INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2016; **00**:1–21
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/nme

Local Proper Generalized Decomposition

A. Badías[1], D. González[1], I. Alfaro[1], F. Chinesta[2] and E.Cueto[1,*]

[1]*Aragon Institute of Engineering Research, Universidad de Zaragoza, Zaragoza, Spain.*
[2] *Institute of High-Performance Computing, ICI, Ecole Centrale de Nantes, France.*

SUMMARY

One of the main difficulties a reduced order method could face is the poor separability of the solution. This problem is common to both *a posteriori* model order reduction (Proper Orthogonal Decomposition, Reduced Basis) and *a priori* (Proper Generalized Decomposition) model order reduction. Early approaches to solve it include the construction of local reduced order models in the framework of POD. We present here an extension of local models in a PGD —and thus, *a priori*— context. Three different strategies are introduced to estimate the size of the different patches or regions in the solution manifold where PGD is applied. As will be noticed, no *gluing* or special technique is needed to deal with the resulting set of local reduced order models, in contrast to most POD local approximations.
The resulting method can be seen as a sort of *a priori* manifold learning or non-linear dimensionality reduction technique. Examples are shown that demonstrate pros and cons of each strategy for different problems.

Received . . .

KEY WORDS: local model order reduction, proper generalized decomposition, kernel principal component analysis, non linear dimensionality reduction.

1. INTRODUCTION

Computer-based modeling and simulation in applied science and technology has become one of the most valuable and helpful tools in design, development and prediction in the world. The possibility of simulating the behavior of virtually any physical system allows us to understand its response ahead of time. Some big disciplines like economics, biology, sociology or engineering are indeed examples of successful application of computer simulation. Today researches in new techniques of simulation will surely be the basic tools used in next years due the fast evolution of the field and the high requirements of the users. Fast adaptation and continuous development of new models and techniques are some known features in simulation, among other things, by the high quantity of people involved.

The joint between human thirst for knowledge and the increased computing power of today computers has triggered a new revolution in simulation that allows us to obtain the behavior of any system involving high quantities of parameters. But our wishes have no limits, and researchers desire to manipulate large data sets to obtain models with a high degree of similarity with the real behavior, something that not always can be satisfied with nowadays computers (nor presumably, any existing computer ever [1]).

Model Order Reduction (MOR) [2–6] encompasses a group of techniques to reduce the complexity of models in the field of computational mechanics. In an algebraic way, it transforms the

*Correspondence to: Elías Cueto. Edificio Betancourt. Universidad de Zaragoza. María de Luna, s.n. E-50018 Zaragoza, Spain. E-mail: ecueto@unizar.es

solution of a given problem from the original basis to a new and reduced one, preserving as many energy of the system as possible. Traditionally, this is known as transforming the solution from the high-dimensional problem to the reduced basis problem by keeping only the relevant information. A classical categorization distinguishes *a posteriori* from *a priori* methods:

- The *a posteriori* methods are built after computing some solutions of the system or the whole set of solutions, and are especially useful when the model has a relatively small number of parameters but many observations. The most classical example is Proper Orthogonal Decomposition (POD) [7], that is based on a statistical procedure called Principal Component Analysis (PCA) [8]. Other examples include the Reduced Basis Method (RB) [9].
- The *a priori* methods are built without the need of computing any solution to the problem. However, avoiding the computation of *snapshots* implies the need for solving a sequence of non-linear problems. A representative example of this kind of methods is the Proper Generalized Decomposition (PGD).

Although both POD and PGD are already well established techniques, we make here a brief overview of the last one for completeness. The interested reader could consult [10–12] for recent surveys on the method.

PGD assumes as a basic principle that the solution of the governing equation can be approximated as a finite sum of products of functions depending only on one variable. Let us assume that we want to solve a problem stated by its governing differential equation with D coordinates/parameters and that the solution admits a separated representation

$$u(x_1, x_2, \dots, x_D) \approx \sum_{i=1}^{N} F_i^1(x_1) \cdot F_i^2(x_2) \cdot \ldots \cdot F_i^D(x_D), \tag{1}$$

where N is the number of sums of functional *modes* and u is the solution of the equation depending on D independent variables. These D variables can be physical space variables like cartesian coordinates or parameters like Young's modulus or Poisson's ratio in linear elasticity. The particular form of the *modes* $F_i^j(x_j)$ is found in a greedy framework in which one sum of the approximation is computed at a time. Within each sum, to found a product of functions (expressed on a finite element basis) any linearization scheme can be used. We usually employ fixed point iterations because of its conceptual simplicity.

Thus, PGD can be used with two different purposes: as a reduction method (by controlling the number of terms in the approximation, N) and as a parametric *vademecum* generator [13] that can be employed in real time (and also, of course, as a combination of both [14]), being actually the same concept but observed from two different perspectives.

PGD has demonstrated to be specially efficient in the solution of high-dimensional or parametric problems [15] [16] [17] [18]. But it has also some limitations that we have to keep in mind:

- On one hand, some problems have, by definition, a solution structure which is intrinsically non-separable. According to Fourier's decomposition [19] any periodic function can be decomposed in a finite (or infinite) sum of a set of simple functions. We can extrapolate this approximation to a non-periodic function, and also to the PGD method with negligible loss of accuracy. So we can approximate a non-separable solution as a sum of the product of separable functions, although the number of required terms can be really high.
- On the other hand, PGD is equivalent to POD —and thus provides optimal modes with respecto to the chosen norm— when solving elliptic differential equations up to two dimensions [20]. For parabolic and hyperbolic differential equations, or elliptic equations with more than 2 dimensions, PGD method is in general able to find a solution but there is not evidence enough to assure that the solution is optimal. The modes thus obtained are no longer orthogonal nor contain the maximum amount of the energy for a prescribed number of modes. But this is not only a PGD problem. Methods based upon POD are in a similar situation, since high order singular value decomposition (HOSVD [21]), one method for computing the Tucker3 decomposition [22], is not optimal for 3rd or more order tensors [23].

DOI: 10.1002/nme

The origin of the poor separability of the solution relies in the manifold structure of the solution. Many MOR methods look for a suitable basis set in order to project the solution. In general, this basis, even if it is globally optimal, can lead to very poor results if the manifold structure of the solution is very intricate. Previous works in the field include [24]. To avoid these problems, the idea of local basis arises naturally. Initial works were done in the framework of POD-based methods [25–28]. After that, PCA was used also in its local basis approximation with data analysis [29], pattern recognition [30] and image interpolation [31] purposes. Kambahtla *et al.* worked also with local PCA with clustering techniques to determine the local regions and they compared the results with neural network implementations of nonlinear PCA [32].

Some works are based in hyper reduction methods [33] on local reduced-order bases [34]. They make use of an *offline-online* scheme to precompute the local bases on the *offline* stage using POD and obtaining the particular solution in the *online* stage by exploiting selectively the local bases. Other model order reduction methods project over a restricted subset of the spatial domain [35] or use weighted functions to determine the most relevant *snapshots* [36]. This last work also established an interesting classification of the POD method into global POD, local POD and adaptive POD. Global POD projects the original system onto a subspace considering all the snapshots of the domain, local POD projects a local region of the system onto a subspace considering only the snapshots in the subdomain, and adaptive POD uses the global domain to create adaptive reduced bases making use of interpolation methods [37]. Of course, in addition to POD, this classification can be used with any other MOR techniques.

The difficulty of using PGD in a local context relies precisely in its *a priori* character, i.e., in the fact that we have absolutely no information on the manifold structure of the solution. In this paper we present a method in which we adaptively unveil this manifold structure on the fly, without the need for previous sampling of the solution.

The structure of the paper has been divided in four sections after this introduction. Section 2 describes PGD briefly and introduces some well-known cases where PGD encounters difficulties in the resolution. This is the case, for instance, of the so-called *Idelsohn's Benchmark*, a transient heat equation with moving source [38].

This paper shows an analysis of the proposed local PGD (hereafter, ℓ-PGD) method and three different approaches to it, where two of them take advantage of the known information at each moment. Section 3 introduces three examples to validate the method: a first example of 2D linear elasticity with moving loads, a second example solving the *Idelsohn's Benchmark* and a third example solving the *Idelsohn's Benchmark* adding the conductivity as a parameter in the PGD formulation. Finally an analysis with the conclusions of the work is covered in Section 3.

2. LOCAL PGD (ℓ-PGD)

PGD assumes a separated variable structure of the solution that may confront with some problems called *non-separable*. As we have seen in Section 1, some physical phenomena have a solution structure that does not fit with a separate variable expression, and PGD may need too many modes to obtain an accurate enough solution with acceptable error rates.

2.1. Difficulties in the reduction process

The sources of non-separability of the solution are manifold. Following [5], we could mention

- The global dimension of the problem. At a given space-time(-parameter) point at the solution manifold we can approximate the solution by the tangent hyperplane, which is in turn spanned by a basis vector of dimension equal to that of the space plus that of the parameter vector and time. However, to approximate well *all* the solution manifold, the number of vectors in the basis may be much higher.
- It also depends on the approximability of the solution manifold by n-dimensional subspaces, the so-called Kolmogorov n-width. It is specially high in the case of the already mentioned Idelsohn's problem, Eq. (2).

DOI: 10.1002/nme

- Differentiability and regularity of the solution map.
- Parametric complexity, i.e., to what extent the solution is affine in the parametric space, thus admitting a separated expression like Eq. (1).

Several methods have been tried so far. One solution would be to employ space-time (thus, non-separated) basis functions. An example is the work of Gerbeau [39], that employed the concept of Lax Pairs [40] to obtain a reduced model with a special application to wave and soliton problems.

In general, we have few options to try to minimize the effect of the poor regularity of the solution map or the parametric complexity. However, there are some options to work with the two first sources of poor separability mentioned before, namely the global dimension of the problem and the high Kolmogorov n-width. If the local dimension of the problem is much lower than the global one, one could reasonably expect that the employ of local PGD approximations could improve the results.

In what follows we discuss three different strategies for the construction of local PGD (ℓ-PGD) approximations to the problem at hand.

2.2. *Constant local partitions (Cℓ-PGD)*

Consider, to begin with, an equation depending solely on space and time, without any parametric dependence. The most naive approach to the problem is to work by partitioning the time domain only, and doing it uniformly. Each resulting two-dimensional sub-region is therefore formed by the same number of nodes both in spatial and temporal axes. In this way, we obtain a local basis for every reduced sub-domain, and a global solution in separate variables where the spatial modes are supported in the entire domain Ω. Time modes, on the contrary, span only a local sub-domain $\mathcal{I}_i := [t_i^0,\, t_i^{\text{end}}]$.

No special procedure is needed to guarantee the continuity of the solution, even if time modes could be discontinuous, see Fig. 16a for an example corresponding to Eq. (2). This example is discussed thoroughly in Section 3.2. The only requirement to guarantee continuity is to take the final temperature values at interval $\mathcal{I}_i$ as initial conditions of the $\mathcal{I}_{i+1}$ interval. Therefore, no special procedure is needed to achieve continuity. The proposed method can be seen as a sequence of PGD problems with no special treatment of initial nor boundary conditions. All the details concerning the imposition of non-homogeneous initial and boundary conditions were already studied at one of the early papers on PGD, see [41].

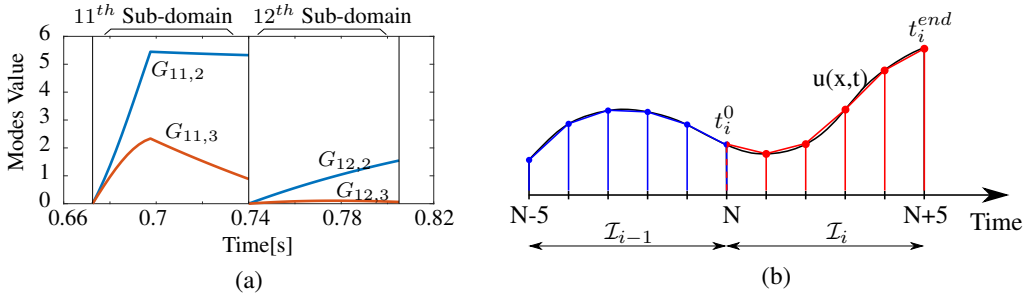


Figure 1. a) Time modes $G_i(t)$ may be discontinuous along time (no partition is made here on the space domain). However, the resulting solution, b), is made continuous by only imposing the temperature field at the end of sub-domain $\mathcal{I}_i$ as initial condition of the subsequent time interval $\mathcal{I}_{i+1}$.

The partition of the domain into constant size sub-regions allows PGD to capture the behavior of the solution in a faster and accurate way in some problems —as will be noticed in Section 3—, but we are not fully exploiting the potential of ℓ-PGD. The number of necessary modes will be highly variable for each subdomain, in general. We could perform a more efficient domain partitioning by employing variable subdomains sizes. Solution manifolds with highly intricate geometries can be described in a better way by using local domains with varying sizes. For example, solution regions with small variation in the original manifold can be approximated with bigger sub-domains ($\mathcal{T}_3(\mathcal{M})$

DOI: 10.1002/nme

in Fig. 2) than regions with higher variation in the original manifold that may need a higher number of modes ($\mathcal{T}_4(\mathcal{M})$ and $\mathcal{T}_5(\mathcal{M})$ in Fig. 2).

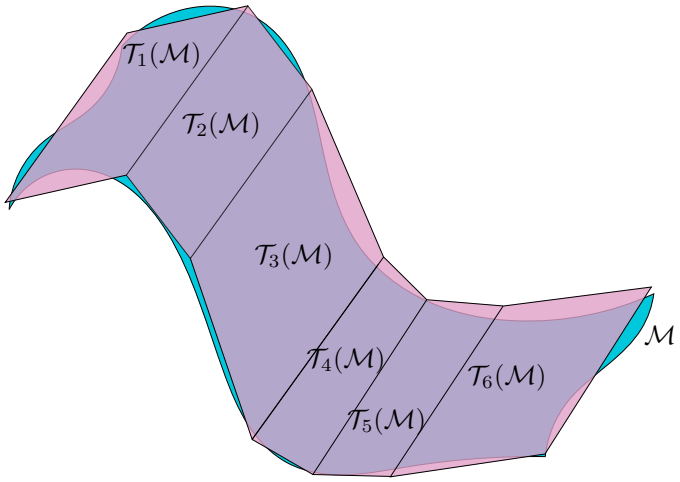


Figure 2. Manifold $\mathcal{M}$ where the solution lies, and tangent hyper-planes $\mathcal{T}_i(\mathcal{M})$ corresponding to local basis approximation.

In order to overcome the already mentioned limitations of constant partitioning, we propose the use of two different alternatives producing an adaptive partitioning of the domain in a smart way. The first one is an iterative method similar to the one proposed in [42] for time domain partitioning. This method computes the size of the subdomain iteratively by imposing equal number of modes within each subdomain and a specific error tolerance. The second method uses Kernel Principal Component Analysis projections (k-PCA, [43]) of the already computed solution to unveil the manifold structure of the already computed part of the solution, and to suggest local divisions depending on the degree of regularity of the manifold.

2.3. ℓ-PGD adaptive partitioning with mode number optimization (MNOℓ-PGD)

Let us assume that the solution of a multidimensional problem remains on a manifold $\mathcal{M}$ as it is shown in Fig. 2. An adaptive partitioning allows us to better describe the manifold $\mathcal{M}$ by projecting the solution onto local bases $\mathcal{T}_i(\mathcal{M})$ that may be thought of as hyperplanes tangent to the manifold. The distance between tangency points may be non constant, since the solution manifold may have an irregular variability.

For space-time problems, we will work on local regions formed by the full spatial domain and a target number of M local partitions in time. Local time-partitioning will be modified iteratively and PGD solved in each step. We start with a seed number p of time nodes, optimizing p to allow the number of local modes to be as close as possible to the sought amount M, which has been fixed previously, and an error tolerance E. Given a number of local modes N_i required to build the solution on a sub-domain i, and assuming a space and time discretization Δx and Δt both changeless, since we can only modify the number of local time nodes T_i, it is possible that the number of modes to obtain M may not be achievable, $N_i \neq M$. We decided to choose the stopping number of modes N_i to be below and closest to the optimal, being $N_i \leq M \; \forall \, i \in [1, k]$ to ensure the maximum number of local modes is at most M, i.e.,

$$\max_i N_i \leq M.$$

The implementation of the adaptive time-partitioning method with mode number optimization is shown in Algorithm 1. We have employed unitary increments and decrements of the number of local modes T_i within each algorithm iteration to assure its simplicity and robustness, but other techniques may be used to accelerate the convergence.

DOI: 10.1002/nme

The difficulty increases if we add parameters to the PGD separated formulation. This disadvantage led us to think in a third method able to extract information about the manifold structure of the solution, so as to allow us to perform adaptive partitions on the fly.

Algorithm 1: Pseudo-code to implement the adaptive partitioning MNOℓ-PGD at a subdomain i.

```
AllowedModes = M;
while LastTimeInstantOfLastPartition < GlobalTimeEnd do
    InitialLocalPartitioning(i);
    while (LocalModesObtained(i) ≠ AllowedModes) & (LoopOut == 0) do
        [SpaceModes(i), TimeModes(i), LocalModesObtained(i)] = LocalPGDImplementation(i);
        if LocalModesObtained(i) == AllowedModes then
            break;
        else if LocalModesObtained(i) ≠ AllowedModes then
            UpdateLocalPartitioning(i);
        end
        if AllowedModesCanNeverBeReached then
            UpdateLocalPartitioning(i);
            LoopOut = 1;
        end
    end
    Solution(i) = SpaceModes(i) * TimeModes(i)';
    SetInitialConditions(i + 1);
    i ← i + 1;
end
```

2.4. kernel-*PCA-based ℓ-PGD ($k\ell$-PGD)*

The application of PGD locally can be carried out in multiple ways. There could be eventually many other methods to find the number of local modes depending on the available information. But our goal remains the same, estimating the size of each local sub-domain to obtain a separated representation of the solution with a minimal amount of terms and, therefore, computational time and storage costs.

With this third method we intend to use manifold learning tools to unveil the manifold structure of the solution. The difficulty, however, remains to be the *a priori* character of PGD. Since PGD starts the approximation to the solution from scratch, without any sampling of the parametric space, we have no information on the manifold structure of the solution. Instead, we start by performing an arbitrary partitioning. We will employ k-PCA methods [44] to check the appropriateness of this partitioning and, eventually, correct it iteratively.

k-PCA methods work, surprisingly, by projecting data onto a higher dimensional (eventually, infinite dimensional) space where data is exactly separable. In this space, standard PCA techniques can be applied. Then, data can be projected back onto the principal components and the first few, more relevant projections, retained. To obtain a better understanding of the manifold structure of the solution, we will measure the distance between different parameter-time-dependent solutions by resorting to the concept of Hausdorff distance [45].

2.4.1. Kernel Principal Component Analysis. As stated before, the idea behind k-PCA methods may seem surprising, but the elegance of the method comes from its conceptual simplicity: data not linearly separable in D dimensions, could be linearly separated if previously projected to a space in $Q > D$ dimensions [46, 47]. Thus, k-PCA begins by projecting the data to an even higher dimensional space. To do it, a mapping is necessary:

$$\phi: \ \mathcal{M} \subset \mathbb{R}^D \to \mathbb{R}^Q, \ \boldsymbol{y} \to \boldsymbol{z} = \phi(\boldsymbol{y}),$$

DOI: 10.1002/nme

where Q may be any dimension. One of the biggest advantages of this tool is that there is no need to explicitly determine the analytical expression of the mapping ϕ. This is known as the *kernel trick* in the machine learning community.

The resulting symmetric matrix $\boldsymbol{\Phi} = \boldsymbol{Z}^T\boldsymbol{Z}$ needs to be decomposed in eigenvalues and eigenvectors. In addition, the mapped data $\boldsymbol{z}_i$ involved in $\boldsymbol{\Phi}$ must be previously centered. The centering can be performed even if the mapping is unknown, in an implicit way by performing the so-called double centering.

Let us denote the mean of the j-th column of $\boldsymbol{\Phi}$ as $\mu_i(\boldsymbol{z}_i \cdot \boldsymbol{z}_j)$, and the mean of its i-th row as $\mu_j(\boldsymbol{z}_i \cdot \boldsymbol{z}_j)$. The mean of all entries of $\boldsymbol{\Phi}$ reads $\mu_{i,j}(\boldsymbol{z}_i \cdot \boldsymbol{z}_j)$. The double centering process gives

$$\boldsymbol{z}_i \cdot \boldsymbol{z}_j - \mu_i(\boldsymbol{z}_i \cdot \boldsymbol{z}_j) - \mu_j(\boldsymbol{z}_i \cdot \boldsymbol{z}_j) + \mu_{i,j}(\boldsymbol{z}_i \cdot \boldsymbol{z}_j).$$

The mentioned eigenvalue-eigenvector decomposition can be performed on the doubly centered matrix, leading to

$$\boldsymbol{\Phi} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T.$$

It is important to realize that the mapping ϕ needs not to be evaluated in the computation of the matrix $\boldsymbol{\Phi}$. If the number of dimensions of the target space, Q, is high (even infinite), the explicit computation of its entries may be prohibitive. The kernel trick allows us to overcome this difficulty by founding a kernel function κ that directly gives the value of the scalar product $\kappa(\boldsymbol{y}_i, \boldsymbol{y}_j) = \boldsymbol{z}_i \cdot \boldsymbol{z}_j$. This property follows from Mercer's theorem [46, 47].

Many different kernels fulfilling Mercer's condition exist. Among them, the most popular ones are:

- Polynomial kernels: $\kappa(\boldsymbol{u}, \boldsymbol{v}) = (\boldsymbol{u} \cdot \boldsymbol{v} + 1)^p$, with p an arbitrary integer;
- Gaussian kernels: $\kappa(\boldsymbol{u}, \boldsymbol{v}) = \exp\left(-\frac{\|\boldsymbol{u}-\boldsymbol{v}\|^2}{2\sigma^2}\right)$ for a real σ;
- Sigmoid kernels: $\kappa(\boldsymbol{u}, \boldsymbol{v}) = \tanh(\boldsymbol{u} \cdot \boldsymbol{v} + b)$ for a real b.

In this work, and without loss of generality, we have employed Gaussian kernels. These provided excellent results in the determination of the manifold structure of the solution. Actually, what we pursue with the application of k-PCA is to determine the true *distance* between different parameter-specific solutions in the manifold. In other words, to unveil how intricate the manifold is.

Note that the use of k-PCA methods implies loosing the *a priori* character of PGD in some sense. k-PCA is indeed an *a posteriori* method, which works on a set of snapshots of the problem. However, as will be clear in the results section, the method is still appealing since it minimizes the amount of CPU time and memory storage with respect to classical PGD methods.

2.4.2. Hausdorff distance.

Our goal is to determine how far each space-time(-parameter) particularization of the solution is from its neighboring particularizations. In other words, to unveil the regularity of the solution manifold. To this end, once the k-PCA structure has been obtained, we determine the Hausdorff distance between particularized solutions. In fact what we look for are actually *trends* in the distance. In other words: to know if the partitioning could be made coarser or, on the contrary, it is necessary to refine it locally.

The Hausdorff distance between two subsets X, Y of a metric space, is defined as

$$d_H(X,Y) = \max\{\sup_{x\in X}\inf_{y\in Y} d(x,y), \sup_{y\in Y}\inf_{x\in X} d(x,y)\}.$$

The Hausdorff distance between particular realizations of the solution reveals the intrinsic variability of the solution. Once the distance between realizations has been determined, an algorithm must be designed to estimate the most appropriate size of the *patches* on which $k\ell$-PGD is applied. To this end, we have chosen the ε-neighborhood tool to cluster the information and extract the limits of local patches.

DOI: 10.1002/nme

2.4.3. ε-neighborhood clustering. The neighborhood of a point p in a metric space $M(X,d)$ is a set V if there exists an open ball with center p and radius $r>0$, such that $B_r(p)$ is contained in V

$$B_r(p) = B(p;r) = \{x \in X \mid d(x,p) < r\}.$$

We consider the ε-neighborhood ε-N_{D_i} of a point p as the set of all points in the real space $\mathbb{R}^D$ that are located at a distance less than ε from p measured in dimension D_i,

$$B(a;\varepsilon) = \{x \in \mathbb{R}^D : |x-p|_{D_i} < \varepsilon\}.$$

In sum, an iterative algorithm has been designed that proceeds by

1. Choosing an arbitrary partition of the space-time(-parameter) space,
2. unveiling its manifold structure by applying k-PGD,
3. measuring the distance between particular time-parameter solutions by means of its Hausdorff distance, and
4. eventually modify the original partition and adapting it to the distance distribution.

This procedure has been schematically represented in Algorithm 2 below.

Algorithm 2: Pseudo-code to implement the *kernel*-PCA-based ℓ-PGD ($k\ell$-PGD).

```
VariablesInitialization;
ConstantLocalSolution = ComputeConstantLocalPGD;
while j < NumberOf-kPCACurves do
    kPCACurves(j) = ComputekPCA(ConstantLocalSolution);
    j ← j + 1;
end
ComputeHausdorffDistance(AllkPCACurves);
LocalkPCADivisions = Clustering ε-neighborhood;
while i < NumberOfLocalkPCADivisions do
    [SpaceModes(i), TimeModes(i)] = ComputeLocalPGD;
    Solution(i) = SpaceModes(i) * TimeModes(i)';
    SetInitialConditions(i+1);
    i ← i + 1;
end
```

In the next Section we present some examples of application of the three different methods and compare their relative performance.

3. EXAMPLES

We introduce here three benchmark examples. ℓ-PGD is applied and results compared with the classical, global PGD, and even the standard, also global, POD method.

The first example considers displacement of a two-dimensional cantilever beam with a moving vertical load along its upper boundary. The second example considers the already mentioned Idelsohn's benchmark. And, finally, the third example is the parametric version of the Idelsohn's benchmark, by adding the conductivity k as a parameter in the PGD formulation. Problems in up to three dimensions have been considered for visualization purposes, but the presented technique is completely general and can be applied to problems in arbitrary dimensions.

3.1. Cantilever beam with a moving load

This problem considers the horizontal and vertical displacement $\boldsymbol{u}(x,y)$ of any point of a bi-dimensional cantilever beam Ω, Fig. 3, with a moving vertical load F applied at a variable location on the upper boundary $\boldsymbol{s} \in \bar{\Gamma} \subset \partial\Omega$ where $\bar{\Gamma}$ represents the portion of the Neumann boundary Γ_t with non-vanishing natural conditions.

DOI: 10.1002/nme

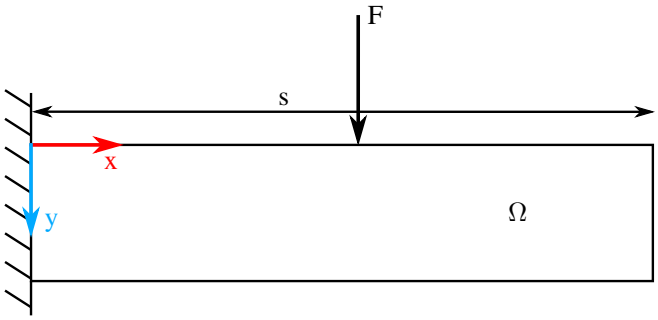


Figure 3. Cantilever beam with a moving vertical load F.

The governing equation for linear elasticity in its strong form is well known [48]: *find* $\boldsymbol{u}(\boldsymbol{x})$ *such that*

$$\begin{cases} \boldsymbol{\nabla}\cdot\boldsymbol{\sigma}+\boldsymbol{b} &= \boldsymbol{0} \quad \text{in } \Omega, \\ \boldsymbol{\sigma}\,\boldsymbol{n} &= \bar{\boldsymbol{t}} \quad \text{on } \Gamma_t, \\ \boldsymbol{u} &= \bar{\boldsymbol{u}} \quad \text{on } \Gamma_u. \end{cases}$$

Since we are interested in finding the parametric solution $\boldsymbol{u}(\boldsymbol{x},s)$, i.e., for any possible load position along $\bar{\Gamma}$, we develop a doubly weak form by integrating not only in Ω, but also on $\bar{\Gamma}$ (details on the Matlab implementation of this problem can be found in [10], see also [11, 12]):

$$\int_{\bar{\Gamma}}\int_{\Omega}\boldsymbol{\nabla}_{\boldsymbol{s}}\boldsymbol{u}^{*}:\boldsymbol{\sigma}\,d\Omega\,d\bar{\Gamma}=\int_{\bar{\Gamma}}\int_{\Gamma_t}\boldsymbol{u}^{*}\,\boldsymbol{t}\,d\Gamma\,d\bar{\Gamma},$$

where $\boldsymbol{u}^{*}\in H_0^1(\Omega)$ is an arbitrary test function and $\boldsymbol{\nabla}_{\boldsymbol{s}}=\frac{1}{2}[\boldsymbol{\nabla}+(\boldsymbol{\nabla})^T]$ is the symmetric gradient operator. Following the standard PGD assumption on the separability of the solution, we postulate the solution $\boldsymbol{u}(\boldsymbol{x},s)$ separated in the following way,

$$\boldsymbol{u}(\boldsymbol{x},s)\approx\sum_{i=1}^{n}\boldsymbol{F}_i(\boldsymbol{x})\circ\boldsymbol{G}_i(s),$$

where "$\circ$" stands for the entry-wise Hadamard or Schur product of vectors, n is the number of terms, $\boldsymbol{F}_i(\boldsymbol{x})$ are the space modes in separate variables and $\boldsymbol{G}_i(s)$ are the modes referred to the position of the load. We consider here s as a scalar, since the boundary of the two-dimensional beam is actually a one-dimensional region.

Standard PGD methods use a greedy algorithm to obtain the n functions $\boldsymbol{F}_i(\boldsymbol{x})$ and $\boldsymbol{G}_i(s)$. Assume that, at iteration p, we look for the $p+1$-th functions $\boldsymbol{R}(\boldsymbol{x})$ and $\boldsymbol{S}(s)$ that produce the enrichment of the solution:

$$\boldsymbol{u}_{p+1}(\boldsymbol{x},s)=\boldsymbol{u}_p(\boldsymbol{x},s)+\boldsymbol{R}(\boldsymbol{x})\circ\boldsymbol{S}(s).$$

Therefore, the test function is

$$\boldsymbol{u}^{*}(\boldsymbol{x},s)=\boldsymbol{R}^{*}(\boldsymbol{x})\circ\boldsymbol{S}(s)+\boldsymbol{R}(\boldsymbol{x})\circ\boldsymbol{S}^{*}(s).$$

The load F has been designed as a unitary force acting along the vertical axis y at a variable position s along the upper boundary of the beam, see Fig. 3,

$$\boldsymbol{t}=F\,\delta(x-s)\boldsymbol{e}_y,$$

where δ represents the Dirac delta function. Function $\boldsymbol{t}$ needs to be expressed in separate form to comply with the PGD method, so

$$t_x=0;\quad t_y\approx\sum_{j=1}^{m}h_j(\boldsymbol{x})\,k_j(s),$$

DOI: 10.1002/nme

where m is the number of sums to approximate function t_y and $h_j(\boldsymbol{x})$, $k_j(s)$ are the functions in space and load position, respectively. We have employed a fixed point algorithm to solve the resulting non-linear problem of finding the pair functions $\boldsymbol{R}(\boldsymbol{x}) \circ \boldsymbol{S}(s)$ at every enrichment iteration, but any other method (noteworthy, Newton) can be used. See [10] for a more precise description of the matrix form of this problem and the implementation code in *MATLAB* language.

In the $C\ell$-PGD framework, uniformly-spaced local partitions have been made in the s domain maintaining the whole space $\boldsymbol{x}$ for every partition. Assuming M partitions in the parameter-space domain, the solution will be at each one of these partitions $u_i(\boldsymbol{x}, s) \in H_1(\Omega_x) \times H_1(\Omega_y) \times L_2(\bar{\Gamma}_i)$ $\forall\ i \in [1, M]$, with $\Omega_x := [0, 9][m]$, $\Omega_y := [0, 1][m]$ and $\bar{\Gamma}_i := [s_i^0, s_i^{end}]$. We employed a mesh with size $\Delta_x = \Delta_y = \Delta_s = 0.1[m]$.

In the $k\ell$-PGD approach, on the contrary, we have applied k-PCA to vectors containing nodal values for the same s position (see Fig. 4b) using the vertical displacement as input. We obtained a set of embedded curves (Fig. 4a) and making use of the Hausdorff distance concept we measured the distance between curves and the origin of the coordinate system so as to estimate the manifold structure of the problem (Fig. 4c). From here, it is possible to divide the s domain in partitions using clustering tools as described in Section 2.4.3.

The resulting vertical displacement field for the load in the end position of the beam is shown in Fig. 5a and Fig. 5b shows the surface obtained for the load applied in any available position.

To determine the accuracy of each proposed local implementation of the PGD, we compare our results with the well-known analytic solution of the 1D Euler-Bernoulli-Navier model of a cantilever beam [49]. Fig. 5c shows the relative error between the analytic solution of the beam with a load applied at the tip and the foloowing techniques: finite element method (FEM), PGD in its global implementation, POD also in its global version, $C\ell$-PGD with constant divisions of domain s and $k\ell$-PGD with variable-sized divisions. Finally, Fig. 5d shows the L_2-norm error computed by comparing the result obtained with each technique against the finite element solution with the same discretization.

It is worth noting that both ℓ-PGD versions provide very similar results, showing that merely two load modes in nine divisions provide with the same error as the finite element solution for that particular location of the load. On the contrary, both global implementations of POD and PGD needed some eighty modes to obtain the same accuracy.

3.2. Idelsohn's benchmark

An already classical example of non separable problems was established by Sergio Idelsohn in a joint Spanish-French workshop held at Jaca, Spain, in 2013. It is based in the transient diffusion equation in one dimension with a moving source. The governing equation and boundary conditions were described in a subsequent technical report by D. Neron and P. Ladeveze [38]:

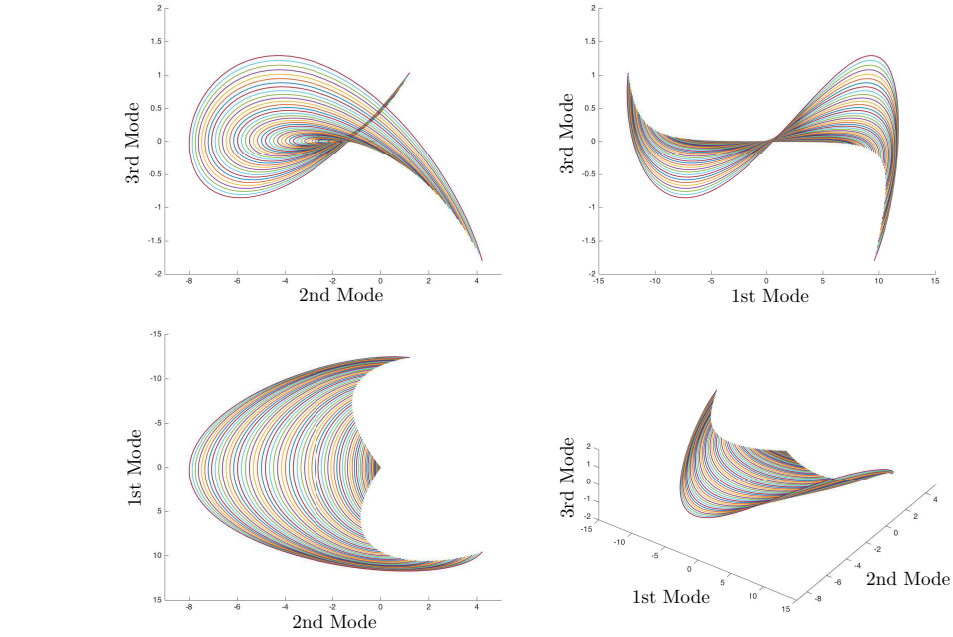
$$\begin{aligned} \rho c_p \partial_t u - \nabla \cdot k \nabla u = f(x,t) \quad & \text{in } \Omega \times \mathcal{I}, \\ u = u_D \quad & \text{on } \Gamma_D \times \mathcal{I}, \\ u = u_0 \quad & \text{on } \Omega \times \{0\}, \end{aligned} \tag{2}$$

with $\Omega := [0, \pi]$, $T = 1$, $u(0,t) = u(\pi,t) = u_D = 0$ and $u(x,0) = u_0 = 0$. The value of the thermal conductivity is $k = 0.05$, density is $\rho = 1$ and the specific heat capacity $c_p = 1$. The source term is described by the equation:

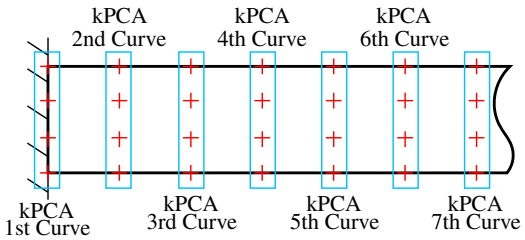
$$f(x,t) = \begin{cases} 0 & t < t_i \ \text{ or } \ t > t_f \\ A \cos\left(\frac{\pi}{L}(x - x_0(t))\right) & t_i < t < t_f \ \text{ and } \ -\frac{L}{2} < x - x_o(t) < \frac{L}{2} \end{cases},$$

where $x_0(t) = x_i + \frac{x_f - x_i}{t_f - t_i}(t - t_i)$, $A = 100$, $L = 0.15$, $t_i = 0.2$, $t_f = 0.7$, $x_i = 2\pi/7$ and $x_f = 5\pi/7$. The source term $f(x,t)$ is the heat added per unit volume. Fig. 6 sketches the problem. In what follows, we consider also the possibility of a parametric problem, where the conductivity k acts itself as a parameter of the problem. This problem has also been analyzed, among others, by Allier *et al.* [50].

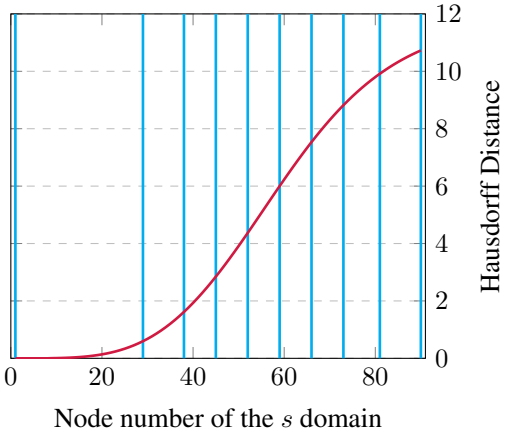
DOI: 10.1002/nme



(a)



(b)



(c)

Figure 4. a) Set of curves obtained after applying k-PCA. b) Sketch of the employed strategy. Red crosses (nodes) are grouped in columns and k-PCA is applied to the resulting vector of nodal displacements. Each column will represent a curve in a). c) Hausdorff distance between k-PCA curves and the origin of the coordinate system. Blue lines indicate the partitions in the s domain.

Idelsohn's benchmark problem has been solved by applying POD and PGD in their global, classical version. These will serve as a reference measure of the degree of reduction attainable by ℓ-PGD.

The temperature distribution of the 1D solid can be plotted in a 3D surface where the first axis is the time coordinate, the second axis comprises physical coordinates of the solid (assumed to be one-dimensional) and the third, vertical, axis shows the temperature distribution in the space-time domain (Fig. 7a). For completeness, we also plot a 2D surface to better appreciate the sharp discontinuity in the solution (Fig. 7b).

The L_2-norm error is computed by comparing the result obtained against the reference finite element solution with the same discretization (i.e., the full-order model). The k-PCA projection of the solution onto the three first eigenvectors is shown in Fig. 8.

DOI: 10.1002/nme

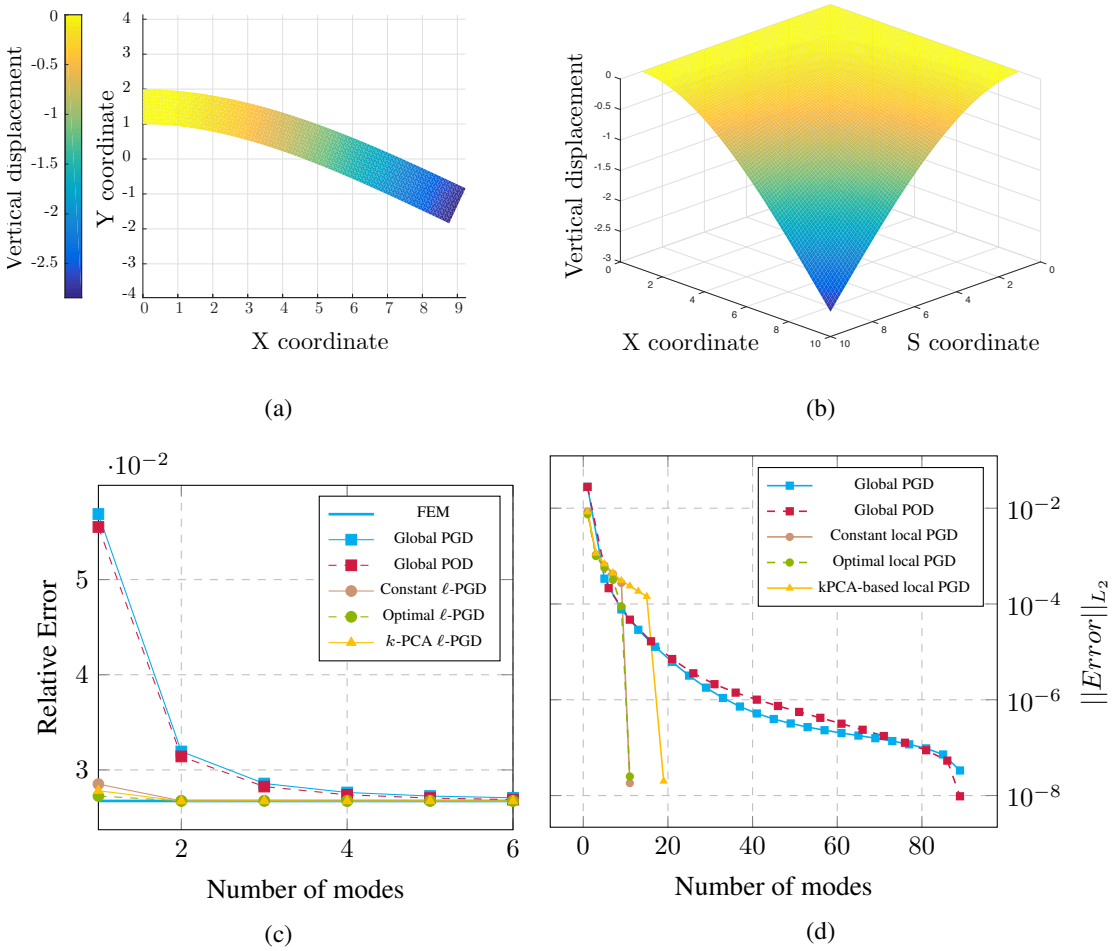


Figure 5. a) Vertical displacement of the cantilever beam solved with FEM with $F = -1[N]$, $E = 1000[N/m^2]$, $\nu = 0.3$, $s = 9[m]$ (this plot corresponds to the case of the load at the end tip). b) Vertical displacement of the cantilever beam solved for every position of the load in the s domain. c) Relative error of the vertical displacement of the end point (load at end tip) for some methods respect to the analytic Euler-Bernoulli-Navier solution. d) L_2-norm error of the vertical displacement for every position of the load in the s domain between each technique and the finite element solution with the same discretization, up to an error $e = 10^{-8}$.

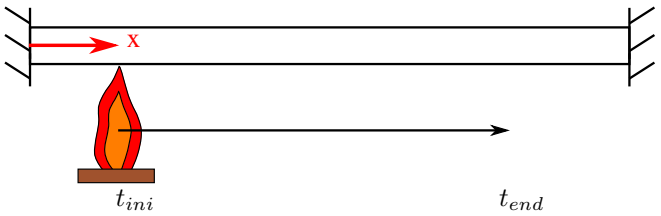


Figure 6. Idelsohn's benchmark in 1D.

Fig. 9 shows the result of the L^2-norm error of the temperature distribution in the space-time surface (Fig. 7). The size of the mesh is N_x=301 space nodes and N_t=401 time nodes. It should be pointed out that local time modes can be arranged (and stored in memory) as global, though discontinuous, time modes. However, since we are taking the full spatial domain to form each sub-domain, spatial modes can not be organized in the same way. This requires more storage space

DOI: 10.1002/nme

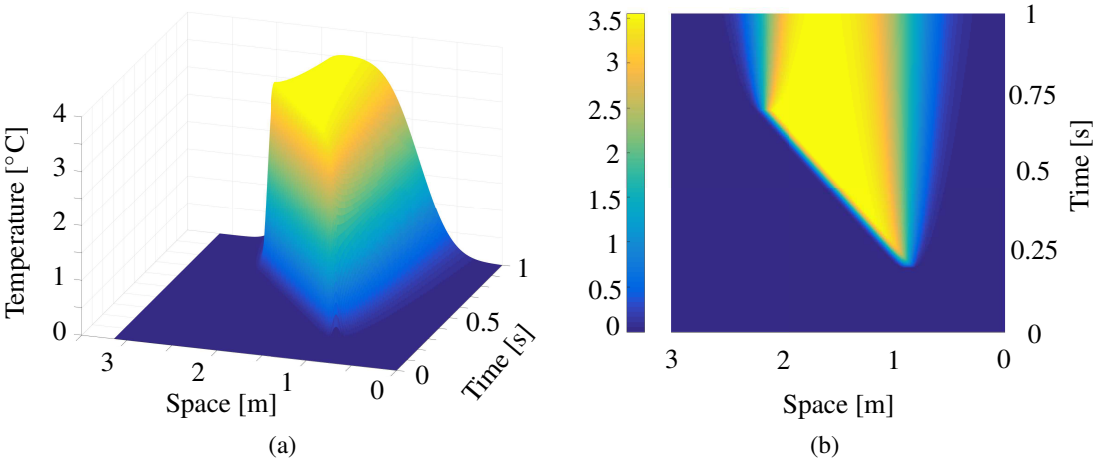


Figure 7. a) 3D surface of the solution of the heat equation with moving source with $xf = 5\pi/7$[m], $k = 0.05$ [W/(m K)]. b) 2D view of the same problem to better appreciate the difficulty of separating a sharp discontinuity running across the diagonal of space-time.

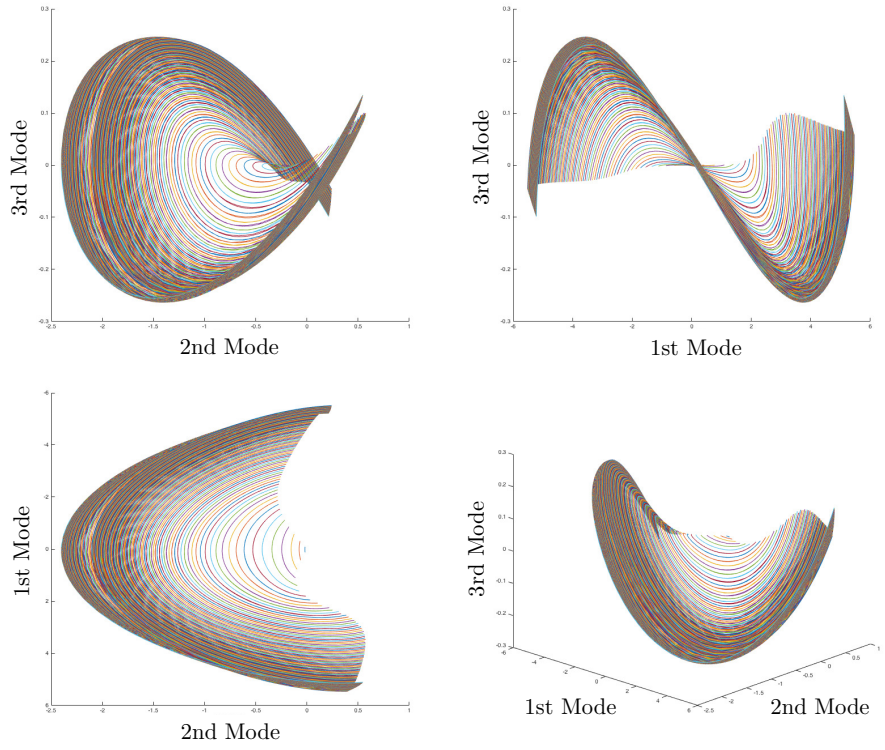


Figure 8. Example of application of k-PCA to the reference solution of the Idelsohn's benchmark. Each curve represents a space-time particularization of the solution to Eq. (2).

when compared with the same number of global modes, but still allows us to obtain a much smaller computation time, as can be shown in Section 3.4.

Fig. 9 shows that ℓ-PGD provides with substantial savings in the number of necessary modes for a prescribed error in the approximation. It is also important, however, to determine the true savings in terms of stored memory or how the number of modes is distributed for each sub-domain. A constant number of modes in all sub-domains denotes that we are using the memory in the best way, and

DOI: 10.1002/nme

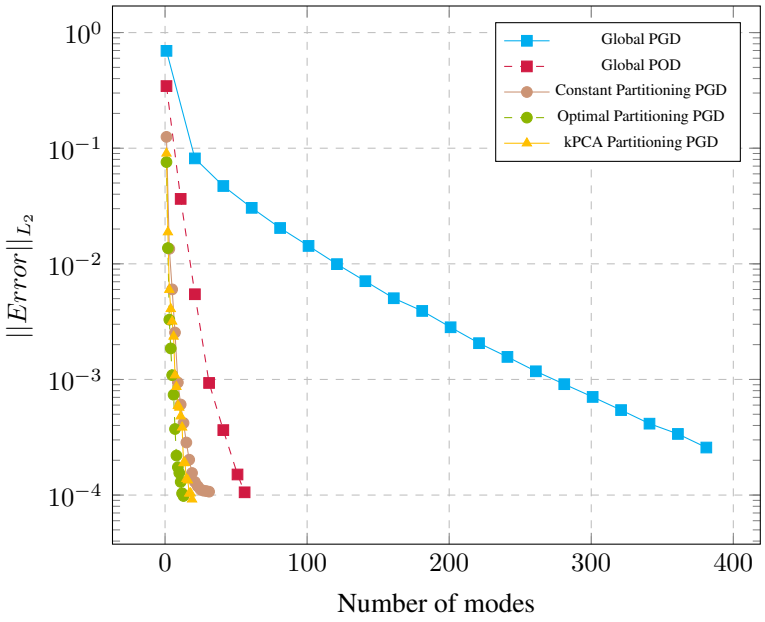


Figure 9. L^2-norm error of the temperature distribution versus the number of modes for global PGD, global POD and ℓ-PGD in its three proposed variants, computed respect FEM solution with the same discretization and up to an error rate of $e = 10^{-4}$.

that the size of the sub-domains is balanced in terms of variability of the manifold structure of the solution. Fig. 10 shows the number of modes per sub-domain for the three ℓ-PGD methods with a value of x_f=2π/7 (Fig. 10a) and x_f=5π/7 (Fig. 10b). The Hausdorff curve that measures the distance between every k-PCA curve and the origin, used with $k\ell$-PGD, is shown in Fig. 11. In this figure we also plot the divisions employed to form each sub-domain in the time axis.

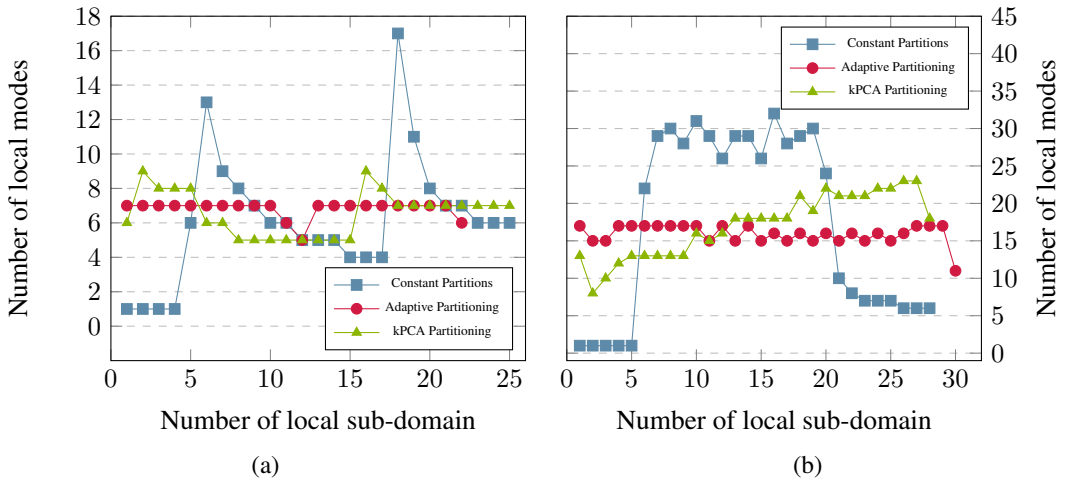


Figure 10. Number of modes per sub-domain for the three ℓ-PGD methods with a) x_f=2π/7 (global PGD needs 211 modes) and b) x_f=5π/7 (global PGD needs 382 modes)

3.3. Parametric transient heat transfer equation

This example shows the addition of the thermal conductivity k into the PGD formulation of the transient heat transfer equation (already considered in Section 3.2) as a parameter. It modifies

DOI: 10.1002/nme

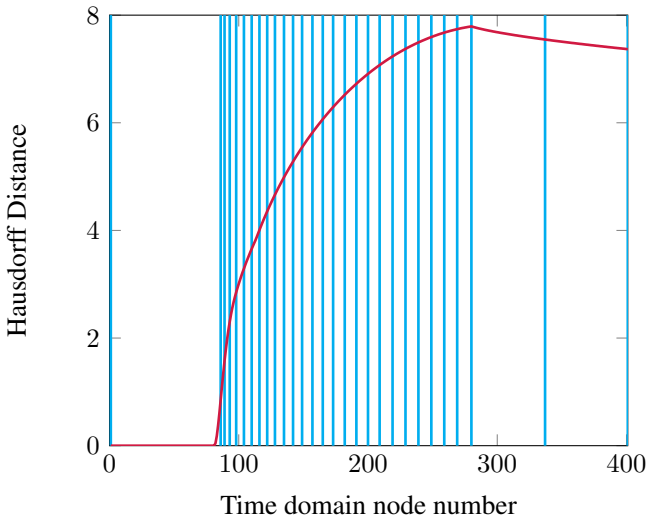


Figure 11. Hausdorff curve measured with respect the coordinate origin in red with sub-domain divisions in blue.

the problem by moving it to a three-dimensional space, where the solution takes now the form $u(x,t,k) \in H_1(\Omega) \times L_2(\mathcal{I}) \times L_2(\mathcal{K}_k)$ where $\mathcal{I},\ \mathcal{K}_k \subset \mathbb{R}$ represent the considered intervals for time and conductivity, respectively. In the spirit of PGD, we assume a separate form for the unknown,

$$u(x,t,k) \approx \sum_{i=1}^{n} F_i(x) \cdot G_i(t) \cdot H_i(k).$$

At iteration $p+1$ we look for an enrichment of the already known approximation at order p (i.e., we employ a greedy algorithm),

$$u_{p+1}(x,t,k) \approx u_p(x,t,k) + R(x) \cdot S(t) \cdot P(k).$$

again, the admissible variation of the solution u^* is

$$u^*(x,t,k) = R^*(x) \cdot S(t) \cdot P(k) + R(x) \cdot S^*(t) \cdot P(k) + R(x) \cdot S(t) \cdot P^*(k).$$

The resulting nonlinear problem is solved by employing a fixed point algorithm.

We present a comparison of the L_2-norm error of the solution for one particular conductivity value (k=0.05 $\frac{W}{mK}$) in Fig. 12.

Note that in this example the Hausdorff distance plot becomes a surface because of the 3-dimensional nature of the problem. One of the advantages of $k\ell$-PGD method is that we can work in a general number of dimensions easily since k-PCA, Hausdorff distances and ε-neighborhoods are well-defined entities in any number of dimensions. The Hausdorff distance plot in this case is shown in Fig. 13. Note that, in general, the distance between results grows monotonically up to $t \approx 0.7$, and then remains constant (for $k = 0.0$) or decreases.

In general, as in previous examples, ℓ-PGD shows much better convergence rates with respect to the global versions of HOSVD and PGD.

3.4. Time and memory cost

It is worth mentioning that the number of obtained modes is not the only relevant variable. Of course, the time employed to obtain the separated representation of the solution (the off-line part of the PGD algorithm), the time employed in reconstructing one particular solution (i.e., the on-line stage) and the amount of memory employed to store the PGD solution are also important values to keep in mind. These are represented in Figs. 14, 15 and 16, respectively. It can be noticed how, specially

DOI: 10.1002/nme

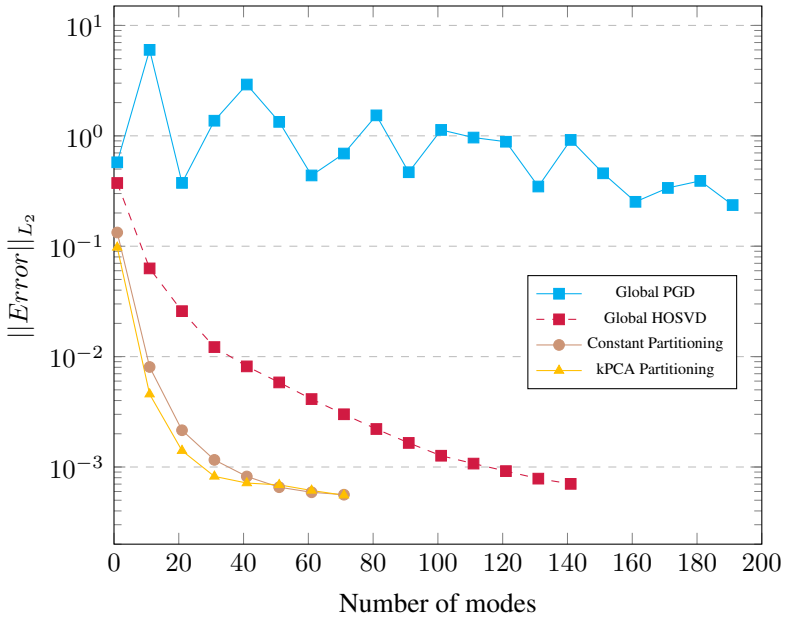


Figure 12. L^2-norm error of the temperature distribution versus the number of modes for global PGD, global POD and ℓ-PGD in its three variants, computed respect FEM solution with the same discretization, in the parametric heat transient example.

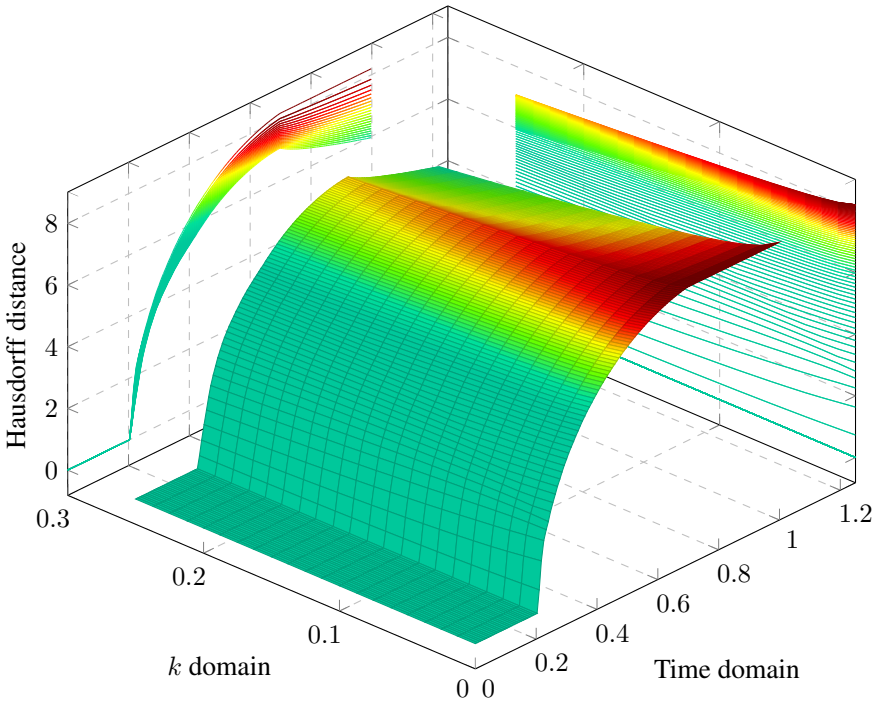


Figure 13. Hausdorff surface obtained from every time-kPCA Hausdorff curve moved along the k domain. Time domain is $]0, 1]$ and k domain is $[0, 0.25]$.

for the parametric case, the employ of local PGD approximations results in a very competitive approach for non-separable problems. Important time and memory savings are obtained that make ℓ-PGD methods an appealing choice for model order reduction in this context.

DOI: 10.1002/nme

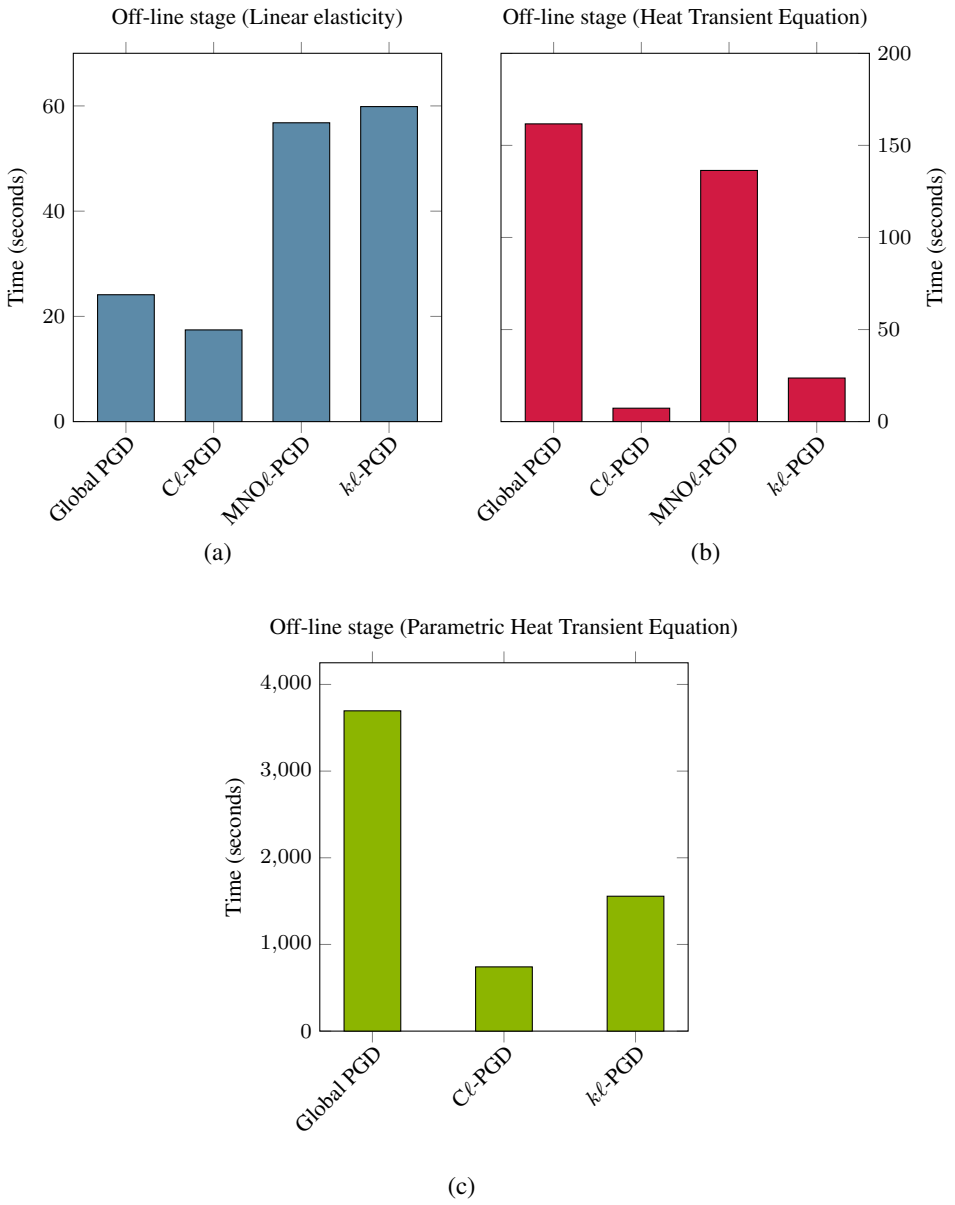


Figure 14. Time needed to compute the off-line stage for the a) Linear Elasticity example, b) Heat Transient Equation and c) Parametric Heat Transient Equation.

4. CONCLUSIONS

It is well known that for problems whose solution presents an intricate manifold structure, local model order reduction strategies provide with better results than global ones. However, very little has been done for *a priori* methods. The main reason for that is precisely the difficulty of determining the manifold structure of the solution *a priori*, with no snapshot of the solution available.

We have proposed here three possible methods for estimating the most appropriate size of the local sub-domains. Of course, many other methods in addition to those proposed may be envisaged. But what we know for sure is that adaptive strategies allow a better capture of the behavior of the solution manifold and look for an as constant as possible size of the reduced bases.

The method of constant division ($C\ell$-PGD) may be a valid option with respect to global implementations, but when compared with the other two adaptive methods shows its *brute force*

DOI: 10.1002/nme

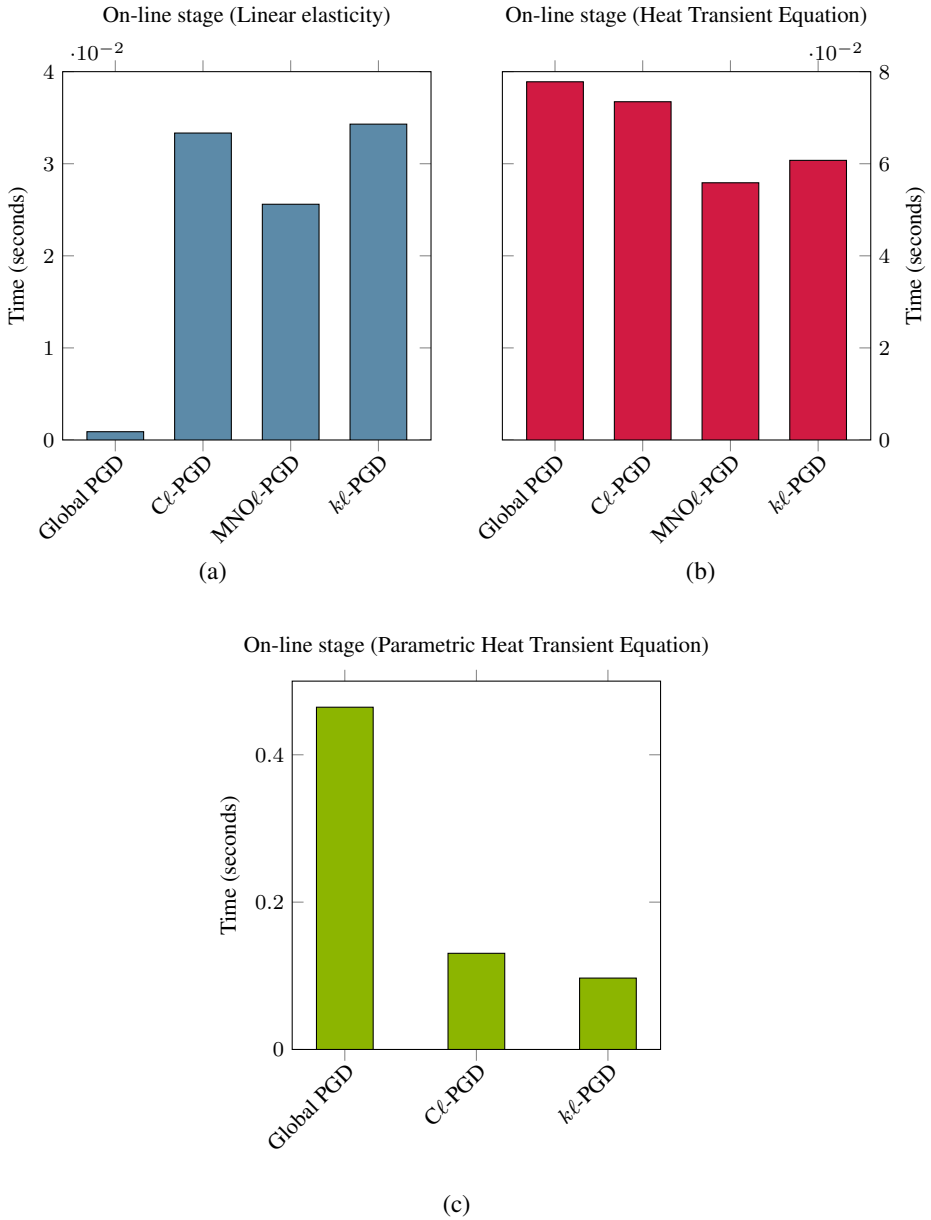


Figure 15. Time needed to compute the on-line stage for the a) Linear Elasticity example, b) Heat Transient Equation and c) Parametric Heat Transient Equation.

nature. The adaptive method with mode number optimization (MNOℓ-PGD) is the optimal method when only one dimension of the whole domain needs to be adjusted to choose the size of the sub-domains. But when the number of dimensions increases we propose the $k\ell$-PGD as the best method to adaptive partitioning the domain with, as we have seen, excellent results with respect to global *a priori* and *a posteriori* strategies.

ACKNOWLEDGEMENT

This work has been supported by the Spanish Ministry of Economy and Competitiveness through Grants number CICYT DPI2014-51844-C2-1-R and DPI2015-72365-EXP and by the Regional Government of Aragon and the European Social Fund, research group T88.

DOI: 10.1002/nme

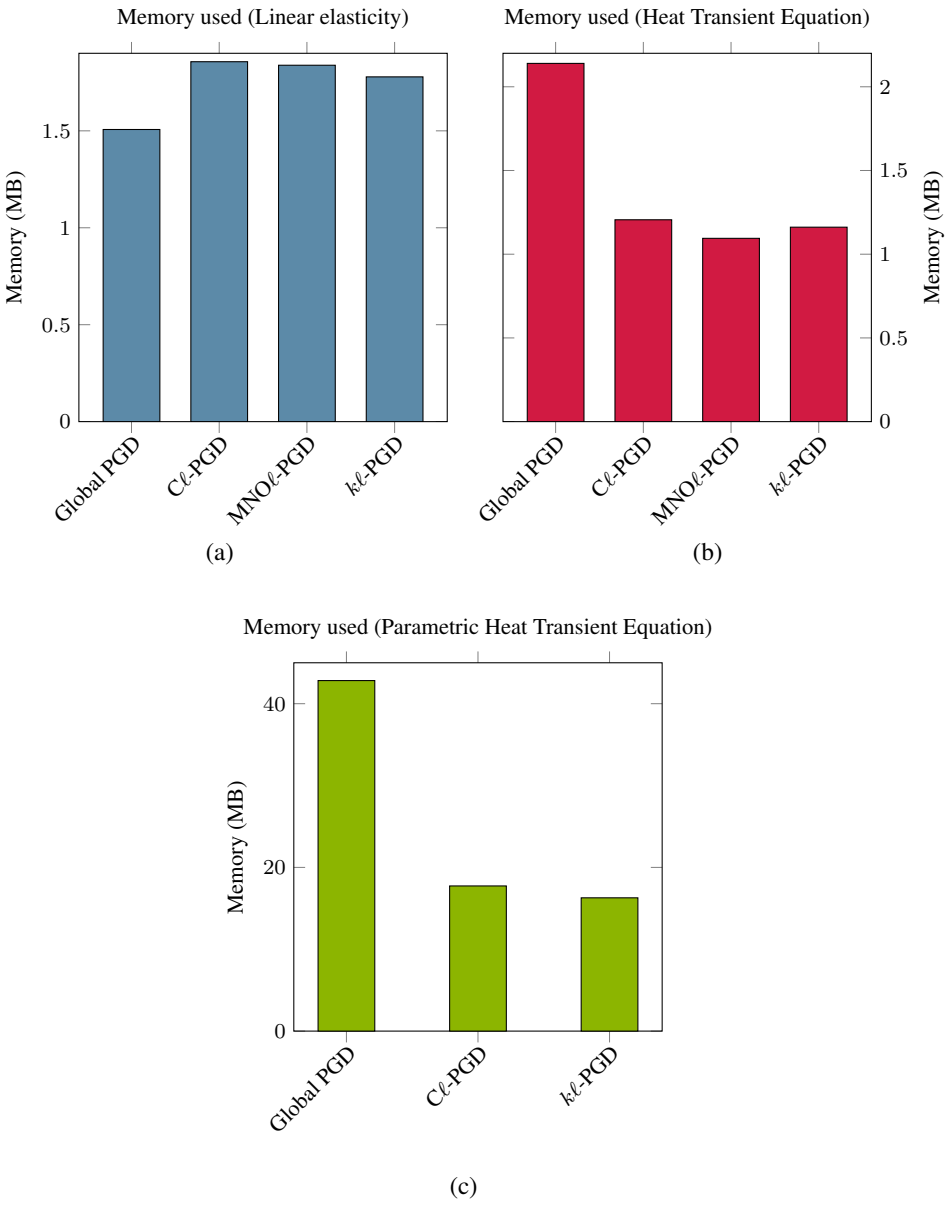


Figure 16. Memory needed to store the separated solution for the a) Linear Elasticity example, b) Heat Transient Equation and c) Parametric Heat Transient Equation.

REFERENCES

1. R. B. Laughlin and David Pines. The theory of everything. *Proceedings of the National Academy of Sciences*, 97(1):28–31, 2000.
2. P. Ladeveze. *Nonlinear Computational Structural Mechanics*. Springer, N.Y., 1999.
3. Sanjay Lall, Petr Krysl, and Jerrold E Marsden. Structure-preserving model reduction for mechanical systems. *Physica D: Nonlinear Phenomena*, 184(1-4):304–318, 2003.
4. David Ryckelynck, Francisco Chinesta, E Cueto, and Amine Ammar. On the a priori model reduction: Overview and recent developments. *Archives of Computational methods in Engineering*, 13(1):91–128, 2006.
5. Alfio Quarteroni, Andrea Manzoni, and Federico Negri. *Reduced Basis Methods for Partial Differential Equations: An Introduction*, volume 92. Springer, 2015.
6. Francisco Chinesta and Pierre Ladevèze. *Separated Representations and PGD-Based Model Reduction: Fundamentals and Applications*, volume 554. Springer, 2014.
7. YC Liang, HP Lee, SP Lim, WZ Lin, KH Lee, and CG Wu. Proper orthogonal decomposition and its applications—part i: Theory. *Journal of Sound and vibration*, 252(3):527–544, 2002.

DOI: 10.1002/nme

8. Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.
9. JP Fink and WC Rheinboldt. On the error behavior of the reduced basis technique for nonlinear finite element approximations. *ZAMMJournal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, 63(1):21–28, 1983.
10. Elías Cueto, David González, and Icíar Alfaro. *Proper generalized decompositions: an introduction to computer implementation with Matlab*. Springer, 2016.
11. Francisco Chinesta and Elias Cueto. *PGD-Based Modeling of Materials, Structures and Processes*. Springer International Publishing Switzerland, 2014.
12. Francisco Chinesta, Roland Keunings, and Adrien Leygue. *The Proper Generalized Decomposition for Advanced Numerical Simulations*. Springer International Publishing Switzerland, 2014.
13. F. Chinesta, A. Leygue, F. Bordeu, J.V. Aguado, E. Cueto, D. Gonzalez, I. Alfaro, A. Ammar, and A. Huerta. PGD-Based Computational Vademecum for Efficient Design, Optimization and Control. *Archives of Computational Methods in Engineering*, 20(1):31–59, 2013.
14. Carlos Quesada, David González, Iciar Alfaro, Elías Cueto, and Francisco Chinesta. Computational vademecums for realtime simulation of surgical cutting in haptic environments. *International Journal for Numerical Methods in Engineering*, 2016.
15. Francisco Chinesta, Pierre Ladeveze, and Elías Cueto. A short review on model order reduction based on proper generalized decomposition. *Archives of Computational Methods in Engineering*, 18(4):395–404, 2011.
16. A. Ammar, F. Chinesta, P. Diez, and A. Huerta. An error estimator for separated representations of highly multidimensional models. *Computer Methods in Applied Mechanics and Engineering*, 199(25-28):1872 – 1880, 2010.
17. Amine Ammar, Antonio Huerta, Francisco Chinesta, Elías Cueto, and Adrien Leygue. Parametric solutions involving geometry: A step towards efficient shape optimization. *Computer Methods in Applied Mechanics and Engineering*, 268(0):178 – 193, 2014.
18. David Modesto, Sergio Zlotnik, and Antonio Huerta. Proper generalized decomposition for parameterized Helmholtz problems in heterogeneous and unbounded domains: Application to harbor agitation. *COMPUTER METHODS IN APPLIED MECHANICS AND ENGINEERING*, 295:127–149, OCT 1 2015.
19. PG Dirichlet. Lejeune, 1829,sur la convergence des séries trigonométriques qui servent àreprésenter une fonction arbitraire entre des limites données. *Journal für die reine und angewandte Mathematik*, 3:157–169.
20. A Falcó, L Hilario, N Montés, and MC Mora. Numerical strategies for the galerkin–proper generalized decomposition method. *Mathematical and Computer Modelling*, 57(7):1694–1702, 2013.
21. Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.
22. Ledyard R Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.
23. Anthony Nouy. A priori model reduction through Proper Generalized Decomposition for solving time-dependent partial differential equations. *COMPUTER METHODS IN APPLIED MECHANICS AND ENGINEERING*, 199(23-24):1603–1626, 2010.
24. Pierre-Eric Allier, Ludovic Chamoin, and Pierre Ladevèze. Proper generalized decomposition computational methods on a benchmark problem: introducing a new strategy based on constitutive relation error minimization. *Advanced Modeling and Simulation in Engineering Sciences*, 2(1):17, 2015.
25. Roger N Shepard. The analysis of proximities: Multidimensional scaling with an unknown distance function. i. *Psychometrika*, 27(2):125–140, 1962.
26. Roger N Shepard. The analysis of proximities: Multidimensional scaling with an unknown distance function. ii. *Psychometrika*, 27(3):219–246, 1962.
27. Joseph B Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.
28. Joseph B Kruskal. Nonmetric multidimensional scaling: a numerical method. *Psychometrika*, 29(2):115–129, 1964.
29. Keinosuke Fukunaga and David R Olsen. An algorithm for finding intrinsic dimensionality of data. *IEEE Transactions on Computers*, 100(2):176–183, 1971.
30. Geoffrey E Hinton, Michael Revow, and Peter Dayan. Recognizing handwritten digits using mixtures of linear models. *Advances in neural information processing systems*, pages 1015–1022, 1995.
31. Christoph Bregler and Stephen M Omohundro. Nonlinear image interpolation using manifold learning. *Advances in neural information processing systems*, pages 973–980, 1995.
32. Nandakishore Kambhatla and Todd K Leen. Dimension reduction by local principal component analysis. *Neural Computation*, 9(7):1493–1516, 1997.
33. D Ryckelynck. A priori hyperreduction method: an adaptive approach. *Journal of computational physics*, 202(1):346–366, 2005.
34. David Amsallem, Matthew J Zahr, and Charbel Farhat. Nonlinear model order reduction based on local reduced-order bases. *International Journal for Numerical Methods in Engineering*, 92(10):891–916, 2012.
35. Patricia Astrid, Siep Weiland, Karen Willcox, and Ton Backx. Missing point estimation in models described by proper orthogonal decomposition. *IEEE Transactions on Automatic Control*, 53(10):2237–2251, 2008.
36. Liqian Peng and Kamran Mohseni. Nonlinear model reduction via a locally weighted pod method. *International Journal for Numerical Methods in Engineering*, 2016.
37. David Amsallem and Charbel Farhat. Interpolation method for adapting reduced-order models and application to aeroelasticity. *AIAA journal*, 46(7):1803–1813, 2008.
38. David Neron and Pierre Ladeveze. Idelsohns benchmark. Technical report, 2013.
39. Jean-Frédéric Gerbeau and Damiano Lombardi. Reduced-order modeling based on approximated lax pairs. *arXiv preprint arXiv:1211.4153*, 2012.

40. Peter D Lax. Integrals of nonlinear equations of evolution and solitary waves. *Communications on pure and applied mathematics*, 21(5):467–490, 1968.
41. D. Gonzalez, A. Ammar, F. Chinesta, and E. Cueto. Recent advances on the use of separated representations. *International Journal for Numerical Methods in Engineering*, 81(5), 2010.
42. Markus Dihlmann, Martin Drohmann, and Bernard Haasdonk. Model reduction of parametrized evolution problems using the reduced basis method with adaptive time-partitioning. *Proc. of ADMOS*, 2011, 2011.
43. Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. pages 583–588. Springer, 1997.
44. Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.
45. Felix Hausdorf and John R Aumann. *Grundzüge der mengenlehre*. Veit, 1914.
46. Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput.*, 10(5):1299–1319, July 1998.
47. B. Scholkopf, A. Smola, and K.R. Muller. Kernel principal component analysis. In *ADVANCES IN KERNEL METHODS - SUPPORT VECTOR LEARNING*, pages 327–352. MIT Press, 1999.
48. Jacob Fish and Ted Belytschko. *A first course in finite elements*. John Wiley & Sons, 2007.
49. CA Truesdell. The rational mechanics of flexible or elastic bodies, 1638-1788, 1960. *Sert d'introduction àO. II*, 10.
50. Pierre-Eric Allier, Ludovic Chamoin, and Pierre Ladevèze. Proper generalized decomposition computational methods on a benchmark problem: introducing a new strategy based on constitutive relation error minimization. *Advanced Modeling and Simulation in Engineering Sciences*, 2(1):1, 2015.

DOI: 10.1002/nme