*, 283:1357 – 1381.

[Glüsmann and Kreuzer, 2009] Glüsmann, P. and Kreuzer, E. (2009). On the application of karhunen–loève transform to transient dynamic systems. *Journal of Sound and Vibration*, 328(4-5):507–519.

[González et al., 2010] González, D., Ammar, A., Chinesta, F., and Cueto, E. (2010). Recent advances on the use of separated representations. *International Journal for Numerical Methods in Engineering*, 81(5):637–659.

[Gonzalez et al., 2012] Gonzalez, D., Masson, F., Poulhaon, F., Leygue, A., Cueto, E., and Chinesta, F. (2012). Proper generalized decomposition based dynamic data driven inverse identification. *Mathematics and Computers in Simulation*, 82(9):1677–1695.

[Grepl et al., 2007] Grepl, M., Maday, Y., Nguyen, N., and Patera, A. (2007). Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations. *Modélisation mathématique et analyse numérique*, 41(3):575–605.

[Guidault et al., 2008] Guidault, P.-A., Allix, O., Champaney, L., and Cornuault, C. (2008). A multi-scale extended finite element method for crack propagation. *Computer Methods in Applied Mechanics and Engineering*, 197(5):381–399.

[Hernandez et al., 2014] Hernandez, J., Oliver, J., Huespe, A., Caicedo, M., and Cante, J. (2014). High-performance model reduction techniques in computational multiscale homogenization. *Computer Methods in Applied Mechanics and Engineering*, 276(1):149–189.

[Heyberger et al., 2011] Heyberger, C., Boucard, P.-A., and Néron, D. (2011). Multiparametric analysis whitin the proper generalized decomposition framework. *Computational Mechanics*, 49(3):277–289.

[Hughes et al., 1998] Hughes, T. J., Feijóo, G. R., Mazzei, L., and Quincy, J.-B. (1998). The variational multiscale method—a paradigm for computational mechanics. *Computer Methods in Applied Mechanics and Engineering*, 166(1–2):3–24. Advances in Stabilized Methods in Computational Mechanics.

[Ibrahimbegovic and Melnyk, 2007] Ibrahimbegovic, A. and Melnyk, S. (2007). Embedded discontinuity finite element method for modeling of localized failure in heterogeneous materials with structured mesh: an alternative to extended finite element method. *Computational Mechanics*, 40(1):149–155.

[Jordan, 1874] Jordan, C. (1874). Mémoire sur les formes bilinéaires. *Journal de mathématiques pures et appliquées 2e série*, 19:35–54.

[Karhunen, 1943] Karhunen (1943). Uber lineare methoden fur wahrscheinigkeitsrechnung. *Annales of Academic Science Fennicae Series A1 Mathematical Physics*, 37:3–79.

[Kerschen et al., 2005] Kerschen, G., Golinval, J., Vakakis, A., and Bergman, L. (2005). The method of proper orthogonal decomposition for dynamical characterization and order reduction of mechanical systems: An overview. *Nonlinear Dynamics*, 1(41):147–169.

[Kunish and Xie, 2005] Kunish, K. and Xie, L. (2005). Pod-based feedback control of the burgers equation by solving the evolutionary hjb equation. *Computers and Mathematics With Applications*, 49(7-8):5730 – 5742.

[Ladevèze, 1985] Ladevèze, P. (1985). Sur une famille d'algorithmes en mécanique des structures. *Comptes Rendus Académie des Sciences. Paris.*, Ser. II(300):41–44.

[Ladevèze, 1989] Ladevèze, P. (1989). La méthode à grand incrément de temps pour l'analyse de structures à comportement non linéaire décrit par variables internes. *Comptes-rendus Académie des Sciences*, série II(11):1095–1099.

[Ladevèze, 1997] Ladevèze, P. (1997). A computational technique for the integrals over the time-space domain in connection with the latin method (in french). Technical Report 193, LMT Cachan.

[Ladevèze, 1999] Ladevèze, P. (1999). *Nonlinear Computational Structural Mechanics - new approaches and non-incremental methods of calculation.* XII. F. F. Ling.

[Ladevèze and Chamoin, 2011] Ladevèze, P. and Chamoin, L. (2011). On the verification of the model reduction methods based on the proper generalized decomposition. *Computer Methods in Applied Mechanics and Engineering*, 200:2032–2047.

[Ladevèze and Dureisseix, 2000] Ladevèze, P. and Dureisseix, D. (2000). A micro / macro approach for parallel computing of heterogeneous structures. *International Journal for Computational Civil and Structural Engineering*, 1:18–28.

[Ladevèze et al., 2001] Ladevèze, P., Loiseau, O., and Dureisseix, D. (2001). A micro-macro and parallel computational strategy for highly heterogeneous structures. *International Journal for Numerical Methods in Engineering*, (52):121–138.

[Ladevèze and Lorong, 1991] Ladevèze, P. and Lorong, P. (1991). A large time increment approach with domain decomposition for mechanical non linear problem. In *Proceedings of the 10th International Conference on Computing Methods in Applied Sciences and Engineering on Computing Methods in Applied Sciences and Engineering*, pages 569–578, Commack, NY, USA. Nova Science Publishers, Inc.

[Ladevèze et al., 2006] Ladevèze, P., Néron, D., and Gosselet, P. (2006). On a mixed and multiscale domain decomposition method. *Comput. Methods Appl. Mech. Engrg.*, 196(1):1526–1540.

[Ladevèze and Nouy, 2003] Ladevèze, P. and Nouy, A. (2003). On a multiscale computational strategy with time and space homogenization for structural mechanics. *Computational Methods Applied Mechanical Engineering*, (192):3061–3087.

[Ladevèze et al., 2009] Ladevèze, P., Passieux, J.-C., and Néron, D. (2009). *On multiscale computational mechanics with time-space homogenization*, pages 247–282. Oxford University Press.

[Ladevèze et al., 2010] Ladevèze, P., Passieux, J.-C., and Néron, D. (2010). The latin multiscale computational method and the proper generalized decomposition. *Computational Methods Applied Mechanical Engineering*, (199):1287–1296.

[Legoll, 2009] Legoll, F. (2009). Multiscale methods coupling atomistic and continuum mechanics: some examples of mathematical analysis. In *Analytical and Numerical Aspects of Partial Differential Equations*, Proceedings in Mathematics, pages 193–245. de Gruyter.

[Liu et al., 2010] Liu, Y., Sheng, J., and Ding, R. (2010). Convergence of stochastic gradient estimation algorithm for multivariable arx-like systems. *Computers and Mathematics with Applications*, 59(8):2615 – 2627.

[Loeve, 1955] Loeve, M. (1955). *Probability Theory: Foundations, Random Sequences*. Pennsylvania state university edition.

[Maday, 2006] Maday, Y. (2006). Reduced basis method for the rapid and reliable solution of partial differential equations. In *Proceedings of International Conference of Mathematicians, Madrid. European Mathematical Society Eds.*

[Maday et al., 2009] Maday, Y., Nguyen, N. C., Patera, A. T., and Pau, G. S. H. (2009). A general multiporpuse interpolation procedure: The magic points. *Communications in pure and applied analysis*, 8(1):383–404.

[Maday and Ronquist., 2004] Maday, Y. and Ronquist., E. M. (2004). The reduced-basis element method: application to a thermal fin problem. *SIAM Journal on Scientific Computing*, 26(1):240–258.

[Mandel, 1993] Mandel, J. (1993). Balancing domain decomposition. *Communications in Numerical Methods in Engineering*, 9(3):233–241.

[Michel and Suquet, 2003] Michel, J. C. and Suquet, P. (2003). Nonuniform transformation field analysis. *International Journal of Solids Structures*, 40:6937–6955.

[Michel-Ponnelle, 2001] Michel-Ponnelle, S. (2001). *Modélisation et simulation des structures élastomères endommageables en grandes transformations*. PhD thesis, Ecole Normale Supérieure de Cachan, 61 av. du Président Wilson, F-94230 Cachan France.

[Miled et al., 2013] Miled, B., Ryckelynck, D., and Cantournet, S. (2013). A priori hyper-reduction method for coupled viscoelastic-viscoplastic composites. *Computer and Structures*, 119:95–103.

[Mokdad et al., 2007] Mokdad, B., Pruliere, E., Ammar, A., and Chinesta, F. (2007). On the simulation of kinetic theory models of complex fluids using the fokker-planck approach. *Applied Rheology*, 17(2):26494, 1–4.

[Monserrat et al., 2001] Monserrat, C., Meier, U., Alcaniz, M., Chinesta, F., and Juan, M. (2001). A new approach for the real-time simulation of tissue deformations in surgery simulation. *Computer Methods and Programs in Biomedicine*, 64(2):77–85.

[Néron et al., 2015] Néron, D., Boucard, P.-A., and Relun, N. (2015). Time-space pgd for the rapid solution of 3d nonlinear parametrized problems in the many-query context. *International Journal for Numerical Methods in Engineering*.

[Néron and Dureisseix, 2008] Néron, D. and Dureisseix, D. (2008). A computational strategy for poroelastic problems with a time interface between coupled physics. *International Journal for Numerical Methods in Engineering*, 73(6):783–804.

[Néron and Ladevèze, 2010] Néron, D. and Ladevèze, P. (2010). Proper generalized decomposition for multiscale and multiphysics problems. *Archives of Computational Methods In Engineering*, 17:351–372.

[Néron and Ladevèze, 2012] Néron, D. and Ladevèze, P. (2012). A data compression approach for pgd reduced-order modeling. In ASME, editor, *11th Biennial Conference On Engineering Systems Design And Analysis*. ESDA 2012.

[Nguyen, 2008] Nguyen, N. (2008). A multiscale reduced-basis method for parametrized elliptic partial differential equations with multiple scales. *Journal of Computational Physics*, 227(1):9807–9822.

[Nguyen et al., 2005] Nguyen, N., Veroy, K., and Patera, A. (2005). *Certified real-time solution of parametrized partial differential equations*, volume Handbook of materials modeling, pages 1523 – 1558. Springer, Berlin.

[Nguyen, 2005] Nguyen, N. C. (2005). *Reduced-Basis Approximations and A Posteriori Error Bounds for Nonaffine and Nonlinear Partial Differential Equations: Application to Inverse Analysis*. PhD thesis, HCMC University of Technology.

[Nguyen and Peraire, 2008] Nguyen, N. C. and Peraire, J. (2008). An efficient reduced-order modeling approach for non-linear parametrized partial differential equations. *International Journal for Numerical Methods in Engineering*, 76:27–55.

[Niroomandi et al., 2010] Niroomandi, S., Alfaro, I., Cueto, E., and Chinesta, F. (2010). Model order reduction for hyperelastic materials. *International Journal for Numerical Methods in Engineering*, 81(9):1180–1206.

[Niroomandi et al., 2008] Niroomandi, S., Alfaro, I., Cueto, E., and et al. (2008). Real-time deformable models of non-linear tissues by model reduction techniques. *Computer Methods and Programs in Biomedicine*, 91(3):223–231.

[Niroomandi et al., 2012a] Niroomandi, S., Alfaro, I., Gonzalez, D., Cueto, E., and Chinesta, F. (2012a). Model order reduction in hyperelasticity: a proper generalized decomposition approach. *International Journal for Numerical Methods in Engineering*, 00(1):1–28.

[Niroomandi et al., 2012b] Niroomandi, S., Alfaro, I., Gonzalez, D., Cueto, E., and Chinesta, F. (2012b). Real time simulation of surgery by reduced order modeling and x-fem techniques. *International Journal of Numerical Methods in Biomedical Engineerings*, 28(5):574–588.

[Noor and Peters, 1980] Noor, A. K. and Peters, J. M. (1980). Reduced basis technique for nonlinear analysis of structures. *AIAA Journal*, 18(1):455–462.

[Nouy, 2009] Nouy, A. (2009). Recent developments in spectral stochastic methods for the numerical solution of stochastic partial differential equations. *Archives of Computational Methods In Engineering*, 16(3):251–285.

[Nouy, 2010] Nouy, A. (2010). A priori model reduction through proper generalized decomposition for solving time-dependent partial differential equations. *Computational Methods Applied Mechanical Engineering*, 199(23-24):1603–1626.

[Oden et al., 1999] Oden, J. T., Vemaganti, K., and Moës, N. (1999). Hierarchical modeling of heterogeneous solids. *Computer Methods in Applied Mechanics and Engineering*, 172(1–4):3 – 25.

[Passieux et al., 2010] Passieux, J.-C., Ladevèze, P., and Néron, D. (2010). A scalable time-space multiscale domain decomposition method: adaptive time scales separation. *Computational Mechanics*, 46(4):621–633.

[Pelle and Ryckelynck, 2000] Pelle, J.-P. and Ryckelynck, D. (2000). An efficient adaptive strategy to master the global quality of viscoplastic analysis. *Computers and Structures*, 78:169–183.

[Prud'homme et al., 2002] Prud'homme, C., Rovas, D., Veroyand, K., Machiels, L., Maday, Y., Patera, A., and Turinici, G. (2002). Reliable real-time solution of parametrized partial differential equations: Reduced-basis output bound methods. *Journal of Fluids Engineering*, 124(1):70–80.

[Relun et al., 2011] Relun, N., Néron, D., and Boucard, P. (2011). A model reduction technique based on the pgd for elastic-viscoplastic computational analysis. *Computational Mechanics*.

[Rewienski and White, 2006] Rewienski, M. and White, J. (2006). Model order reduction for nonlinear dynamical systems based on trajectory piecewise-linear approximations. *Linear Algebra and its Applications*, 415(1):426–454.

[Rheinboldt, 1993] Rheinboldt, W. (1993). On the theory and error estimation of the reduced basis method for multi-parameter problems. *Nonlinear Analysis: Theory, Methods and Applications*, 21:849–858.

[Roulet et al., 2011] Roulet, V., Champaney, L., and Boucard, P.-A. (2011). A parallel strategy for the multiparametric analysis of structures with large contact and friction surfaces. *Advances in Engineering Software*, 42(6):347 – 358.

[Rozza, 2004] Rozza, G. (2004). Reduced-basis methods for elliptic equations in sub-domains with a posteriori error bounds and adaptivity. *Applied Numerical Mathematics*, (55):403–424.

[Rozza, 2006] Rozza, G. (2006). Reduced basis methods for stokes equations in domains with non-affine parameter dependence. *Computing and Visualization in Science*, (12):23–35.

[Rozza, 2011] Rozza, G. (2011). Reduced basis approximation and error bounds for potential flows in parametrized geometries. *Communications In Computational Physics*, 9(1):1–48.

[Rozza and Patera, 2008] Rozza, G. and Patera, A. T. (2008). Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations. *Archives of Computational Methods In Engineering*, (15):229–275.

[Ryckelynck, 2002] Ryckelynck, D. (2002). Réduction a priori de modèles thermomécaniques. *Compte Rendu Mecanique*, 330:499–505.

[Ryckelynck, 2005] Ryckelynck, D. (2005). A priori hypereduction method: an adaptive approach. *International Journal of Computational Physics*, 202(1):346–366.

[Ryckelynck, 2009] Ryckelynck, D. (2009). Hyper-reduction of mechanical models involving internal variables. *International Journal for Numerical Methods in Engineering*, 77(1):75–89.

[Ryckelynck and Benziane, 2010] Ryckelynck, D. and Benziane, D. M. (2010). Multi-level a priori hyper-reduction of mechanical models involving internal variables. *Computer Methods in Applied Mechanics and Engineering*, 199(17–20):1134 – 1142.

[Ryckelynck et al., 2012] Ryckelynck, D., Vincent, F., and Cantournet, S. (2012). Multidimensional a priori hyper-reduction of mechanical models involving internal variables. *Computer Methods in Applied Mechanics and Engineering*, (225-228):28 – 43.

[Sanchez-Palencia, 1974] Sanchez-Palencia, E. (1974). Comportements local et macroscopique d'un type de milieux physiques hétérogène. *International Journal of Engineering Science*, 12:331–351.

[Schmidt, 1907] Schmidt, E. (1907). Zur theorie der linearen und nichtlinearen integral gleichungen. i teil. entwicklung will kiirlichen funktionen nach system vorgeschriebener. *Mathematische Annalen*, 63:433–476.

[Sylvester, 1889] Sylvester, J. J. (1889). A new proof that a general quadric may be reduced to its canonical form (that is, a linear function of squares) by means of a real orthogonal substitution. *Messenger of Mathematics*, 19:1–5.

[Veroy and Patera, 2005] Veroy, K. and Patera, A. T. (2005). Certified real-time solution of the parametrized steady incompressible navier-stokes equations: rigorous reduced-basis a posteriori error bounds. *International Journal for Numerical Methods in Fluids*, 47(1):773–788.

[Willcox, 2006] Willcox, K. (2006). Unsteady flow sensing and estimation via the proper orthogonal decomposition. *Computers and Fluids*, 35(2):208–226.

[Wu et al., 2003] Wu, C., Liang, Y., Lin, W., Lee, H., and Lim, S. (2003). A note on equivalence of proper orthogonal decomposition methods. *Journal of Sound and Vibration*, (265):1103–1110.

[Zhao and Ling, 2003] Zhao, H. and Ling, G. (2003). Low-dimensional dynamical systems for a wake-type shear flow with vortex dislocations. *Parallel Computational Fluid Dynamics*, 33(3):299–312.

[Zimmermann, 2012] Zimmermann, R. (2012). Gradient-enhanced surrogate modeling based on proper orthogonal decomposition. *Journal of Computational and Applied Mathematics*, (237):403–418.

Titre : Un nouveau cadre d'approximation dédié à la strategie de calcul PGD pour problèmes non-linéaires

Mots clés : Réduction des modèles, Méthode LaTIn, PGD POD et reduced-basis, Analyse multi-échelle, méthode d'hyper-reduction, méthode par points de référence

Résumé : Le but de ce travail est d'introduire un cadre d'approximation, la Reference Points Method, afin de réduire la complexité de calcul des opérations algébriques lorsqu'elles concernent des approximations à variables séparées dans le cadre de la Proper Generalized Decomposition. La PGD a été introduite dans le cadre de la méthode LaTIn pour résoudre efficacement des équations différentielles non linéaires et dépendantes du temps en mécanique des structures. La technique consiste à chercher la solution d'un problème dans une base d'ordre réduit (ROB) qui est générée automatiquement et à la volée par la méthode LaTIn. Les techniques de réduction de modèle sont particulièrement efficaces lorsque le ROM a besoin d'être construit qu'une seule fois. Ce n'est pas le cas pour des problèmes non linéaires. En effet, dans un tel cas, les opérateurs qui sont impliqués dans la construction du ROM varient au cours du processus itératif et des calculs préliminaires ne peuvent pas être effectués à l'avance pour accélérer le processus 'online'. Par conséquent, la construction du ROM est un élément coûteux de la stratégie en terme de temps de calcul. Il en découle la nécessité d'évaluer, à chaque itération, la fonction non linéaire de grande dimension (et éventuellement sa jacobienne) et ensuite sa projection pour obtenir les opérateurs réduits. Cela représente un point de blocage des stratégies de réduction de modèle dans le cadre non linéaire. Le présent travail a comme but une réduction ultérieure du coût de calcul, grâce à l'introduction d'un nouveau cadre d'approximation dédiée à la strategie de calcul LaTIn-PGD. Il est basé sur la notion de temps, de points et de paramètres de référence et permet de définir une version compressée des données. Le RPM est introduit dans le solveur LaTIn-PGD non linéaire pour calculer certaines opérations répétitives. Ces opérations sont liées à la résolution du problème du temps/paramètre qui implique la mise à jour de l'opérateur tangent et la projection de ce dernier sur la base réduite. La RPM permet de simplifier et de réduire le nombre d'opérations nécessaires.

Title : A new approximation framework for PGD-based nonlinear solvers

Keywords : Reduced-order modeling, LaTIn method, PGD; POD and reduced-basis, Multi-scale analysis, hyper-reduction method, Reference Points Method

Abstract : The aim of this work is to introduce an approximation framework, called Reference Points Method (RPM), in order to decrease the computational complexity of algebraic operations when dealing with separated variable approximations in the Proper Generalized Decomposition (PGD) framework.
The PGD has been introduced in the context of the LATIN method to solve efficiently time dependent and/or parametrized nonlinear partial differential equations in structural mechanics. Roughly, the PGD technique consists in seeking the solution of a problem in a relevant Reduced-Order Basis (ROB) which is generated automatically and on-the-fly by the LATIN method. However, model reduction techniques are particularly efficient when the ROM needs one construction only. This is not the case for the model reduction techniques when they are addressed to nonlinear problems. Indeed, in such a case, the operators which are involved in the construction of the ROM change all along the iterative process and no preliminary computations can be performed in advance to speed up the online process. The Hence, the construction of the ROM is an expensive part of the calculation strategy in terms of CPU. It ensues from the need to evaluate the high-dimensional nonlinear function (and eventually its Jacobian) and then to project it to get the low-dimensional operators at each computational step of a solution algorithm. This amounts to being the bottleneck of nonlinear model reduction strategies. The present work is then focused on a further reduction of the computational cost, thanks to the introduction of a new approximation framework dedicated to PGD-based nonlinear solver. It is based on the concept of reference times, points and parameters and allows to define a compressed version of the data. The RPM is introduced in the PGD-based nonlinear solver to compute some repetitive operations. These operations are related to the resolution of the time/parameter problem that involves the update of the tangent operator (for nonlinear problems) and the projection of this latter on the Reduced Order Basis. For that the RPM allows to simplify and reduce the number of operations needed.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France