



Science Arts & Métiers (SAM)

is an open access repository that collects the work of Arts et Métiers Institute of Technology researchers and makes it freely available over the web where possible.

This is an author-deposited version published in: <https://sam.ensam.eu>
Handle ID: <http://hdl.handle.net/10985/10241>

To cite this version :

Francisco CHINESTA SORIA, Adrien LEYGUE, Felipe BORDEU, Jose Vicente AGUADO, Elias CUETO, David GONZALEZ, Icíar ALFARO, Amine AMMAR, Antonio HUERTA - PGD-Based Computational Vademecum for Efficient Design, Optimization and Control - Archives of Computational Methods in Engineering - Vol. 20, n°1, p.31-59 - 2013

Any correspondence concerning this service should be sent to the repository

Administrator : scienceouverte@ensam.eu



PGD-Based *Computational Vademecum* for Efficient Design, Optimization and Control

F. Chinesta · A. Leygue · F. Bordeu · J.V. Aguado ·
E. Cueto · D. Gonzalez · I. Alfaro · A. Ammar ·
A. Huerta

Abstract In this paper we are addressing a new paradigm in the field of simulation-based engineering sciences (SBES)

to face the challenges posed by current ICT technologies. Despite the impressive progress attained by simulation capabilities and techniques, some challenging problems remain today intractable. These problems, that are common to many branches of science and engineering, are of different nature. Among them, we can cite those related to high-dimensional problems, which do not admit mesh-based approaches due to the exponential increase of degrees of freedom. We developed in recent years a novel technique, called Proper Generalized Decomposition (PGD). It is based on the assumption of a separated form of the unknown field and it has demonstrated its capabilities in dealing with high-dimensional problems overcoming the strong limitations of classical approaches. But the main opportunity given by this technique is that it allows for a completely new approach for classic problems, not necessarily high dimensional. Many challenging problems can be efficiently cast into a multi-dimensional framework and this opens new possibilities to solve old and new problems with strategies not envisioned until now. For instance, parameters in a model can be set as additional extra-coordinates of the model. In a PGD framework, the resulting model is solved once for life, in order to obtain a general solution that includes all the solutions for every possible value of the parameters, that is, a sort of *computational vademecum*. Under this rationale, optimization of complex problems, uncertainty quantification, simulation-based control and real-time simulation are now at hand, even in highly complex scenarios, by combining an off-line stage in which the general PGD solution, the vademecum, is computed, and an on-line phase in which, even on deployed, handheld, platforms such as smartphones or tablets, real-time response is obtained as a result of our queries.

1 Introduction

1.1 Motivation

Six unique initiatives have been recently selected (and funded with 100 millions of euros per year) by the European Research Council based solely on their potential for realizing scientific breakthroughs and influencing Europe's social and industrial challenges, including health. Their aim will then be to deliver major breakthroughs in information and communication technologies (ICT), with the potential to provide solutions to some of society's biggest challenges. Despite being different there is a common ingredient to all of them, which is to emphasize the necessity of making use of advanced simulation-driven sciences and engineering, as will be highlighted below. The six contenders, from which the two flagship initiatives will be selected, are [37]:

1. Guardian Angels for a Smarter Life [38]: a project aimed at developing tiny devices without batteries that act like thinking and autonomous personal assistants, providing information and communication technologies to assist people in all sorts of complex situations delivering features and characteristics that go well beyond human capabilities.
2. The Human Brain Project [39] whose goal is to understand the way the human brain works. The long-term goal of the Human Brain Project is to build the informatics, modeling, and supercomputing technologies that are needed to simulate and understand the human brain.
3. IT Future of Medicine [40] proposes a data-driven, individualized medicine of the future, based on the molecular/physiological/anatomical data from individual patients. The project outcomes will enable data-driven real-time calculation of health, disease, therapy and its effects for individual patients.
4. Robot Companions for Citizens [41]: a project devoted to developing soft-skinned and intelligent robots with highly developed perceptive, cognitive and emotional skills. Robot Companions for Citizens will be based on the novel solid articulated structures with flexible properties displaying soft behavior, haptic devices and simulation based real time control in deployed systems. These companions will also have new levels of perceptual, cognitive and emotive capabilities and be aware of their physical and social surroundings and respond accordingly.
5. FuturICT Knowledge Accelerator and Crisis-Relief System [42]: What if global scale computing facilities were available that could analyze most of the data available in the world? What insights could scientists gain about the way society functions? What new laws of nature would be revealed? Could society discover a more sustainable way of living? ICT (Information and Communication

Technology) can analyze vast amounts of data and complex situations so as to better predict natural disasters, or manage and respond to man-made disasters that cross national borders or continents.

6. Graphene Science and technology for ICT and beyond [43]: Graphene is a new substance developed by atomic and molecular scale manipulation that could replace silicon as the wonder material of the 21st century. This aims to explore revolutionary potentials, in terms of both conventional as well as radically new fields of Information and Communication Technologies applications.

It is now well known [39] that the human brain consumes 4 watts for performing some tasks that today's computers will require the power of several nuclear plants. It is then clear that our computers and algorithms for addressing the models encountered in science and engineering are definitively suboptimal. The above six flagship projects share some key aspects related to efficient computational sciences. It is expected that these projects will reach a certain number of breakthroughs, but all of them will face important limitations of today's computer capabilities and, notably, simulation techniques.

All these society needs require fast and accurate solutions, in general data-driven, of very complex models, involving an unimaginable amount of information, in most cases in real time and on deployed platforms. Up to now, the solution of complex models, preferably fast and accurate, is addressed by using high performance computing and hyper powerful computing platforms. Obviously the consecution of the above "dreams" will require as much as computational power (supercomputing) as possible, and consequently, advances in hardware and software for high-performance computing will be necessary. But at the same time, there is a need for a new generation simulation techniques, beyond high-performance computing or nowadays approaches (most of them proposed 40 years ago), to simply improve efficiency or to allow getting results when other alternatives fail in the above challenging scenarios.

All the above challenging problems are data-driven. The importance of Dynamic Data-Driven Application Systems—DDDAS—in the forthcoming decades has been already noticed by the NSF Blue Ribbon Panel on Simulation Based Engineering Sciences report, that in 2006 included DDDAS as one of the five core issues or challenges in the field for the next decade (together with multi-scale simulation, model validation and verification, handling large data and visualization). This panel concluded that "Dynamic data-driven application systems will rewrite the book on the validation and verification of computer predictions" and that "research is needed to effectively use and integrate data-intensive computing systems, ubiquitous sensors and high-resolution detectors, imaging devices, and other data-gathering storage and distribution devices, and to develop methodologies and

theoretical frameworks for their integration into simulation systems” [29, 65, 66]. Moreover, the NSF believes that “. . . The DDDAS community needs to reach a critical mass both in terms of numbers of investigators, and in terms of the depth, breadth and maturity of constituent technologies . . .” [65].

1.2 Nowadays Computational Issues

Today many problems in science and engineering remain intractable, in spite of the impressive progresses attained in modeling, numerical analysis, discretization techniques and computer science during the last decade, because their numerical complexity, or the restrictions imposed by different requirements (real-time on deployed platforms, for instance) make them unaffordable for today’s technologies.

We can enumerate different challenging scenarios for efficient numerical simulations:

- The first one concerns models that are defined in high dimensional spaces, usually encountered in quantum chemistry describing the structure and mechanics of materials [4, 21], the kinetic theory description of complex materials [16, 52], social dynamics and economic systems, vehicular traffic flow phenomena, complex biological systems involving mutation and immune competition, crowds and swarms encountered in congested and panic flows, among many other unimaginable possibilities (see [13] and the references therein); the chemical modeling in too dilute systems where the concept of concentration cannot be used, that results in the so-called chemical master equation governing for example cell signaling and other phenomena in molecular biology [10].

Models defined in high dimensional spaces suffer the so-called curse of dimensionality. If one proceeds to the solution of a model defined in a space of dimension d by using a standard mesh based discretization technique, where \mathcal{M} nodes are used for discretizing each space coordinate, the resulting number of nodes reaches the astronomical value of \mathcal{M}^d . With $\mathcal{M} \approx 10^3$ (a very coarse description in practice) and $d \approx 30$ (a very simple model) the numerical complexity results 10^{90} . It is important to recall that 10^{80} is the presumed number of elementary particles in the universe!

Traditionally, high dimensional models were addressed by using stochastic simulations. However these techniques have their own challenges: variance reduction is always an issue and the construction of distribution functions in high dimensional spaces remains in most cases unaffordable. It is also quite difficult within the stochastic framework to implement parametric or sensitivity analysis that go beyond the brute force approach of computing a large number of expensive, individual simulations.

- Online control can be carried out following different approaches. The most common one consists in considering systems as a black box whose behavior is modeled by a transfer function relating certain inputs to certain outputs. This modeling that may seem poor has as main advantage the possibility of proceeding rapidly due to its simplicity. This compromise between accuracy and rapidity was often used in the past and this pragmatic approach has allowed us to control processes and to optimize them, once the transfer function modeling the system is established.

The establishment of such goal-oriented transfer function is the trickiest point. For this purpose, it is possible to proceed from a sometimes overly simplified physical model or directly from experiments (allowing us to extract a phenomenological goal-oriented transfer function) or from a well-balanced mixture of both approaches. In all cases, the resulting modeling can only be applied within the framework that served to derive it. However, on one hand, the fine description of systems requires a sufficiently detailed description of them and, in that case, traditional goal-oriented simplified modeling becomes inapplicable. On the other hand, actual physical models result, in general, in complex mathematical objects, non-linear and strongly coupled partial differential equations. Such mathematical objects are representing physical reality up to a certain degree of accuracy. However, the available numerical tools capable of solving these complex models require the use of powerful computers that can require hours, days and weeks to solve them. Known as numerical simulation, its output solution is very rich but it seems inapplicable for control purposes that require fast responses, often in real-time.

Until now, numerical simulation has been used offline but in some cases it allows us to define simplified models (with their inherent limitations and drawbacks) running in real-time that could be used online but such simplified modeling has the previously quoted drawbacks.

- Many problems in parametric modeling, inverse identification, and process or shape optimization, usually require, when approached with standard techniques, the direct computation of a very large number of solutions of the concerned model for particular values of the problem parameters. When the number of parameters increases such a procedure becomes inapplicable.
- Traditionally, Simulation-based Engineering Sciences—SBES—relied on the use of static data inputs to perform the simulations. These data could be parameters of the model(s) or boundary conditions. The word static is intended to mean here that these data could not be modified during the simulation. A new paradigm in the field of Applied Sciences and Engineering has emerged in the last decade. Dynamic Data-Driven Application Systems

(DDDAS) constitute nowadays one of the most challenging applications of simulation-based Engineering Sciences. By DDDAS we mean a set of techniques that allow the linkage of simulation tools with measurement devices for real-time control of simulations. DDDAS entails the ability to dynamically incorporate additional data into an executing application, and in reverse, the ability of an application to dynamically steer the measurement process.

In this context, real time simulators are needed in many applications. One of the most challenging situations is that of haptic devices, where forces must be translated to the peripheral device at a rate of 500 Hz. Control, malfunctioning identification and reconfiguration of malfunctioning systems also need to run in real time. All these problems can be seen as typical examples of DDDAS.

- Augmented reality is another area in which efficient (fast and accurate) simulation is urgently needed. The idea is supplying in real time appropriate information to the reality perceived by the user. Augmented reality could be an excellent tool in many branches of science and engineering. In this context, light computing platforms are appealing alternatives to heavy computing platforms that in general are expensive and whose use requires technical knowledge.
- Inevitable uncertainty. In science and engineering, in its widest sense, it now seems obvious that there are many causes of variability. The introduction of such variability, randomness and uncertainty is a priority for the next decade. Although it was a priority in the preceding decade, the practical progress attained seems fairly weak.

While the previous list is by no means exhaustive, it includes a set of problems with no apparent relationship between them that can however be treated in a unified manner as will be shown in what follows. Their common ingredient is our lack of capabilities (or knowledge) to solve them numerically in a direct, traditional way.

2 Fast Calculations from a Historical Perspective

The human being throughout the history developed several facilities for giving fast responses to a variety of questions. Thus, abaci were used 2700 years B.C. in Mesopotamia. This abacus was a sort of counting frame primarily used for performing arithmetic calculations. We associate this abacus to a bamboo frame with beads sliding on wires, however, originally they were beans or stones moved in grooves in sand or on tablets of wood, stone, or metal. The abacus was in use centuries before the adoption of the written modern numeral system and is still widely used by readers. There are many variants, the Mesopotamian abacus, the Egyptian, Persian, Greek, Roman, Chinese, Indian, Japanese, Korean, native American, Russian, etc.

However, the initial arithmetic needs were rapidly complemented with more complex representations. We are considering some few variants:

- Charts appeared for graphical representation of data with multiple meanings. However, there are common features that provide the chart with its ability to extract meaning from data. In general a chart is graphical, containing very little text, since humans infer meaning from pictures quicker than from text. A particular variant of charts in the Nomogram.
- Nomography, is the graphical representation of mathematical relationships or laws. It is an area of practical and theoretical mathematics invented in 1880 by Philbert Maurice d'Ocagne and used extensively for many years to provide engineers with fast graphical calculations of complicated formulas to a practical precision. Thus, a nomogram can be considered as a graphical calculating device. There are thousands of examples on the use of nomograms in all the fields of sciences and engineering.

The former facilities allowed for fast calculations and data manipulations. Nomograms can be easily constructed when the mathematical relationships that they express are purely algebraic, eventually non-linear. In those cases it was easy to represent some outputs as a function of some inputs. The calculation of these data representations was performed off-line and then used on-line in many branches of engineering sciences for design and optimization.

However, the former procedures fail when addressing more complex scenarios. Thus, sometimes engineers manipulate not properly understood physics and in that case the construction of nomograms based on a too coarse modelling could be dangerous. In that cases one could proceed by making several experiments from which defining a sort of experiment-based nomogram. In other cases the mathematical object to be manipulated consists of a system of complex coupled non-linear partial differential equations, whose solution for each possible combination of the values of the parameters that it involves is simply unimaginable for the nowadays computational availabilities. In these cases experiments or expensive computational solutions are performed for some possible states of the system, from which a simplified model linking the inputs to the outputs of interest is elaborated. These simplified models have different names: surrogate models, metamodels, response surface methodologies, ... Other associated tricky questions are the one that concerns the best sampling strategy (Latin hypercube, ...) and also the one concerning the appropriate interpolation techniques for estimating the response at an unmeasured position from observed values at surrounding locations. Many possibilities exist, being Kriging one of the most widely used for interpolating data.

All these techniques allow defining a sort of numerical or graphical handbook. One of the earliest and most widely

known within engineering practice is that of Bernoulli [14]. However, we must accept a certain inevitable inaccuracy when estimating solutions from the available data. It is the price to pay if neither experimental measurements nor numerical solutions of the fine but expensive model are achievable for each possible scenario.

Recently model order reduction opened new possibilities. First, proper orthogonal decompositions (POD) allows extracting the most significant characteristic of the solution, that can be then applied for solving models slightly different to the ones that served to defined the reduced approximation bases. There is an extensive literature. The interested readers can refer to [2, 5, 15, 20, 36, 58, 60–63, 67, 74] and the numerous references therein. The extraction of the reduced basis is the tricky point when using POD-based model order reduction, as well its adaptivity when addressing scenarios far from the ones considered when constructing the reduced basis [72, 73]. Another issue lies in the error control, and its connection with verification and validation.

The calculation of the reduced basis is not unique. There are many alternatives. Some ones introduce some improvements on the POD methodology, as is the case of the Goal Oriented Model Constrained Optimization approach (see [19] and the references therein) or the modal identification method (see [33] and the references therein). The Branch Eigenmodes Reduction Method combined with the amalgam method is another appealing constructor of reduced bases [77].

Another family of model reduction techniques lies in the used of reduced basis constructed by combining a greedy algorithm and a priori error indicator. It needs for some amount off-line work but then the reduced basis can be used on-line for solving different models with a perfect control of the solution accuracy because the availability of error bounds. When the error is inadmissible, the reduced basis can be enriched by invoking again the same greedy algorithm. The interested readers can refer to [56, 57, 71, 76] and the references therein. The main drawback of such an approach is the amount of data that must be computed, stored and then manipulated.

Separated representations were introduced in the 80s by Pierre Ladeveze that proposed a space-time separated representation of transient solutions involved in strongly non-linear models, defining a non-incremental integration procedure. The interested reader can refer to the numerous Ladeveze’s works [44–49, 59, 68]. Later, separated representations were employed in the context of stochastic modelling [64] as well as for solving multidimensional models suffering the so-called curse of dimensionality, some of them never solved before [1]. The techniques making use of separated representations computed on the fly were called Proper Generalized Decompositions—PGD—.

PGD constitutes an efficient multidimensional solver that allows introducing model parameters (boundary conditions,

initial conditions, geometrical parameters, material and process parameters ...) as extra-coordinates. Then by solving only once and off-line the resulting multidimensional model we have access to the parametric solution that can be viewed as a sort of handbook or vademecum than can be then used on-line.

In what follows, we are describing within the PGD approach the way of introducing extra-coordinates of different nature. Later, we will prove the potentiality of such an approach for the efficient solution of a variety of problems.

2.1 PGD at a Glance

Consider a problem defined in a space of dimension d for the unknown field $u(x_1, \dots, x_d)$. Here, the coordinates x_i denote any usual coordinate (scalar or vectorial) related to physical space, time, or conformation space in microscopic descriptions [1, 4], for example, but they could also include, as we illustrate later, problem parameters such as boundary conditions or material parameters. We seek a solution for $(x_1, \dots, x_d) \in \Omega_1 \times \dots \times \Omega_d$.

The PGD yields an approximate solution in the separated form:

$$u(x_1, \dots, x_d) \approx \sum_{i=1}^N X_i^1(x_1) \cdots X_i^d(x_d) = \sum_{i=1}^N \prod_{j=1}^d X_i^j(x_j) \quad (1)$$

The PGD approximation is thus a sum of N functional products involving each a number d of functions $X_i^j(x_j)$ that are unknown *a priori*. It is constructed by successive enrichment, whereby each functional product is determined in sequence. At a particular enrichment step $n + 1$, the functions $X_i^j(x_j)$ are known for $i \leq n$ from the previous steps, and one must compute the new product involving the d unknown functions $X_{n+1}^j(x_j)$. This is achieved by invoking the weak form of the problem under consideration. The resulting problem is non-linear, which implies that iterations are needed at each enrichment step. A low-dimensional problem can thus be defined in Ω_j for each of the d functions $X_{n+1}^j(x_j)$.

If \mathcal{M} nodes are used to discretize each coordinate, the total number of PGD unknowns is $N \cdot \mathcal{M} \cdot d$ instead of the \mathcal{M}^d degrees of freedom involved in standard mesh-based discretizations. We will come back later to the issues related to the convergence and optimality of the separated representations.

2.2 Parametric Solution-Based Vademecum

In the case of a field depending on the physical space $\mathbf{x} \in \Omega_x \subset \mathcal{R}^3$, the time $t \in \mathcal{I}_t \subset \mathcal{R}$ and Q parameters $p^1, \dots, p^Q, p^j \in \Omega_{p^j}, j = 1, \dots, Q$, the solution is sought

under the separated form

$$u(\mathbf{x}, t, p^1, \dots, p^Q) \approx \sum_{i=1}^N X_i(\mathbf{x}) \cdot T_i(t) \cdot \prod_{j=1}^Q P_i^j(p^j) \quad (2)$$

As soon as this solution is available, after solving the multidimensional model within the PGD framework, we can have access to any possible solution. In fact the representation (2) implies an approximation of each function. Thus the space functions $X_i(\mathbf{x})$ could be approximated for example by using a finite element interpolation that implies knowing the value of each space function $X_i(\mathbf{x})$ at the \mathcal{M}_x nodes \mathbf{x}_k ($k = 1, \dots, \mathcal{M}_x$) of the mesh used for approximating them, i.e. $X_i(\mathbf{x}_k)$. The functions $T_i(t)$ depending on time will be expressed from the values of those functions at \mathcal{M}_t time instants t_l ($l = 1, \dots, \mathcal{M}_t$), i.e. $T_i(t_l)$. Finally, the functions depending on the different parameters $P_i^j(p^j)$ will be expressed from the values of those functions at \mathcal{M}_{p_j} values of each parameter p_r^j ($r = 1, \dots, \mathcal{M}_{p_j}$), i.e. $P_i^j(p_r^j)$. Thus, the solution consists of N vectors of size \mathcal{M}_x that contain the discrete representation of functions $X_i(\mathbf{x})$, N vectors of size \mathcal{M}_t that contain the discrete representation of functions $T_i(t)$ and N vectors of size \mathcal{M}_{p_j} that contain the discrete representation of functions $P_i^j(p^j)$, $j = 1, \dots, Q$.

If we imagine that $\mathcal{M}_x = \mathcal{M}_t = \mathcal{M}_{p^1} = \dots = \mathcal{M}_{p^Q} = \mathcal{M}$, then the solution representation involves $N \cdot (Q + 2)$ vectors of size \mathcal{M} , that is $N \cdot (Q + 2) \cdot \mathcal{M}$ values instead of the \mathcal{M}^{2+Q} involved by an equivalent mesh. If for a while we imagine $Q = 10$, $\mathcal{M} = 100$ and $N = 10$ the separated representation will involve 10^4 values instead of 10^{20} involved in a hypothetical equivalent mesh.

Thus, the PGD solver allowing the construction of the separated representation (2) can be viewed as a solver that constructs on the fly a compressed representation of the model solution. In the case of 2D models, involving two generic coordinates c_1 and c_2 and symmetric and positive definite differential operators, the separated representation constructed by the PGD solver reads:

$$u(c_1, c_2) \approx \sum_{i=1}^N C_i^1(c_1) \cdot C_i^2(c_2) \quad (3)$$

where the number of modes N corresponds to the one involved by the best separated representation of the model solution $u(c_1, c_2)$ obtained by applying on it a singular value decomposition—SVD—. In higher dimensions, the higher-order SVD counterpart, the so-called High Order SVD (HOSVD) is no more optimal. In any case the PGD constructor can be viewed as a sort of “a priori” HOSVD. Some mathematical results can be found in [7, 53] and the review [26] and the references therein.

When the differential operator involved in the model is non symmetric, the separated representation obtained by applying any of the nowadays available PGD solvers is suboptimal, that is, the number of terms in the finite sum N is higher than the number of modes that the SVD (in 2D) or its higher order counterpart would require when performing an “a posteriori” decomposition of the model solution.

In any case, even when PGD calculates suboptimal separated representations, at least it allows calculating solutions of highly multidimensional models that cannot be solved by using more experienced mesh-based discretizations. Thus the possibility of solving a problem, even when its solution representation involves sometimes more terms than strictly needed, is in any case a real success. Moreover, in general the solution of models involving many parameters as extra-coordinates is performed only once and off-line, and then it is particularized on-line. In these cases the optimality issue is not crucial. However, when the solution involves too many terms, with respect to an hypothetical optimal representation, i.e. $N \gg N^{opt}$, one can proceed to an “a posteriori” data compression, in order to alleviate as much as possible post-processing, that is vital when real time responses are envisaged. For this purpose, we assume that the suboptimal solution, calculated from the differential model by applying a PGD solver, reads

$$u(x_1, \dots, x_d) \approx \sum_{i=1}^N \prod_{j=1}^d X_i^j(x_j) \quad (4)$$

and we look for an enhanced representation $u^{enh}(x_1, \dots, x_d)$

$$u^{enh}(x_1, \dots, x_d) \approx \sum_{i=1}^{\tilde{N}} \prod_{j=1}^d \tilde{X}_i^j(x_j) \quad (5)$$

that verifies

$$u^{enh}(x_1, \dots, x_d) = u(x_1, \dots, u_d) \quad (6)$$

whose integral counterpart writes

$$\int_{\Omega_1 \times \dots \times \Omega_d} u^* \cdot (u^{enh}(x_1, \dots, x_d) - u(x_1, \dots, u_d)) dx_1 \dots dx_d = 0 \quad (7)$$

that is solved by invoking again the PGD solver.

$u^{enh}(x_1, \dots, x_d)$ results a more compact representation, i.e. $\tilde{N} \leq N$, sometimes $\tilde{N} \ll N$. The problem (4)–(7) is equivalent to the application of a HOSVD on the suboptimal solution (4). This post-compression allows substantial storage savings, an important factor when using deployed devices, and also significant CPU time savings when manipulating data in post-processing tasks. In real time applications this post-compression can be decisive.

Until now, we presented the PGD as an efficient solver, however it can be also viewed as a model reduction strategy when computing only the most relevant terms of the separated representation. In that case an error is introduced, but because the few number of terms retained, storage and data post-processing can be efficiently performed.

In any case, as soon as a suboptimal or an enhanced separated representation is available, it can be viewed as a computational handbook, a metamodel, containing an unimaginable amount of information. For example, if the solution (2) applies for a model involving 10 parameters, all them considered as extra-coordinates, i.e. $Q = 10$, and 10 discrete values are considered for describing each parametric dependency, i.e. $\mathcal{M}_1 = \dots = \mathcal{M}_{10} = 10$ the solution (2) contains the information associated to 10^{10} possible scenarios. Obtaining an equivalent amount of information would require the solution of 10^{10} transient 3D problems. Now, from this extremely rich metamodel, one could obtain the solution for any possible scenario $u(\mathbf{x}, t, p^1, \dots, p^{10})$ from the parametric solution (2).

3 Parametric PGD-Based Vademecums: Going Beyond the Nowadays Computational Limits

In this section we revisit the computational issues enumerated in the first section from the perspective of parametric PGD based vademecums described in the previous section. For this purpose, we are selecting some scenarios, that prove the potentiality of the proposed approach. We first expose the main ideas through simple academic examples based on the solution of the heat equation, and then we address in the next section more complex situations.

3.1 Material Parameters as Extra-Coordinates

In this section, we illustrate the PGD by considering the following parametric heat transfer equation:

$$\frac{\partial u}{\partial t} - k \cdot \Delta u - f = 0 \quad (8)$$

with homogeneous initial and boundary conditions. The enforcement of non-homogeneous initial and boundary conditions was deeply treated in [34] and [23].

Here $(\mathbf{x}, t, k) \in \Omega \times \mathcal{I}_t \times \mathcal{I}_k$, with $\Omega \subset \mathcal{R}^3$, $\mathcal{I}_t \subset \mathcal{R}$ and $\mathcal{I}_k \subset \mathcal{R}$. For the sake of simplicity and without loss of generality the source term f is assumed constant. The conductivity k is viewed as a new coordinate defined in the interval \mathcal{I}_k . Thus, instead of solving the thermal model for different discrete values of the conductivity parameter, we wish to solve at once a more general problem, the price to pay being an increase of the problem dimensionality. However, as the complexity of the PGD scales only linearly (and not

exponentially) with the space dimension, consideration of the conductivity as a new coordinate still allows one to efficiently obtain an accurate solution.

The weighted residual form related to Eq. (8) reads:

$$\int_{\Omega \times \mathcal{I}_t \times \mathcal{I}_k} u^* \cdot \left(\frac{\partial u}{\partial t} - k \cdot \Delta u - f \right) d\mathbf{x} \cdot dt \cdot dk = 0 \quad (9)$$

for all test functions u^* selected in an appropriate functional space.

The PGD solution is sought in the form:

$$u(\mathbf{x}, t, k) \approx \sum_{i=1}^N X_i(\mathbf{x}) \cdot T_i(t) \cdot K_i(k) \quad (10)$$

At enrichment step n of the PGD algorithm, the following approximation is already known:

$$u^{n-1}(\mathbf{x}, t, k) = \sum_{i=1}^{n-1} X_i(\mathbf{x}) \cdot T_i(t) \cdot K_i(k) \quad (11)$$

We wish to compute the next functional product $X_n(\mathbf{x}) \cdot T_n(t) \cdot K_n(k)$, which we write as $R(\mathbf{x}) \cdot S(t) \cdot W(k)$ for notational simplicity.

Thus, the solution at enrichment step n reads

$$u^n = u^{n-1} + R(\mathbf{x}) \cdot S(t) \cdot W(k) \quad (12)$$

We propose the simplest choice for the test functions u^* used in Eq. (9):

$$u^* = R^*(\mathbf{x}) \cdot S(t) \cdot W(k) + R(\mathbf{x}) \cdot S^*(t) \cdot W(k) + R(\mathbf{x}) \cdot S(t) \cdot W^*(k) \quad (13)$$

With the trial and test functions given by Eqs. (12) and (13) respectively, Eq. (9) is a non-linear problem that must be solved by means of a suitable iterative scheme. In our earlier papers [1] and [3], we used Newton's method. Simpler linearization strategies can also be applied, however. The simplest one is an alternated directions fixed-point algorithm, which was found remarkably robust in the present context. Each iteration consists of three steps that are repeated until reaching convergence, that is, until reaching the fixed point. The first step assumes $S(t)$ and $W(k)$ known from the previous iteration and compute an update for $R(\mathbf{x})$ (in this case the test function reduces to $R^*(\mathbf{x}) \cdot S(t) \cdot W(k)$).

In order to perform this step, the weak form is integrated in $\mathcal{I}_t \times \mathcal{I}_k$ because all the functions involving the time t and the conductivity k are at the present step assumed known. When the integral involves a separated form the integration is very cheap. To understand it we consider the integral of a generic function:

$$\int_{\mathcal{I}_t \times \mathcal{I}_k} F(\mathbf{x}, t, k) dt \cdot dk \quad (14)$$

For computing the integral numerically one should compute an integral in $\mathcal{I}_t \times \mathcal{I}_k$ for each value of \mathbf{x} . Even when considering a discrete number of points \mathbf{x}_k , the integration complexity scales with the number of points \mathbf{x}_k .

On the contrary, when the integral concerns a function that can be expressed in a separated form, the integral reduces to:

$$\begin{aligned} & \int_{\mathcal{I}_t \times \mathcal{I}_k} F(\mathbf{x}, t, k) dt \cdot dk \\ &= \int_{\mathcal{I}_t \times \mathcal{I}_k} \sum_{j=1}^M F_j^x(\mathbf{x}) \cdot F_j^t(t) \cdot F_j^k(k) dt \cdot dk \\ &= \sum_{j=1}^M F_j^x(\mathbf{x}) \cdot \left(\int_{\mathcal{I}_t} F_j^t(t) dt \right) \cdot \left(\int_{\mathcal{I}_k} F_j^k(k) dk \right) \end{aligned} \quad (15)$$

that implies $2 \cdot M$ one-dimensional integrals.

Now, from the just-updated $R(\mathbf{x})$ and the previously-used $W(k)$, we can update $S(t)$ (with $u^* = R(\mathbf{x}) \cdot S^*(t) \cdot W(k)$). Finally, from the just-computed $R(\mathbf{x})$ and $S(t)$, we update $W(k)$ (with $u^* = R(\mathbf{x}) \cdot S(t) \cdot W^*(k)$). Again, the separability of the functions to be integrated becomes a key point from the computational point of view.

This iterative procedure continues until reaching convergence. The converged functions $R(\mathbf{x})$, $S(t)$ and $W(k)$ yield the new functional product at the current enrichment step: $X_n(\mathbf{x}) = R(\mathbf{x})$, $T_n(t) = S(t)$ and $K_n(k) = W(k)$. The explicit form of these operations was described in many of our former works [23]. For the sake of completeness they are given in the annex.

There are other constructors of the separated representation more efficient when applying to non-symmetric differential operators, as the one based on the residual minimization, among many others [24]. The issue related to the optimality of the separated representations obtained by applying standard strategies will be addressed in the next section. The enrichment is stopped as soon as the equation residual or any other error estimator based on a quantity of interest [8, 50] is small enough.

We have seen that at each enrichment step the construction of the new functional product in Eq. (10) requires nonlinear iterations. If m_i denotes the number of iterations needed at enrichment step i , the total number of iterations involved in the construction of the PGD approximation is $m = \sum_{i=1}^N m_i$. In the above example, the entire procedure thus involves the solution of m three-dimensional problems for the functions $X_i(\mathbf{x})$, m one-dimensional problems for the functions $T_i(t)$ and m algebraic systems for the functions $K_i(k)$. We can conclude that the complexity of the PGD procedure to compute the approximation (10) is of some tens of 3D steady-state problems (the cost related to the 1D and algebraic problems being negligible with respect to the

3D problems). In a classical approach, one must solve for each particular value of the parameter k a 3D problem at each time step. In usual applications, this often implies the computation of several millions of 3D solutions. Clearly, the CPU time savings by applying the PGD can be of several orders of magnitude.

3.2 Boundary Conditions as Extra-Coordinates

For the sake of simplicity we first consider the steady state heat equation

$$\nabla \cdot (\mathbf{K} \cdot \nabla u(\mathbf{x})) + f(\mathbf{x}) = 0 \quad (16)$$

with $\mathbf{x} \in \Omega \subset \mathcal{R}^3$, subjected to the boundary conditions:

$$\begin{cases} u(\mathbf{x} \in \Gamma_d) = u_g \\ (-\mathbf{K} \cdot \nabla u)|_{\mathbf{x} \in \Gamma_n} \cdot \mathbf{n} = \mathbf{q}_g \cdot \mathbf{n} = q_g \end{cases} \quad (17)$$

with \mathbf{K} the conductivity tensor and \mathbf{n} the outwards unit vector defined in the domain boundary Γ_n , with $\partial\Omega \equiv \Gamma = \Gamma_d \cup \Gamma_n$ and $\Gamma_d \cap \Gamma_n = \emptyset$.

3.2.1 Neumann Boundary Condition as Extra-Coordinate

First, imagine that we are interested in knowing the model solution for \mathcal{M}_q values of the heat flux $q_g^1, \dots, q_g^{\mathcal{M}_q}$ prescribed on the domain boundary Γ_n , i.e. $u(\mathbf{x}; q_g^i)$, $i = 1, \dots, \mathcal{M}_q$. The space approximation is assumed given by a standard finite element interpolation defined from \mathcal{M}_x nodes. As discussed previously one possibility lies in the solution of the thermal model for the different values of the prescribed flux, from which one could define a metamodel. Thus, we must calculate the solution of \mathcal{M}_q 3D steady state heat problems and then store these solutions that involve $\mathcal{M}_x \cdot \mathcal{M}_q$ data.

Another possibility consists of considering the prescribed heat flux q_g as an extra-coordinate defined in the interval $\mathcal{I}_q = [q_g^-, q_g^+]$ (with eventually $q_g^- = q_g^1$ and $q_g^+ = q_g^{\mathcal{M}_q}$) and then solving only once the resulting 4D heat equation for calculating the general parametric solution $u(\mathbf{x}, q)$. For this purpose the solution is sought in the separated form

$$u(\mathbf{x}, q_g) \approx \sum_{i=1}^N X_i(\mathbf{x}) \cdot Q_i(q_g) \quad (18)$$

In order to enforce the prescribed Dirichlet boundary condition $u(\mathbf{x} \in \Gamma_d) = u_g$ the simplest procedure consists of choosing the first functional couple $X_1(\mathbf{x}) \cdot Q_1(q_g)$ in order to ensure that $u^1(\mathbf{x} \in \Gamma_d, q_g) = X_1(\mathbf{x} \in \Gamma_d) \cdot Q_1(q_g) = u_g$. Thus, the remaining terms of the finite sum $X_i(\mathbf{x})$, $i > 1$, will be subjected to homogeneous essential boundary conditions, i.e. $X_i(\mathbf{x} \in \Gamma_d) = 0$, $i > 1$. Alternative possibilities

for addressing Dirichlet boundary conditions consist of using penalty or Lagrange multiplier formulations. Other possibilities were considered in [34].

In order to use the approximation (18) we start by considering the weak form related to Eq. (16), that writes: Find $u(\mathbf{x}) \in H^1(\Omega)$, verifying $u(\mathbf{x} \in \Gamma_d) = u_g$, such that

$$\begin{aligned} \int_{\Omega} \nabla u^* \cdot (\mathbf{K} \cdot \nabla u) d\mathbf{x} &= \int_{\Gamma_n} u^* \cdot (\mathbf{K} \cdot \nabla u) \cdot \mathbf{n} d\mathbf{x} \\ &+ \int_{\Omega} u^* \cdot f(\mathbf{x}) d\mathbf{x} \end{aligned} \quad (19)$$

is verified $\forall u^* \in H^1(\Omega)$, with $u^*(\mathbf{x} \in \Gamma_d) = 0$.

By introducing the Neumann condition given in (17) into (19) it results

$$\int_{\Omega} \nabla u^* \cdot (\mathbf{K} \cdot \nabla u) d\mathbf{x} = - \int_{\Gamma_n} u^* \cdot q_g d\mathbf{x} + \int_{\Omega} u^* \cdot f(\mathbf{x}) d\mathbf{x} \quad (20)$$

For using the approximation (18) we must consider the extended-weak form defined in the domain $\Omega \times \mathcal{I}_q$

$$\begin{aligned} \int_{\Omega \times \mathcal{I}_q} \nabla u^* \cdot (\mathbf{K} \cdot \nabla u) d\mathbf{x} \cdot dq_g \\ = - \int_{\Gamma_n \times \mathcal{I}_q} u^* \cdot q_g d\mathbf{x} \cdot dq_g \\ + \int_{\Omega \times \mathcal{I}_q} u^* \cdot f(\mathbf{x}) d\mathbf{x} \cdot dq_g \end{aligned} \quad (21)$$

that assuming at iteration n :

$$\begin{cases} u^n(\mathbf{x}, q_g) = \sum_{i=1}^{n-1} X_i(\mathbf{x}) \cdot Q_i(q_g) + X_n(\mathbf{x}) \cdot Q_n(q_g) \\ \quad = u^{n-1}(\mathbf{x}, q_g) + X_n(\mathbf{x}) \cdot Q_n(q_g) \\ u^* = X^*(\mathbf{x}) \cdot Q_n(q_g) + X_n(\mathbf{x}) \cdot Q^*(q_g) \end{cases} \quad (22)$$

with $u^{n-1}(\mathbf{x}, q_g)$ known, we can compute the new couple of unknown functions $X_n(\mathbf{x})$ and $Q_n(q_g)$ by applying the alternated directions fixed point algorithm just summarized and described in the annex for a generic parametric problem, that is assumed to reach the fixed point with a prescribed precision in m_n iterations.

If we assume that the solution needs N terms in the finite sum, the solution will involve $N \cdot (\mathcal{M}_x + \mathcal{M}_q)$ data. If we define $m = \sum_{i=1}^N m_i$ the separated representation solution need the solution of m 3D problems for calculating functions $X_i(\mathbf{x})$, $i = 1, \dots, N$, and m 1D problems for calculating the functions $Q_i(q_g)$, $i = 1, \dots, N$. The computing cost related to the solution of the 1D problems can be

neglected with respect to the one associated with the solution of the 3D problems. Thus, if $m < \mathcal{M}_q$, PGD will proceed faster than the solution of the model for the different values of the parameter q_g^i , $i = 1, \dots, \mathcal{M}_q$. From the point of view of the data storage, PGD is superior as soon as $N \cdot (\mathcal{M}_x + \mathcal{M}_q) < \mathcal{M}_x \cdot \mathcal{M}_q$.

When considering only one parameter as extra-coordinate the superiority of PGD with respect to standard procedures is not crucial, but as discussed previously, when the number of extra-coordinates increases the benefit in using the PGD is impressive.

3.2.2 Dirichlet Boundary Condition as Extra-Coordinate

In this section we consider that we are interested in considering the solution of model (16) for any value of u_g in (17) in a certain interval $\mathcal{I}_u = [u_g^-, u_g^+]$. For this purpose we consider the function $\varphi(\mathbf{x})$ continuous in $\overline{\Omega}$ such that $\Delta\varphi \in L_2(\Omega)$ and $\varphi(\mathbf{x} \in \Gamma_d) = 1$. Thus, we can define the change of variable [34]

$$u(\mathbf{x}) = v(\mathbf{x}) + u_g \cdot \varphi(\mathbf{x}) \quad (23)$$

that allows rewriting Eqs. (16) and (17) as:

$$\nabla \cdot (\mathbf{K} \cdot \nabla v(\mathbf{x})) + u_g \cdot \nabla \cdot (\mathbf{K} \cdot \nabla \varphi(\mathbf{x})) + f(\mathbf{x}) = 0 \quad (24)$$

subjected to the boundary conditions:

$$\begin{cases} v(\mathbf{x} \in \Gamma_d) = 0 \\ (-\mathbf{K} \cdot \nabla v)|_{\mathbf{x} \in \Gamma_n} \cdot \mathbf{n} = u_g \cdot (\mathbf{K} \cdot \nabla \varphi)|_{\mathbf{x} \in \Gamma_n} \cdot \mathbf{n} + q_g \end{cases} \quad (25)$$

that results in the weak form

$$\begin{aligned} \int_{\Omega} \nabla v^* \cdot (\mathbf{K} \cdot \nabla v) d\mathbf{x} \\ = - \int_{\Omega} \nabla v^* \cdot u_g \cdot (\mathbf{K} \cdot \nabla \varphi) d\mathbf{x} + \int_{\Omega} v^* \cdot f(\mathbf{x}) d\mathbf{x} \\ - \int_{\Gamma_n} v^* \cdot q_g d\mathbf{x} - \int_{\Gamma_n} v^* \cdot u_g \cdot (\mathbf{K} \cdot \nabla \varphi) \cdot \mathbf{n} d\mathbf{x} \end{aligned} \quad (26)$$

that in fact only requires \mathcal{C}^0 continuity of the function $\varphi(\mathbf{x})$.

We can now introduce u_g as extra-coordinate, searching the solution in the separated form:

$$v(\mathbf{x}, u_g) \approx \sum_{i=1}^N X_i(\mathbf{x}) \cdot U_i(u_g) \quad (27)$$

that needs for the extended weak-form

$$\begin{aligned} \int_{\Omega \times \mathcal{I}_u} \nabla v^* \cdot (\mathbf{K} \cdot \nabla v) d\mathbf{x} \cdot du_g \\ = - \int_{\Omega \times \mathcal{I}_u} \nabla v^* \cdot u_g \cdot (\mathbf{K} \cdot \nabla \varphi) d\mathbf{x} \cdot du_g \end{aligned}$$

$$\begin{aligned}
& + \int_{\Omega \times \mathcal{I}_u} v^* \cdot f(\mathbf{x}) d\mathbf{x} \cdot du_g \\
& - \int_{\Gamma_n \times \mathcal{I}_u} v^* \cdot q_g d\mathbf{x} \cdot du_g \\
& - \int_{\Gamma_n \times \mathcal{I}_u} v^* \cdot u_g \cdot (\mathbf{K} \cdot \nabla \varphi) \cdot \mathbf{n} d\mathbf{x} \cdot du_g \quad (28)
\end{aligned}$$

on which the alternated directions fixed point algorithm applies again to calculate the parametric solution (27).

3.2.3 Mixed Boundary Conditions as Extra-Coordinates

From the extended weak form

$$\begin{aligned}
& \int_{\Omega \times \mathcal{I}_u \times \mathcal{I}_q} \nabla v^* \cdot (\mathbf{K} \cdot \nabla v) d\mathbf{x} \cdot du_g \cdot dq_g \\
& = - \int_{\Omega \times \mathcal{I}_u \times \mathcal{I}_q} \nabla v^* \cdot u_g \cdot (\mathbf{K} \cdot \nabla \varphi) d\mathbf{x} \cdot du_g \cdot dq_g \\
& + \int_{\Omega \times \mathcal{I}_u \times \mathcal{I}_q} v^* \cdot f(\mathbf{x}) d\mathbf{x} \cdot du_g \cdot dq_g \\
& - \int_{\Gamma_n \times \mathcal{I}_u \times \mathcal{I}_q} v^* \cdot q_g d\mathbf{x} \cdot du_g \cdot dq_g \\
& - \int_{\Gamma_n \times \mathcal{I}_u \times \mathcal{I}_q} v^* \cdot u_g \cdot (\mathbf{K} \cdot \nabla \varphi) \cdot \mathbf{n} d\mathbf{x} \cdot du_g \cdot dq_g \quad (29)
\end{aligned}$$

one could compute a parametric solution involving Dirichlet and Neumann boundary conditions as extra-coordinates, i.e. $v(\mathbf{x}, u_g, q_g)$ according to:

$$v(\mathbf{x}, u_g, q_g) \approx \sum_{i=1}^N X_i(\mathbf{x}) \cdot U_i(u_g) \cdot Q_i(q_g) \quad (30)$$

3.2.4 Non Constant Neumann Boundary Conditions

We consider that in Eq. (17) $q_g = q_g(\mathbf{x})$, with $\mathbf{x} \in \Gamma_n$. First we assume that $q_g(\mathbf{x})$ can be approximated on Γ_n from:

$$q_g(\mathbf{x}) \approx \sum_{k=1}^{S_q} Q_g^k \cdot \xi_k(\mathbf{x}) \quad (31)$$

where $\mathbf{x} \in \Gamma_n$ and Q_g^k represents the prescribed nodal fluxes, i.e. $Q_g^k = g_g(\mathbf{x}_k)$ at the nodal positions $\mathbf{x}_k \in \Gamma_n$.

Now, introducing the approximation (31) into the weak form (20) it results

$$\begin{aligned}
\int_{\Omega} \nabla u^* \cdot (\mathbf{K} \cdot \nabla u) d\mathbf{x} & = - \int_{\Gamma_n} u^* \cdot \left(\sum_{k=1}^{S_q} Q_g^k \cdot \xi_k(\mathbf{x}) \right) d\mathbf{x} \\
& + \int_{\Omega} u^* \cdot f(\mathbf{x}) d\mathbf{x} \quad (32)
\end{aligned}$$

If the nodal fluxes Q_g^k , $k = 1, \dots, S_q$, can take values into the intervals \mathcal{I}_Q^k and we are interested to calculating the general parametric solution $u(\mathbf{x}, Q_g^1, \dots, Q_g^{S_q})$, it suffices to consider the extended weak form related to Eq. (32)

$$\begin{aligned}
& \int_{\Omega \times \mathcal{I}_Q^1 \times \dots \times \mathcal{I}_Q^{S_q}} \nabla u^* \cdot (\mathbf{K} \cdot \nabla u) d\mathbf{x} \cdot dQ_g^1 \dots dQ_g^{S_q} \\
& = - \int_{\Gamma_n \times \mathcal{I}_Q^1 \times \dots \times \mathcal{I}_Q^{S_q}} u^* \\
& \quad \cdot \left(\sum_{k=1}^{S_q} Q_g^k \cdot \xi_k(\mathbf{x}) \right) d\mathbf{x} \cdot dQ_g^1 \dots dQ_g^{S_q} \\
& + \int_{\Omega \times \mathcal{I}_Q^1 \times \dots \times \mathcal{I}_Q^{S_q}} u^* \cdot f(\mathbf{x}) d\mathbf{x} \cdot dQ_g^1 \dots dQ_g^{S_q} \quad (33)
\end{aligned}$$

and the solution separated representation

$$u(\mathbf{x}, Q_g^1, \dots, Q_g^{S_q}) \approx \sum_{i=1}^N X_i(\mathbf{x}) \cdot \prod_{j=1}^{S_q} G_i^j(Q_g^j) \quad (34)$$

Then, from Eqs. (33) and (34) we can compute the parametric solution. In this case the use of the separated representation is compulsory because the curse of dimensionality that model (33) implies when S_q increases. Standard discretization strategies fail for solving the multi-dimensional model (33) and the sampling of the parametric space becomes inefficient when its dimensionality increases.

3.2.5 Non Constant Dirichlet Boundary Conditions

In this section we consider the solution of model (16) for u_g in Eq. (17) depending on the space, i.e. $u_g(\mathbf{x})$, $\mathbf{x} \in \Gamma_d$. If $u_g(\mathbf{x})$ can be approximated on Γ_d from

$$u_g(\mathbf{x}) \approx \sum_{k=1}^{S_u} U_g^k \cdot \eta_k(\mathbf{x}) \quad (35)$$

where U_g^k represents the prescribed nodal temperatures, i.e. $U_g^k = u_g(\mathbf{x}_k)$ at the nodal positions $\mathbf{x}_k \in \Gamma_d$ taking values in the intervals \mathcal{I}_U^k .

To compute the parametric solution $u(\mathbf{x}, U_g^1, \dots, U_g^{S_u})$ we first define the functions $\varphi_k(\mathbf{x})$ continuous in $\overline{\Omega}$ and verifying $\Delta \varphi_k(\mathbf{x}) \in L_2(\Omega)$, such that $\varphi_k(\mathbf{x} \in \Gamma_d) = \eta_k(\mathbf{x})$. Thus, we can define the change of variable

$$u(\mathbf{x}) = v(\mathbf{x}) + \sum_{i=1}^{S_u} U_g^k \cdot \varphi_k(\mathbf{x}) \quad (36)$$

that leads to the weak form

$$\begin{aligned}
& \int_{\Omega} \nabla v^* \cdot (\mathbf{K} \cdot \nabla v) d\mathbf{x} \\
&= - \int_{\Omega} \nabla v^* \cdot \left(\sum_{k=1}^{S_u} U_g^k \cdot (\mathbf{K} \cdot \nabla \varphi_k) \right) d\mathbf{x} \\
&+ \int_{\Omega} v^* \cdot f(\mathbf{x}) d\mathbf{x} \\
&- \int_{\Gamma_n} v^* \cdot q_g d\mathbf{x} \\
&- \int_{\Gamma_n} v^* \cdot \left(\sum_{k=1}^{S_u} U_g^k \cdot (\mathbf{K} \cdot \nabla \varphi_k) \cdot \mathbf{n} \right) d\mathbf{x} \quad (37)
\end{aligned}$$

We can consider now the separated form of the parametric solution

$$u(\mathbf{x}, U_g^1, \dots, U_g^{S_u}) \approx \sum_{i=1}^N X_i(\mathbf{x}) \cdot \prod_{j=1}^{S_u} F_i^j(U_g^j) \quad (38)$$

that will be constructed using some standard procedure (alternated directions fixed point or residual minimization) from the extended weak form:

$$\begin{aligned}
& \int_{\Omega \times \mathcal{I}_U^1 \times \dots \times \mathcal{I}_U^{S_u}} \nabla v^* \cdot (\mathbf{K} \cdot \nabla v) d\mathbf{x} \cdot dU_g^1 \dots dU_g^{S_u} \\
&= - \int_{\Omega \times \mathcal{I}_U^1 \times \dots \times \mathcal{I}_U^{S_u}} \nabla v^* \\
&\cdot \left(\sum_{k=1}^{S_u} U_g^k \cdot (\mathbf{K} \cdot \nabla \varphi_k) \right) d\mathbf{x} \cdot dU_g^1 \dots dU_g^{S_u} \\
&+ \int_{\Omega \times \mathcal{I}_U^1 \times \dots \times \mathcal{I}_U^{S_u}} v^* \cdot f(\mathbf{x}) d\mathbf{x} \cdot dU_g^1 \dots dU_g^{S_u} \\
&- \int_{\Gamma_n \times \mathcal{I}_U^1 \times \dots \times \mathcal{I}_U^{S_u}} v^* \cdot q_g d\mathbf{x} \cdot dU_g^1 \dots dU_g^{S_u} \\
&- \int_{\Gamma_n \times \mathcal{I}_U^1 \times \dots \times \mathcal{I}_U^{S_u}} v^* \\
&\cdot \left(\sum_{k=1}^{S_u} U_g^k \cdot (\mathbf{K} \cdot \nabla \varphi_k) \cdot \mathbf{n} \right) d\mathbf{x} \cdot dU_g^1 \dots dU_g^{S_u} \quad (39)
\end{aligned}$$

3.3 Initial Conditions as Extra-Coordinates

We consider in this section the transient heat equation in a homogeneous and isotropic medium

$$\rho \cdot C_p \cdot \frac{\partial u}{\partial t} = k \cdot \Delta u + f \quad (40)$$

$t \in \mathcal{I}_t = (0, \Theta] \subset \mathcal{R}$, $\mathbf{x} \in \Omega \subset \mathcal{R}^3$ and $f = cte$. The initial and boundary conditions read:

$$\begin{cases} u(\mathbf{x} \in \Gamma_d) = u_g \\ (-k \cdot \nabla u)|_{\mathbf{x} \in \Gamma_n} \cdot \mathbf{n} = q_g \\ u(\mathbf{x}, t = 0) = u^0(\mathbf{x}) \end{cases} \quad (41)$$

The associated weak form reads:

$$\begin{aligned}
& \int_{\Omega} u^* \cdot \rho \cdot C_p \cdot \frac{\partial u}{\partial t} d\mathbf{x} + \int_{\Omega} \nabla u^* \cdot k \cdot \nabla u d\mathbf{x} \\
&= - \int_{\Gamma_n} u^* \cdot q_g d\mathbf{x} + \int_{\Omega} u^* \cdot f(\mathbf{x}) d\mathbf{x} \quad (42)
\end{aligned}$$

that includes explicitly the natural (Neumann) boundary conditions. To prescribe both the initial and the essential (Dirichlet) boundary conditions we proceed to define the following functions:

$$\hat{u}^0(\mathbf{x}) = \begin{cases} u^0(\mathbf{x}), & \mathbf{x} \in \Omega \\ 0, & \mathbf{x} \in \Gamma \end{cases} \quad (43)$$

$$\Upsilon(t) = \begin{cases} 1, & t > 0 \\ 0, & t = 0 \end{cases} \quad (44)$$

and $\varphi(\mathbf{x})$ continuous in $\overline{\Omega}$, verifying $\Delta \varphi \in L_2(\Omega)$ and the essential boundary conditions

$$\varphi(\mathbf{x} \in \Gamma_d) = u_g \quad (45)$$

We could define the function $\Sigma(\mathbf{x}, t)$ expressed in the separated form

$$\Sigma(\mathbf{x}, t) = \hat{u}^0(\mathbf{x}) + \varphi(\mathbf{x}) \cdot \Upsilon(t) \quad (46)$$

that verifies the initial and essential boundary conditions. However, functions \hat{u}^0 and $\Upsilon(t)$ are not regular enough to be employed in the weak form of the problem. A direct regularization consists in defining these functions at the nodal positions and then define interpolations with the required regularity. Thus, the discrete counterpart of functions \hat{u}^0 and $\Upsilon(t)$ are given by:

$$\hat{u}^0(\mathbf{x}_k) = \begin{cases} u^0(\mathbf{x}_k), & \mathbf{x}_k \in \Omega \\ 0, & \mathbf{x}_k \in \Gamma \end{cases} \quad (47)$$

and

$$\Upsilon(t_l) = \begin{cases} 1, & t_l > 0 \\ 0, & t_l = 0 \end{cases} \quad (48)$$

with $k = 1, \dots, 1, \dots, \mathcal{M}_x$; $l = 1, \dots, \mathcal{M}_t$. Now, standard interpolation is applied to the define functions $\hat{u}^0(\mathbf{x})$ and $\Upsilon(t)$ everywhere from their nodal values expressed by Eqs. (47) and (48).

By applying now the change of variable:

$$u(\mathbf{x}, t) = v(\mathbf{x}, t) + \Sigma(\mathbf{x}, t) = v(\mathbf{x}, t) + \hat{u}^0(\mathbf{x}) + \varphi(\mathbf{x}) \cdot \Upsilon(t) \quad (49)$$

the weak form (57) results:

$$\begin{aligned} & \int_{\Omega} v^* \cdot \rho \cdot C_p \cdot \frac{\partial v}{\partial t} d\mathbf{x} + \int_{\Omega} \nabla v^* \cdot k \cdot \nabla v d\mathbf{x} \\ &= - \int_{\Omega} v^* \cdot \rho \cdot C_p \cdot \varphi \cdot \frac{\partial \Upsilon}{\partial t} d\mathbf{x} \\ & \quad - \int_{\Omega} \nabla v^* \cdot k \cdot \nabla \hat{u}^0 d\mathbf{x} \\ & \quad - \int_{\Gamma_n} v^* \cdot k \cdot \nabla \hat{u}^0 \cdot \mathbf{n} d\mathbf{x} \\ & \quad - \int_{\Gamma_n} v^* \cdot q_g d\mathbf{x} - \int_{\Omega} \nabla v^* \cdot k \cdot \Upsilon \cdot \nabla \varphi d\mathbf{x} \\ & \quad - \int_{\Gamma_n} v^* \cdot k \cdot \Upsilon \cdot \nabla \varphi \cdot \mathbf{n} d\mathbf{x} + \int_{\Omega} v^* \cdot f(\mathbf{x}) d\mathbf{x} \end{aligned} \quad (50)$$

If the initial condition is approximated by

$$\hat{u}^0(\mathbf{x}) \approx \sum_{k=1}^{S_0} U_0^k \cdot \eta_k(\mathbf{x}) \quad (51)$$

after introducing it into the weak form (50) we obtain:

$$\begin{aligned} & \int_{\Omega} v^* \cdot \rho \cdot C_p \cdot \frac{\partial v}{\partial t} d\mathbf{x} + \int_{\Omega} \nabla v^* \cdot k \cdot \nabla v d\mathbf{x} \\ &= - \int_{\Omega} v^* \cdot \rho \cdot C_p \cdot \varphi \cdot \frac{\partial \Upsilon}{\partial t} d\mathbf{x} \\ & \quad - \int_{\Omega} \nabla v^* \cdot k \cdot \left(\sum_{k=1}^{S_0} U_0^k \cdot \nabla \eta_k(\mathbf{x}) \right) d\mathbf{x} \\ & \quad - \int_{\Gamma_n} v^* \cdot k \cdot \left(\sum_{k=1}^{S_0} U_0^k \cdot \eta_k(\mathbf{x}) \cdot \mathbf{n} \right) d\mathbf{x} \\ & \quad - \int_{\Gamma_n} v^* \cdot q_g d\mathbf{x} - \int_{\Omega} \nabla v^* \cdot k \cdot \Upsilon \cdot \nabla \varphi d\mathbf{x} \\ & \quad - \int_{\Gamma_n} v^* \cdot k \cdot \Upsilon \cdot \nabla \varphi \cdot \mathbf{n} d\mathbf{x} + \int_{\Omega} v^* \cdot f(\mathbf{x}) d\mathbf{x} \end{aligned} \quad (52)$$

that allows us looking for a parametric solution $v(\mathbf{x}, U_0^1, \dots, U_0^{S_0})$, with $U_0^j \in \mathcal{I}_0^j = [(U_0^j)^-, (U_0^j)^+]$

$$u(\mathbf{x}, U_0^1, \dots, U_0^{S_0}) \approx \sum_{i=1}^N X_i(\mathbf{x}) \cdot \prod_{j=1}^{S_0} \mathcal{U}_i^j(U_0^j) \quad (53)$$

from the extended weak form

$$\begin{aligned} & \int_{\Omega \times \mathcal{I}_0^1 \times \dots \times \mathcal{I}_0^{S_0}} v^* \cdot \rho \cdot C_p \cdot \frac{\partial v}{\partial t} d\mathbf{x} \cdot dU_0^1 \dots dU_0^{S_0} \\ & \quad + \int_{\Omega \times \mathcal{I}_0^1 \times \dots \times \mathcal{I}_0^{S_0}} \nabla v^* \cdot k \cdot \nabla v d\mathbf{x} \cdot dU_0^1 \dots dU_0^{S_0} \\ &= - \int_{\Omega \times \mathcal{I}_0^1 \times \dots \times \mathcal{I}_0^{S_0}} v^* \cdot \rho \cdot C_p \cdot \varphi \cdot \frac{\partial \Upsilon}{\partial t} d\mathbf{x} \cdot dU_0^1 \dots dU_0^{S_0} \\ & \quad - \int_{\Omega \times \mathcal{I}_0^1 \times \dots \times \mathcal{I}_0^{S_0}} \nabla v^* \cdot k \\ & \quad \cdot \left(\sum_{k=1}^{S_0} U_0^k \cdot \nabla \eta_k(\mathbf{x}) \right) d\mathbf{x} \cdot dU_0^1 \dots dU_0^{S_0} \\ & \quad - \int_{\Gamma_n \times \mathcal{I}_0^1 \times \dots \times \mathcal{I}_0^{S_0}} v^* \cdot k \\ & \quad \cdot \left(\sum_{k=1}^{S_0} U_0^k \cdot \eta_k(\mathbf{x}) \cdot \mathbf{n} \right) d\mathbf{x} \cdot dU_0^1 \dots dU_0^{S_0} \\ & \quad - \int_{\Gamma_n \times \mathcal{I}_0^1 \times \dots \times \mathcal{I}_0^{S_0}} v^* \cdot q_g d\mathbf{x} \cdot dU_0^1 \dots dU_0^{S_0} \\ & \quad - \int_{\Omega \times \mathcal{I}_0^1 \times \dots \times \mathcal{I}_0^{S_0}} \nabla v^* \cdot k \cdot \Upsilon \cdot \nabla \varphi d\mathbf{x} \cdot dU_0^1 \dots dU_0^{S_0} \\ & \quad - \int_{\Gamma_n \times \mathcal{I}_0^1 \times \dots \times \mathcal{I}_0^{S_0}} v^* \cdot k \cdot \Upsilon \cdot \nabla \varphi \cdot \mathbf{n} d\mathbf{x} \cdot dU_0^1 \dots dU_0^{S_0} \\ & \quad + \int_{\Omega \times \mathcal{I}_0^1 \times \dots \times \mathcal{I}_0^{S_0}} v^* \cdot f(\mathbf{x}) d\mathbf{x} \cdot dU_0^1 \dots dU_0^{S_0} \end{aligned} \quad (54)$$

3.4 Geometrical Parameters as Extra-Coordinates

For the sake of clarity and without loss of generality we are addressing in this section the transient one-dimensional heat equation

$$\frac{\partial u}{\partial t} = \alpha \cdot \frac{\partial^2 u}{\partial x^2} + f \quad (55)$$

with $t \in \mathcal{I}_t = (0, \Theta] \subset \mathcal{R}$, $x \in \Omega = (0, L) \subset \mathcal{R}$, $f = cte$ and $u(x=0, t) = u(x=L, t) = u(x, t=0) = 0$.

The associated space-time weak form reads:

$$\begin{aligned} & \int_{\Omega \times \mathcal{I}_t} u^* \cdot \frac{\partial u}{\partial t} dx \cdot dt = -\alpha \cdot \int_{\Omega \times \mathcal{I}_t} \frac{\partial u^*}{\partial x} \cdot \frac{\partial u}{\partial x} dx \cdot dt \\ & \quad + \int_{\Omega \times \mathcal{I}_t} u^* \cdot f dx \cdot dt \end{aligned} \quad (56)$$

If we are interested in computing the solution $u(x, t)$ in many domains of length $L \in [L^-, L^+]$ and for many time intervals of length $\Theta = [\Theta^-, \Theta^+]$, more than solving the

model for many possible choices in order to define a meta-model, it is preferable to compute the parametric solution $u(x, t, L, \Theta)$.

This parametric solution is sought in the separated form

$$u(x, t, L, \Theta) \approx \sum_{i=1}^N X_i(x) \cdot T_i(t) \cdot \mathcal{L}_i(L) \cdot \mathcal{T}_i(\Theta) \quad (57)$$

However, Eq. (56) does not involve an explicit dependence on the extra-coordinates L and Θ , both defining the domain of integration. In order to explicit this dependence, we consider the coordinates transformation

$$\begin{cases} t = \tau \cdot \Theta, & \tau \in [0, 1] \\ x = \lambda \cdot L, & \lambda \in [0, 1] \end{cases} \quad (58)$$

In this case the weak form (56) reads:

$$\begin{aligned} & \int_{[0,1]^2} u^* \cdot \frac{\partial u}{\partial \tau} \cdot L \, d\lambda \cdot d\tau \\ &= -\alpha \cdot \int_{[0,1]^2} \frac{\partial u^*}{\partial \lambda} \cdot \frac{\partial u}{\partial \lambda} \cdot \frac{\Theta}{L} \, d\lambda \cdot d\tau \\ &+ \int_{[0,1]^2} u^* \cdot f \cdot L \cdot \Theta \, d\lambda \cdot d\tau \end{aligned} \quad (59)$$

that allows calculating the parametric solution derived from (57) after applying the change of coordinates

$$u(\lambda, \tau, L, \Theta) \approx \sum_{i=1}^N \tilde{X}_i(\lambda) \cdot \tilde{T}_i(\tau) \cdot \mathcal{L}_i(L) \cdot \mathcal{T}_i(\Theta) \quad (60)$$

4 Accounting for Non-linearities

The treatment of non-linear models, compulsory for obtaining the handbook related to parametric non-linear models usually encountered in industrial applications, was addressed in some of our former works [6, 70] where standard linearization procedures were extended to the PGD framework. We start here by considering such procedures in order to point out their inherent limitations before considering alternative approaches.

We consider the same model that we considered in the just referred woks:

$$\begin{cases} \frac{\partial u}{\partial t} - k \cdot \Delta u = -u^2 + f(\mathbf{x}, t) & \text{in } \Omega \times (0, T_{max}) \\ u(\mathbf{x}, t) = 0 & \text{on } \partial\Omega \times (0, T_{max}) \\ u(\mathbf{x}, 0) = 0 & \text{in } \Omega \end{cases} \quad (61)$$

where $\Omega \subset \mathbb{R}^d$, $d \geq 1$, $T_{max} > 0$ and $k > 0$ is the diffusion coefficient. To build-up the approximated solution of (61)

by using a separated representation, we considered two approaches: (i) an incremental linearization and (ii) a Newton linearization, both described below.

We write the solution of problem (61) in the separated form

$$u(\mathbf{x}, t) \approx \sum_{i=1}^N X_i(\mathbf{x}) \cdot T_i(t)$$

We suppose that at iteration n , with $n < N$, the n first modes (X_i, T_i) , $i = 1, \dots, n$, are already known and that at present iteration we search the new enrichment functional product $R(\mathbf{x}) \cdot S(t)$ such that the updated approximation writes

$$\begin{aligned} u^{n+1}(\mathbf{x}, t) &= u^n(\mathbf{x}, t) + R(\mathbf{x}) \cdot S(t) \\ &= \sum_{i=1}^n X_i(\mathbf{x}) \cdot T_i(t) + R(\mathbf{x}) \cdot S(t) \end{aligned} \quad (62)$$

The alternating directions scheme proceed by calculating $R(\mathbf{x})$ from the temporal function $S(t)$ just computed, and then, updating $R(\mathbf{x})$ from the just computed $S(t)$ as we described in the previous sections. The iteration procedure should continue until reaching convergence.

– Incremental linearization

When using the simplest incremental linearization the non-linear term u^2 is calculated from the solution at the previous enrichment iteration, that is, from u^n

$$u^2 \approx \left(\sum_{i=1}^n X_i(\mathbf{x}) \cdot T_i(t) \right)^2 \quad (63)$$

– Newton linearization

From the solution at iteration n , u^n

$$u^n(\mathbf{x}, t) = \sum_{i=1}^n X_i(\mathbf{x}) \cdot T_i(t) \quad (64)$$

the solution at the next iteration can be written as $u^{n+1} = u^n + \tilde{u}$ where \tilde{u} is the solution of the linearized problem

$$\frac{\partial \tilde{u}}{\partial t} - k \cdot \Delta \tilde{u} + 2u^n \tilde{u} = -\mathcal{R}(u^n) \quad (65)$$

where the residual $\mathcal{R}(u^n)$ reads

$$\mathcal{R}(u^n) \equiv \left(\frac{\partial u^n}{\partial t} - k \cdot \Delta u^n + (u^n)^2 - f(\mathbf{x}, t) \right) \quad (66)$$

4.1 Discussion

Both procedures converge but no significant differences in the number of required iterations were noticed. The convergence rate and the computing time were similar. As dis-

cussed in [55] even when the exact solution can be represented by a single functional product, i.e.

$$u^{ex}(\mathbf{x}, t) = X^{ex}(\mathbf{x}) \cdot T^{ex}(t) \quad (67)$$

the non linear solver produces a solution composed of many sums

$$u(\mathbf{x}, t) \approx \sum_{i=1}^N X_i(\mathbf{x}) \cdot T_i(t) \quad (68)$$

with $N > 1$. The main reason is that the number of sums is in this case subsidiary of the convergence rate of the non-linear solver.

In [70] we analyzed other linearization schemes. When we considered the improved fixed point, in which the non-linear term is approximated at iteration q of the enrichment step n , according to:

$$u^2 \approx (u^n + R^{(q-1)}(\mathbf{x}) \cdot S^{(q-1)}(t))^2 \quad (69)$$

then we proved, in the case described above whose exact solution consists of a single product, that the solver converges after computing the first functional couple. In that sense the solver is optimal but the computing time is similar to the one required by using the standard fixed point or the Newton strategy previously described.

The main difficulty related to the use of standard linearizations lies in the necessity of evaluating the non-linear term. Because the necessity of arriving to a separated representation of such term, one possibility consists in performing a polynomial expansion and then compute the different powers of the solution at the previous step. However, it is easy to understand that the power p of u^n involves too many terms when p , n or the number of the coordinates d involved in the model increase as can be noticed from:

$$(u^n(x_1, \dots, x_d))^p = \left(\sum_{i=1}^n F_i^1(x_1) \times \dots \times F_i^d(x_d) \right)^p \quad (70)$$

The issue related to the high powers could be alleviated by introducing new variables v^i such that $v^{(2)} = u \cdot u$, $v^{(3)} = u^3 = v^{(2)} \cdot u$, ... that allows never compute more than powers of degree 2.

However, in general the differential operators related to the searched functional product involve the more and more terms. Thus, when d , p or n increases the solution procedure described in the previous sections become rapidly inefficient.

In [55] we explored the asymptotic numerical method that guarantees a constant number of terms to represent the differential operator. It proceeds when addressing model (61)

by introducing the loading parameter λ affecting the non-linear term

$$\frac{\partial u}{\partial t} - k \cdot \Delta u = -\lambda \cdot u^2 + f(\mathbf{x}, t) \quad (71)$$

We denote by u_0 the solution related to $\lambda = \lambda_0 = 0$ that can be computed easily because it corresponds to the solution of the linear problem. The searched solution is the one related to $\lambda = 1$. Now, we define an asymptotic expansion of the unknown field u as well as of the loading parameter λ by considering powers of the expansion parameter a :

$$\begin{cases} u = u_0 + a \cdot u_1 + a^2 \cdot u_2 + \dots \\ \lambda = \lambda_0 + a \cdot \lambda_1 + a^2 \cdot \lambda_2 + \dots \end{cases} \quad (72)$$

The non linear term can be written as:

$$u^2 = (u^2)_0 + a \cdot (u^2)_1 + a^2 \cdot (u^2)_2 + \dots + a^p \cdot (u^2)_p + \dots \quad (73)$$

where $(u^2)_p$ reads:

$$(u^2)_p = \sum_{i=0}^p u_i \cdot u_{p-i} = 2 \cdot u_0 \cdot u_p + \sum_{i=1}^{p-1} u_i \cdot u_{p-i} \quad (74)$$

Introducing (72) and (74) into (61) and identifying the different powers of a , it results a sequence of problems, one at each order (power of a), all them having the same differential operator, and whose right hand members depend on the solutions computed at lower orders (see [55] for additional details). Again, for higher powers, the introduction of new variables avoids the computation of powers higher than 2. Despite all these subtle advantages, in the general multi-parametric case the computational complexity increases too much, and moreover, these methods need to be coupled with a continuation strategy when going beyond the convergence disc of the expansion.

A promising alternative consists of interpolating the non-linear term as proposed in [12], whose discrete counterpart, the Discrete Empirical Interpolation Method—DEIM—was given in [22], and then introducing such an interpolation within the separated representation PGD constructor.

4.2 DEIM Based PGD for the Efficient Solution of Non-linear Models

We consider the solution of

$$\frac{\partial u}{\partial t} - k \cdot \Delta u = \mathcal{L}(u) + f(\mathbf{x}, t) \quad (75)$$

with homogeneous initial and boundary conditions and where $\mathcal{L}(u)$ represents a nonlinear function of u .

We first address the solution $u^0(\mathbf{x}, t)$ of its linear counterpart:

$$\frac{\partial u^0}{\partial t} - k \cdot \Delta u^0 = f(\mathbf{x}, t) \quad (76)$$

whose solution is found in a separated form by applying the PGD. Thus, the solution of Eq. (76) writes:

$$u^0(\mathbf{x}, t) \approx \sum_{i=1}^{N^0} X_i^0(\mathbf{x}) \cdot T_i^0(t) \quad (77)$$

that allows to define the reduced approximation basis $\mathcal{B}^0 = \{\tilde{X}_1^0 \cdot \tilde{T}_1^0, \dots, \tilde{X}_{N^0}^0 \cdot \tilde{T}_{N^0}^0\}$ that contains the normalized functions: $\tilde{X}_i^0 = \frac{X_i^0}{\|X_i^0\|}$ and $\tilde{T}_i^0 = \frac{T_i^0}{\|T_i^0\|}$.

Now, we could define an interpolation of the nonlinear function $\mathcal{L}(u)$ by using the basis \mathcal{B}^0 . For this purpose we consider N^0 points (\mathbf{x}_j^0, t_j^0) , $j = 1, \dots, N^0$, and we enforce:

$$\mathcal{L}(u^0(\mathbf{x}_j^0, t_j^0)) = \sum_{i=1}^{N^0} \xi_i^0 \cdot \tilde{X}_i^0(\mathbf{x}_j^0) \cdot \tilde{T}_i^0(t_j^0), \quad j = 1, \dots, N^0 \quad (78)$$

that represents a linear system of size N^0 whose solution allows calculating the coefficients ξ_i^0 .

As soon as these coefficients ξ_i^0 are known, the interpolation of the nonlinear term is fully defined:

$$\mathcal{L}^0 \equiv \mathcal{L}(u^0(\mathbf{x}, t)) \approx \sum_{i=1}^{N^0} \xi_i^0 \cdot \tilde{X}_i^0(\mathbf{x}) \cdot \tilde{T}_i^0(t) \quad (79)$$

that is introduced into the original nonlinear problem leading to the linear problem involving now $u^1(\mathbf{x}, t)$

$$\frac{\partial u^1}{\partial t} - k \cdot \Delta u^1 = \mathcal{L}^0(u) + f(\mathbf{x}, t) \quad (80)$$

Now for calculating the solution $u^1(\mathbf{x}, t)$ many choices exist, being the most direct ones, among many others:

– Restart the separated representation, i.e.:

$$u^1(\mathbf{x}, t) \approx \sum_{i=1}^{N_s^1} X_i^1(\mathbf{x}) \cdot T_i^1(t) \quad (81)$$

– Reuse the solution u^0 :

$$u^1(\mathbf{x}, t) \approx u^0(\mathbf{x}, t) + \sum_{i=1}^{N_r} X_i^1(\mathbf{x}) \cdot T_i^1(t) = \sum_{i=1}^{N_r^1} X_i^1(\mathbf{x}) \cdot T_i^1(t) \quad (82)$$

– Reuse by projecting. In this case first we consider

$$u^{1,0}(\mathbf{x}, t) \approx \sum_{i=1}^{N^0} \eta_i \cdot \tilde{X}_i^0(\mathbf{x}) \cdot \tilde{T}_i^0(t) \quad (83)$$

that introduced into (80) allows computing the coefficients η_i . Then the approximation is enriched by considering

$$\begin{aligned} u^1(\mathbf{x}, t) &\approx \sum_{i=1}^{N^0} \eta_i \cdot \tilde{X}_i^0(\mathbf{x}) \cdot \tilde{T}_i^0(t) + \sum_{i=1}^{N_p} X_i^1(\mathbf{x}) \cdot T_i^1(t) \\ &= \sum_{i=1}^{N_p^1} X_i^1(\mathbf{x}) \cdot T_i^1(t) \end{aligned} \quad (84)$$

We define both, N^1 as the number of final approximation functions, N_s^1 , N_r^1 or N_p^1 depending on the previous choice, and the associated reduced approximation basis $\mathcal{B}^1 = \{\tilde{X}_1^1 \cdot \tilde{T}_1^1, \dots, \tilde{X}_{N^1}^1 \cdot \tilde{T}_{N^1}^1\}$. Now the nonlinear term is interpolated again from N^1 points (\mathbf{x}_j^1, t_j^1) , $j = 1, \dots, N^1$:

$$\mathcal{L}^1 \equiv \mathcal{L}(u^1(\mathbf{x}, t)) \approx \sum_{i=1}^{N^1} \xi_i^1 \cdot \tilde{X}_i^1(\mathbf{x}) \cdot \tilde{T}_i^1(t) \quad (85)$$

that is introduced into the original nonlinear problem leading to the linear problem involving now $u^2(\mathbf{x}, t)$. The just described procedure is repeated until reaching convergence.

The only point that deserves additional comments is the one related to the choice of the interpolation points (\mathbf{x}_j^k, t_j^k) , $j = 1, \dots, N^k$ at iteration k . At this iteration the reduced approximation basis reads:

$$\mathcal{B}^k = \{\tilde{X}_1^k \cdot \tilde{T}_1^k, \dots, \tilde{X}_{N^k}^k \cdot \tilde{T}_{N^k}^k\} \quad (86)$$

Following [12] and [22] we consider

$$(\mathbf{x}_1^k, t_1^k) = \underset{\mathbf{x}, t}{\operatorname{argmax}} |\tilde{X}_1^k(\mathbf{x}) \cdot \tilde{T}_1^k(t)| \quad (87)$$

then we compute d_1 from

$$d_1 \cdot \tilde{X}_1^k(\mathbf{x}_1^k) \cdot \tilde{T}_1^k(t_1^k) = \tilde{X}_2^k(\mathbf{x}_1^k) \cdot \tilde{T}_2^k(t_1^k) \quad (88)$$

that allows defining $r_2^k(\mathbf{x}, t)$

$$r_2^k(\mathbf{x}, t) = \tilde{X}_2^k(\mathbf{x}) \cdot \tilde{T}_2^k(t) - d_1 \cdot \tilde{X}_1^k(\mathbf{x}) \cdot \tilde{T}_1^k(t) \quad (89)$$

from which computing point (\mathbf{x}_2^k, t_2^k) according to:

$$(\mathbf{x}_2^k, t_2^k) = \underset{\mathbf{x}, t}{\operatorname{argmax}} |r_2^k(\mathbf{x}, t)| \quad (90)$$

As by construction $r_2^k(\mathbf{x}_1^k, t_1^k) = 0$ we can ensure $(\mathbf{x}_2^k, t_2^k) \neq (\mathbf{x}_1^k, t_1^k)$.

The procedure is generalized for obtaining the other points involved in the interpolation procedure. Thus for obtaining point (\mathbf{x}_j^k, t_j^k) , $j \leq k$, we consider

$$r_j^k(\mathbf{x}, t) = \tilde{X}_j^k(\mathbf{x}) \cdot \tilde{T}_j^k(t) - \sum_{i=1}^{j-1} d_i \cdot \tilde{X}_i^k(\mathbf{x}) \cdot \tilde{T}_i^k(t) \quad (91)$$

whose maximum results the searched point (\mathbf{x}_j^k, t_j^k) , i.e.

$$(\mathbf{x}_j^k, t_j^k) = \underset{\mathbf{x}, t}{\operatorname{argmax}} |r_j^k(\mathbf{x}, t)| \quad (92)$$

The coefficients d_1, \dots, d_{j-1} must be chosen for ensuring that $(\mathbf{x}_j^k, t_j^k) \neq (\mathbf{x}_i^k, t_i^k)$, $\forall i < j \leq k$. For this purpose we enforce that the residual $r_j^k(\mathbf{x}, t)$ vanishes at each location (\mathbf{x}_i^k, t_i^k) with $i < j$, that is:

$$\begin{aligned} r_j^k(\mathbf{x}_l^k, t_l^k) \\ = 0 = \tilde{X}_j^k(\mathbf{x}_l^k) \cdot \tilde{T}_j^k(t_l^k) - \sum_{i=1}^{j-1} d_i \cdot \tilde{X}_i^k(\mathbf{x}_l^k) \cdot \tilde{T}_i^k(t_l^k), \\ l = 1, \dots, j-1 \end{aligned} \quad (93)$$

that constitutes a linear system whose solution results the searched coefficients d_1, \dots, d_{j-1} .

4.3 DEIM-PGD Numerical Test

In this section we consider the one-dimensional model

$$\frac{\partial u}{\partial t} - k \cdot \frac{\partial^2 u}{\partial x^2} = -u^2 + f(x, t), \quad (x, t) \in (0, 1) \times (0, 1] \quad (94)$$

where the source term $f(x, t)$ is chosen in order to ensure that the exact solution writes

$$u(x, t) = x \cdot t + x^2 \cdot t^2 \quad (95)$$

that represents a separated solution involving two terms.

With this choice the initial condition reads $u(x, t=0) = 0$ whereas the boundary conditions are given by $u(x=0, t) = 0$ and $\frac{\partial u}{\partial x}(x=1, t) = t + 2 \cdot t^2$.

Using the notation introduced in the previous section and the strategy that reuses the previous reduced bases (see Eq. (82)), the convergence was reached after the construction of 5 reduced bases ($k=5$) in which the nonlinear term was interpolated. The final solution involved 40 ($N^k=40$) functional products $X_i^k(x) \cdot T_i^k(t)$, $i=1, \dots, 40$. Figures 1 and 2 depict the six first space and time modes respectively. Then Fig. 3 compares the time evolution at different locations obtained with the DEIM based PGD and the exact solution. Finally Fig. 4 shows the space-time DEIM based PGD solution. From these results we can conclude on the potentiality of the proposed technology for solving non-linear eventually multi-parametric models.

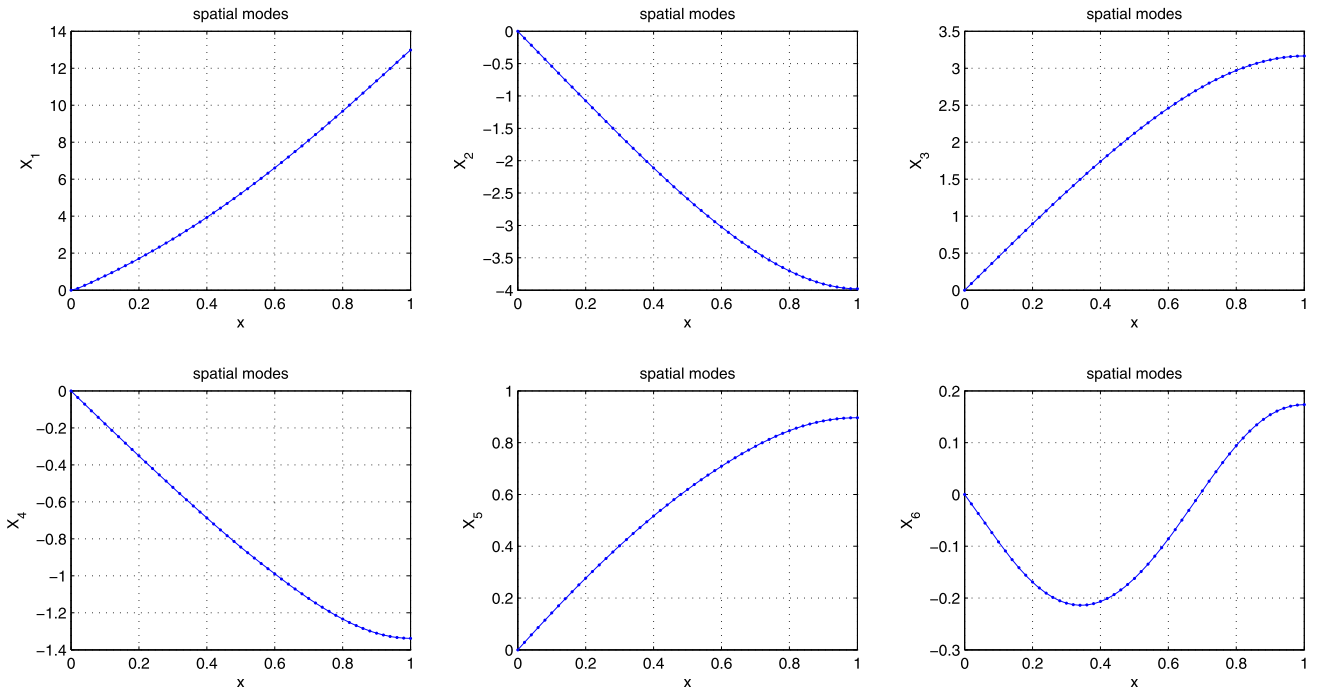


Fig. 1 Space modes: $X_i^k(x)$, $i=1, \dots, 6$

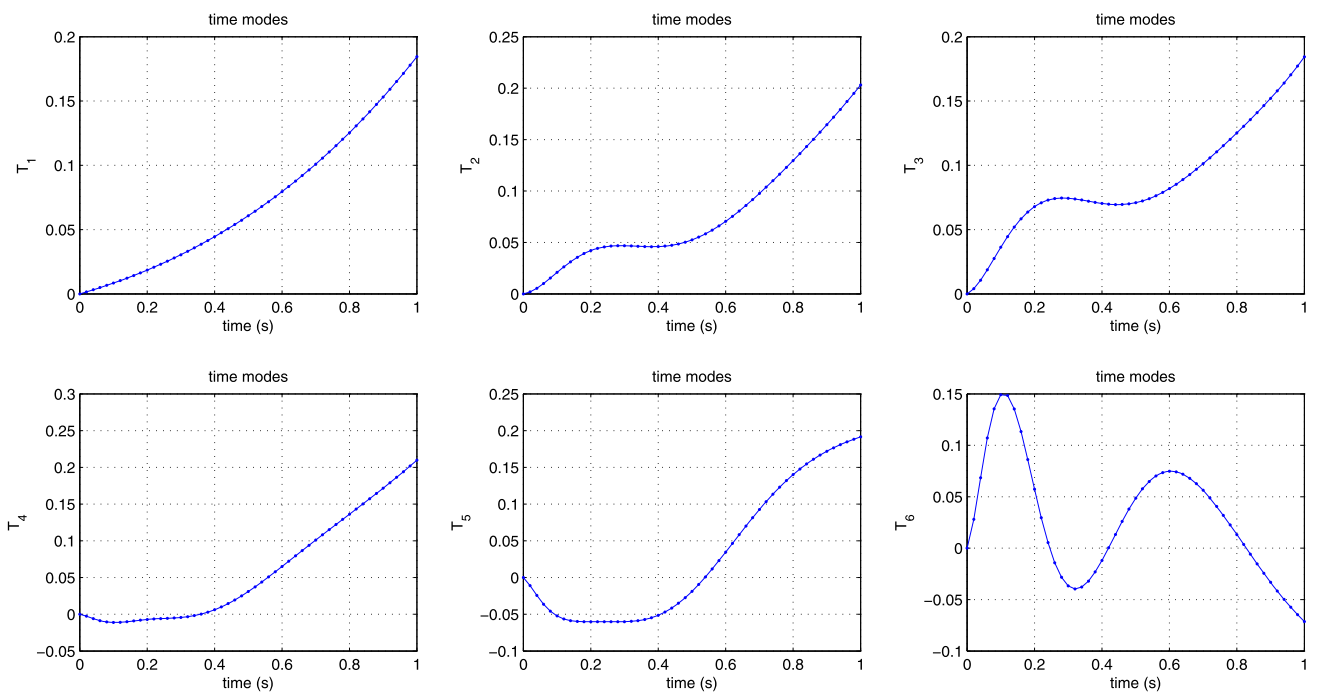


Fig. 2 Time modes: $T_i^k(t)$, $i = 1, \dots, 6$

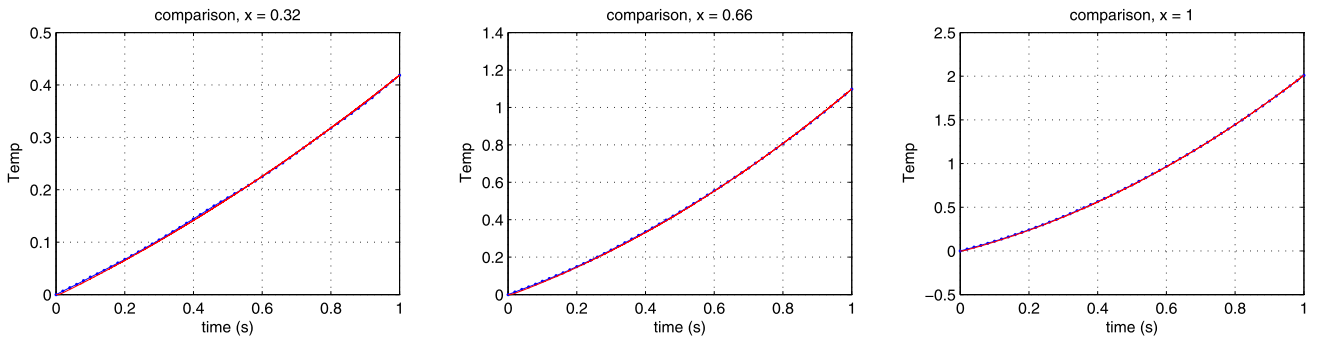
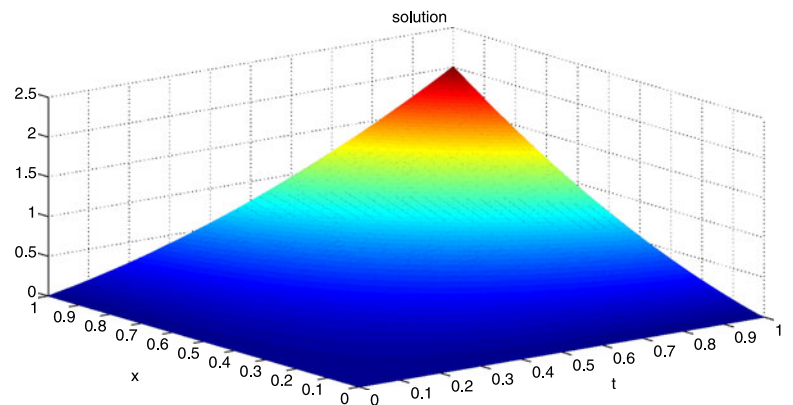


Fig. 3 DEIM based PGD solution versus the exact one

Fig. 4 Space-time reconstructed DEIM based PGD solution



5 Vademecums for Industrial Applications

As just illustrated usual computational mechanics models could be enriched by introducing several extra-coordinates. Thus, adding some new coordinates to models initially non high-dimensional, could lead to new, never before explored insights in the physics as previously illustrated in the context of a parametric thermal models.

Next, we review some of the most representative examples explored so far.

5.1 Geometrical and Material Parameters

Classical design strategies consider given parameters and then solve the mechanical problem. A cost function is evaluated as soon as the solution is available. If the solution is not good enough, parameters are updated by using an appropriate optimization strategy and then the model is solved again, and the process continues until reaching convergence. The main drawback lies in the fact that numerous resolutions are generally needed with the consequent impact in terms of the computing time.

As explained before, if all the parameters involved in the design process are considered as extra-coordinates (just like space and time in standard models) a unique solution of the resulting multidimensional model allows knowing the solution for any choice of the parameters considered as extra-coordinates. The price to pay is the solution of a multidimensional model. However, this solution is feasible by invoking the PGD solver and its inherent separated representation. This allows circumventing the curse of dimensionality.

This kind of parametric modelling was addressed in [9, 11, 25, 70] where material parameters were introduced as extra-coordinates. In [51], thermal conductivities, macroscopic temperature and its time evolution were introduced as extra-coordinates for computing linear and non-linear homogenization.

In [17] we proved that the PGD method with separated space coordinates is a very efficient way to compute 3D elastic problems defined in degenerated domains (plate or shells) with a numerical cost that scales like 2D. The key point for such an approach is to use a separated representation for each quantity of the model as a sum of products of functions of each coordinate or group of coordinates. In the case of a plate the retained separated representation of a generic function $u(x, y, z)$ reads:

$$u(x, y, z) \approx \sum_{i=1}^N X_i(x, y) \cdot Z_i(z) \quad (96)$$

In this work, we consider additional model parameters as extra-coordinates. In addition to the 3 dimensions describing the physical space, we add new coordinates related to the Young's modulus E , to the Poisson's coefficient ν and to the

geometrical parameter e depicted in Fig. 5. Thus separated representations write:

$$u(x, y, z, E, \nu, e) \approx \sum_{i=1}^N X_i(x, y) \cdot Z_i(z) \cdot F_i(E) \cdot H_i(\nu) \cdot G_i(e) \quad (97)$$

For an efficient solution of the mechanical model making use of a separated representation we must ensure a separated representation of all the fields involved in the model. However, there is a technical difficulty because the coordinates e and z are not independent. In order to perform a fully separated representation we could consider the following transformation $z \rightarrow z'$:

$$\begin{cases} z' = \frac{z}{e} & z \in [0, e] \\ z' = 1 + \frac{z-e}{h} & z \in [e, e+h] \\ z' = 2 + \frac{z-e-h}{e} & z \in [e+h, e+h+e] \end{cases} \quad (98)$$

Thus, finally $z' \in [0, 3]$ and $e \in \Omega_e$, both being independents, lead to a fully separated representation. The components of the Jacobian matrix are $\frac{1}{e}$ or $\frac{1}{h}$ that facilitates the change of variable in the resulting weak form related to the elastic model.

In the numerical example here addressed we considered $\nu \in [0, 0.5]$, $E \in [5, 500]$ (GPa) and $e \in [5, 20]$ (mm) that allow to describe a large variety of isotropic material: plastics, metals, alloys, ...

As soon the parametric solution is computed by solving only once the resulting multidimensional model (defined in this case in a space of dimension 6) we can particularize it for different materials (by choosing appropriate values of E or ν) or for different geometries (by choosing e). Figure 6

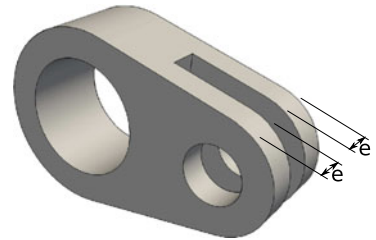


Fig. 5 Parametrized part

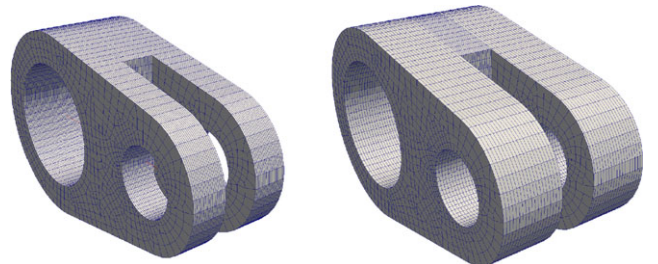


Fig. 6 Parts related to different choices of the model parameter e

Fig. 7 Add-on developed for the open source post-processing code ParaView. The three sliders on the bottom-right menu control, respectively, the Poisson coefficient, Young's modulus and thickness e

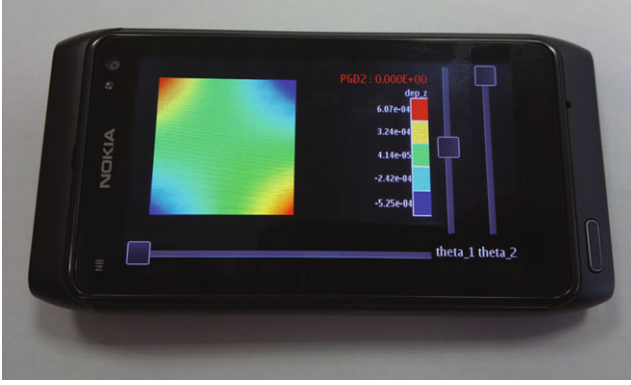
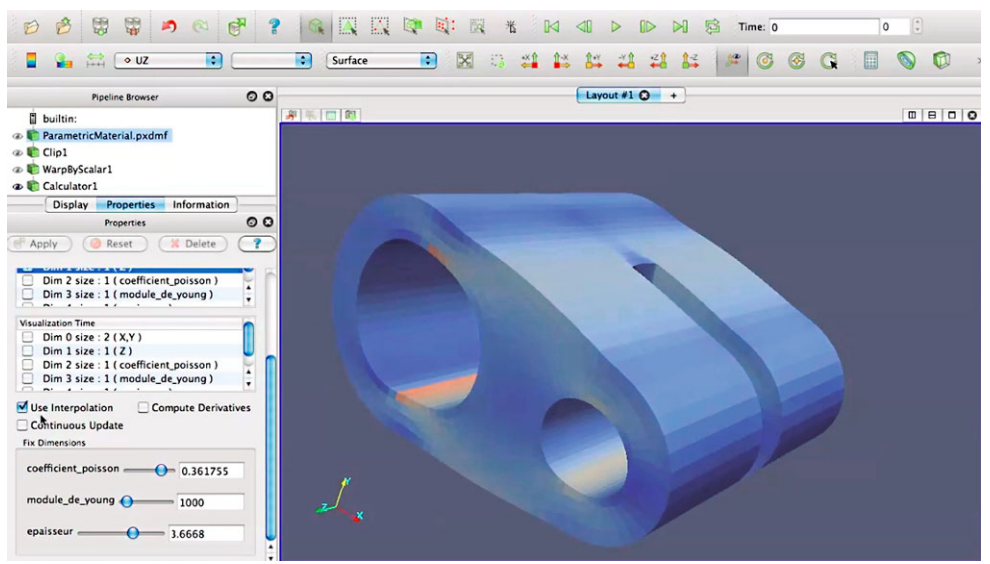


Fig. 8 Composite laminate analysis on a smartphone

illustrates the same part for two values of the parameter e , while Fig. 7 shows the appearance of the application.

In [17], the anisotropy directions of plies involved in a composite laminate were considered as extra-coordinates. As soon as the separated representation of the parametric solution was computed off-line, its on-line use only needs to particularize such solution for a desired set of parameters. Obviously, this task can be performed very fast, many times in real time, and by using light computing platforms, as smartphones or tablets. Figure 8 illustrates a smartphone application [17] in which the elastic solution of a two-ply composite laminate was computed by introducing the fiber orientation in each ply, θ_1 and θ_2 , as extra-coordinates

$$u_j(x, y, z, \theta_1, \theta_2)$$

$$\approx \sum_{i=1}^N X_i^j(x, y) \cdot Z_i^j(z) \cdot \Theta_i^{j,1}(\theta_1) \cdot \Theta_i^{j,2}(\theta_2) \quad (99)$$

Then one can visualize each component of the displacement field, by particularizing the z -coordinate from the hor-

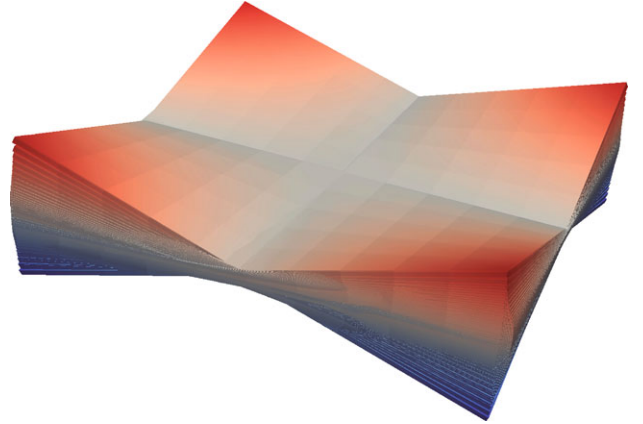


Fig. 9 Deformation envelope generated by all combinations of the reinforcement orientations of the top and bottom plies

izontal slider as well as the orientation of the reinforcement in both plies from both vertical sliders. Obviously when the laminate is equilibrated there is no noticeable deformations and the plate remains plane, but as soon as we consider an unbalanced laminate by acting on both vertical sliders, the plate exhibits a residual distortion. By assuming a certain uncertainty in the real orientation of such plies, one can evaluate the envelope of the resulting distorted structures due to the thermomechanical coupling as depicted in Fig. 9.

5.2 Inverse Identification and Optimization

It is easy to understand that after performing this type of calculations, in which parameters are considered advantageously as new coordinates of the model, a posteriori inverse identification or optimization can be easily handled. This new PGD framework allows us to perform this type of calculations very efficiently, because in fact all possible solutions

have been previously computed in the form of a separated, high-dimensional solution so that they constitute a simple post-processing of this general solution. Process optimization was considered in [31], for instance. Shape optimization was performed in [54] by considering all the geometrical parameters as extra-coordinates, leading to the model solution in any of the geometries generated by the parameters considered as extra-coordinates.

We consider the Laplace equation defined in the parametrized domain Ω^r described from 12 control points \mathcal{P}_i^r , $i = 1, \dots, 12$, with coordinates

$$\begin{cases} \mathcal{P}_1^r = (0, 0) \\ \mathcal{P}_2^r = (1, 0) \\ \mathcal{P}_3^r = (2, 0) \\ \mathcal{P}_4^r = (3, 0) \\ \mathcal{P}_5^r = (4, 0) \\ \mathcal{P}_6^r = (5, 0) \\ \mathcal{P}_7^r = (5, 1) \\ \mathcal{P}_8^r = (4, 1) \\ \mathcal{P}_9^r = (3, 1) \\ \mathcal{P}_{10}^r = (2, 1) \\ \mathcal{P}_{11}^r = (1, 1) \\ \mathcal{P}_{12}^r = (0, 1) \end{cases} \quad (100)$$

Different polygonal domains Ω are obtained by moving vertically points \mathcal{P}_i^r , $i = 7, \dots, 12$, being defined by:

$$\begin{cases} \mathcal{P}_1 = (0, 0) \\ \mathcal{P}_2 = (1, 0) \\ \mathcal{P}_3 = (2, 0) \\ \mathcal{P}_4 = (3, 0) \\ \mathcal{P}_5 = (4, 0) \\ \mathcal{P}_6 = (5, 0) \\ \mathcal{P}_7 = (5, 1 + \theta_1) \\ \mathcal{P}_8 = (4, 1 + \theta_2) \\ \mathcal{P}_9 = (3, 1 + \theta_3) \\ \mathcal{P}_{10} = (2, 1 + \theta_4) \\ \mathcal{P}_{11} = (1, 1 + \theta_5) \\ \mathcal{P}_{12} = (0, 1 + \theta_6) \end{cases} \quad (101)$$

with $\theta_i \in [-0.3, 0.3]$, $i = 1, \dots, 6$.

The resulting separated representation of the solution involves 70 terms

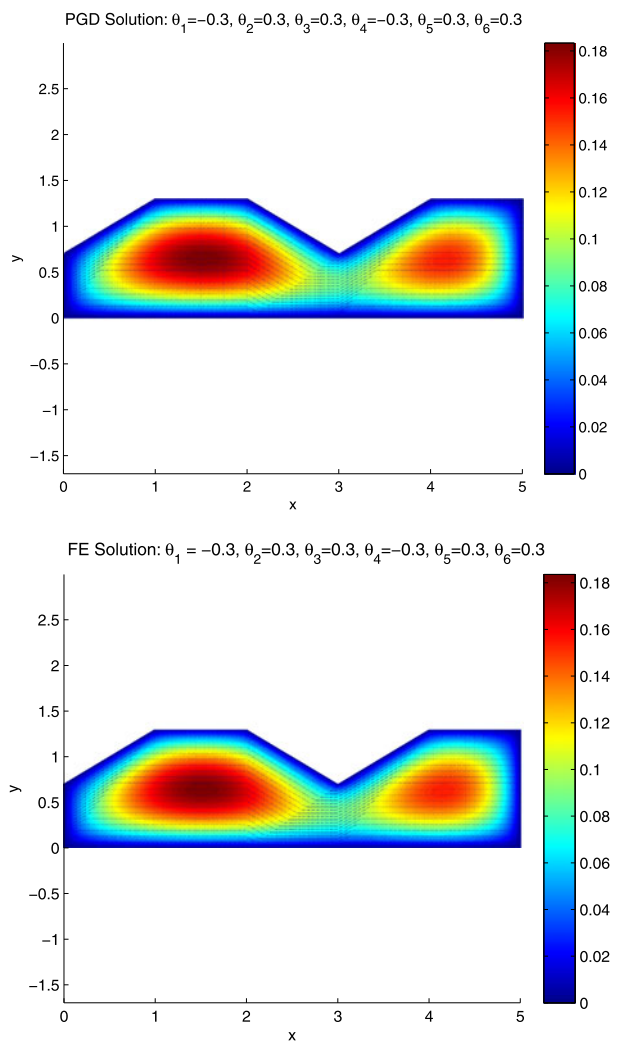


Fig. 10 Comparing $u(\mathbf{x}, \theta_1 = -0.3, \theta_2 = 0.3, \theta_3 = 0.3, \theta_4 = -0.3, \theta_5 = 0.3, \theta_6 = 0.3)$ with the finite element solution $u(\mathbf{x})$, $\mathbf{x} \in \Omega$, with Ω defined by $\theta_1 = -0.3, \theta_2 = 0.3, \theta_3 = 0.3, \theta_4 = -0.3, \theta_5 = 0.3, \theta_6 = 0.3$

$$\begin{aligned} u(\mathbf{x}, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6) \\ \approx \sum_{i=1}^{70} F_i(\mathbf{x}) \cdot \Theta_{1i}(\theta_1) \cdot \Theta_{2i}(\theta_2) \cdot \Theta_{3i}(\theta_3) \cdot \Theta_{4i}(\theta_4) \\ \cdot \Theta_{5i}(\theta_5) \cdot \Theta_{6i}(\theta_6) \end{aligned} \quad (102)$$

Figure 10 compares the particularization of the general solution (102) when considering the geometry defined by $(\theta_1, \dots, \theta_6) = (-0.3, 0.3, 0.3, -0.3, 0.3, 0.3)$, that is $u(\mathbf{x}, \theta_1 = -0.3, \theta_2 = 0.3, \theta_3 = 0.3, \theta_4 = -0.3, \theta_5 = 0.3, \theta_6 = 0.3)$ with the finite element solution in such a domain. We can conclude that both solutions are in perfect agreement. It is important to notice that as the interval in which coordinates θ_i ($i = 1, \dots, 6$) are defined $[-0.3, 0.3]$ were discretized by using 13 nodes uniformly distributed, the separated representation (102) represents the solution for 13^6 different geometries, that is, for 4,826,809 possible do-

main geometries. Again, the analysis can be performed in deployed devices like smartphones or tablets, in real time.

5.3 PGD Based Dynamic Data Driven Application Systems

Inverse methods in the context of real-time simulations were addressed in [35] and were coupled with control strategies in [32] as a first step towards DDDAS (dynamic data-driven application systems). Moreover, because the general parametric solution was pre-computed off-line, it can be used on-line under real time constraints and using light computing platforms like smartphones [17, 32], that constitutes a first step towards the use of this kind of representation in augmented reality platforms.

Traditionally, Simulation-based Engineering Sciences (SBES) relied on the use of static data inputs to perform the simulations. These data could be parameters of the model(s) or boundary conditions, outputs at different time instants, etc., traditionally obtained through experiments. The word static is intended here to mean that these data could not be modified during the simulation.

A new paradigm in the field of Applied Sciences and Engineering has emerged in the last decade. Dynamic Data-Driven Application Systems (DDDAS) constitute nowadays one of the most challenging applications of SBES. By DDDAS we mean a set of techniques that allow the linkage of simulation tools with measurement devices for real-time control of simulations and applications. As defined by the U.S. National Science Foundation, “DDDAS entails the ability to dynamically incorporate additional data into an executing application, and in reverse, the ability of an application to dynamically steer the measurement process” [75].

An important issue encountered in DDDAS, related to process control and optimization, inverse analysis, etc., lies in the necessity of solving many direct problems. Thus, for example, process optimization implies the definition of a cost function and the search of optimum process parameters, which minimize the cost function. In most engineering optimization problems the solution of the model is the most expensive step. Real-time computations with zero-order optimization techniques can not be envisioned except for very particular cases. The computation of sensitivity matrices and adjoint approaches also hampers fast computations. Moreover, global minima are only ensured under severe conditions, which are not (or cannot be) verified in problems of engineering interest.

Multidimensionality offers an alternative getaway to avoid too many direct solutions. In this section the main ideas related to casting the model into a multidimensional framework, followed by process optimization, are introduced. For the sake of clarity in what follows we consider the thermal model related to a material flowing into a heated

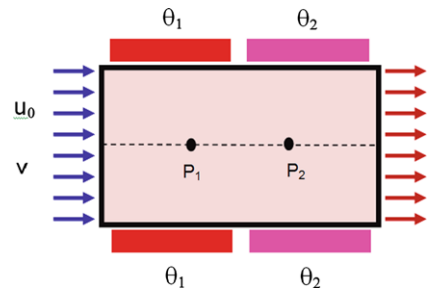


Fig. 11 Thermal process consisting of two heating devices located on the die walls where the temperature is enforced to the values θ_1 and θ_2 respectively

die. Despite the apparent simplicity, the strategy here described can be extended to address more complex scenarios.

For illustrative purposes we consider the 2D thermal process sketched in Fig. 11. The material flows with a velocity v inside a die Ω of length L and width H . The temperature of the material at the die entrance is u_0 . The die is equipped with two heating devices of lengths L_1 and L_2 respectively, whose temperatures θ_1 and θ_2 respectively, can range within an interval $[\theta_{min}, \theta_{max}]$.

The die is equipped with two heating devices as depicted in Fig. 11 whose temperatures constitute the process parameters to be optimized and, eventually, controlled. For the sake of simplicity the internal heat generation Q is assumed constant, as well as the velocity v and the inlet temperature u_0 .

Different values of prescribed temperatures at both heating devices can be considered. The resulting 2D heat transfer equation can be then solved. As noted earlier, optimization or inverse identification will require many direct solutions or, as named in the introduction, static data computations. Obviously, when the number of the process parameters involved in the model is increased, standard approaches fail to compute optimal solutions in a reasonable time. Thus, for a large number of process parameters, real-time computations are precluded and, moreover, performing “on-line” optimization or inverse analysis is a challenging issue.

The method proposed in [32] consists on introducing both process parameters, i.e. temperatures of the heating devices, θ_1 and θ_2 , as extra coordinates.

To circumvent the curse of dimensionality related to the high dimensional space in which the temperature field $u(x, y, \theta_1, \theta_2)$ is defined—which we retain to be four-dimensional for the ease of exposition—we consider a separated representation of that field:

$$u(x, y, \theta_1, \theta_2) \approx \sum_{i=1}^N F_i(x, y) \Theta_i^1(\theta_1) \Theta_i^2(\theta_2) \quad (103)$$

where all the functions involved in such separated representation are computed by applying the Proper Generalized Decomposition technique, described previously.

Optimization procedures look for optimal parameters minimizing an appropriate single or multi objective cost function (sometimes subjected to many constraints). In this work we consider a simple scenario, in which the cost function only involves the coldest thermal history of an imaginary material particle traversing the die, it is expressed as:

$$\mathcal{C}(\theta_1, \theta_2) = \frac{1}{2} \left(\int_0^L u \left(x, \frac{H}{2}, \theta_1, \theta_2 \right) dx - \beta \right)^2, \quad (104)$$

where β denotes the optimal value of the thermal history able to ensure a certain material transformation. Values lower than β imply that the material has not received the necessary amount of heat, whereas values higher than β imply an unnecessary extra-heating.

Now, optimal process parameters θ_1^{opt} and θ_2^{opt} must be calculated by minimizing the cost function. There exist many techniques for such minimization. The interested reader can refer to any book on optimization. Many of them proceed by evaluating the gradient of the cost function and then moving on that direction. The gradient computation involves the necessity of performing first derivatives of the cost function with respect to the process parameters. Other techniques involve the calculation of second derivatives. To this end, one should calculate the derivatives of the problem solution with respect to the optimization parameters.

It is important to note that separated representations of the process parameters drastically simplifies this task because as the solution depends explicitly on the parameters its derivation is straightforward, namely,

$$\frac{\partial u}{\partial \theta_1}(x, y, \theta_1, \theta_2) \approx \sum_{i=1}^N F_i(x, y) \frac{\partial \Theta_i^1}{\partial \theta_1}(\theta_1) \Theta_i^2(\theta_2),$$

and

$$\frac{\partial u}{\partial \theta_2}(x, y, \theta_1, \theta_2) \approx \sum_{i=1}^N F_i(x, y) \Theta_i^1(\theta_1) \frac{\partial \Theta_i^2}{\partial \theta_2}(\theta_2)$$

Note that second derivatives are also similarly obtained. The calculation of the solution derivatives is a tricky point when proceeding from standard discretization techniques because the parametric dependency of the solution is, in general, not explicit.

In the simulations carried out in [17], the minimization of the cost function was performed by using a Levenberg-Marquardt algorithm, see [30] for further details.

By performing an inverse analysis it is also possible to determine a hypothetical malfunctioning of the system, along with the determination of the broken heater. This inverse identification can easily be done in real-time by minimizing a new cost function involving the distance of the measurements to the optimal solution obtained before. The

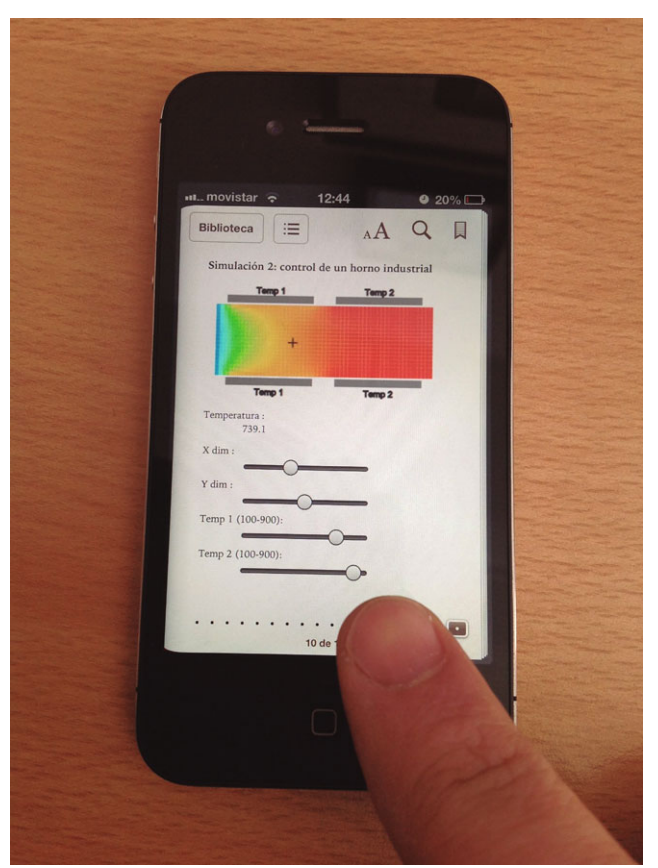


Fig. 12 Implementation of the technique described before on an iPhone. Simple formats such as the epub open format, that enables javascript, suffices implement this technique

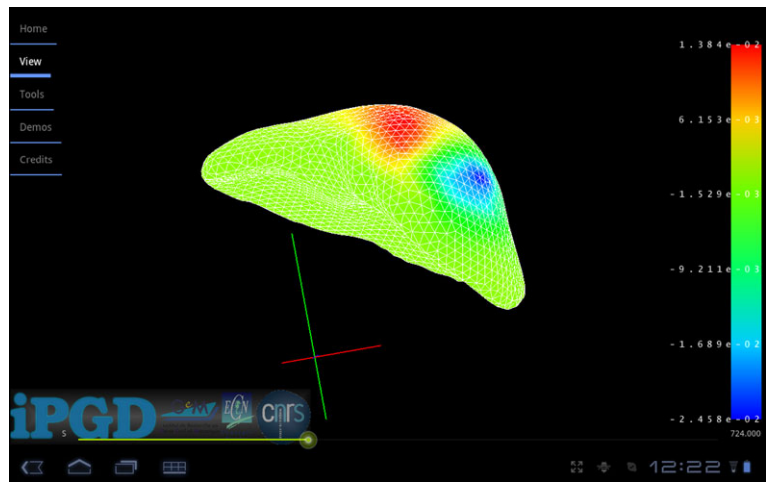
last step consists in the reconfiguration of the system, assuming that the broken heater cannot be replaced for a while. Again, a minimization procedure of the cost function, Eq. (104), this time with one fixed temperature (that of the broken heater) serves to this purpose. An implementation of this procedure on a smartphone can be done easily, see Fig. 12.

5.4 Surgery Simulators

As mentioned before, surgical simulators must provide feedback response frequencies higher than 500 Hz. This means that we must solve problems involving material and geometrical nonlinearities close to one thousand times per second. It is now clear that the use of model reduction seems to be an appealing alternative for reaching such performances. However, techniques based on the use of POD, POD with interpolation (PODI), even combined with asymptotic numerical methods to avoid the computation of the tangent stiffness matrix [28, 54], exhibit serious difficulties to fulfil such requirements as discussed in [60–63].

Here, parametric solutions are envisaged in which the applied load \mathbf{p} and its point of application \mathbf{y} are considered

Fig. 13 Towards real time surgical simulations based on parametric PGD-based vademecums



as extra-coordinates, allowing the off-line calculation of the parametric solution:

$$u_j(\mathbf{x}, \mathbf{p}, \mathbf{y}) \approx \sum_{i=1}^N X_i^j(\mathbf{x}) \cdot P_i^j(\mathbf{p}) \cdot Y_i^j(\mathbf{y}) \quad (105)$$

Again, the obtained, off-line, solution is exploited in real time even on smartphones and tablets, see Fig. 13 for an Android implementation.

For a liver palpation simulation, for instance, model's solution was composed by a total of $N = 167$ functional pairs. The third component (thus $j = 3$) of the first six spatial modes $\mathbf{X}(\mathbf{x})$ is depicted in Fig. 14. The same is done in Fig. 15 for functions \mathbf{Y} , although in this case they are defined only on the boundary of the domain, i.e., $\bar{\Gamma} = \partial\Omega$.

In this case, an explicit linearization of the resulting system of equations was employed, although other more sophisticated techniques could equally be employed.

Noteworthy, both \mathbf{X} and \mathbf{Y} sets of functions present a structure similar to that generated by Proper Orthogonal Decompositions methods, despite the fact that they are not, in general, optimal. Note how the frequency content of each pair of functions increases as we increase the number of the function, k .

The solution provided by the method agrees well with reference FE solutions obtained employing full-Newton-Raphson iterative schemes (following the same tendency than that shown for the beam bending problem). But, notably, the computed solution can be stored in a so compact form that an implementation of the method is possible on handheld devices such as smartphones and tablets. For more sophisticated requirements, such as those dictated by haptic peripherals, a simple laptop (in our case a MacBook pro running MAC OSX 10.7.4, equipped with 4 Gb RAM and an Intel core i7 processor at 2.66 GHz) is enough to achieve this performance, see Fig. 16.

5.5 Other Industrial Applications

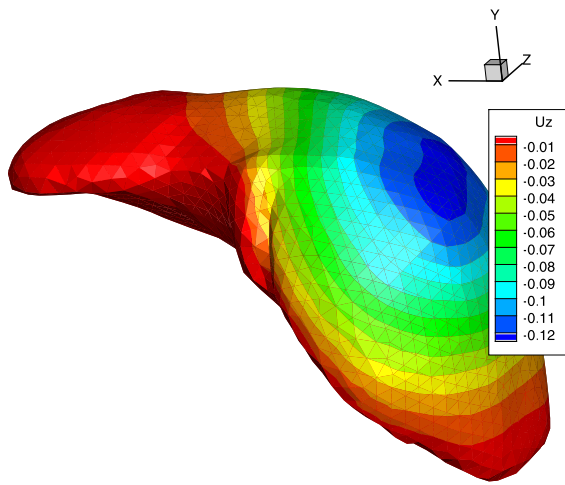
In [27, 69] authors addressed an industrial application for on-line simulation and material and process characterization of automated tape placement for composite forming processes. This application is at present running at the industrial level in different platforms: laptop, tablets and smartphones. Its application for training purposes is being explored, and the first accomplishments were reported in [18].

6 Conclusions

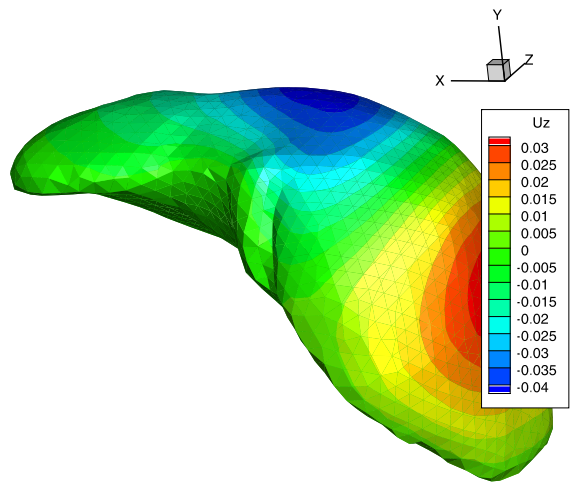
In this paper we proved that models can be enriched by introducing model parameters as extra-coordinates. Thus, one can introduce boundary conditions, material or process parameters, initial conditions, geometrical parameters, ... as extra-coordinates in order to compute general parametric solutions that define a sort of handbook or metamodels, much more rich than the ones obtained by sampling the parametric space. The price to be paid is the increase of the model dimensionality, but the separated representations involved in the so called PGD method allows circumventing efficiently this numerical illness. Moreover, the parametric solution is calculated in a sort of compressed format allowing for cheap storage and post-treatment. Thus, only one off-line heavy solution is needed for computing the parametric solution that constitutes the computational vademecum that is then used on-line, sometimes in real time, in deployed devices as tablets or smartphones.

This off-lie/on-line approach opens numerous possibilities in the context of simulation based engineering for simulating, optimizing or controlling materials, processes and systems.

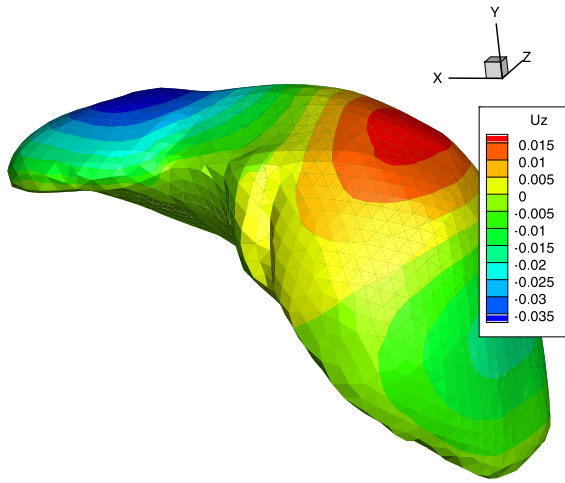
Until now, the results obtained are very encouraging, however a major difficulty persists: the one related to the solution of parametric non-linear models involving multi-scale



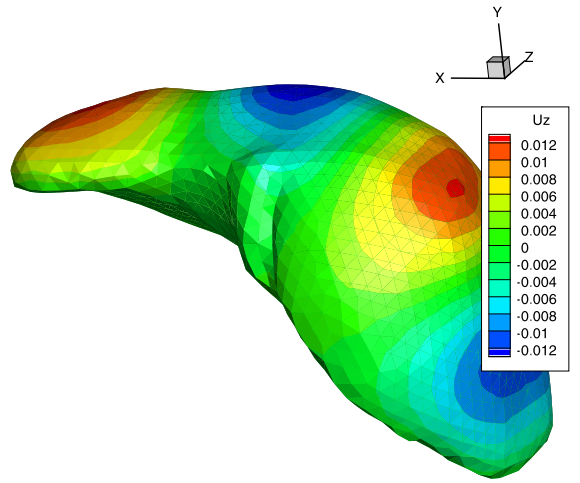
(a)



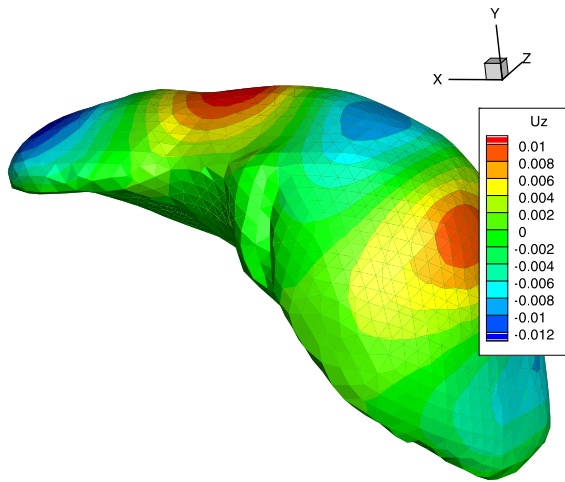
(b)



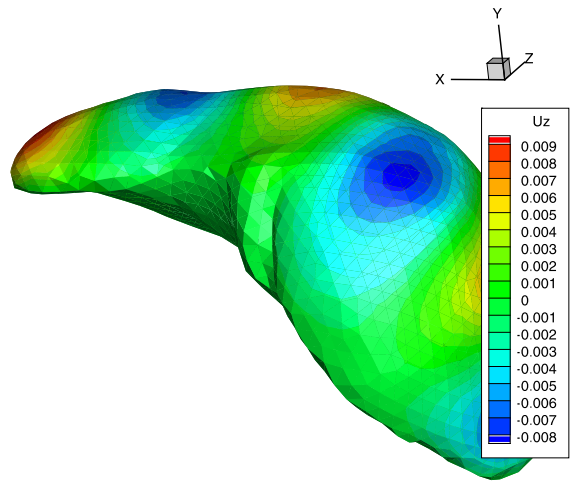
(c)



(d)

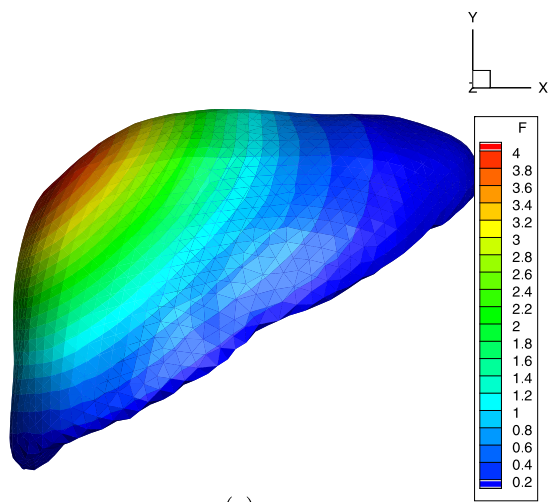


(e)

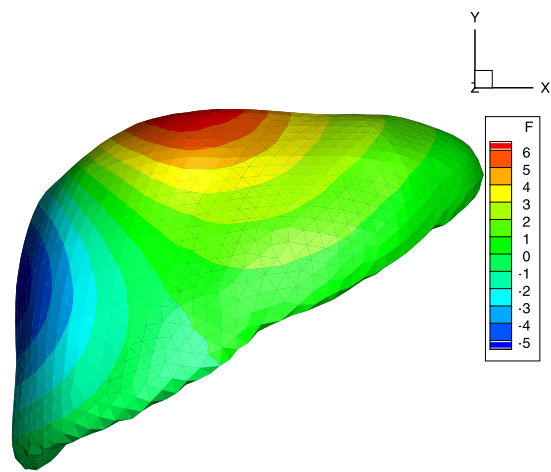


(f)

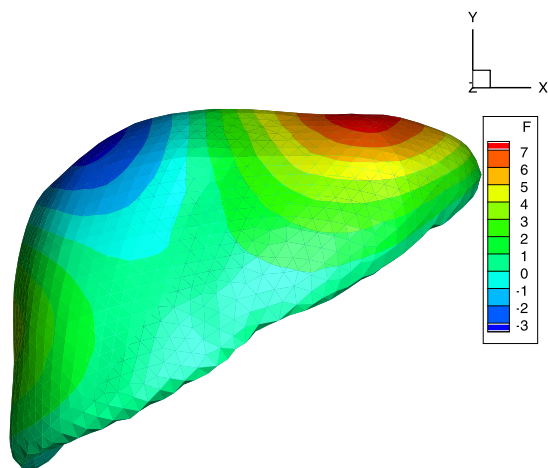
Fig. 14 Six first functions $\mathbf{X}(\mathbf{x})$, $k = 1, \dots, 6$, for the simulation of the liver



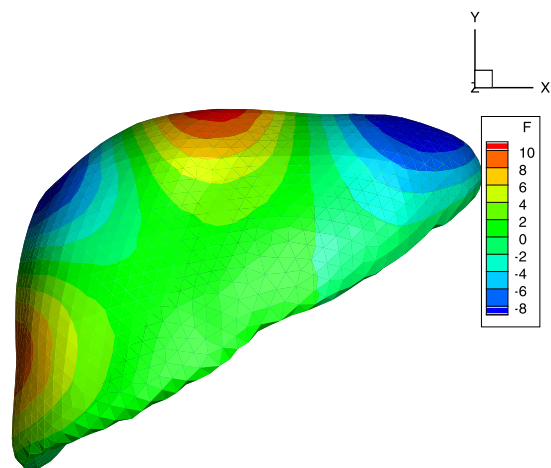
(a)



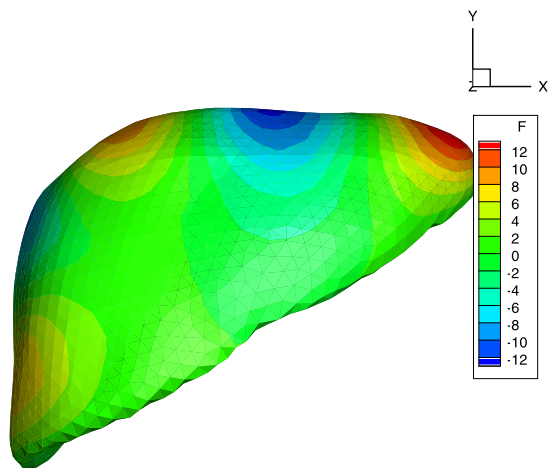
(b)



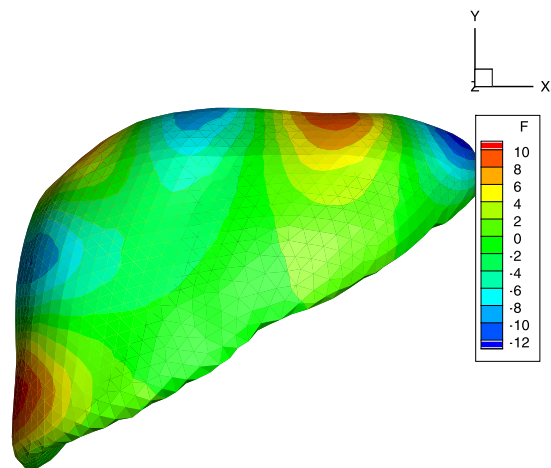
(c)



(d)



(e)



(f)

Fig. 15 Six first functions $Y(y)$, $k = 1, \dots, 6$, for the simulation of the liver. Note that, in this case, functions $Y(y)$ are defined on the boundary of the liver only

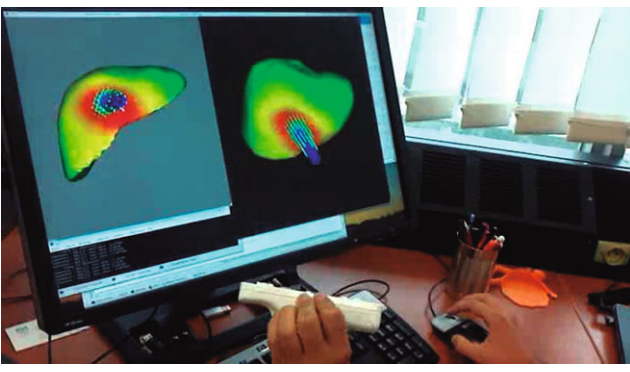


Fig. 16 Implementation of the proposed technique on a PC

and multi-physics complex couplings. For this purpose different alternatives have been analyzed.

If the next future non-linear parametric models can be addressed with the same simplicity than the linear ones, parametric PGD based vademecums could open a new age for the XXI century design, optimization and control of materials, processes and systems, revolutionizing the ICTs technologies.

Appendix: Alternating Directions Separated Representation Constructor

A.1 Computing $R(\mathbf{x})$ from $S(t)$ and $W(k)$

We consider the extended weighted residual form of Eq. (8):

$$\int_{\Omega \times \mathcal{I}_t \times \mathcal{I}_k} u^* \left(\frac{\partial u}{\partial t} - k \cdot \Delta u - f \right) d\mathbf{x} \cdot dt \cdot dk = 0 \quad (106)$$

where the trial and test functions write respectively:

$$u^n(\mathbf{x}, t, k) = \sum_{i=1}^{n-1} X_i(\mathbf{x}) \cdot T_i(t) \cdot K_i(k) + R(\mathbf{x}) \cdot S(t) \cdot W(k) \quad (107)$$

and, assuming S and W known from the previous iteration,

$$u^*(\mathbf{x}, t, k) = R^*(\mathbf{x}) \cdot S(t) \cdot W(k) \quad (108)$$

Introducing (107) and (108) into (106) it results:

$$\begin{aligned} & \int_{\Omega \times \mathcal{I}_t \times \mathcal{I}_k} R^* \cdot S \cdot W \\ & \cdot \left(R \cdot \frac{\partial S}{\partial t} \cdot W - k \cdot \Delta R \cdot S \cdot W \right) d\mathbf{x} \cdot dt \cdot dk \\ & = - \int_{\Omega \times \mathcal{I}_t \times \mathcal{I}_k} R^* \cdot S \cdot W \cdot \mathcal{R}^{n-1} d\mathbf{x} \cdot dt \cdot dk \end{aligned} \quad (109)$$

where \mathcal{R}^{n-1} defines the residual related to $u^{n-1}(\mathbf{x}, t, k)$:

$$\mathcal{R}^{n-1} = \sum_{i=1}^{n-1} X_i \cdot \frac{\partial T_i}{\partial t} \cdot K_i - \sum_{i=1}^{n-1} k \cdot \Delta X_i \cdot T_i \cdot K_i - f \quad (110)$$

Once all functions involving time and conductivity have been determined, we can integrate Eq. (109) along its respective domains $\mathcal{I}_t \times \mathcal{I}_k$, and by taking into account the following notations:

$$\left[\begin{array}{lll} w_1 = \int_{\mathcal{I}_k} W^2 dk & s_1 = \int_{\mathcal{I}_t} S^2 dt & r_1 = \int_{\Omega} R^2 d\mathbf{x} \\ w_2 = \int_{\mathcal{I}_k} k W^2 dk & s_2 = \int_{\mathcal{I}_t} S \cdot \frac{dS}{dt} dt & r_2 = \int_{\Omega} R \cdot \Delta R d\mathbf{x} \\ w_3 = \int_{\mathcal{I}_k} W dk & s_3 = \int_{\mathcal{I}_t} S dt & r_3 = \int_{\Omega} R d\mathbf{x} \\ w_4^i = \int_{\mathcal{I}_k} W \cdot K_i dk & s_4^i = \int_{\mathcal{I}_t} S \cdot \frac{dT_i}{dt} dt & r_4^i = \int_{\Omega} R \cdot \Delta X_i d\mathbf{x} \\ w_5^i = \int_{\mathcal{I}_k} k W \cdot K_i dk & s_5^i = \int_{\mathcal{I}_t} S \cdot T_i dt & r_5^i = \int_{\Omega} R \cdot X_i d\mathbf{x} \end{array} \right] \quad (111)$$

Equation (109) is reduced to:

$$\begin{aligned} & \int_{\Omega} R^* \cdot (w_1 \cdot s_2 \cdot R - w_2 \cdot s_1 \cdot \Delta R) d\mathbf{x} \\ & = - \int_{\Omega} R^* \cdot \left(\sum_{i=1}^n w_4^i \cdot s_4^i \cdot X_i \right. \\ & \quad \left. - \sum_{i=1}^n w_5^i \cdot s_5^i \cdot \Delta X_i - w_3 \cdot s_3 \cdot f \right) d\mathbf{x} \end{aligned} \quad (112)$$

Equation (112) defines an elliptic steady-state boundary value problem that can be solved by using any discretization technique operating on the weak form of the problem (finite elements, finite volumes, ...). Another possibility consists in coming back to the strong form of Eq. (112):

$$\begin{aligned} & w_1 \cdot s_2 \cdot R - w_2 \cdot s_1 \cdot \Delta R \\ & = - \left(\sum_{i=1}^n w_4^i \cdot s_4^i \cdot X_i - \sum_{i=1}^n w_5^i \cdot s_5^i \cdot \Delta X_i - w_3 \cdot s_3 \cdot f \right) \end{aligned} \quad (113)$$

that could be solved by using any classical collocation technique (finite differences, SPH, ...).

A.2 Computing $S(t)$ from $R(\mathbf{x})$ and $W(k)$

In the present case the test function is written as:

$$u^*(\mathbf{x}, t, k) = S^*(t) \cdot R(\mathbf{x}) \cdot W(k) \quad (114)$$

Now, the weighted residual form becomes:

$$\begin{aligned} & \int_{\Omega \times \mathcal{I}_t \times \mathcal{I}_k} S^* \cdot R \cdot W \\ & \cdot \left(R \cdot \frac{\partial S}{\partial t} \cdot W - k \cdot \Delta R \cdot S \cdot W \right) d\mathbf{x} \cdot dt \cdot dk \\ & = - \int_{\Omega \times \mathcal{I}_t \times \mathcal{I}_k} S^* \cdot R \cdot W \cdot \mathcal{R}^{n-1} d\mathbf{x} \cdot dt \cdot dk \end{aligned} \quad (115)$$

that integrating in the space $\Omega \times \mathcal{I}_k$ and by taking into account the notation (111) results:

$$\begin{aligned} & \int_{\mathcal{I}_t} S^* \cdot (w_1 \cdot r_1 \cdot \frac{dS}{dt} - w_2 \cdot r_2 \cdot S) dt \\ &= - \int_{\mathcal{I}_t} S^* \cdot \left(\sum_{i=1}^n w_4^i \cdot r_5^i \cdot \frac{dT_i}{dt} \right. \\ & \quad \left. - \sum_{i=1}^n w_5^i \cdot r_4^i \cdot T_i - w_3 \cdot r_3 \cdot f \right) dt \end{aligned} \quad (116)$$

Equation (116) represents the weak form of the ODE defining the time evolution of the field S that can be solved by using any stabilized discretization technique (SU, Discontinuous Galerkin, ...). The strong form of Eq. (116) reads:

$$\begin{aligned} & w_1 \cdot r_1 \cdot \frac{dS}{dt} - w_2 \cdot r_2 \cdot S \\ &= - \left(\sum_{i=1}^n w_4^i \cdot r_5^i \cdot \frac{dT_i}{dt} - \sum_{i=1}^n w_5^i \cdot r_4^i \cdot T_i - w_3 \cdot r_3 \cdot f \right) \end{aligned} \quad (117)$$

Equation (117) can be solved by using backward finite differences, or higher order Runge-Kutta schemes, among many other possibilities.

A.3 Computing $W(k)$ from $R(\mathbf{x})$ and $S(t)$

In this part of the algorithm, the test function is written as:

$$u^*(\mathbf{x}, t, k) = W^*(k) \cdot R(\mathbf{x}) \cdot S(t) \quad (118)$$

Now, the weighted residual form becomes:

$$\begin{aligned} & \int_{\Omega \times \mathcal{I}_t \times \mathcal{I}_k} W^* \cdot R \cdot S \\ & \cdot \left(R \cdot \frac{\partial S}{\partial t} \cdot W - k \cdot \Delta R \cdot S \cdot W \right) d\mathbf{x} \cdot dt \cdot dk \\ &= - \int_{\Omega \times \mathcal{I}_t \times \mathcal{I}_k} W^* \cdot R \cdot S \cdot \mathcal{R}^{n-1} d\mathbf{x} \cdot dt \cdot dk \end{aligned} \quad (119)$$

Integrating Eq. (119) in $\Omega \times \mathcal{I}_t$ and considering the notations given by Eq. (111) leads to:

$$\begin{aligned} & \int_{\mathcal{I}_k} W^* \cdot (r_1 \cdot s_2 \cdot W - r_2 \cdot s_1 \cdot k \cdot W) dk \\ &= - \int_{\mathcal{I}_k} W^* \cdot \left(\sum_{i=1}^n r_5^i \cdot s_4^i \cdot K_i \right. \\ & \quad \left. - \sum_{i=1}^n r_4^i \cdot s_5^i \cdot k \cdot K_i - r_3 \cdot s_3 \cdot f \right) dk \end{aligned} \quad (120)$$

Equation (120) does not involve any differential operator. The strong form of Eq. (120) is:

$$\begin{aligned} & (r_1 \cdot s_2 - r_2 \cdot s_1 \cdot k) \cdot W \\ &= - \left(\sum_{i=1}^n (r_5^i \cdot s_4^i - r_4^i \cdot s_5^i \cdot k) \cdot K_i - r_3 \cdot s_3 \cdot f \right) \end{aligned} \quad (121)$$

Equation (121) represents an algebraic equation because the original model does not involve derivatives with respect to the conductivity. Thus, despite the introduction of parameters as additional model coordinates, the computational complexity remains essentially the same, however, the introduction of extra-coordinates implies in general the increase of the number of modes involved by the separated representation, and consequently the computing time.

References

1. Ammar A, Mokdad B, Chinesta F, Keunings R (2006) A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids. *J Non-Newton Fluid Mech* 139:153–176
2. Ammar A, Ryckelynck D, Chinesta F, Keunings R (2006) On the reduction of kinetic theory models related to finitely extensible dumbbells. *J Non-Newton Fluid Mech* 134:136–147
3. 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 representation. *J Non-Newton Fluid Mech* 144:98–121
4. Ammar A, Chinesta F, Joyot P (2008) The nanometric and micro-metric scales of the structure and mechanics of materials revisited: an introduction to the challenges of fully deterministic numerical descriptions. *Int J Multiscale Comput Eng* 6(3):191–213
5. Ammar A, Pruliere E, Chinesta F, Laso M (2009) Reduced numerical modeling of flows involving liquid-crystalline polymers. *J Non-Newton Fluid Mech* 160:140–156
6. Ammar A, Normandin M, Daim F, Gonzalez D, Cueto E, Chinesta F (2010) Non-incremental strategies based on separated representations: applications in computational rheology. *Commun Math Sci* 8(3):671–695
7. Ammar A, Chinesta F, Falco A (2010) On the convergence of a greedy rank-one update algorithm for a class of linear systems. *Arch Comput Methods Eng* 17(4):473–486
8. Ammar A, Chinesta F, Diez P, Huerta A (2010) An error estimator for separated representations of highly multidimensional models. *Comput Methods Appl Mech Eng* 199:1872–1880
9. Ammar A, Normandin M, Chinesta F (2010) Solving parametric complex fluids models in rheometric flows. *J Non-Newton Fluid Mech* 165:1588–1601
10. Ammar A, Cueto E, Chinesta F (2012) Reduction of the chemical master equation for gene regulatory networks using proper generalized decompositions. *Int J Numer Methods Biomed Eng* 28(9):960–973
11. Ammar A, Cueto E, Chinesta F Non-incremental PGD solution of parametric uncoupled models defined in evolving domains. *Int J Numer Methods Eng*. doi:10.1002/nme.4413
12. Barrault M, Maday Y, Nguyen NC, Patera AT (2004) An “empirical interpolation” method: application to efficient reduced-basis discretization of partial differential equations. *C R Math* 339(9):667–672

13. Bellomo N (2008) Modeling complex living systems. Birkhäuser, Basel
14. Bernoulli Ch (1836) Vademecum des Mechanikers. Cotta, Stuttgart
15. Bialecki RA, Kassab AJ, Fic A (2005) Proper orthogonal decomposition and modal analysis for acceleration of transient FEM thermal analysis. *Int J Numer Methods Eng* 62:774–797
16. Bird BB, Curtiss CF, Armstrong RC, Hassager O (1987) Dynamics of polymeric liquids. In: Kinetic theory, vol 2. Wiley, New York
17. Bognet B, Leygue A, Chinesta F, Poitou A, Bordeu F (2012) Advanced simulation of models defined in plate geometries: 3D solutions with 2D computational complexity. *Comput Methods Appl Mech Eng* 201:1–12
18. Bordeu F, Leygue A, Modesto D, Gonzalez D, Cueto E, Chinesta F Real-time simulation techniques for augmented learning in science and engineering higher education. A PGD approach. *Arch Comput Methods Eng*, submitted
19. Bui-Thanh T, Willcox K, Ghattas O, Van Bloemen Waanders B (2007) Goal-oriented, model-constrained optimization for reduction of large-scale systems. *J Comput Phys* 224(2):880–896
20. Burkardt J, Gunzburger M, Lee H-C (2006) POD and CVT-based reduced-order modeling of Navier-Stokes flows. *Comput Methods Appl Mech Eng* 196:337–355
21. Cancès E, Defranceschi M, Kutzelnigg W, Le Bris C, Maday Y (2003) Computational quantum chemistry: a primer. Handbook of numerical analysis, vol X. Elsevier, Amsterdam, pp 3–270
22. Chaturantabut S, Sorensen DC (2010) Nonlinear model reduction via discrete empirical interpolation. *SIAM J Sci Comput* 32:2737–2764
23. Chinesta F, Ammar A, Cueto E (2010) Proper generalized decomposition of multiscale models. *Int J Numer Methods Eng* 83(8–9):1114–1132
24. Chinesta F, Ammar A, Cueto E (2010) Recent advances and new challenges in the use of the proper generalized decomposition for solving multidimensional models. *Arch Comput Methods Eng* 17(4):327–350
25. Chinesta F, Ammar A, Leygue A, Keunings R (2011) An overview of the proper generalized decomposition with applications in computational rheology. *J Non-Newton Fluid Mech* 166:578–592
26. Chinesta F, Ladeveze P, Cueto E (2011) A short review in model order reduction based on proper generalized decomposition. *Arch Comput Methods Eng* 18:395–404
27. Chinesta F, Leygue A, Bognet B, Ghnatios Ch, Poulhaon F, Bordeu F, Barasinski A, Poitou A, Chatel S, Maison-Le-Poec S (2012) First steps towards an advanced simulation of composites manufacturing by automated tape placement. *Int J Mater Forming*. doi:10.1007/s12289-012-1112-9
28. Cochelin B, Damlé N, Potier-Ferry M (1994) The asymptotic numerical method: an efficient perturbation technique for nonlinear structural mechanics. *Rev Eur Elem Finis* 3:281–297
29. Darema F (1994) Engineering/scientific and commercial applications: differences, similarities, and future evolution. In: Proceedings of the second Hellenic European conference on mathematics and informatics. HERMIS, Paris, vol 1, pp 367–374
30. Dennis JE Jr., Schnabel RB (1996) Numerical methods for unconstrained optimization and nonlinear equations. Classics in applied mathematics, vol 16. Society for Industrial and Applied Mathematics (SIAM), Philadelphia. Corrected reprint of the 1983 original
31. Ghnatios Ch, Chinesta F, Cueto E, Leygue A, Breitkopf P, Villon P (2011) Methodological approach to efficient modeling and optimization of thermal processes taking place in a die: application to pultrusion. *Composites, Part A* 42:1169–1178
32. Ghnatios Ch, Masson F, Huerta A, Cueto E, Leygue A, Chinesta F (2012) Proper generalized decomposition based dynamic data-driven control of thermal processes. *Comput Methods Appl Mech Eng* 213:29–41
33. Girault M, Videcoq E, Petit D (2010) Estimation of time-varying heat sources through inversion of a low order model built with the modal identification method from in-situ temperature measurements. *Int J Heat Mass Transf* 53:206–219
34. Gonzalez D, Ammar A, Chinesta F, Cueto E (2010) Recent advances in the use of separated representations. *Int J Numer Methods Eng* 81(5):637–659
35. Gonzalez D, Masson F, Poulhaon F, Leygue A, Cueto E, Chinesta F (2012) Proper generalized decomposition based dynamic data-driven inverse identification. *Math Comput Simul* 82(9):1677–1695
36. Gunzburger MD, Peterson JS, Shadid JN (2007) Reduced-order modeling of time-dependent PDEs with multiple parameters in the boundary data. *Comput Methods Appl Mech Eng* 196:1030–1047
37. <http://www.epractice.eu/en/news/5304734>
38. <http://www.ga-project.eu/>
39. <http://www.humanbrainproject.eu/>
40. <http://www.itfom.eu/>
41. <http://robotcompanions.eu>
42. <http://www.futurict.eu>
43. <http://www.graphene-flagship.eu/>
44. Ladevèze P (1989) The large time increment method for the analyze of structures with nonlinear constitutive relation described by internal variables. *C R Acad Sci Paris* 309:1095–1099
45. Ladevèze P, Nouy A (2002) A multiscale computational method with time and space homogenization. *C R, Méc* 330(10):683–689
46. Ladevèze P, Nouy A, Loiseau O (2002) A multiscale computational approach for contact problems. *Comput Methods Appl Mech Eng* 191(43):4869–4891
47. Ladevèze P, Nouy A (2003) On a multiscale computational strategy with time and space homogenization for structural mechanics. *Comput Methods Appl Mech Eng* 192(28–30):3061–3087
48. Ladevèze P, Néron D, Gosselet P (2007) On a mixed and multiscale domain decomposition method. *Comput Methods Appl Mech Eng* 96:1526–1540
49. Ladevèze P, Passieux J-C, Néron D (2010) The Latin multiscale computational method and the proper generalized decomposition. *Comput Methods Appl Mech Eng* 199(21–22):1287–1296
50. Ladevèze P, Chamoin L (2011) On the verification of model reduction methods based on the proper generalized decomposition. *Comput Methods Appl Mech Eng* 200:2032–2047
51. Lamari H, Ammar A, Cartraud P, Legrain G, Jacquemin F, Chinesta F (2010) Routes for efficient computational homogenization of non-linear materials using the proper generalized decomposition. *Arch Comput Methods Eng* 17(4):373–391
52. Lamari H, Ammar A, Leygue A, Chinesta F (2012) On the solution of the multidimensional Langer’s equation by using the proper generalized decomposition method for modeling phase transitions. *Model Simul Mater Sci Eng* 20(1):015007
53. Le Bris C, Lelièvre T, Maday Y (2009) Results and questions on a nonlinear approximation approach for solving high-dimensional partial differential equations. *Constr Approx* 30:621–651
54. Leygue A, Verron E (2010) A first step towards the use of proper general decomposition method for structural optimization. *Arch Comput Methods Eng* 17(4):465–472
55. Leygue A, Chinesta F, Beringhier M, Nguyen TL, Grandidier JC, Pasavento F, Schrefler B Towards a framework for non-linear thermal models in shell domains. *Int J Numer Methods Heat Fluid Flow*. doi:10.1108/09615531311289105
56. Maday Y, Ronquist EM (2002) A reduced-basis element method. *C R Acad Sci Paris, Ser I* 335:195–200
57. Maday Y, Patera AT, Turinici G (2002) A priori convergence theory for reduced-basis approximations of single-parametric elliptic partial differential equations. *J Sci Comput* 17(1–4):437–446

58. Maday Y, Ronquist EM (2004) The reduced basis element method: application to a thermal fin problem. *SIAM J Sci Comput* 26(1):240–258
59. Néron D, Ladevèze P (2010) Proper generalized decomposition for multiscale and multiphysics problems. *Arch Comput Methods Eng* 17(4):351–372
60. Niroomandi S, Alfaro I, Cueto E, Chinesta F (2008) Real-time deformable models of non-linear tissues by model reduction techniques. *Comput Methods Programs Biomed* 91:223–231
61. Niroomandi S, Alfaro I, Cueto E, Chinesta F (2010) Model order reduction for hyperelastic materials. *Int J Numer Methods Eng* 81(9):1180–1206
62. Niroomandi S, Alfaro I, Cueto E, Chinesta F (2012) Accounting for large deformations in real-time simulations of soft tissues based on reduced order models. *Comput Methods Programs Biomed* 105:1–12
63. Niroomandi S, Alfaro I, Gonzalez D, Cueto E, Chinesta F (2012) Real time simulation of surgery by reduced order modelling and X-FEM techniques. *Int J Numer Methods Biomed Eng* 28(5):574–588
64. Nouy A (2010) Proper generalized decompositions and separated representations for the numerical solution of high dimensional stochastic problems. *Arch Comput Methods Eng* 17:403–434
65. NSF Final Report (2006) DDDAS Workshop 2006, Arlington, VA, USA
66. Oden JT, Belytschko T, Fish J, Hughes TJR, Johnson C, Keyes D, Laub A, Petzold L, Srolovitz D, Yip S (2006) Simulation-based engineering science: revolutionizing engineering science through simulation. NSF Blue Ribbon Panel on SBES
67. Park HM, Cho DH (1996) The use of the Karhunen-Loève decomposition for the modelling of distributed parameter systems. *Chem Eng Sci* 51:81–98
68. Passieux J-C, Ladevèze P, Néron D (2010) A scalable time-space multiscale domain decomposition method: adaptive time scale separation. *Comput Mech* 46(4):621–633
69. Pruliere E, Ferec J, Chinesta F, Ammar A (2010) An efficient reduced simulation of residual stresses in composites forming processes. *Int J Mater Forming* 3(2):1339–1350
70. Pruliere E, Chinesta F, Ammar A (2010) On the deterministic solution of multidimensional parametric models by using the proper generalized decomposition. *Math Comput Simul* 81:791–810
71. Rozza G, Huynh DBP, Patera AT (2008) Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations—application to transport and continuum mechanics. *Arch Comput Methods Eng* 15(3):229–275
72. Ryckelynck D, Hermanns L, Chinesta F, Alarcon E (2005) An efficient a priori model reduction for boundary element models. *Eng Anal Bound Elem* 29:796–801
73. Ryckelynck D, Chinesta F, Cueto E, Ammar A (2006) On the a priori model reduction: overview and recent developments. *Arch Comput Methods Eng* 13(1):91–128
74. Schmidt F, Pirc N, Mongeau M, Chinesta F (2011) Efficient mould cooling optimization by using model reduction. *Int J Mater Forming* 4(1):71–82
75. Various authors (2006) Final report. DDDAS workshop 2006 at Arlington, VA, USA Technical report, National Science Foundation
76. Veroy K, Patera A (2005) Certified real-time solution of the parametrized steady incompressible Navier-Stokes equations: rigorous reduced-basis a posteriori error bounds. *Int J Numer Methods Fluids* 47:773–788
77. Videcoq E, Quemener O, Lazard M, Neveu A (2008) Heat source identification and on-line temperature control by a branch eigenmodes reduced model. *Int J Heat Mass Transf* 51:4743–4752