Local Proper Generalized Decomposition

A. Badías[1, a)], D. González[1,b)], I. Alfaro[1,c)], F. Chinesta[2,d)] and E. Cueto[1,e)]

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

[a)] Corresponding author: abadias@unizar.es
[b)] gonzal@unizar.es
[c)] iciar@unizar.es
[d)] francisco.chinesta@ec-nantes.fr
[e)] ecueto@unizar.es

Abstract. Local model order reduction methods provide better results than global ones to problems with intricate manifold solution structure. *A posteriori* methods (e.g. Proper Orthogonal Decomposition) have been many times applied locally, but *a priori* methods (e.g. Proper Generalized Decomposition) have the difficulty of determining the manifold structure of the solution in a previous way. We propose three strategies for estimating the appropriate size of the local sub-domains where afterwards local PGD (ℓ-PGD) is applied. It can be seen as a sort of a priori manifold learning or non-linear dimensionality reduction technique. Finally, three examples support the work.

INTRODUCTION

Computer-based simulation has become one of the most relevant tools to design, control or predict the behavior of any system in engineering. Any physical problem can be approximated by a model, which may be very simple or very complex, but allows us to obtain a numerical simulation of the behavior of the system.

The response time of a system is the time taken for the system to generate output from associated input [1]. When a physical problem has a response time very small, and with an input to the system the response is produced *immediately*, it is said that the problem works in *real-time*. But actually the word *real-time* means otherwise in computer science. A *real-time* system can be any information processing activity or system that has to respond to externally generated input stimuli within a finite and specified period [2]. This does not mean that the response time must be very small; it means that response time must be under control and we must guarantee the response within specified time constraints, often referred to as *deadlines*.

The high computing capacity of our computers allows us to simulate the behavior of complex systems using models with large amounts of parameters. Doing it in *real-time* may sometimes cause bad situations due to the large amount of information. And in some cases, we are not able to meet the deadline time. Therefore, the idea of using model order reduction methods was born to reduce the dimensionality of our problem, as well as the amount of data needed, working on a reduced basis.

Model order reduction methods can be classified into *a posteriori* methods (e.g., Proper Orthogonal Decomposition [3], Reduced Basis [4]) and *a priori* methods (e.g., Proper Generalized Decomposition [5,6]). *A posteriori* methods are built after computing some solutions of the system or the whole set of solutions. They are especially useful when the model has a relatively small number of parameters but many observations. *A posteriori* methods try to obtain a projected solution in a space of smaller dimensionality to the starting space.

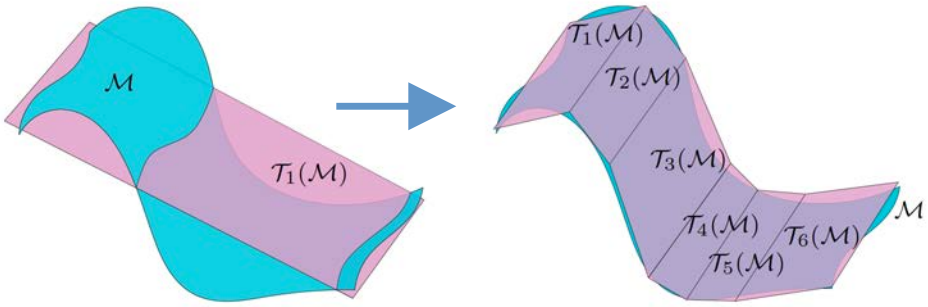
On the other side, *a priori* methods are built without the need of pre-computing any solution of the problem and are based in the knowledge of the equation that governs the problem. *A priori* methods iteratively look for the solution of the problem but directly on a reduced basis.

In multiparametric problems with a high number of dimensions (each parameter spans one dimension) it may be impractical to pre-calculate the solution of the problem and then obtain the reduced solution. For this reason, a priori methods are the ideal tool for multiparametric problems.

One of the most relevant *a priori* methods is Proper Generalized Decomposition (PGD). This method consists in obtaining the solution of a multiparametric problem in separate variables, where each variable depends only on one parameter. Let $u(x_1, x_2, \dots, x_D)$ be the solution of a given problem with D dependent variables $(x_1, x_2, \dots, x_D)$. PGD method makes an approximation u^{aprox} to the real solution as the sum of N terms, being each term the product of D separate functions.

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

PGD method has experienced a strong development in recent years achieving great results. However, some physical problems called non-separable problems may cause the number of modes necessary to approximate the solution to be quite high. These non-separable problems are often characterized by having an intricate or complex manifold (hyper-surface where the solution lies). One solution to address this problem is to reduce the complexity making partitions in the global domain obtaining local sub-domains that better approximate the solution (see Fig. 1). Applying PGD in local sub-domains (ℓ-PGD) mixes the *a priori* benefits with the simplicity of using local implementations.



a) b)

FIGURE 1. Manifold $\mathcal{M}$ where the solution lies and tangent hyper-planes $\mathcal{T}_i(\mathcal{M})$ where the solution is projected. a) Global approximation. b) Local approximation.

LOCAL PGD (ℓ-PGD)

There are multiple ways to apply the PGD method locally. Divisions in local sub-domains can be made in any of the dimensions of the problem to solve (and also in all dimensions). In addition, the choice of the division boundaries of the sub-domains can also be made in multiple ways. We propose 3 strategies to do this task that are shown in next sections.

Constant Local Partitions (Cℓ-PGD)

The first and simplest strategy makes partitions of regular size fixing either the number of sub-domains or the size of each sub-domain. It is a non-intelligent way of making partitions as there is no evidence of the complexity of each sub-domain. The number of local modes can be very different in the sub-domains.

ℓ-PGD Adaptive Partitioning with Mode Number Optimization (MNOℓ-PGD)

The second strategy tries to obtain a number of local modes M_i for each sub-domain the most similar to a number N previously set.

$$M_i \approx N \quad \forall \quad i = 1, \dots, n_d$$

where n_d is the number of local sub-domains. It is based in an iterative algorithm similar to the one proposed in [7].

Kernel-PCA-based ℓ-PGD (kℓ-PGD)

The third strategy that we propose uses the kernel Principal Component Analysis (k-PCA) [8] technique to transform the original data and obtain the hidden and nonlinear complexity of the problem to perform divisions depending on the complexity of the manifold. It consists in a 2-steps method where the solution is pre-computed with the Cℓ-PGD strategy to apply, later, the k-PCA transformation and estimate the best way of making divisions. Finally the solution is computed with the final sizes of the local sub-domains.

Kernel Principal Component Analysis

The purpose of using k-PCA is to obtain the principal components of the data in a new space of Q dimensions after applying a non-linear transformation, where $Q > D$ and D is the size of the original space. This transformation is carried out applying a non-linear mapping ϕ.

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

The analytical expression of the mapping ϕ can be replaced by the use of the *kernel trick* simplifying the procedure. We have used a *Gaussian* kernel (1).

$$\kappa(u, v) = \exp(-\frac{\|u-v\|^2}{2\,\sigma^2}) \text{ for a real } \sigma \quad (1)$$

After applying the kernel, a decomposition in principal components (PCA) is applied to project the data on a new space of dimensions d, where $d \ll Q$. This new space of dimensions is characterized by maximizing the variance of the data. Finally, using the *Hausdorff distance* we are able to quantify the complexity (hidden variability) of the data.

Hausdorff distance

The *Hausdorff* distance reveals the intrinsic variability of the solution between particular realizations. 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 serves to know the variability of the manifold, but to really make the partitions and set the boundaries of the local sub-domains we have used the *ε-neighborhood clustering*.

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 $Br(p)$ 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,

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

NUMERICAL RESULTS

Cantilever Beam with a Moving Load

The first example of the numerical results consists in solving the displacement $u(x,y)$ of any point of a bidimensional cantilever beam Ω with a moving load modeled with linear elasticity. The load has a constant value and is applied on the upper face of the beam ($s \in \overline{\Gamma}$), as can be seen in Fig. 2.a. The linear elasticity governing equation in its strong way is

$$u(\boldsymbol{x}) = \begin{cases} \nabla \cdot \sigma + b &= 0 \quad on\ \Omega, \\ \sigma\, n &= \bar{t} \quad on\ \Gamma_t, \\ u &= \bar{u} \quad on\ \Gamma_u \end{cases} \tag{2}$$

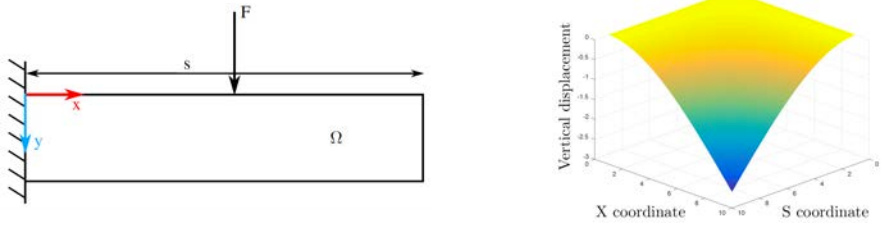
We are looking for the parametric solution $u(\boldsymbol{x}, s)$ that depends on (x, y, s) so applying the PGD formalism [9] to equation (2) we obtain

$$\int_{\overline{\Gamma}} \int_{\Omega} \nabla_s\, u^* : \sigma\, d\Omega\, d\overline{\Gamma} = \int_{\overline{\Gamma}} \int_{\Gamma_t} u^*\, t\ d\Gamma\, d\overline{\Gamma}$$

where $\nabla_s = \frac{1}{2}[\nabla + (\nabla)^T]$ is the symmetric gradient operator and $u^* \in H_0^1(\Omega)$ is an arbitrary test function. We force the solution $u(\boldsymbol{x}, s)$ in separate variables as

$$u(\boldsymbol{x}, s) \approx \sum_{i=1}^{n} [F_i(\boldsymbol{x}) \cdot G_i(s)]$$

where n is the number of sums, $F_i(\boldsymbol{x})$ are the space modes and $G_i(s)$ are the modes of the position of the load in separate variables. The interested reader can find more details of this example in [9]. The result in the 3 most relevant dimensions after applying k-PCA to the vertical displacement of the beam is shown in Fig. 3. The separation between curves computed using the Hausdorff distance is shown in Fig. 6.a. Finally, Fig. 7.a shows the L^2-norm error of the surface solution (Fig. 2.b) computed with several methods with respect the FEM solution



a) b)

FIGURE 2. a) Sketch of the cantilever beam with a moving load in its upper face. b) Vertical displacement of the cantilever beam for every position of the load creating the solution surface.

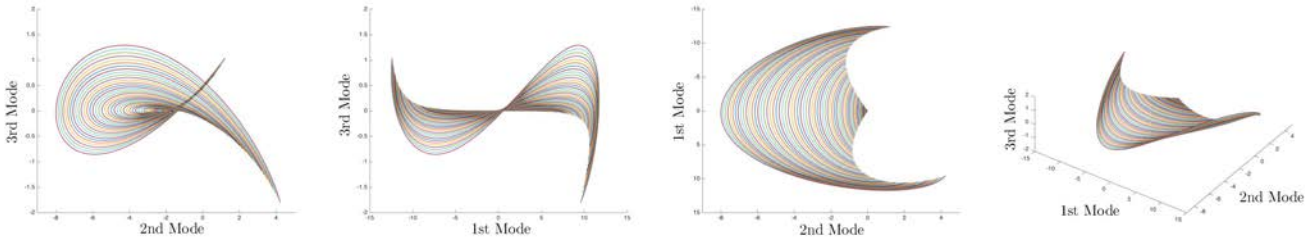
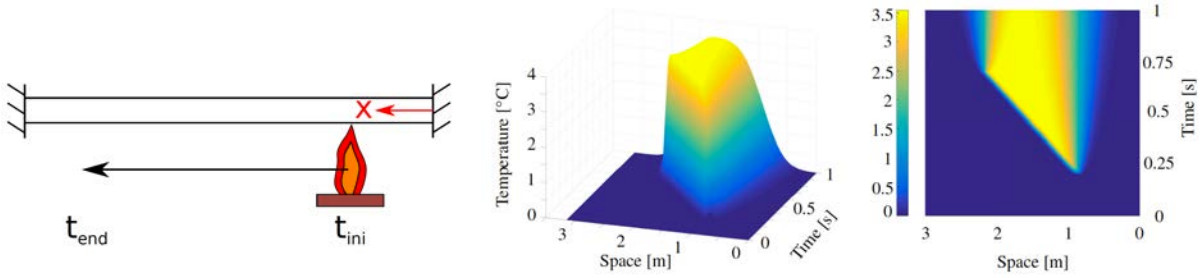


FIGURE 3. Set of curves obtained after applying k-PCA.

Heat Transient Equation

The second example is the heat transient equation applied in a 1D beam with a moving load, as it is shown in Fig. 4.



a) b) c)

FIGURE 4. a) 1D beam where the heat transient equation is applied. The source starts heating at (x_{ini}, t_{ini}) and stops adding heat in (x_{end}, t_{end}). b) 3D surface of the solution. c) 2D view of the same problem to better appreciate the difficulty of separating a sharp discontinuity running across the diagonal of space-time.

The governing equation of the problem is

$$u(x,t) = \begin{cases} \rho\, c_p \dfrac{\partial u(x,t)}{\partial t} - \kappa \dfrac{\partial^2 u(x,t)}{\partial x^2} & = f(x,t) \quad in\ \Omega\ x\ \mathcal{I}, \\ u & = u_D \quad on\ \Gamma_D\ x\ \mathcal{I}, \\ u & = u_0 \quad on\ \Omega\ x\ \{0\} \end{cases}$$

where u is the solution (temperature), ρ is the material density, c_p is the specific heat capacity and κ is the thermal conductivity. Using the PGD formalism the structure in separate variables of the solution and the source term is

$$u(x,t) \approx \sum_{i=1}^{n} [F_i(x) \cdot G_i(t)], \qquad f(x,t) \approx \sum_{h=1}^{m} [f_{a_h}(x) \cdot f_{b_h}(t)]$$

The result after applying k-PCA to the temperature distribution of the beam is shown in Fig. 5. The separation between curves computed using the Hausdorff distance is shown in Fig. 6.b. Fig. 7.b shows the L^2-norm error of the temperature distribution (Fig. 4.b) computed with several methods with respect the FEM solution.

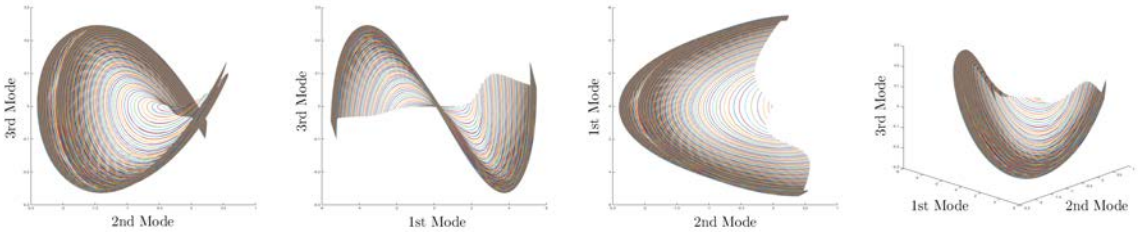


FIGURE 5. Set of curves obtained after applying k-PCA to the Heat Transient example.

Heat Transient Equation (k parametric)

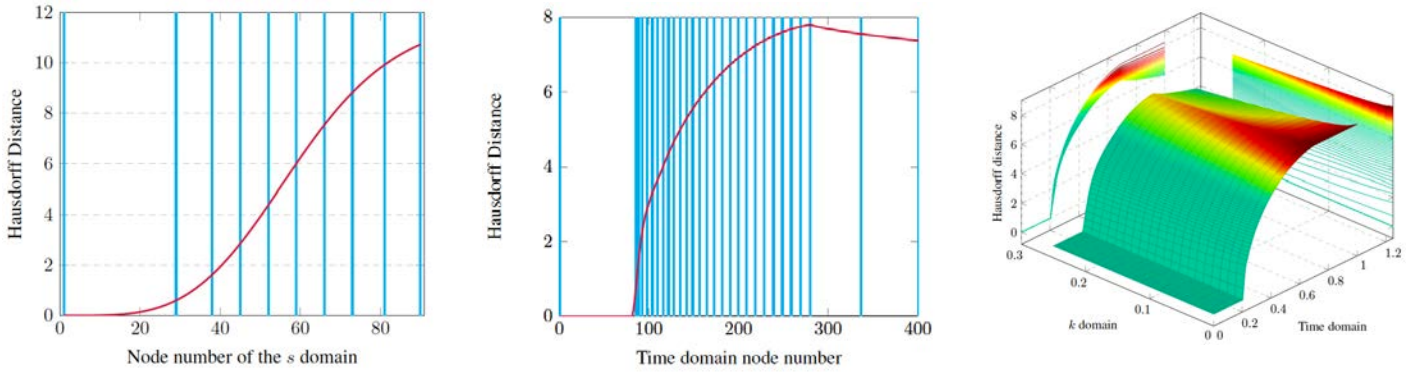
The third example is the heat transient equation again but we add the thermal conductivity κ as a parameter. The solution in separate variables is

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

The Hausdorff distance creates a surface as we are making divisions in two dimensions (Fig. 6.c). In Fig. 7.c we can see the L^2-norm error of the temperature distribution of the same surface of Fig. 2.b.

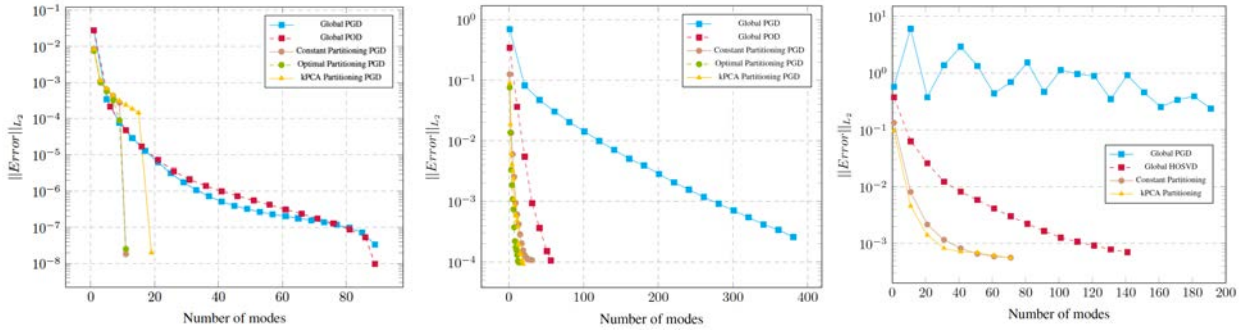
CONCLUSIONS

We have proposed three methods for estimating the most appropriate size of the local sub-domains. MNOℓ-PGD is the optimal method when only one dimension needs to be adjusted to choose the size of the sub-domains. When the number of dimensions increases we propose the kℓ-PGD as the best method to adaptive partitioning the domain with excellent results with respect to global *a priori* and *a posteriori* strategies.



a) b) c)

FIGURE 6. Hausdorff distance and sub-domain separation (vertical blue lines) in the: a) cantilever beam example and b) heat transient problem. c) Hausdorff distance creating a surface in the heat transient (k parametric) example.



a) b) c)

FIGURE 7. L^2-norm error of the a) vertical displacement of the cantilever beam with a moving load, b) temperature distribution of the heat transient example, c) temperature distribution of the heat transient example with parametric κ.

ACKNOWLEDGMENTS

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.

REFERENCES

1. Burns, A., & Wellings, A. J. (2010). *Real-time systems and programming languages* (Vol. 2097). Addison-Wesley.
2. Young, S. J. (1982). *Real time languages*. Horwood.
3. Liang, Y. C., Lee, H. P., Lim, S. P., Lin, W. Z., Lee, K. H., & Wu, C. G. (2002). Proper orthogonal decomposition and its applications—Part I: Theory. *Journal of Sound and vibration*, *252*(3), 527-544.
4. Fink, J. P., & Rheinboldt, W. C. (1983). On the error behavior of the reduced basis technique for nonlinear finite element approximations. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, *63*(1), 21-28.
5. Chinesta, F., & Cueto, E. (2014). *PGD-based modeling of materials, structures and processes*. Heidelberg: Springer.
6. Chinesta, F., Keunings, R., & Leygue, A. (2013). *The proper generalized decomposition for advanced numerical simulations: a primer*. Springer Science & Business Media.
7. Dihlmann, M., Drohmann, M., & Haasdonk, B. (2011). Model reduction of parametrized evolution problems using the reduced basis method with adaptive time-partitioning. *Proc. of ADMOS*, *2011*.
8. Schölkopf, B., Smola, A., & Müller, K. R. (1997, October). Kernel principal component analysis. In *International Conference on Artificial Neural Networks*(pp. 583-588). Springer Berlin Heidelberg.
9. Cueto, E., González, D., & Alfaro, I. (2016). *Proper generalized decompositions: an introduction to computer implementation with Matlab*. Springer.