Integration of PGD-virtual charts into an engineering design process

Amaury Courard, David Néron, Pierre Ladevèze, Ludovic Ballère

Integration of PGD-virtual charts into an engineering design process

Amaury Courard[1] · **David Néron**[2] · **Pierre Ladevèze**[2] · **Ludovic Ballere**[1]

Abstract This article deals with the efficient construction of approximations of fields and quantities of interest used in geometric optimisation of complex shapes that can be encountered in engineering structures. The strategy, which is developed herein, is based on the construction of virtual charts that allow, once computed offline, to optimise the structure for a negligible online CPU cost. These virtual charts can be used as a powerful numerical decision support tool during the design of industrial structures. They are built using the proper generalized decomposition (PGD) that offers a very convenient framework to solve parametrised problems. In this paper, particular attention has been paid to the integration of the procedure into a genuine engineering design process. In particular, a dedicated methodology is proposed to interface the PGD approach with commercial software.

Keywords Model reduction · PGD · Geometric parameters · Virtual chart · Shape optimisation

1 Introduction

Due the rapid and constant increase in computing power, particularly with the development of High Performance Computing, it is possible to run simulations of complex structures, which was unimaginable a few decades ago. It is, then, possible for some structures to run a parametric simulation (for all the values of parameters) and to choose the right set of parameters for the structure to design afterwards.

Nevertheless the tendency nowadays is to introduce more and more physics into modelling, i.e. dealing with very complex material laws, geometric non-linearities due, especially, to the development of composite materials in many engineering fields (aeronautics, automotive...). Simulations of these models can take a substantial amount of time (weeks, months). Thus considering each new structure, i.e. a new set of parameters, as a new problem during the design stage leads to highly expensive simulations. Consequently, structures tend to be oversized in order to reduce the design time (the design stage is stopped before finding the optimal set of parameters) which is an issue, for instance, in aeronautics where weight reduction is an engineering challenge.

To fix ideas, the optimisation problem would be the minimisation of the mass M of an aeronautical structure with respect to a set of design parameters $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n) \in \boldsymbol{\mathcal{A}} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_n \subset \mathbb{R}^n$. Meanwhile, the Von Mises stress σ_{VM} must remain under a given threshold σ_0 in order to guaranty the safety of the structure. The constrained optimisation problem can be written as follows: Find $\boldsymbol{\alpha}^* = \left(\alpha_1^*, \ldots, \alpha_n^*\right) \in \boldsymbol{\mathcal{A}} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_n$ such that

$$\boldsymbol{\alpha}^* = arg \min_{\substack{\boldsymbol{\alpha} \in \boldsymbol{\mathcal{A}} \\ \sigma_{VM}(\boldsymbol{\alpha}) \leqslant \sigma_0}} M(\boldsymbol{\alpha}) \tag{1}$$

which can be rewritten by restraining the space $\boldsymbol{\mathcal{A}}$ to $\boldsymbol{\mathcal{A}}_{\sigma_0}$, the space where the parameters associated to structures, whose Von Mises stress do not exceed σ_0, belong. Hence, the optimisation problem becomes: Find $\boldsymbol{\alpha}^* = \left(\alpha_1^*, \ldots, \alpha_n^*\right) \in \boldsymbol{\mathcal{A}}_{\sigma_0}$ such that

$$\boldsymbol{\alpha}^* = arg \min_{\boldsymbol{\alpha} \in \boldsymbol{\mathcal{A}}_{\sigma_0}} M(\boldsymbol{\alpha}) \tag{2}$$

✉ David Néron
neron@lmt.ens-cachan.fr

[1] AIRBUS Defence & Space, B.P. 20011 33165, Saint-Médard-en-Jalles Cedex, France

[2] LMT, ENS Cachan, CNRS, Université Paris-Saclay, 61 avenue du Président Wilson, 94235 Cachan Cedex, France

However, the difficulty lies in the definition of the search space $\mathcal{A}_{\sigma_0}$ because it implies evaluating the stress field $\boldsymbol{\sigma}$ for all $\boldsymbol{\alpha} \in \mathcal{A}$, which can be out-of-reach of classic methods in terms of computation time.

The aim of this work is not to derive any new approach on the optimisation process itself because numerous algorithms are already available but to propose a strategy to build efficiently the fields needed for evaluation (stress, strain, displacement, quantities of interest, etc.)

It is worth noticing that a non negligible part of structural design is repetitive since many structures are similar in shape. So it could be helpful to store the results obtained for previous structures to use them for new ones. That is why the idea of handbooks is still of topical interest. Handbooks were, for centuries, powerful decision support tools in many fields such as architecture, engineering (*Vademecum des Mechaniker* [7] by Christoph Bernoulli) and science in general.

A novel idea is to update the notion of handbooks to current engineering requirements. These new kind of handbooks are called *virtual charts* [17] or *computational vademecum* [11]. Varying the values of a set of geometric parameters, it is possible to construct a whole family of structures from a reference one. Results of numerical simulations are stored once and for all in virtual charts for families of structures. Hence the end-user has solely to seek the sets of parameters that meet the specifications among all the sets considered in a certain range. The construction of a virtual chart may be time consuming but, since it is done once and for all, it is worth loosing time at this stage with regards to the one won during the practical use, above all for repetitive tasks.

However, as it was previously mentioned, the construction of virtual charts is still out of reach of standard Finite Elements approaches using "brute force" due to the prohibitive computation time.

The path, which is followed in this work, is to build the charts by mean of a reduced-order modelling technique. The community of model reduction is very active and many strategies are, nowadays, available. One of the most popular approaches relies on the use of the proper orthogonal decomposition (POD), which is based on some preliminary computations, called *snapshots*, of the high-fidelity problem for given values of the parameters (see e.g. [6,15,19,20,31]). The reduced basis method (RB, see e.g. [27,30]) adds an automatic selection of these snapshots by a greedy algorithm based on some error indicators. Finally, the (PGD, see e.g. [4,16] or [10] for a review of the method) follows a different path as it builds progressively an approximated separated representation of the solution, without assuming any snapshot or basis.

The PGD was considered, in our former works, in the context of the LATIN method [16], in particular for the analysis of elastic-viscoplastic problems [29], multiscale problems [12], multimodel problems [22], multiphysics problems [24] or parametrised studies [23]. The PGD has also been widely developed by Chinesta and co-authors, who proposed very efficient implementations of the method (see [9] for an overview). To focus only on some very recent works, one can cite the examples of real-time simulations in surgery [2,14], real-time monitoring of thermal process [1] or the simulation of viscoelastic models [5].

Concerning the case of geometric variations for shape optimization, the literature in the field of model reduction is poorer as it involves some technical difficulties that will be discussed along this paper. Some POD based approaches deal with this issue such as [21,30], where geometric parameters are taken into account in the RB framework and [28], where the authors handled it through a combination of POD (called Principal Component Analysis here) and Diffusion approximation. The PGD approach is, obviously, a natural framework to deal with geometric variations by handling geometric parameters in the separated variables decomposition and then solve the problem for any set. The first demonstration of this approach has been done in [3] on rather academic geometries of thermal problems and reused in [32] very recently.

The aim of the present work is to address more complex problems encountered in industry such as axisymmetric geometries and shapes defined by non-straight lines such as splines and to take care of the implementation in the engineering process as well as the coupling with engineering software. In our knowledge, the coupling between PGD and commercial software is quite new in the literature due to the high degree of intrusiveness of the PGD method and marrying the new algorithms with the techniques and tools that are used in engineering design offices is a clearly mandatory. This work is a first demonstrator of what can be envisaged in the future to introduce PGD in industry.

The article is organised as follows: in Sect. 2, we present the engineering case ; in Sect. 3, the notion of *virtual chart* is defined and the Proper Generalized Decomposition is introduced ; Sect. 4 deals with geometric parameters and how the PGD has to be modified to take them into account ; Sect. 5 treats the integration of the PGD into an engineering design process using commercial software ; finally, Sect. 6 exemplified the approach on the demonstrator proposed by AIRBUS Defence & Space. In Appendix, some discussions on the PGD can be found for the novice reader.

2 Presentation of the engineering structure

The problem that will be studied in detail in Sect. 6 is an engineering case. During the design process, different parameters can be played with such as loads, material parameters (Young's modulus, Poisson's ratio...) and geometric parameters. For our study, we focus on geometric parameters.

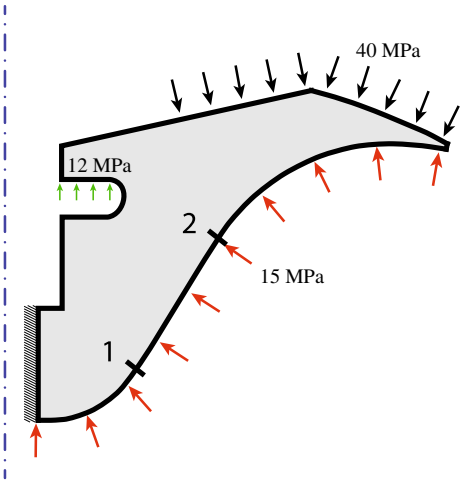

Fig. 1 Considered structure

As shown in Fig. 1, the considered structure is an axisymmetric geometry in isotropic, linear elastic material, whose boundary is defined by splines and parametrised by the positions of control points 1 and 2. In other words, there is a total of four geometric parameters (two per point). The set of parameters is denoted as $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \alpha_3, \alpha_4) = (r_1, z_1, r_2, z_2)$, which belongs to design space $\boldsymbol{\mathcal{A}}$. One of the quantities of interest sought for design is the size of the plastic zones. For the sake of demonstration, the computations performed in this article are linear elastic and the plastic zones are estimated by defining a threshold and considering the zones whose strain is superior to this threshold (see Sect. 6.2 for further details). An example of virtual chart representing this quantity of interest will be constructed and exposed in Sect. 6.2.

In the current approach, the problem is solved, after discretisation in space, through a FE procedure governed by an equation of the form:

$$\forall\, \boldsymbol{\alpha} \in \boldsymbol{\mathcal{A}}, \quad \mathbb{K}\left(\boldsymbol{\alpha}\right) \mathbf{U}\left(\boldsymbol{\alpha}\right) = \mathbf{F}\left(\boldsymbol{\alpha}\right) \tag{3}$$

where

- $\mathbb{K}\left(\boldsymbol{\alpha}\right)$: stiffness matrix
- $\mathbf{U}\left(\boldsymbol{\alpha}\right)$: displacement vector
- $\mathbf{F}\left(\boldsymbol{\alpha}\right)$: load vector

The brute force approach consists in solving (3) for each set of parameters and checking whether the structure meets the mechanical requirements or not. If not, a new simulation is run and so on. The issue is that numerous simulations may be needed before finding the right set of parameters increasing the computation time and conception costs. That is why reduced order modelling and, in particular, PGD was considered. Difficulties will be encountered due to axisymmetric modelling and the complex geometry considered (splines). A methodology to take care of them will presented.

3 PGD-virtual chart

3.1 Virtual chart

Coming back to the continuous framework, a PGD-virtual chart of the displacement $\boldsymbol{u}$, depending on a set of parameters $\boldsymbol{\alpha} \in \boldsymbol{\mathcal{A}}$, can be seen as a separated variables approximation of $\boldsymbol{u}$:

$$\begin{aligned} &\forall\, \boldsymbol{X} \in \Omega,\ \forall\, \boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_m) \in \boldsymbol{\mathcal{A}}, \\ &\boldsymbol{u}\left(\boldsymbol{\alpha}, \boldsymbol{X}\right) \approx \boldsymbol{u}_n\left(\boldsymbol{\alpha}, \boldsymbol{X}\right) = \sum_{k=1}^{n} \lambda_k\left(\boldsymbol{\alpha}\right) \boldsymbol{\Lambda}_k\left(\boldsymbol{X}\right) \end{aligned} \tag{4}$$

where $\lambda_k\left(\boldsymbol{\alpha}\right) = \prod_{j=1}^{m} \lambda_k^j\left(\alpha_j\right)$ and n is the number of "modes" used in the approximation. λ_k^j are scalar functions and $\boldsymbol{\Lambda}_k$ are vectors. All is remaining for the final user is to particularise this solution for a given set of parameters.

For parameters such as material parameters (Young's moduli, Poisson's ratios, diffusion coefficients...), the PGD method furnishes a separated variables representation thanks to a *greedy algorithm* [18]. For geometric parameters, the method cannot be applied directly as is and must be slightly modified as it was exemplified in [3] (see Subsect. 4).

The construction of the virtual chart, using a greedy algorithm, is classic and recalled in Appendix 1 for the reader who is not familiar with the method. The important point is that, roughly, the greedy algorithm is based upon a variational formulation of the form: Find $\boldsymbol{u} \in \mathcal{I} \otimes \mathcal{V}$ such that

$$\forall\, \boldsymbol{v} \in \mathcal{I} \otimes \mathcal{V}, \quad \int_{\boldsymbol{\mathcal{A}}} \int_{\Omega} \boldsymbol{\sigma}\left(\boldsymbol{u}\right) : \boldsymbol{\varepsilon}\left(\boldsymbol{v}\right) \mathrm{d}\Omega\, \mathrm{d}\boldsymbol{\mathcal{A}} = \int_{\boldsymbol{\mathcal{A}}} \int_{\partial_d \Omega} \boldsymbol{f} \cdot \boldsymbol{v}\, \mathrm{d}S\, \mathrm{d}\boldsymbol{\mathcal{A}} \tag{5}$$

where $\otimes$ stands for the tensor product. The whole PGD algorithm is exposed in Algorithm 1. An example of application for material parameters is given in Appendix 2.

4 Geometric parameters

Geometric parameters are of different nature regarding to other parameters as, for instance, material parameters (Young's moduli, Poisson's ratios...) due to the fact that their variations affect directly the integration domain $\Omega\left(\boldsymbol{\alpha}\right)$. Let us come back to the problem shown in (5). This time, we will assume that the geometry depends on a set of parameters $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_m) \in \boldsymbol{\mathcal{A}} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_m$. Equation (5) becomes:

Algorithm 1: PGD greedy algorithm

Input: A sequence of parameters
$\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_m) \in \boldsymbol{\mathcal{A}} = \mathcal{A}_1 \times \ldots \times \mathcal{A}_m$

Level 1: Greedy algorithm;
while $\|\mathbf{R}(\boldsymbol{u}_\mathbf{n})\| >$ *threshold (with* $\mathbf{R}(\boldsymbol{u}_\mathbf{n})$ *being the residual)* **do**
  Level 2: Fixed-point algorithm;
  Initialisation of the $(m+1)$-uplet $\left(\lambda^1, \ldots, \lambda^m, \boldsymbol{\Lambda}\right)$;
  for $k = 1$ ***to*** k_{max} **do**
    for $j = 1$ ***to*** m **do**
      solve the parametric problem associated to λ^j
    solve the spatial problem associated to $\boldsymbol{\Lambda}$
  Update of the functions $\left(\lambda_i^1, \ldots, \lambda_i^m\right)_{i \in \{1\,..\,n\}}$;
  $\boldsymbol{u}_n \leftarrow \boldsymbol{u}_{n-1} + \prod_{r=1}^{m} \lambda^r \boldsymbol{\Lambda}$;
  $n \leftarrow n+1$;

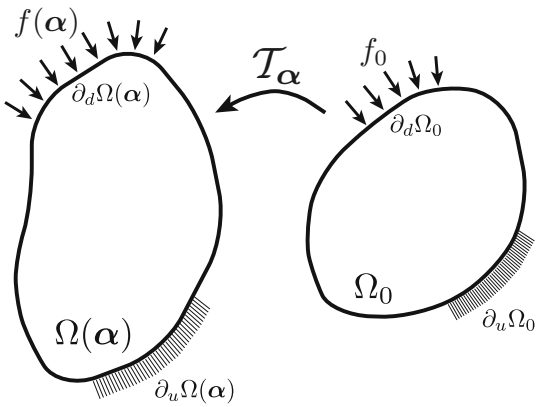

Fig. 2 Definition of the problem on a reference structure Ω_0

Find $\boldsymbol{u} \in \mathcal{I} \otimes \mathcal{V}(\boldsymbol{\alpha})$ such that

$$\forall\, \boldsymbol{v} \in \mathcal{I} \otimes \mathcal{V}(\boldsymbol{\alpha}),\ \int_{\mathcal{A}} \int_{\Omega(\boldsymbol{\alpha})} \boldsymbol{\sigma}(\boldsymbol{u}) : \boldsymbol{\varepsilon}(\boldsymbol{v})\, \mathrm{d}\Omega\, \mathrm{d}\boldsymbol{\mathcal{A}} = \int_{\mathcal{A}} \int_{\partial_d \Omega(\boldsymbol{\alpha})} \boldsymbol{f} \cdot \boldsymbol{v}\, \mathrm{d}S\, \mathrm{d}\boldsymbol{\mathcal{A}} \tag{6}$$

with $\mathcal{V}(\boldsymbol{\alpha}) = H_0^1(\Omega(\boldsymbol{\alpha})) = \left\{\boldsymbol{v} \in H^1(\Omega(\boldsymbol{\alpha})) \,\middle|\, \boldsymbol{v}_{|\partial_u \Omega(\boldsymbol{\alpha})} = \mathbf{0}\right\}$

This complicates the calculations of the different integrals since they cannot be computed independently from one another any more. The dependency of the integration domain on the geometric parameters has to be, in some way, skirted. The solution, introduced in [3], is to use a change of variables, i.e. to define a reference structure Ω_0 and, from the current structure $\Omega(\boldsymbol{\alpha})$, where the problem is initially defined, to come back to that reference structure Ω_0 thanks to a geometric transformation $\mathcal{T}_{\boldsymbol{\alpha}}$ (see Fig. 2). Thus,

$$\boldsymbol{X} = \mathcal{T}_{\boldsymbol{\alpha}}(\boldsymbol{X}_0) \tag{7}$$

$$\boldsymbol{u}(\boldsymbol{\alpha}, \boldsymbol{X}) = \boldsymbol{u}(\boldsymbol{\alpha}, \mathcal{T}_{\boldsymbol{\alpha}}(\boldsymbol{X}_0)) \tag{8}$$

The methodology is recalled, hereafter, for the sake of consistency and to highlight the technical difficulties of dealing with axisymmetric modelling. One can write (6) applying the change of variables:
Find $\boldsymbol{u} \in \mathcal{I} \otimes \mathcal{V}_0$ such that

$$\begin{aligned}\forall\, \boldsymbol{v} \in \mathcal{I} \otimes \mathcal{V}_0,\ &\int_{\boldsymbol{\mathcal{A}}} \int_{\Omega_0} \boldsymbol{\sigma}\left(\boldsymbol{u}\left(\boldsymbol{\alpha}, \mathcal{T}_{\boldsymbol{\alpha}}(\boldsymbol{X}_0)\right)\right) \\ &: \boldsymbol{\varepsilon}\left(\boldsymbol{v}\left(\boldsymbol{\alpha}, \mathcal{T}_{\boldsymbol{\alpha}}(\boldsymbol{X}_0)\right)\right) J_{\mathcal{T}_{\boldsymbol{\alpha}}}(\boldsymbol{X}_0)\, \mathrm{d}\Omega_0\, \mathrm{d}\boldsymbol{\mathcal{A}} \\ &= \int_{\boldsymbol{\mathcal{A}}} \int_{\partial_d \Omega_0} \boldsymbol{f}\left(\boldsymbol{\alpha}, \mathcal{T}_{\boldsymbol{\alpha}}(\boldsymbol{X}_0)\right) \cdot \boldsymbol{v}\left(\boldsymbol{\alpha}, \mathcal{T}_{\boldsymbol{\alpha}}(\boldsymbol{X}_0)\right) J_{\mathcal{T}_{\boldsymbol{\alpha}}}(\boldsymbol{X}_0)\, \mathrm{d}S_0\, \mathrm{d}\boldsymbol{\mathcal{A}}\end{aligned} \tag{9}$$

with $\mathcal{V}_0 = H_0^1(\Omega_0) = \left\{\boldsymbol{v} \in H^1(\Omega_0) \,\middle|\, \boldsymbol{v}_{|\partial_u \Omega_0} = \mathbf{0}\right\}$ and $J_{\mathcal{T}_{\boldsymbol{\alpha}}}$ is the modulus of the determinant of the Jacobian matrix of geometric transformation $\mathcal{T}_{\boldsymbol{\alpha}}$.

In this formulation, the dependency on the geometric parameters has been transferred from the integration domain to the integrand through the Jacobian. Now a PGD approximation of the field $\boldsymbol{u}$ can be sought using the PGD procedure already exposed in Sect. 3.

4.1 Proper generalized decomposition for geometric parameters

Let us come back to the problem defined in Sect. 2. One applies the principle of stationary potential energy and integrates it on the geometric parameter space, which reads after discretisation in space:

$$\forall\, \mathbf{U}^*(\boldsymbol{\alpha}) \in \mathcal{I} \otimes \mathbb{R}^N,\ \int_{\boldsymbol{\mathcal{A}}} \mathbf{U}^T(\boldsymbol{\alpha})\, \mathbb{K}(\boldsymbol{\alpha})\, \mathbf{U}^*(\boldsymbol{\alpha})\, \mathrm{d}\boldsymbol{\mathcal{A}} = \int_{\boldsymbol{\mathcal{A}}} \mathbf{F}^T(\boldsymbol{\alpha})\, \mathbf{U}^*(\boldsymbol{\alpha})\, \mathrm{d}\boldsymbol{\mathcal{A}} \tag{10}$$

where N is the number of degrees of freedom of the discretisation in space.

From this formulation, the PGD-approximation of displacement vector $\mathbf{U}$ is calculated:

$$\mathbf{U}(\boldsymbol{\alpha}) = \mathbf{U}(\alpha_1, \ldots, \alpha_m) = \sum_{k=1}^{n} \prod_{j=1}^{m} \lambda_k^j(\alpha_j)\, \tilde{\mathbf{U}}_k \tag{11}$$

At iteration n, the following test function vector is used for the computation of the new $(m+1)$-uplet:

$$\mathbf{U}^*(\boldsymbol{\alpha}) = \mathbf{U}^*(\alpha_1, \ldots, \alpha_m) = \prod_{j=1}^{m} \lambda^j(\alpha_j)\, \tilde{\mathbf{U}}^* + \sum_{i=1}^{m} \lambda^{i*}(\alpha_i) \prod_{\substack{j=1 \\ j \neq i}}^{m} \lambda^j(\alpha_j)\, \tilde{\mathbf{U}} \tag{12}$$

with $\tilde{\mathbf{U}}^* \in \mathbb{R}^N$ and $\lambda^{i*} \in \mathcal{I}_i$ for $i = 1..m$.

Using Algorithm 1 defined previously, one solves alternatively the linear systems:

$$\begin{aligned}
&\forall r \in \{1..m\}, \left\{ \tilde{\mathbf{U}}^T \left[\int_{\bar{\boldsymbol{\mathcal{A}}}_r} \prod_{\substack{i=1 \\ i\neq r}}^{m} \left(\lambda^i(\alpha_i)\right)^2 \mathbb{K}(\boldsymbol{\alpha})\, \mathrm{d}\bar{\boldsymbol{\mathcal{A}}}_r \right] \tilde{\mathbf{U}} \right\} \lambda^r(\alpha_r) \\
&= \left[\int_{\bar{\boldsymbol{\mathcal{A}}}_r} \prod_{\substack{i=1 \\ i\neq r}}^{m} \lambda^i(\alpha_i)\, \mathbf{F}^T(\boldsymbol{\alpha})\, \mathrm{d}\bar{\boldsymbol{\mathcal{A}}}_r \right] \tilde{\mathbf{U}} \\
&\quad - \sum_{k=1}^{n-1} \left\{ \tilde{\mathbf{U}}_k^T \left[\int_{\bar{\boldsymbol{\mathcal{A}}}_r} \prod_{\substack{i=1 \\ i\neq r}}^{m} \prod_{\substack{j=1 \\ j\neq r}}^{m} \lambda^i(\alpha_i)\, \lambda_k^j(\alpha_j)\, \mathbb{K}(\boldsymbol{\alpha})\, \mathrm{d}\bar{\boldsymbol{\mathcal{A}}}_r \right] \tilde{\mathbf{U}} \right\} \lambda_k^r(\alpha_r)
\end{aligned} \tag{13}$$

with $\bar{\boldsymbol{\mathcal{A}}}_r = \mathcal{A}_1 \times \cdots \times \mathcal{A}_{r-1} \times \mathcal{A}_{r+1} \times \cdots \times \mathcal{A}_m$

$$\begin{aligned}
\left[\int_{\boldsymbol{\mathcal{A}}} \prod_{i=1}^{m} \left(\lambda^i(\alpha_i)\right)^2 \mathbb{K}(\boldsymbol{\alpha})\, \mathrm{d}\boldsymbol{\mathcal{A}} \right] \tilde{\mathbf{U}} &= \int_{\boldsymbol{\mathcal{A}}} \prod_{i=1}^{m} \lambda^i(\alpha_i)\, \mathbf{F}(\boldsymbol{\alpha}) \mathrm{d}\boldsymbol{\mathcal{A}} \\
&\quad - \sum_{k=1}^{n-1} \left[\int_{\boldsymbol{\mathcal{A}}} \prod_{i=1}^{m} \prod_{j=1}^{m} \lambda^i(\alpha_i)\, \lambda_k^j(\alpha_j)\, \mathbb{K}(\boldsymbol{\alpha})\, \mathrm{d}\boldsymbol{\mathcal{A}} \right] \tilde{\mathbf{U}}_k
\end{aligned} \tag{14}$$

Let us remark that the computation of the various integrations can be greatly facilitated if the stiffness matrix $\mathbb{K}$ is written in a separated variables form, which is the aim of the next subsection.

4.2 Stiffness matrix computation in the axisymmetric case

Let us start writing the expression of the stiffness matrix $\mathbb{K}^e(\boldsymbol{\alpha})$ of an element for the current structure. Applying a change of variables, it will be defined on the reference structure:

$$\mathbb{K}^e(\boldsymbol{\alpha}) = 2\pi \int_{\Omega^e(\boldsymbol{\alpha})} \mathbb{B}^T(\boldsymbol{\alpha}, \boldsymbol{X})\, \mathbb{C}\, \mathbb{B}(\boldsymbol{\alpha}, \boldsymbol{X})\, r(\boldsymbol{\alpha}, \boldsymbol{X})\, \mathrm{d}\Omega \tag{15}$$

$$= 2\pi \int_{\Omega_0^e} \hat{\mathbb{B}}^T(\boldsymbol{\alpha}, \boldsymbol{X}_0)\, \mathbb{C}\, \hat{\mathbb{B}}(\boldsymbol{\alpha}, \boldsymbol{X}_0)\, \hat{r}(\boldsymbol{\alpha}, \boldsymbol{X}_0)\, J_{\mathcal{T}_{\boldsymbol{\alpha}}}(\boldsymbol{X}_0)\, \mathrm{d}\Omega_0 \tag{16}$$

where

- $\mathbb{B}$: matrix of the partial derivatives of the shape functions
- r: radius
- $\mathbb{C}$: Hooke's matrix (here independent on $\boldsymbol{\alpha}$ because we are not dealing with material parameters)
- $\hat{\mathbb{B}}(\boldsymbol{\alpha}, \boldsymbol{X}_0) = \mathbb{B}(\mathcal{T}_{\boldsymbol{\alpha}}(\boldsymbol{X}_0))$ and $\hat{r}(\boldsymbol{\alpha}, \boldsymbol{X}_0) = r(\mathcal{T}_{\boldsymbol{\alpha}}(\boldsymbol{X}_0))$

It can be immediately noticed that axisymmetric modelling adds an extra difficulty with respect to a classic two dimensional modelling since $\hat{r}$, which depends on the geometric parameters, appears in the integrand. To obtain a separated variables form of $\mathbb{K}^e(\boldsymbol{\alpha})$, one must first write $\hat{\mathbb{B}}$ in the separated way.
Let us write the matrix $\hat{\mathbb{B}}$:

$$\hat{\mathbb{B}} = \left[\hat{\mathbb{B}}_1, \hat{\mathbb{B}}_2, \cdots, \hat{\mathbb{B}}_p \right] \tag{17}$$

where p corresponds to the number of shape functions used and:

$$\forall I \in \{1..p\},\ \hat{\mathbb{B}}_I = \begin{bmatrix} N_{I,r} & 0 \\ 0 & N_{I,z} \\ \frac{N_I}{r} & 0 \\ N_{I,z} & N_{I,r} \end{bmatrix} \tag{18}$$

It has to be recalled that the problem is defined on the reference structure Ω_0. Therefore $N_{I,r}$ and $N_{I,z}$ are unknown on the contrary to N_{I,r_0} and N_{I,z_0}. Obviously, it is possible to express $N_{I,r}$ and $N_{I,z}$ in function of N_{I,r_0} and N_{I,z_0}:

$$\begin{bmatrix} N_{,r_0} \\ N_{,z_0} \end{bmatrix} = \underbrace{\begin{bmatrix} r_{,r_0} & z_{,r_0} \\ r_{,z_0} & z_{,z_0} \end{bmatrix}}_{\mathbb{J}_{\mathcal{T}_{\boldsymbol{\alpha}}}^T} \begin{bmatrix} N_{,r} \\ N_{,z} \end{bmatrix} \tag{19}$$

Hence,

$$\begin{bmatrix} N_{,r} \\ N_{,z} \end{bmatrix} = \mathbb{J}_{\mathcal{T}_{\boldsymbol{\alpha}}}^{-T} \begin{bmatrix} N_{,r_0} \\ N_{,z_0} \end{bmatrix} \tag{20}$$

Introducing the adjugate matrix of $\mathbb{J}_{\mathcal{T}_{\boldsymbol{\alpha}}}$:

$$\begin{bmatrix} N_{,r} \\ N_{,z} \end{bmatrix} = \frac{1}{J_{\mathcal{T}_{\boldsymbol{\alpha}}}} \mathrm{adj}\left(\mathbb{J}_{\mathcal{T}_{\boldsymbol{\alpha}}}\right)^T \begin{bmatrix} N_{,r_0} \\ N_{,z_0} \end{bmatrix} \tag{21}$$

(18) becomes, defining $\mathrm{Adj} = \mathrm{adj}\left(\mathbb{J}_{\mathcal{T}_{\boldsymbol{\alpha}}}\right)^T$:

$$\forall I \in \{1..p\},\quad \hat{\mathbb{B}}_I = \frac{1}{J_{\mathcal{T}_{\boldsymbol{\alpha}}}} \begin{bmatrix} \mathrm{Adj}_{1,1} N_{I,r_0} + \mathrm{Adj}_{1,2} N_{I,z_0} & 0 \\ 0 & \mathrm{Adj}_{2,1} N_{I,r_0} + \mathrm{Adj}_{2,2} N_{I,z_0} \\ \frac{N_I}{r} & 0 \\ \mathrm{Adj}_{2,1} N_{I,r_0} + \mathrm{Adj}_{2,2} N_{I,z_0} & \mathrm{Adj}_{1,1} N_{I,r_0} + \mathrm{Adj}_{1,2} N_{I,z_0} \end{bmatrix} \tag{22}$$

Jacobian matrix $\mathbb{J}_{\mathcal{T}_{\boldsymbol{\alpha}}}$ of the geometric transformation depends on both space variables and geometric parameters. Therefore obtaining a separated variable representation is not trivial. It can be considered to obtain a separated form of the Jacobian through a *higher-order singular value decomposition* (HOSVD [13]) but due to the large number of HOSVD needed (four for each elements), it would highly increase

the computation time. The proposed solution is to avoid the dependency on space variables of the Jacobian matrix. To do so, the simplest way is to use solely affine geometric transformations with respect to space variables. During the computation of the Jacobian matrix, the space variables will disappear naturally. For that purpose, the structure is divided into sub-domains in which is defined an affine geometric transformation. This solution was also proposed in [3] where geometries are decomposed in triangular sub-domains.

In Sect. 6, a possible subdivision is detailed for the engineering case. In particular, it will be shown that triangles can be limited in some cases of complex geometries such as splines and hence are not a panacea.

An affine geometric transformation is written in the section as:

$$\mathcal{T}_{\boldsymbol{\alpha}}\left(\boldsymbol{X}_0\right) = \begin{pmatrix} \mathcal{T}^r_{\boldsymbol{\alpha}}\left(\boldsymbol{X}_0\right) \\ \mathcal{T}^z_{\boldsymbol{\alpha}}\left(\boldsymbol{X}_0\right) \end{pmatrix} \tag{23}$$

$$= \begin{pmatrix} a_r\left(\boldsymbol{\alpha}\right) r_0 + b_r\left(\boldsymbol{\alpha}\right) z_0 + c_r\left(\boldsymbol{\alpha}\right) \\ a_z\left(\boldsymbol{\alpha}\right) r_0 + b_z\left(\boldsymbol{\alpha}\right) z_0 + c_z\left(\boldsymbol{\alpha}\right) \end{pmatrix} \tag{24}$$

with $\boldsymbol{X}_0 = (r_0, z_0)$. Hence the Jacobian and Adj matrices become:

$$\mathbb{J}_{\mathcal{T}_{\boldsymbol{\alpha}}} = \begin{bmatrix} \dfrac{\partial \mathcal{T}^r_{\boldsymbol{\alpha}}}{\partial r_0} & \dfrac{\partial \mathcal{T}^r_{\boldsymbol{\alpha}}}{\partial z_0} \\ \dfrac{\partial \mathcal{T}^z_{\boldsymbol{\alpha}}}{\partial r_0} & \dfrac{\partial \mathcal{T}^z_{\boldsymbol{\alpha}}}{\partial z_0} \end{bmatrix} \tag{25}$$

$$= \begin{bmatrix} a_r\left(\boldsymbol{\alpha}\right) & a_z\left(\boldsymbol{\alpha}\right) \\ b_r\left(\boldsymbol{\alpha}\right) & b_z\left(\boldsymbol{\alpha}\right) \end{bmatrix} \tag{26}$$

$$\mathrm{Adj} = \begin{bmatrix} b_z\left(\boldsymbol{\alpha}\right) & -b_r\left(\boldsymbol{\alpha}\right) \\ -a_z\left(\boldsymbol{\alpha}\right) & a_r\left(\boldsymbol{\alpha}\right) \end{bmatrix} \tag{27}$$

The Jacobian does not depend on the space variables any more but it is still not obvious to give a separated variables representation of matrix $\hat{\mathbb{B}}$. For this purpose, let us define the following matrices:

$$\mathbb{B}^I_1\left(\boldsymbol{X}_0\right) = \begin{bmatrix} N_{I,r_0}\left(\boldsymbol{X}_0\right) & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & N_{I,r_0}\left(\boldsymbol{X}_0\right) \end{bmatrix}$$

$$\mathbb{B}^I_2\left(\boldsymbol{X}_0\right) = \begin{bmatrix} N_{I,z_0}\left(\boldsymbol{X}_0\right) & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & N_{I,z_0}\left(\boldsymbol{X}_0\right) \end{bmatrix}$$

$$\mathbb{B}^I_3\left(\boldsymbol{X}_0\right) = \begin{bmatrix} 0 & 0 \\ 0 & N_{I,r_0}\left(\boldsymbol{X}_0\right) \\ 0 & 0 \\ N_{I,r_0}\left(\boldsymbol{X}_0\right) & 0 \end{bmatrix}$$

$$\mathbb{B}^I_4\left(\boldsymbol{X}_0\right) = \begin{bmatrix} 0 & 0 \\ 0 & N_{I,z_0}\left(\boldsymbol{X}_0\right) \\ 0 & 0 \\ N_{I,z_0}\left(\boldsymbol{X}_0\right) & 0 \end{bmatrix}$$

$$\mathbb{B}^I_5\left(\boldsymbol{X}_0\right) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ N_I\left(\boldsymbol{X}_0\right) & 0 \\ 0 & 0 \end{bmatrix} \tag{28}$$

and coefficients:

$$\beta_1 = b_z\left(\boldsymbol{\alpha}\right),\ \beta_2 = -a_z\left(\boldsymbol{\alpha}\right),\ \beta_3 = -b_r\left(\boldsymbol{\alpha}\right),\ \beta_4 = a_r\left(\boldsymbol{\alpha}\right) \tag{29}$$

So that $\hat{\mathbb{B}}$ can be conveniently written:

$$\hat{\mathbb{B}}\left(\boldsymbol{\alpha}, \boldsymbol{X}_0\right) = \frac{1}{J_{\mathcal{T}_{\boldsymbol{\alpha}}}} \sum_{i=1}^{4} \beta_i\left(\boldsymbol{\alpha}\right) \mathbb{B}_i\left(\boldsymbol{X}_0\right) + \frac{1}{\hat{r}\left(\boldsymbol{\alpha}, \boldsymbol{X}_0\right)} \mathbb{B}_5\left(\boldsymbol{X}_0\right) \tag{30}$$

Due to the definition of $\mathbb{B}_5$, one can remark:

$$\forall\, i \neq 5,\ \mathbb{B}^T_5\, \mathbb{C}\, \mathbb{B}_i = \mathbb{O} \tag{31}$$

which leads to the following expression of $\mathbb{K}^e$:

$$\begin{aligned} \mathbb{K}^e\left(\boldsymbol{\alpha}\right) = \frac{1}{J_{\mathcal{T}_{\boldsymbol{\alpha}}}} \sum_{i=1}^{4} \sum_{i=1}^{4} \beta_i\left(\boldsymbol{\alpha}\right) \beta_j\left(\boldsymbol{\alpha}\right) & \int_{\Omega^e_0} \mathbb{B}^T_i\left(\boldsymbol{X}_0\right) \mathbb{C}\, \mathbb{B}_j\left(\boldsymbol{X}_0\right) \\ & \times \hat{r}\left(\boldsymbol{\alpha}, \boldsymbol{X}_0\right) \mathrm{d}\Omega_0 \\ & + J_{\mathcal{T}_{\boldsymbol{\alpha}}} \int_{\Omega^e_0} \frac{1}{\hat{r}\left(\boldsymbol{\alpha}, \boldsymbol{X}_0\right)} \mathbb{B}^T_5\left(\boldsymbol{X}_0\right) \mathbb{C}\, \mathbb{B}_5\left(\boldsymbol{X}_0\right) \mathrm{d}\Omega_0 \end{aligned} \tag{32}$$

This expression of $\mathbb{K}^e$ satisfies only *partially* the desired objective, i.e.obtaining a separated variables formulation. Partially because $\hat{r}$ and $1/\hat{r}$ contain both space variables and geometric parameters. Since the choice of an affine geometric transformation was made, $\hat{r}$ is already separated and is written as $\hat{r} = a_r r_0 + b_r z_0 + c_r$.

The case of $1/\hat{r}$ is slightly trickier since $1/\hat{r}$ has no immediate separated form. For this reason, a linear approximation of $1/\hat{r}$ is computed in a neighbourhood of the barycentre (r_{bar}, z_{bar}) of the sub-domain to which the element belongs:

$$\begin{aligned} \frac{1}{\hat{r}} &= \frac{1}{a_r r_0 + b_r z_0 + c_r} \\ &\approx \frac{1}{a_r r_{bar} + b_r z_{bar} + c_r} - \frac{a_r}{\left(a_r r_{bar} + b_r z_{bar} + c_r\right)^2}\left(r_0 - r_{bar}\right) \\ &\quad - \frac{b_r}{\left(a_r r_{bar} + b_r z_{bar} + c_r\right)^2}\left(z_0 - z_{bar}\right) \end{aligned} \tag{33}$$

In Sect. 6, an evaluation of the error for this approximation will be given in the case of the industrial demonstrator. Finally the elementary stiffness matrix $\mathbb{K}^e$ can be written in a fully separated formulation:

$$\mathbb{K}^e(\boldsymbol{\alpha}) = \sum_{i=1}^{4}\sum_{j=1}^{4}\left(m^{ij}(\boldsymbol{\alpha})\,\mathbb{M}^{ij} + m^{ij}_{r_0}(\boldsymbol{\alpha})\,\mathbb{M}^{ij}_{r_0} + m^{ij}_{z_0}(\boldsymbol{\alpha})\,\mathbb{M}^{ij}_{z_0}\right) + m^{55}(\boldsymbol{\alpha})\,\mathbb{M}^{55} + m^{55}_{r_0}(\boldsymbol{\alpha})\,\mathbb{M}^{55}_{r_0} + m^{55}_{z_0}(\boldsymbol{\alpha})\,\mathbb{M}^{55}_{z_0} \quad (34)$$

with

- $m^{ij} = \frac{c_r}{J_{\mathcal{T}_{\boldsymbol{\alpha}}}}\beta_i\,\beta_j$ and $\mathbb{M}^{ij} = \int_{\Omega_0^e} \mathbb{B}_i^T\mathbb{C}\,\mathbb{B}_j\,\mathrm{d}\Omega_0$
- $m^{ij}_{r_0} = \frac{a_r}{J_{\mathcal{T}_{\boldsymbol{\alpha}}}}\beta_i\,\beta_j$ and $\mathbb{M}^{ij}_{r_0} = \int_{\Omega_0^e} r_0\mathbb{B}_i^T\mathbb{C}\,\mathbb{B}_j\,\mathrm{d}\Omega_0$
- $m^{ij}_{z_0} = \frac{b_r}{J_{\mathcal{T}_{\boldsymbol{\alpha}}}}\beta_i\,\beta_j$ and $\mathbb{M}^{ij}_{z_0} = \int_{\Omega_0^e} z_0\,\mathbb{B}_i^T\mathbb{C}\,\mathbb{B}_j\,\mathrm{d}\Omega_0$
- $m^{55} = \frac{J_{\mathcal{T}_{\boldsymbol{\alpha}}}}{a_r r_{bar}+b_r z_{bar}+c_r}$ and $\mathbb{M}^{55} = \int_{\Omega_0^e} \mathbb{B}_5^T\mathbb{C}\,\mathbb{B}_5\,\mathrm{d}\Omega_0$
- $m^{55}_{r_0} = -\frac{a_r J_{\mathcal{T}_{\boldsymbol{\alpha}}}}{(a_r r_{bar}+b_r z_{bar}+c_r)^2}$ and $\mathbb{M}^{55}_{r_0} = \int_{\Omega_0^e} (r_0 - r_{bar})\;\mathbb{B}_5^T\mathbb{C}\,\mathbb{B}_5\,\mathrm{d}\Omega_0$
- $m^{55}_{z_0} = -\frac{b_r J_{\mathcal{T}_{\boldsymbol{\alpha}}}}{(a_r r_{bar}+b_r z_{bar}+c_r)^2}$ and $\mathbb{M}^{55}_{z_0} = \int_{\Omega_0^e} (z_0 - z_{bar})\;\mathbb{B}_5^T\mathbb{C}\,\mathbb{B}_5\,\mathrm{d}\Omega_0$

The different matrices that appear in the decomposition of $\mathbb{K}^e(\boldsymbol{\alpha})$ are to be computed for each element and the coefficients depending on the geometric parameters for each sub-domain. The number of terms belonging to decomposition of the stiffness matrix is fixed with respect to the discretisation of the parameter space $\boldsymbol{\mathcal{A}}$.

5 Integration into the engineering design process

As it was previously said, particular attention has been paid to inserting the methodology developed into a genuine engineering design process. Due to engineering requirements and, in particular, the coupling with other parts of the system that affect the loadings applied on the structure presented Fig. 1, the actual design process is performed using the commercial software SAMCEF.

It is important to note that a single optimisation of the structure with respect to geometric parameters should have certainly been done efficiently using the embedded SAMCEF optimisation procedures. However, this work is the first step of a more complex design process in which some other parameters will be taken into account. Further works are in progress to take also into account various types of loadings that can be applied on the structure and, then, the total number of situations considered justify the offline building of a virtual chart of solutions.

The proposed PGD approach must, then, be included in the design workflow with a limited number of modifications to take advantage of the existing facilities. The main problem of the approach detailed in Subsect. 4.2 is the high degree of intrusiveness of the PGD. Indeed, the linear system defined in (14) is not a classic finite elements problem since the stiffness matrix and the load vector are averaged on the parameter space A. To limit the intrusiveness in the design workflow, different approaches have been envisaged, in collaboration with SAMTECH, the SIEMENS subsidiary which develops SAMCEF software. The first was a complete embedding of the PGD approach in SAMCEF, calculating the averaged stiffness matrices and averaged load vectors (see Eq. (14)):

$$-\int_{\boldsymbol{\mathcal{A}}} \prod_{i=1}^{m}\prod_{j=1}^{m} \lambda^i(\alpha_i)\,\lambda_k^j\left(\alpha_j\right)\;\mathbb{K}(\boldsymbol{\alpha})\,\mathrm{d}\boldsymbol{\mathcal{A}} \quad (35)$$

$$-\int_{\boldsymbol{\mathcal{A}}} \prod_{i=1}^{m} \lambda^i(\alpha_i)\;\mathbf{F}^T(\boldsymbol{\alpha})\,\mathrm{d}\boldsymbol{\mathcal{A}} \quad (36)$$

directly in the commercial code. This approach has been given up because it would have need the development of a new structure of fields in SAMCEF which has been considered a too consequent job. Finally, the technique which has been chosen and is demonstrated herein is to "encapsulate" SAMCEF into an in-house MATLAB code that deals with the particular operators associated to PGD. To sum up, on the one hand, SAMCEFs utilities are used to defined the geometry, the partition, the loads, the mesh and the materials, as well as to solve the space finite elements systems mandatory to build the space modes $\boldsymbol{\Lambda}_i$ of the PGD. On the other hand, the MATLAB in-house code is used to build the specific PGD operators Eqs. (35) and (36), as well as the problems corresponding to parametric modes λ_i.

The communication between the two codes has needed the development by SAMTECH of specific interface functions which takes advantage of the possibility offered by SAMCEF to stop it at a defined step and to restart it from this step afterwards. The whole procedure is detailed in Algorithm 2 and a scheme of the communication between SAMCEF and MATLAB is given in Fig. 3. The overall process of coupling is not optimised and this work only constitutes a demonstrator of the approach that will be extended in the future.

Algorithm 2: SAMCEF-PGD algorithm

Input: A sequence of parameters
$\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_m) \in \boldsymbol{\mathcal{A}} = \mathcal{A}_1 \times \ldots \times \mathcal{A}_m$

Definition of the problem with SAMCEF;
Definition of the geometry, materials, partition, loads and mesh;
START PGD-algorihm in MATLAB;
Load data from SAMCEF
Level 1: Greedy algorithm;
while $\|\mathbf{R}(\mathbf{U_n})\| >$ *threshold (with* $\mathbf{R}(\boldsymbol{u_n})$ *being the residual)* **do**
  Level 2: Fixed-point algorithm;
  Initialisation of the $(m+1)$-uplet $\left(\lambda^1, \ldots, \lambda^m, \tilde{\mathbf{U}}\right)$;
  for $k = 1$ ***to*** k_{max} **do**
    Computation of the averaged elementary stiffness matrices and load vectors;
    for $j = 1$ ***to*** m **do**
      Resolution of the parametric problem associated to λ^j;
    Transfer stiffness matrices and load vectors to SAMCEF
    Resolution of the spatial problem using SAMCEF
  Update of the functions $\left(\lambda_i^1, \ldots, \lambda_i^m\right)_{i \in \{1\,..\,n\}}$;
  $\mathbf{U}_n \leftarrow \mathbf{U}_{n-1} + \prod_{r=1}^{m} \lambda^r \tilde{\mathbf{U}}$;
  $n \leftarrow n + 1$;

6 Application

6.1 Geometry partition

In order to apply what has been exposed in Sect. 3, a division of the geometry of the engineering structure has to be chosen. The division has to verify the sufficient condition defined before, viz. that the geometric transformation associated to each sub-domain must be affine with respect to the space variables.

An example of a possible division of the geometry is represented in Fig. 4. The level of error due to the approximation of $\frac{1}{\tilde{r}}$ Eq. (33) goes from 1.28 % (sub-domain 1) up to 6.32 % (sub-domain 2). The sub-domains are mainly triangles where the geometric transformations are quite immediate to compute. To construct an affine geometric transformation in the neighbourhood of the fillet is trickier. In fact, the geometry must be accurately fitted, so triangles are to be avoided. One can think about using a high number of triangles to fit the fillet but here we face a problem of computation time because the more sub-domains there are, the longer it takes to compute the stiffness matrix (see Subsect. 4.2).

The problem is that six-nodes triangles cannot be considered either, even though they will accurately fit the curved geometry because the geometric transformation associating a six-nodes triangle to another one is quadratic and not affine.

That is why it was decided to use modified three-nodes triangles with two straight edges and a curved one as shown in the close-up of Fig. 4. To do so, the fillet in the reference structure Ω_0 is modelled as an arc of a circle. Let us suppose that for each geometric configuration, the positions of points A and B (used here as control points), together with the tangents at these points, are known and point C is "free". First let us seek the inverse geometric transformation $\mathcal{T}_{\boldsymbol{\alpha}}^{-1}$ (from the current structure $\Omega(\boldsymbol{\alpha})$ to the reference structure Ω_0) associated to the displacements of points A and B which is also affine.

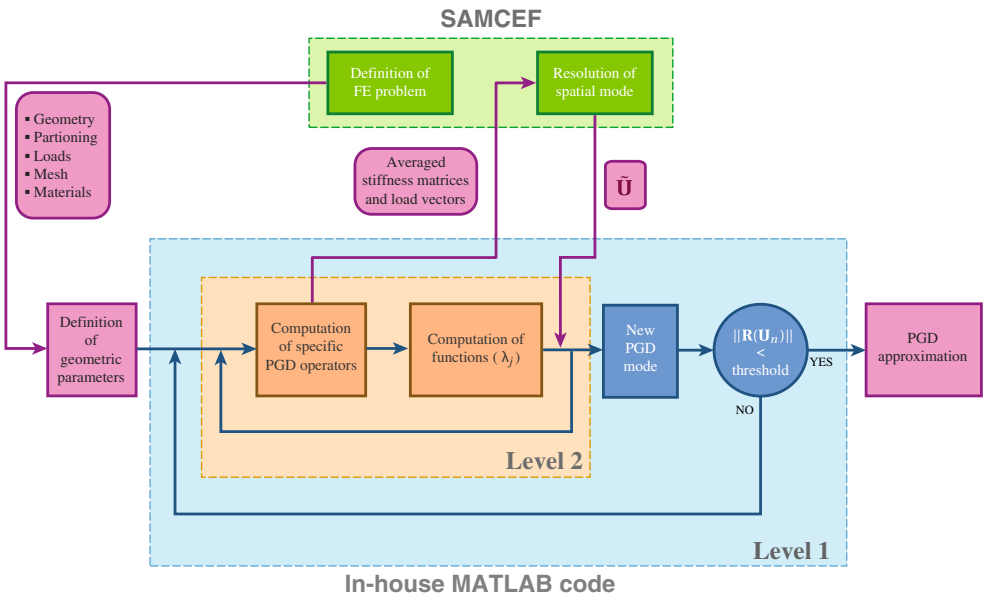

Fig. 3 Communication between SAMCEF and the in-house MATLAB code

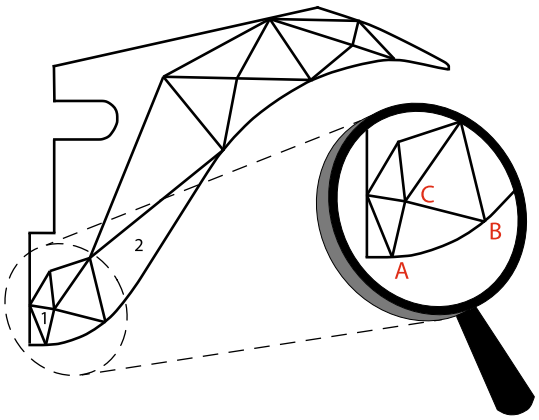

Fig. 4 A possible partition of the geometry

The affine inverse geometric transformation can be written as follows:

$$\mathcal{T}_{\boldsymbol{\alpha}}^{-1}(r_0, z_0) = \begin{pmatrix} \lambda_r\, r + \gamma_r\, z + \delta_r \\ \lambda_z\, r + \gamma_z\, z + \delta_z \end{pmatrix} \tag{37}$$

It remains to determine the six coefficients λ_r, γ_r, δ_r, λ_z, γ_z and δ_z. Four equations are given by (37) for the positions of A and B are known for both current and reference structures.

$$\begin{cases} r_0^A = \lambda_r\, r_A + \gamma_r\, z_A + \delta_r \\ z_0^A = \lambda_z\, r_A + \gamma_z\, z_A + \delta_z \\ r_0^B = \lambda_r\, r_B + \gamma_r\, z_B + \delta_r \\ z_0^B = \lambda_z\, r_B + \gamma_z\, z_B + \delta_z \end{cases} \tag{38}$$

As it was previously mentioned, the fillet boundary is modelled by an arc of a circle further referred as (Γ_0) in the reference structure. Thus,

$$\forall\,(r_0, z_0) \in \Gamma_0,\ (r_0 - r_{centre})^2 + (z_0 - z_{centre})^2 = R_0^2 \tag{39}$$

Substituting (37) in (39), it leads to:

$$\begin{aligned} &(\lambda_r r + \gamma_r z + \delta_r - r_{centre})^2 + (\lambda_z r + \gamma_z z \\ &\quad + c_z - z_{centre})^2 = R_0^2 \end{aligned} \tag{40}$$

Let us derive it with respect to r:

$$\begin{aligned} &(\lambda_r r + \gamma_r z + \delta_r - r_{centre}) \left(\lambda_r + \gamma_r \frac{dz}{dr}\right) \\ &\quad + (\lambda_z r + \gamma_z z + \delta_z - z_{centre}) \left(\lambda_z + \gamma_z \frac{dz}{dr}\right) = 0 \end{aligned} \tag{41}$$

or

$$(r_0 - r_{centre}) \left(\lambda_r + \gamma_r \frac{dz}{dr}\right) + (z_0 - z_{centre}) \left(\lambda_z + \gamma_z \frac{dz}{dr}\right) = 0 \tag{42}$$

Which gives the last two missing equations:

$$\begin{cases} \left(r_0^A - r_{centre}\right) \left(\lambda_r + \gamma_r \left.\frac{dz}{dr}\right|_{r=r_A}\right) \\ \quad + \left(z_0^A - z_{centre}\right) \left(\lambda_z + \gamma_z \left.\frac{dz}{dr}\right|_{r=r_A}\right) = 0 \\ \left(r_0^B - r_{centre}\right) \left(\lambda_r + \gamma_r \left.\frac{dz}{dr}\right|_{r=r_B}\right) \\ \quad + \left(z_0^B - z_{centre}\right) \left(\lambda_z + \gamma_z \left.\frac{dz}{dr}\right|_{r=r_B}\right) = 0 \end{cases} \tag{43}$$

The six coefficients λ_r, γ_r, δ_r, λ_z, γ_z and δ_z of the inverse geometric transformation being determined, the geo-

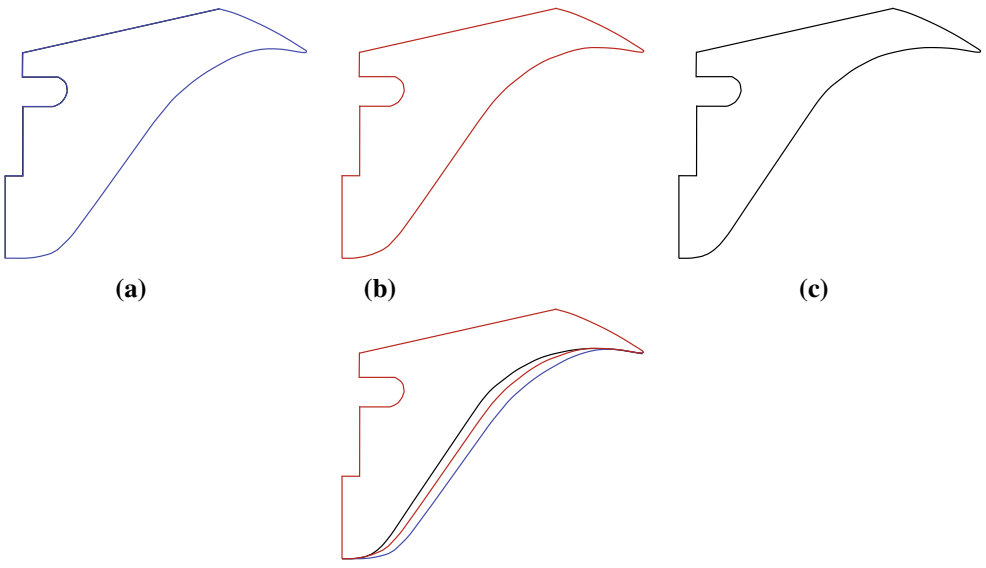

Fig. 5 The variations of geometry are limited by structures (**a**) and (**c**)

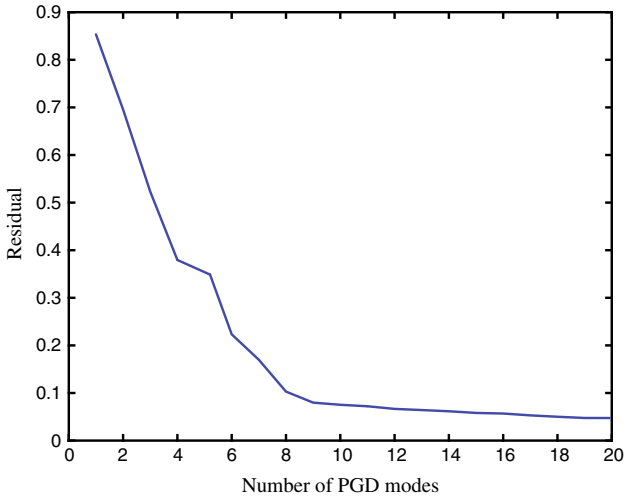

Fig. 6 Convergence curve

metric transformation can be easily obtained by a simple inversion.

6.2 Results

Let us come back to the engineering case presented in Sect. 2. Figure 5 shows the variations of geometry, we are dealing with. The convergence curve of the simulation can be seen in Fig. 6. It is possible to improve the convergence rate of the method following strategies presented in [26], what will be done in future developments. Anyway a PGD approximation of the displacement was obtained with 17 modes with an error on the residual of 5.6 % and from it, it was also deduced a PGD approximation of the strain $\boldsymbol{\varepsilon}$. Comparisons between the PGD solution and FE simulations show a good accuracy of the method developed (Fig. 7).

The question now is: how to explore efficiently the virtual charts? Indeed, this is a real difficulty because one must still have in mind that the final user need a decision support tool

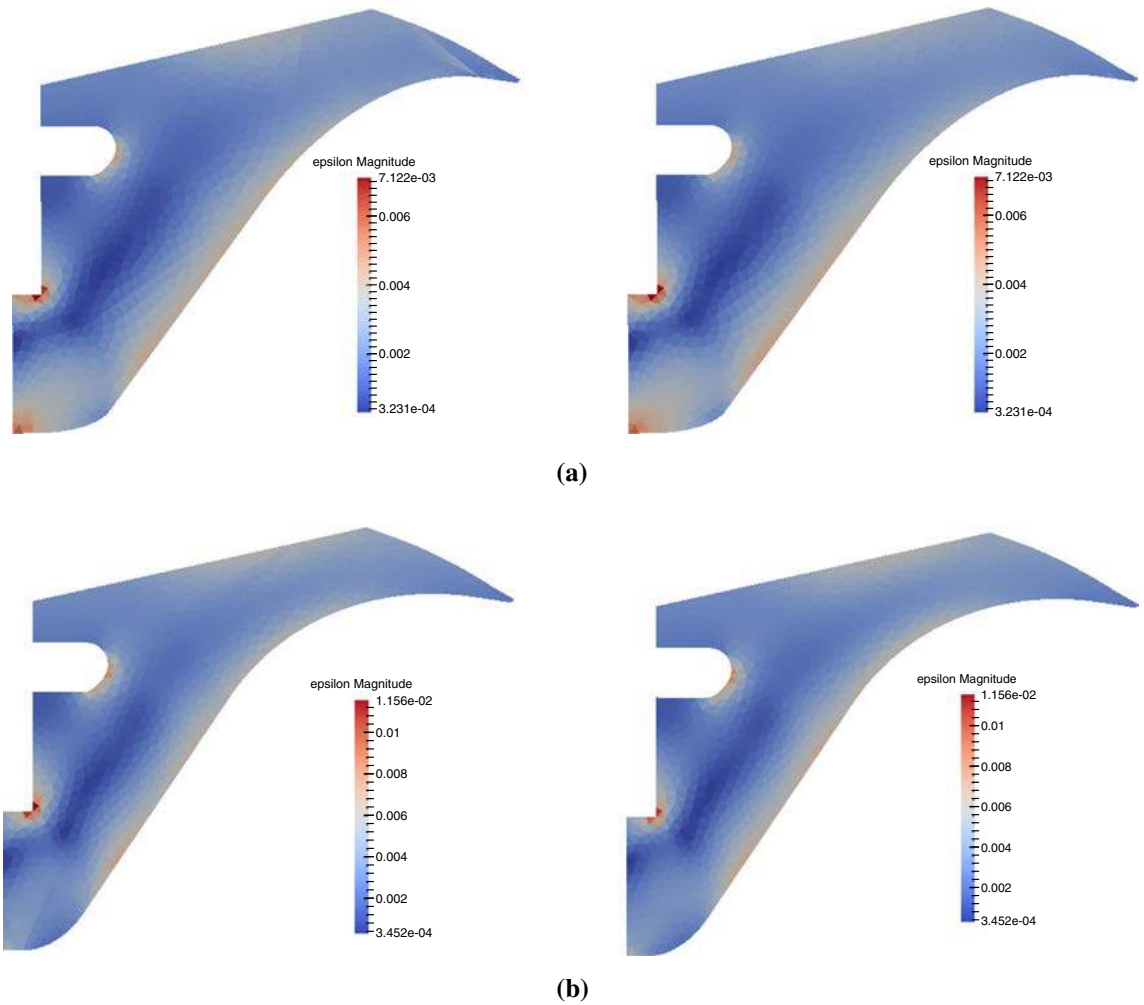

Fig. 7 Comparison between the solutions (strain magnitude) obtained with 17 PGD modes (*left*) and Finite Elements (*right*) for two given sets of parameters. **a** Geometric configuration defined in Fig. 5a. **b** Geometric configuration defined in Fig. 5c

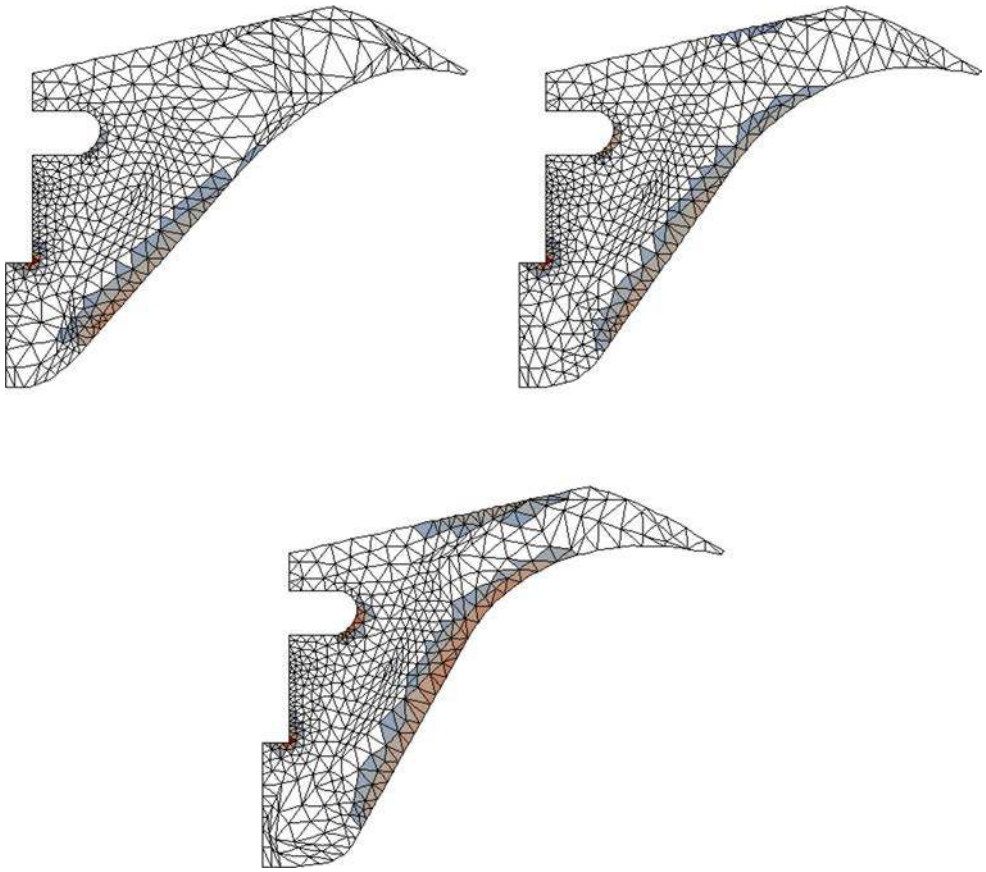

Fig. 8 Example of virtual chart representing the plastic zones for three different values of the set of parameters

as easy to use as a handbook. He must be able to look over all the possible geometries and solutions in an easy and practical way. For that purpose, we exploit the plug-in PXDMF for PARAVIEW developed at Ecole Centrale de Nantes which allows to plot separated variables data [8]. Given the modes associated to each considered field (displacement, strain...), PARAVIEW reconstructs the full fields in real-time by doing summations and multiplications. It allows the visualisation of the evolution of the fields varying each geometric bluepa-rameter.

The criteria used for the design is the size of the plastic zones. For the sake of demonstration in this paper, the calculations performed were linear elastic and the plastic zones are roughly evaluated by defining a threshold and considering the elements whose strain is superior to this value. However the proposed strategy could be extended to truly non-linear computation using the framework proposed in [23]. Figure 8 shows an example of the exploration of the virtual chart using the PXDMF plugin. It is important to note that, the virtual chart being computed, this exploration can be done in real time by the user.

7 Conclusions

In this work, geometric parameters were introduced into the *Proper Generalized Decomposition* following [3] and extended to complex geometries defined by splines and axisymmetric modelling. The approach allows to efficiently build a virtual chart of the solution that can be used, afterwards, in an optimisation process. Particular attention has been paid to integrating the methodology developed into a engineering design process. Due to the high degree of intrusiveness of the PGD, SAMCEF was interfaced with an in-house MATLAB code. The specific PGD operators are now computed by the MATLAB code and the resolution is done by SAMCEF after receiving these operators. We recall that a single optimisation of the structure with respect to geometric parameters should have certainly been done efficiently using the embedded SAMCEF optimisation procedures. However, this work is the first step of a more complex design process in which some other parameters, such as loadings, will be taken into account, which justifies the offline building of a virtual chart of solutions. This target

demonstrator will be composed of parts, whose behaviours can be highly non-linear (large strains, large displacements, non-linear materials) leading to many configurations of loadings that will be taken into account.

Acknowledgments This work was carried out in collaboration with Alain Bergerot from AIRBUS Defence & Space. We would like to thank SAMTECH (SIEMENS Company) for their help and availability so as to interface the method developed with SAMCEF.

Appendix 1

The *Proper Generalized Decomposition* is illustrated through a linear elastic problem where the constitutive law is parametrised by a set of Young's moduli $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_m) \in \boldsymbol{\mathcal{A}} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_m$. The problem is defined by the following governing equations and conditions:

– Equilibrium equations

$$\begin{cases} \boldsymbol{\nabla} \cdot \boldsymbol{\sigma} = \mathbf{0} & \text{in} \quad \Omega \times \boldsymbol{\mathcal{A}} \\ \boldsymbol{\sigma} \cdot \boldsymbol{n} = \boldsymbol{f} & \text{on} \quad \partial_d \Omega \times \boldsymbol{\mathcal{A}} \end{cases} \tag{44}$$

– Constitutive law

$$\boldsymbol{\sigma} = \mathbb{C}(\boldsymbol{\alpha})\, \boldsymbol{\varepsilon} \quad \text{in} \quad \Omega \times \boldsymbol{\mathcal{A}} \tag{45}$$

$$= \mathbb{C}(\boldsymbol{\alpha})\, \boldsymbol{\nabla}_s(\boldsymbol{u}) \quad \text{in} \quad \Omega \times \boldsymbol{\mathcal{A}} \tag{46}$$

where $\boldsymbol{\nabla}_s$ stands for the symmetrical part of $\boldsymbol{\nabla}$ and $\mathbb{C}$ is the Hooke's operator.

– Homogeneous boundary condition

$$\boldsymbol{u} = \mathbf{0} \quad \text{on} \quad \partial_u \Omega \times \boldsymbol{\mathcal{A}} \tag{47}$$

One writes the full variational formulation of the problem, i.e. the problem is not only integrated on space but also on parameters. For that purpose, we introduce the following functional spaces $\mathcal{V} = H_0^1(\Omega) = \left\{ \boldsymbol{v} \in H^1(\Omega) \,|\, \boldsymbol{v}_{|\partial_u \Omega} = \mathbf{0} \right\}$, $\boldsymbol{\mathcal{I}} = L^2(\boldsymbol{\mathcal{A}})$ and $\mathcal{I}_j = L^2(\mathcal{A}_j)$ for $j = 1, \ldots, m$. The problem therefore writes:
Find $\boldsymbol{u} \in \boldsymbol{\mathcal{I}} \otimes \mathcal{V}$ such that

$$\forall\, \boldsymbol{v} \in \boldsymbol{\mathcal{I}} \otimes \mathcal{V}, \quad \int_{\boldsymbol{\mathcal{A}}} \int_{\Omega} \boldsymbol{\sigma}(\boldsymbol{u}) : \boldsymbol{\varepsilon}(\boldsymbol{v})\, \mathrm{d}\Omega\, \mathrm{d}\boldsymbol{\mathcal{A}} = \int_{\boldsymbol{\mathcal{A}}} \int_{\partial_d \Omega} \boldsymbol{f} \cdot \boldsymbol{v}\, \mathrm{d}S\, \mathrm{d}\boldsymbol{\mathcal{A}} \tag{48}$$

where $\otimes$ stands for the tensor product.
One seeks a PGD approximation of the displacement.

$$\boldsymbol{u}(\boldsymbol{\alpha}, \boldsymbol{X}) \approx \boldsymbol{u}_n(\boldsymbol{\alpha}, \boldsymbol{X}) = \sum_{i=1}^{n} \lambda_i(\boldsymbol{\alpha}) \boldsymbol{\Lambda}_i(\boldsymbol{X}) = \sum_{i=1}^{n} \prod_{j=1}^{m} \lambda_i^j(\alpha_j) \boldsymbol{\Lambda}_i(\boldsymbol{X}) \tag{49}$$

The different modes are computed through an iterative algorithm. At the enrichment step n, we suppose the separated variables representation $\boldsymbol{u}_n$ known. The new $(m+1)$-uplet $\left(\lambda^1, \ldots, \lambda^m, \boldsymbol{\Lambda}\right) \in \mathcal{I}_1 \times \cdots \times \mathcal{I}_m \times \mathcal{V}$ is then computed at enrichment step $n+1$:

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \prod_{r=1}^{m} \lambda^r \boldsymbol{\Lambda} \tag{50}$$

To do so, the following expression of the test function $\boldsymbol{v}$ is chosen:

$$\boldsymbol{v} = \prod_{r=1}^{m} \lambda^r \boldsymbol{\Lambda}^* + \sum_{j=1}^{m} \lambda^{j*} \prod_{\substack{r=1 \\ r \neq j}}^{m} \lambda^r \boldsymbol{\Lambda} \tag{51}$$

with $\boldsymbol{\Lambda}^* \in \mathcal{V}$ and $\lambda^{j*} = \mathcal{I}_j$ for $j = 1, \ldots, m$. (5) becomes: Find $\left(\lambda^1, \ldots, \lambda^m, \boldsymbol{\Lambda}\right) \in \mathcal{I}_1 \times \cdots \times \mathcal{I}_m \times \mathcal{V}$ such that

$$\begin{aligned} &\forall \left(\lambda^{1*}, \ldots, \lambda^{m*}, \boldsymbol{\Lambda}^*\right) \in \mathcal{I}_1 \times \cdots \times \mathcal{I}_m \times \mathcal{V}, \\ &\int_{\boldsymbol{\mathcal{A}}} \int_{\Omega} \mathbb{C}(\boldsymbol{\alpha})\, \boldsymbol{\nabla}_s \left(\boldsymbol{u}_n + \prod_{r=1}^{m} \lambda^r \boldsymbol{\Lambda} \right) : \boldsymbol{\nabla}_s \left(\prod_{r=1}^{m} \lambda^r \boldsymbol{\Lambda}^* \right. \\ &\left. + \sum_{j=1}^{m} \lambda^{j*} \prod_{\substack{r=1 \\ r \neq j}}^{m} \lambda^r \boldsymbol{\Lambda} \right) \mathrm{d}\Omega\, \mathrm{d}\boldsymbol{\mathcal{A}} \\ &= \int_{\boldsymbol{\mathcal{A}}} \int_{\partial_d \Omega} \boldsymbol{f} \cdot \left(\prod_{r=1}^{m} \lambda^r \boldsymbol{\Lambda}^* + \sum_{j=1}^{m} \lambda^{j*} \prod_{\substack{r=1 \\ r \neq j}}^{m} \lambda^r \boldsymbol{\Lambda} \right) \mathrm{d}S\, \mathrm{d}\boldsymbol{\mathcal{A}} \end{aligned} \tag{52}$$

The computations of the different integrals are highly facilitated by the separated variables representation of the integrand since they can be done independently from one another. However, the problem is not linear any more with respect to the test function given in (51). The $(m+1)$-uplet $\left(\lambda^1, \ldots, \lambda^m, \boldsymbol{\Lambda}\right)$ is consequently computed thanks to a fixed-point algorithm. The parametric and space problems are solved alternatively. In practice, the fixed-point algorithm is stopped before reaching convergence (only 2 or 3 iterations are performed). The space basis $(\boldsymbol{\Lambda}_i)$ is, then, conserved and the functions $\left(\lambda_i^1, \ldots, \lambda_i^m\right)$ are computed, once again, during the so-called update step [25].

Appendix 2

Let us illustrate the procedure described in Appendix 1 through a simple numerical example. The considered geometry is a Γ-shaped structure (see Fig. 9) which is clamped

at one side and subjected to an upward load of 1000 N at the other side. The structure is partitioned into three different sub-domains and the Young's modulus of each domain, being considered as a parameter, can vary from 50 to 1000 GPa. The Poisson's ratio is homogeneous and equal to 0.3.

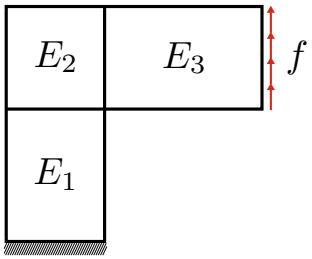

Fig. 9 The test structure is partitioned into three sub-domains, the Young's modulus of each domain being considered as a parameter

A PGD approximation of the generalised displacement field $\boldsymbol{u}$ is sought:

$$\begin{aligned}\boldsymbol{u}(\boldsymbol{\alpha}, \boldsymbol{X}) &= \boldsymbol{u}(E_1, E_2, E_3, \boldsymbol{X}) \\ &= \sum_{i=1}^{m} \lambda_i^1(E_1)\,\lambda_i^2(E_2)\,\lambda_i^3(E_3)\,\tilde{\boldsymbol{u}}_i(\boldsymbol{X})\end{aligned} \tag{53}$$

which gives after space discretisation:

$$\mathbf{U}(E_1, E_2, E_3) = \sum_{i=1}^{m} \lambda_i^1(E_1)\,\lambda_i^2(E_2)\,\lambda_i^3(E_3)\,\tilde{\mathbf{U}}_i \tag{54}$$

The modes, i.e. the functions λ_i^1, λ_i^2, λ_i^3 and $\tilde{\mathbf{U}}_i$ are computed thanks to Algorithm 1. The four first modes associated to the Young's moduli E_1, E_2 and E_3 are plotted in Fig. 10 and the four first spatial modes are represented in Fig. 11.

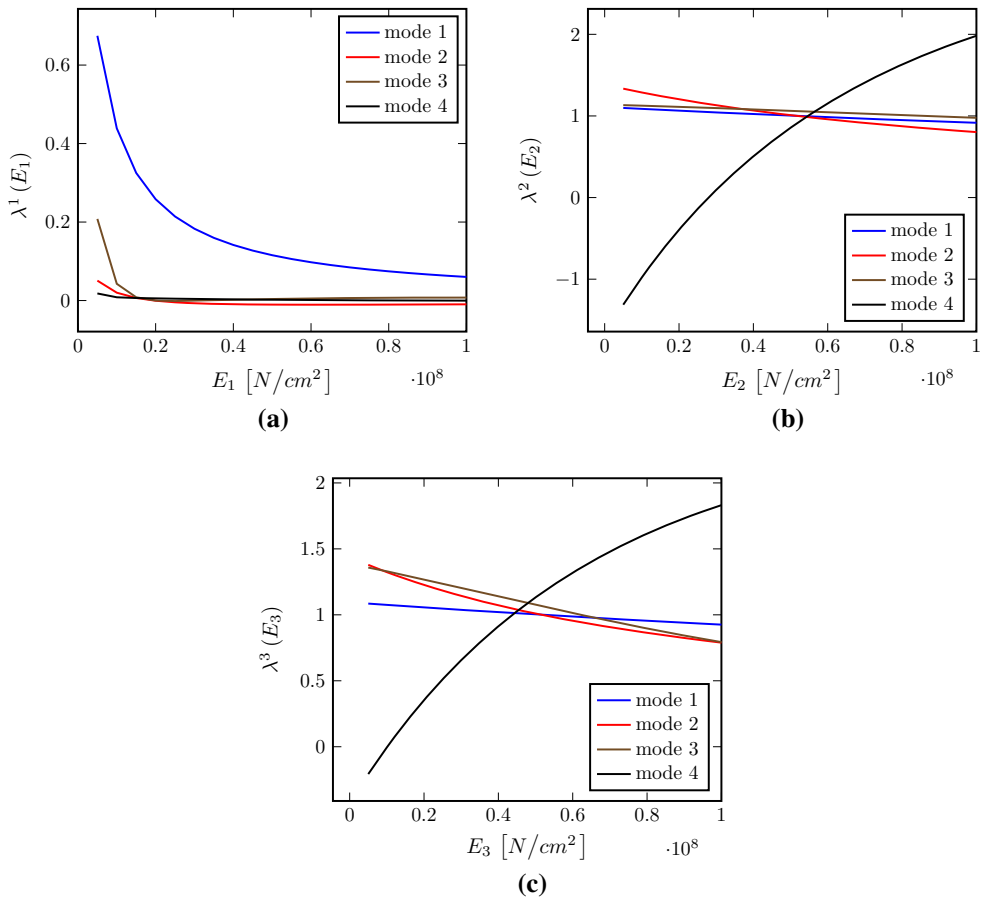

Fig. 10 First modes associated to each Young's modulus. Modes associated to the Young's modulus E_1 (**a**), modes associated to the Young's modulus E_2 (**b**) and msodes associated to the Young's modulus E_3 (**c**) considered as parameters

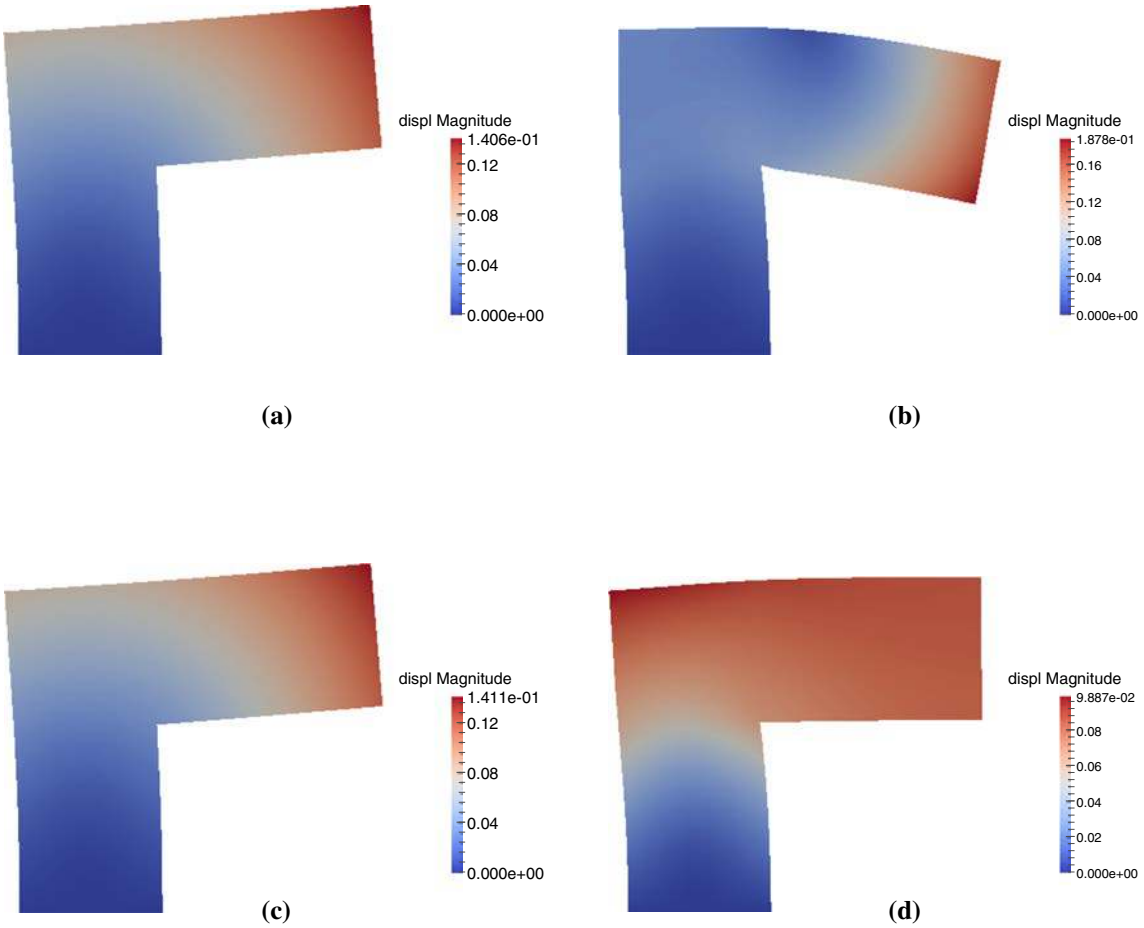

Fig. 11 First spatial modes. **a** First spatial mode $\tilde{\mathbf{U}}_1$, **b** second spatial mode $\tilde{\mathbf{U}}_2$, **c** third spatial mode $\tilde{\mathbf{U}}_3$, **d** fourth spatial mode $\tilde{\mathbf{U}}_4$

References

1. Aguado JV, Huerta A, Chinesta F, Cueto E (2015) Real-time monitoring of thermal processes by reduced-order modeling. Int J Numer Methods Eng 102(5):991–1017
2. Alfaro I, González D, Bordeu F, Leygue A, Ammar A, Cueto E, Chinesta F (2014) Real-time in silico experiments on gene regulatory networks and surgery simulation on handheld devices. J Comput Surg 1(1):1
3. Ammar A, Huerta A, Chinesta F, Cueto E, Leygue A (2014) Parametric solutions involving geometry: a step towards efficient shape optimization. Comput Methods Appl Mech Eng 268:178–193
4. Ammar A, Mokdad B, Chinesta F, Keunings R (2007) A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids: Part II: Transient simulation using space-time separated representations. J Non-Newton Fluid Mech 144(2–3):98–121
5. Ammar A, Zghal A, Morel F, Chinesta F (2015) On the space-time separated representation of integral linear viscoelastic models. Comptes Rendus de Mécanique 343(4):247–263
6. Barrault MYM, Nguyen N, Patera A (2004) An "empirical interpolation" method: application to efficient reduced-basis discretization of partial differential equations. Comptes Rendus de l'Académie des Sciences Paris 339:667–672
7. Bernoulli C (1836) Vademecum des Mechanikers. J. G. Cotta, Stuttgart und Tübingen
8. Bognet B, Leygue A, Chinesta F, Bordeu F (2012) Parametric shape and material simulations for optimized parts design. In: ASME 2012 11th biennial conference on engineering systems design and analysis, American Society of Mechanical Engineers, pp 217–218
9. Chinesta F, Keunings R, Leygue A (2014) The Proper generalized decomposition for advanced numerical simulations: a Primer., SpringerBriefs in Applied Sciences and TechnologySpringer, Cham
10. Chinesta F, Ladevèze P, Cueto E (2011) A short review on model order reduction based on proper generalized decomposition. Arch Comput Methods Eng 18(4):395–404
11. Chinesta F, Leygue A, Bordeu F, Aguado J, Cueto E, Gonzalez D, Alfaro I, Ammar A, Huerta A (2013) PGD-based computational vademecum for efficient design, optimization and control. Arch Comput Methods Eng 20(1):31–59
12. Cremonesi M, Néron D, Guidault PA, Ladevèze P (2013) A PGD-based homogenization technique for the resolution of nonlinear multiscale problems. Comput Methods Appl Mech Eng 267:275–292
13. De Lathauwer L, De Moor B, Vandewalle J (2000) A multilinear singular value decomposition. SIAM J Matrix Anal Appl 21(4):1253–1278
14. González D, Alfaro I, Quesada C, Cueto E, Chinesta F (2015) Computational vademecums for the real-time simulation of haptic collision between nonlinear solids. Comput Methods Appl Mech Eng 283(1):210–223

15. Gunzburger M, Peterson J, Shadid J (2007) Reduced-order modeling of time-dependent PDEs with multiple parameters in the boundary data. Comput Methods Appl Mech Eng 196:1030–1047
16. Ladevèze P (1999) Nonlinear computationnal structural mechanics—new approaches and non-incremental methods of calculation. Springer, New York
17. Ladevèze P (2013) The Virtual chart concept in computational structural mechanics (plenary lecture). In: COMPLAS XII, 3–5 Sept 2013, Barcelona (Spain)
18. Ladevèze P (2014) PGD in linear and nonlinear Computational Solid Mechanics. In: Separated Representations and PGD-Based Model Reduction, Springer, pp 91–152
19. Lieu T, Farhat C, Lesoinne A (2006) Reduced-order fluid/structure modeling of a complete aircraft configuration. Comput Methods Appl Mech Eng 195(41–43):5730–5742
20. Maday Y, Ronquist E (2004) The reduced-basis element method: application to a thermal fin problem. J Sci Comput 26(1):240–258
21. Manzoni A, Quarteroni A, Rozza G (2012) Model reduction techniques for fast blood flow simulation in parametrized geometries. Int J Numer Methods Biomed Eng 28(6–7):604–625
22. Néron D, Ben Dhia H, Cottereau R (2015) A decoupled strategy to solve reduced-order multimodel problems in the PGD and Arlequin frameworks. Comput Mech. doi:10.1007/s00466-015-1236-0
23. Néron D, Boucard PA, Relun N (2015) Time-space PGD for the rapid solution of 3D nonlinear parametrized problems in the many-query context. Int J Numer Methods Eng 103(4):275–292
24. Néron D, Dureisseix D (2008) A computational strategy for poroelastic problems with a time interface between coupled physics. Int J Numer Methods Eng 73(6):783–804
25. Néron D, Ladevèze P (2010) Proper generalized decomposition for multiscale and multiphysics problems. Arch Comput Methods Eng 17(4):351–372
26. Nouy A (2010) A priori model reduction through proper generalized decomposition for solving time-dependent partial differential equations. Comput Methods Appl Mech Eng 199(23):1603–1626
27. Patera AT, Rozza G (2006) Reduced Basis Approximation and A Posteriori Error Estimation for Parametrized Partial Differential Equations. Version 1.0. Copyright MIT 2006, to appear in (tentative rubric) MIT Pappalardo Graduate Monographs in Mechanical Engineering. http://augustine.mit.edu
28. Raghavan B, Breitkopf P, Tourbier Y, Villon P (2013) Towards a space reduction approach for efficient structural shape optimization. Struct Multidiscip Optim 48(5):987–1000
29. Relun N, Néron D, Boucard PA (2013) A model reduction technique based on the PGD for elastic-viscoplastic computational analysis. Comput Mech 51(1):83–92
30. Rozza G, Veroy K (2007) On the stability of the reduced basis method for Stokes equations in parametrized domains. Comput Methods Appl Mech Eng 196(7):1244–1260
31. Ryckelynck DFC, Cueto E, Ammar A (2006) On the a priori model reduction: overview and recent developments. Arch Comput Methods Eng 13(1):91–128
32. Zlotnik S, Díez P, Modesto D, Huerta A (2015) Proper generalized decomposition of a geometrically parametrized heat problem with geophysical applications. Int J Numer Methods Eng 103(10):737–758