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/10985/8499

To cite this version :

Amine AMMAR, Francisco CHINESTA, Elias CUETO, Manuel DOBLARÉ - Proper generalized decomposition of time-multiscale models - International Journal for Numerical Methods in Engineering - Vol. 90, n°5, p.569-596 - 2012

Any correspondence concerning this service should be sent to the repository Administrator : scienceouverte@ensam.eu

Arts et Métiers Institute of Technology

Proper generalized decomposition of time-multiscale models

Amine Ammar[1], Francisco Chinesta[2,*,†], Elías Cueto[3] and Manuel Doblaré[3]

[1]*Arts et Métiers ParisTech, 2 Boulevard du Ronceray, BP 93525, F-49035 Angers cedex 01, France*
[2]*EADS Corporate Foundation International Chair, École Centrale de Nantes, 44300 Nantes, France*
[3]*Aragón Institute of Engineering Research (I3A), Universidad de Zaragoza, Edificio Betancourt, María de Luna, s.n. 50018 Zaragoza, Spain*

SUMMARY

Models encountered in computational mechanics could involve many time scales. When these time scales cannot be separated, one must solve the evolution model in the entire time interval by using the finest time step that the model implies. In some cases, the solution procedure becomes cumbersome because of the extremely large number of time steps needed for integrating the evolution model in the whole time interval. In this paper, we considered an alternative approach that lies in separating the time axis (one-dimensional in nature) in a multidimensional time space. Then, for circumventing the resulting curse of dimensionality, the proper generalized decomposition was applied allowing a fast solution with significant computing time savings with respect to a standard incremental integration.

KEY WORDS: time multiscale, proper generalized decomposition, separated representations, transient models

1. INTRODUCTION

Many problems in science and engineering remain intractable despite the impressive progresses in computer science and the computational resources available today, because their numerical complexity is simply unimaginable. Among the models that remain intractable today, we can distinguish two main families:

- The first family of models consists of standard models usually encountered in computational mechanics, defined in large and complex three-dimensional (3D) geometries, involving many multiphysics couplings, many scales (in space and time), strong nonlinearities, and whose transient simulation needs extremely small time steps. These models are usually encountered in the mechanics of structures, but they are also present in many other fields. To illustrate this scenario, one could imagine the simple reaction-diffusion model that describes the degradation of plastic materials, where the characteristic time of the chemical reaction involved in the material degradation is of some microseconds, and the one related to the diffusion of chemical substances (that also represents the material degradation characteristic time itself) is of the order of years. In these cases, standard incremental techniques must be replaced by other more efficient techniques.
- The other family of challenging models concerns those models defined in highly dimensional spaces. This kind of models appears naturally, for example, in the modeling of the structure and properties of materials at the finest scales. Thus, models in quantum chemistry (e.g., the

*Correspondence to: Francisco Chinesta, EADS Corporate International Chair, École Centrale de Nantes, 1 rue de la Noë, 44300 Nantes, France.
†E-mail: Francisco.Chinesta@ec-nantes.fr

Schroedinger or Dirac equations), the kinetic theory description of materials (e.g., the Fokker–Planck equation), models in financial mathematics, genetics (the chemical master equation) ... are defined in spaces involving hundreds, thousands, or millions of dimensions. These models exhibit the redoubtable curse of dimensionality when the usual mesh-based discretization techniques are applied. The curse of dimensionality can be easily illustrated. Imagine a model defined in a space involving D dimensions. If one proceeds to the solution of such model by using a standard mesh-based discretization technique, where M nodes are used for discretizing each space coordinate, the resulting number of nodes reaches the astronomical value of M^D. With $M \approx 10^3$ (a very coarse description in practice) and $D \approx 30$ (a quite simple system in practice), the numerical complexity results 10^{90}. It is important to recall that 10^{80} represents the presumed number of elementary particles in the whole universe. Moreover, as we illustrate later, many usual models can be expressed by introducing many other extra coordinates that will be called configurational or conformational coordinates. For example, model parameters (e.g., the material conductivity involved in a thermal model, the applied loads, the initial or boundary conditions, parameters describing the geometry ...) could be considered as extra coordinates. Thus, by solving the resulting multidimensional model, we could have access to the model solution at any point of the space domain, at any time, and for any value of the thermal conductivity, source term, parameter describing the geometry ... The price to be paid is the solution of a problem that, until now, no numerical technique was able to perform.

Recently, a novel efficient technique able to circumvent the challenging issues just described was proposed. It was coined as proper generalized decomposition (PGD). It is based on a separated representation of the unknown field. Thus, for circumventing the numerical issues associated with the first family of models (previously described), one could consider the transient solution under the separated form:

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

representation, which was originally proposed in the 1980s by Pierre Ladeveze as one of the main ingredients of the LATIN (nonlinear and non-incremental solver), and which was called "radial approximation" [1–5]. If we look at the performance of such separated representation-based discretization techniques, the verdict is for many models implacable. If one considers a standard transient model defined in a 3D physical space, and if one considers P time steps, usual incremental strategies must solve P (in general nonlinear) 3D problems (do not forget that P can be millions). However, if space–time PGD (radial approximation) is considered, we should solve around $N \cdot m$ 3D problems for computing the space functions and $N \cdot m$ one-dimensional (1D) problems for computing the time functions (m being the number of iterations needed for computing each term of the finite sum because of the nonlinear nature of the definition of the PGD). As $m \approx 10$ and $N \approx 10$ for many models, the computing time savings can reach many orders of magnitude.

If we come back to the second family of models, an appealing choice consists of expressing the unknown field as a finite sum of functional products, that is, expressing a generic multidimensional function as

$$u(x_1, \cdots, x_N) \approx \sum_{i=1}^{N} X_i^1(\mathrm{x}_1) \cdot \ldots \cdot X_i^D(\mathrm{x}_D) \tag{2}$$

where x_i could represent any coordinate, scalar or vector, involving the physical space, the time, or any other conformation coordinate. It could also represent random variables for the modeling of parametric uncertainties. This separated representation was proposed by A. Ammar and F. Chinesta some years ago in the context of multidimensional models encountered in the description of complex fluids within the kinetic theory framework [6, 7], as well as by A. Nouy in the context of stochastic models [8]. The reader can refer to [9, 10] and the references therein for some more recent contributions on this topic.

Within the PGD framework for solving a problem defined in a space of dimension D, if M nodes are used to discretize each coordinate, the total number of unknowns involved in the solution is $M \cdot D$ instead of the M^D degrees of freedom involved in mesh based discretizations. We must recall that these functions are not known "a priori" but are computed by introducing the approximate separated representation into the model weak form and then by solving the resulting nonlinear problem associated with the construction of the separated representation. As it can be noticed in the expression of the separated representation, the complexity scales linearly with the dimension of the space in which the model is defined, instead of the exponential growing characteristic of mesh-based discretization strategies. In general, for many models, the number of terms N in the finite sum is quite reduced (a few tens).

We would like to recall that, using these approaches, we have reduced the computing time related to the solution of problems belonging to the first family of models (defined earlier) in several orders of magnitude, and on the other hand, we solved successfully highly multidimensional models because they were considered suffering of the irremediable curse of dimensionality [10, 11].

Because of the novelty and the youth of the PGD method, many aspects have not yet been addressed. The aim of this work is to push back the limits of these methods. Definitively, these developments could lead to a real change of paradigm in computational mechanics. Imagine the possibility of solving efficiently and with a controlled accuracy any parametric (including parameterized geometries) multiscale and multiphysics model. Inverse identification and optimization would be a simple post-treatment.

In some of our recent works, we focused on multiphysics models involving different characteristic times and behaviors (local and global). This problem was addressed in particular in [12]. In the present work, we focus in a slightly different issue, the one related to models involving non-separable time scales that should be solved by assuming the finest time step concerned by the model solution.

Time multiscale problems involve similar difficulties than those models that are multiscale in space but add some extra issues that must be solved with caution. This is partly caused by the inherent sequential nature of time, which is not present in standard multiscale models. Many works have been devoted in the last years to the topic of establishing efficient time multiscale numerical methods. These approaches are also very different in nature.

For instance, in [13], a method is developed that applies the same principles of spatial homogenization to time multiscale problems. To that end, a neat separation of time scales is mandatory, as in traditional homogenization approaches.

Particularly noteworthy is the topic of establishing parallel approaches in time, needed for computational efficiency, or simply when different physics with different intrinsic time scales are considered. Here, again the sequential nature of time plays an important role and poses important difficulties. [14] and [15] are two particular instances of the application of the so-called time-parallel approaches to time-multiscale problems. In [16], however, the principles of the variational multiscale method are applied to time multiscale integration. In [17, 18], on the contrary, a method based on the LATIN approach is developed that couples different physical problems with different intrinsic time scales in a space–time framework that can be considered as the origin of the approach presented herein.

We consider, however, an alternative approach that lies in separating the time axis (one-dimensional in nature) in a multidimensional time space. Then, for circumventing the resulting curse of dimensionality, the PGD is applied allowing for a fast solution with significant computing time savings with respect to a standard incremental integration.

In the next section, we revisit the main ideas of the PGD whose implementation is illustrated through the solution of an academic problem. Then, a tensor formulation of the PGD is presented in Section 3. Section 4 concerns the introduction of constraints in the solution process from the use of penalty or Lagrange multiplier techniques. Section 5 makes use of this procedure for decomposing the time axis into a two-dimensional (2D) time domain procedure that is applied in a quite simple transient ODE for illustrating the capabilities of the proposed technique. In that section, we consider also the solution of a transient parabolic PDE in which the time coordinate is transformed into a 2D time space. In the last case, the resulting model includes a space coordinate and two time

coordinates. Finally, Section 6 presents an alternative formulation for enforcing the solution continuity in time without using either penalty or Lagrange multipliers. The continuity is enforced a posteriori.

2. THE PROPER GENERALIZED DECOMPOSITION REVISITED

In what follows, the construction of the PGD is illustrated by considering the parametric heat transfer equation:

$$\frac{\partial u}{\partial t} - k\Delta u - f = 0 \tag{3}$$

where $(\mathbf{x}, t, k) \in \Omega \times I \times \Im$ and, for the sake of simplicity, the source term is assumed constant, that is, $f = cst$. Because the conductivity is considered unknown, it is assumed as a new coordinate defined in the interval $\Im$. Thus, instead of solving the thermal model for different values of the conductivity parameter, we prefer introducing it as a new coordinate. The price to pay is the increase of the model dimensionality; however, as the complexity of PGD scales linearly with the space dimension, the consideration of the conductivity as a new coordinate remains compatible with fast and cheap solutions.

The solution of Equation (3) is sought under the form:

$$u(\mathbf{x}, t, k) \approx \sum_{i=1}^{i=N} X_i(\mathbf{x}) \cdot T_i(t) \cdot K_i(k) \tag{4}$$

In what follows, we are assuming that the approximation at iteration n is already achieved:

$$u^n(\mathbf{x}, t, k) = \sum_{i=1}^{i=n} X_i(\mathbf{x}) \cdot T_i(t) \cdot K_i(k) \tag{5}$$

and that, at current iteration, we look for the next functional product $X_{n+1}(\mathbf{x}) \cdot T_{n+1}(t) \cdot K_{n+1}(k)$ that, for alleviating the notation, will be denoted by $R(\mathbf{x}) \cdot S(t) \cdot W(k)$. Prior to solving the resulting nonlinear model related to the calculation of these three functions, a model linearization is compulsory. The simplest choice consists in using an alternating directions fixed point algorithm. It proceeds by assuming $S(t)$ and $W(k)$ given at the previous iteration of the nonlinear solver and then computing $R(\mathbf{x})$. From the just updated $R(\mathbf{x})$ and $W(k)$, we can update $S(t)$, and finally from the just computed $R(\mathbf{x})$ and $S(t)$, we compute $W(k)$. The procedure continues until reaching convergence. The converged functions $R(\mathbf{x})$, $S(t)$, and $W(k)$ allow defining the searched functions: $X_{n+1}(\mathbf{x}) = R(\mathbf{x})$, $T_{n+1}(t) = S(t)$ and $K_{n+1}(k) = W(k)$. We are illustrating each one of the just referred steps.

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

We consider the global weak form of Equation (3):

$$\int_{\Omega \times I \times \Im} u^* \left(\frac{\partial u}{\partial t} - k\Delta u - f \right) d\mathbf{x}\, dt\, dk = 0 \tag{6}$$

where the trial and test functions write, respectively:

$$u(\mathbf{x}, t, k) = \sum_{i=1}^{i=n} X_i(\mathbf{x}) \cdot T_i(t) \cdot K_i(k) + R(\mathbf{x}) \cdot S(t) \cdot W(k) \tag{7}$$

and

$$u^*(\mathbf{x}, t, k) = R^*(\mathbf{x}) \cdot S(t) \cdot W(k) \tag{8}$$

Introducing Equations (7) and (8) into Equation (6), it results in

$$\begin{aligned}&\int\limits_{\Omega\times I\times\Im} R^*\cdot S\cdot W\cdot\left(R\cdot\frac{\partial S}{\partial t}\cdot W-k\cdot\Delta R\cdot S\cdot W\right)\,d\mathbf{x}\,dt\,dk=\\&=-\int\limits_{\Omega\times I\times\Im} R^*\cdot S\cdot W\cdot\mathcal{R}^n\,d\mathbf{x}\,dt\,dk\end{aligned}\tag{9}$$

where $\mathcal{R}^n$ defines the residual at iteration n that reads:

$$\mathcal{R}^n=\sum_{i=1}^{i=n} X_i\cdot\frac{\partial T_i}{\partial t}\cdot K_i-\sum_{i=1}^{i=n} k\cdot\Delta X_i\cdot T_i\cdot K_i.-f\tag{10}$$

Now, knowing all the functions involving the time and the parametric coordinate, we can integrate Equation (9) in its respective domains $I\times\Im$. Integrating in $I\times\Im$ and taking into account the following notations

$$\left[\begin{array}{lll}w_1=\int\limits_{\Im}W^2dk & s_1=\int\limits_{I}S^2dt & r_1=\int\limits_{\Omega}R^2d\mathbf{x}\\ w_2=\int\limits_{\Im}kW^2dk & s_2=\int\limits_{I}S\cdot\frac{dS}{dt}dt & r_2=\int\limits_{\Omega}R\cdot\Delta R\,d\mathbf{x}\\ w_3=\int\limits_{\Im}W\,dk & s_3=\int\limits_{I}S\,dt & r_3=\int\limits_{\Omega}R\,d\mathbf{x}\\ w_4^i=\int\limits_{\Im}W\cdot K_i\,dk & s_4^i=\int\limits_{I}S\cdot\frac{dT_i}{dt}dt & r_4^i=\int\limits_{\Omega}R\cdot\Delta X_i\,d\mathbf{x}\\ w_5^i=\int\limits_{\Im}kW\cdot K_i\,dk & s_5^i=\int\limits_{I}S\cdot T_i\,dt & r_5^i=\int\limits_{\Omega}R\cdot X_i\,d\mathbf{x}\end{array}\right]\tag{11}$$

Equation (9) reduces to:

$$\begin{aligned}&\int\limits_{\Omega}R^*\cdot\left(w_1\cdot s_2\cdot R-w_2\cdot s_1\cdot\Delta R\right)\,d\mathbf{x}=\\&=-\int\limits_{\Omega}R^*\cdot\left(\sum_{i=1}^{i=n}w_4^i\cdot s_4^i\cdot X_i-\sum_{i=1}^{i=n}w_5^i\cdot s_5^i\cdot\Delta X_i-w_3\cdot s_3\cdot f\right)\,d\mathbf{x}\end{aligned}\tag{12}$$

Equation (12) defines an elliptic steady state boundary value problem that can be solved by using any discretization technique operating on the model weak form (finite elements, finite volumes, ...). Another possibility consists in coming back to the strong form of Equation (12):

$$\begin{aligned}&w_1\cdot s_2\cdot R-w_2\cdot s_1\cdot\Delta R=\\&=-\left(\sum_{i=1}^{i=n}w_4^i\cdot s_4^i\cdot X_i-\sum_{i=1}^{i=n}w_5^i\cdot s_5^i\cdot\Delta X_i-w_3\cdot s_3\cdot f\right)\end{aligned}\tag{13}$$

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

Computing $S\,(t)$ from $R\,(\mathbf{x})$ and $W\,(k)$:

In the present case, the test function writes:

$$u^*\,(\mathbf{x},t,k)=S^*\,(t)\cdot R\,(\mathbf{x})\cdot W\,(k)\tag{14}$$

Now, the weak form reads

$$\begin{aligned}&\int\limits_{\Omega\times I\times\Im}S^*\cdot R\cdot W\cdot\left(R\cdot\frac{\partial S}{\partial t}\cdot W-k\cdot\Delta R\cdot S\cdot W\right)\,d\mathbf{x}\,dt\,dk=\\&=-\int\limits_{\Omega\times I\times\Im}S^*\cdot R\cdot W\cdot\mathcal{R}^n\,d\mathbf{x}\,dt\,dk\end{aligned}\tag{15}$$

that integrating in the space $\Omega \times \Im$ and taking into account the notation (11) results in:

$$\begin{aligned} &\int_I S^* \cdot \left(w_1 \cdot r_1 \cdot \frac{dS}{dt} - w_2 \cdot r_2 \cdot S \right) dt = \\ &= - \int_I S^* \cdot \left(\sum_{i=1}^{i=n} w_4^i \cdot r_5^i \cdot \frac{dT_i}{dt} - \sum_{i=1}^{i=n} w_5^i \cdot r_4^i \cdot T_i - w_3 \cdot r_3 \cdot f \right) dt \end{aligned} \tag{16}$$

Equation (16) 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 Equation (16) reads:

$$\begin{aligned} &w_1 \cdot r_1 \cdot \frac{dS}{dt} - w_2 \cdot r_2 \cdot S = \\ &= - \left(\sum_{i=1}^{i=n} w_4^i \cdot r_5^i \cdot \frac{dT_i}{dt} - \sum_{i=1}^{i=n} w_5^i \cdot r_4^i \cdot T_i - w_3 \cdot r_3 \cdot f \right) \end{aligned} \tag{17}$$

than can be solved by using backward finite differences, or higher order Runge–Kutta schemes, among many other possibilities.

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

In the present case, the test function writes:

$$u^*(\mathbf{x}, t, k) = W^*(k) \cdot R(\mathbf{x}) \cdot S(t) \tag{18}$$

Now, the weak form reads

$$\begin{aligned} &\int_{\Omega \times I \times \Im} W^* \cdot R \cdot S \cdot \left(R \cdot \frac{\partial S}{\partial t} \cdot W - k \cdot \Delta R \cdot S \cdot W \right) d\mathbf{x}\, dt\, dk = \\ &= - \int_{\Omega \times I \times \Im} W^* \cdot R \cdot S \cdot \mathcal{R}^n \, d\mathbf{x}\, dt\, dk \end{aligned} \tag{19}$$

that integrating in the space $\Omega \times I$ and taking into account the notation (11) results in:

$$\begin{aligned} &\int_{\Im} W^* \cdot (r_1 \cdot s_2 \cdot W - r_2 \cdot s_1 \cdot k \cdot W)\, dk = \\ &= - \int_{\Im} W^* \cdot \left(\sum_{i=1}^{i=n} r_5^i \cdot s_4^i \cdot K_i - \sum_{i=1}^{i=n} r_4^i \cdot s_5^i \cdot k \cdot K_i - r_3 \cdot s_3 \cdot f \right) dk \end{aligned} \tag{20}$$

Equation (20) does not involve any differential operator. The strong form of Equation (20) reads:

$$(r_1 \cdot s_2 - r_2 \cdot s_1 \cdot k) \cdot W = - \left(\sum_{i=1}^{i=n} \left(r_5^i \cdot s_4^i - r_4^i \cdot s_5^i \cdot k \right) \cdot K_i - r_3 \cdot s_3 \cdot f \right) \tag{21}$$

that represents an algebraic equation. Thus, the introduction of parameters such as additional model coordinates has no noticeable effect in the computational cost, because the original equation does not contain derivatives with respect to those parameters.

Finally, note that other minimization strategies have been proposed, leading to more robust and faster convergence for building up the PGD [10]. The issues related to nonlinear behaviors, as for example, a thermal conductivity depending on the temperature field, was deeply considered in [5] and [19]. On the other hand, the issue related to the enforcement of nonhomogeneous boundary conditions was treated in [20].

Remark 1
The construction of each term in the sum (4) needs a certain number of iterations because of the nonlinearity of the problem related with the approximation (5). We denote by m_i the number of iterations that were needed for computing the i-sum in Equation (4). Let $m = \sum_{i=1}^{i=N} m_i$ be the total number of iterations involved in the construction of the separated approximation (4). It is easy to note that the solution procedure needs the solution of m $3D$ problems related to the construction of the space functions $X_i(\mathbf{x})$, $i = 1, \cdots, N$, m $1D$ ODEs related to the construction of functions $T_i(t)$ and m linear systems related to the definition of functions $K_i(k)$. In general, m_i rarely exceeds ten. On the other hand, the number of terms in the sum N needed to approximate the solution of a given problem depends on the solution regularity itself, but all the experiments carried out until now reveal that this number ranges from a few tens to a few hundreds. Thus, we can conclude that the complexity of the solution procedure is of some hundreds of $3D$ solutions (the cost related to the 1D problems being negligible with respect to the one related to the $3D$ problems). Now, if we assume a classical approach, one should solve a $3D$ problem at each time step and for each value of the parameter k. In usual applications, the complexity reaches millions of $3D$ solutions. In [12], we proved that the CPU time savings, by applying the PGD, can be of several orders of magnitude.

Remark 2
We noticed that, depending on the initial choice of functions $R(\mathbf{x})$, $S(t)$, and $W(k)$, the nonlinear iteration algorithm converges to different functions, but when they were normalized by dividing all of them by their respective norms, the resulting functions where the same independently of the initial choice.

3. PROPER GENERALIZED DECOMPOSITION TENSOR FORM

The procedure described in Section 2 can be generalized by using a tensor notation. We assume that the discrete problem that we write formally as $\mathcal{U}^{*T}\mathcal{A}\mathcal{U} = \mathcal{U}^{*T}\mathcal{B}$, can be expressed in a separated form:

$$\begin{aligned}
\mathcal{A} &= \sum_{i=1}^{n_A} \boldsymbol{A}_1^i \otimes \boldsymbol{A}_2^i \otimes \cdots \otimes \boldsymbol{A}_D^i \\
\mathcal{B} &= \sum_{i=1}^{n_B} \boldsymbol{B}_1^i \otimes \boldsymbol{B}_2^i \otimes \cdots \otimes \boldsymbol{B}_D^i \\
\mathcal{U} &\approx \sum_{i=1}^{N} \boldsymbol{u}_1^i \otimes \boldsymbol{u}_2^i \otimes \cdots \otimes \boldsymbol{u}_D^i
\end{aligned} \tag{22}$$

where $\boldsymbol{A}_i$, $\boldsymbol{B}_i$, and $\boldsymbol{u}_i$ involve only the coordinate x_i.

For an efficient application of the PGD, the differential operators must be separated. A separated representation can be used even if the operators are not separable, but in that case, the integration of the week form becomes cumbersome.

The separated representation of $\mathcal{A}$ and $\mathcal{B}$ comes directly from the differential operators involved in the PDE weak form.

At iteration n, vectors $\boldsymbol{u}_j^i$, $\forall i \leq n$, and $\forall j \leq D$ are assumed known. Now, we are looking for an enrichment:

$$\mathcal{U} = \sum_{i=1}^{n} \boldsymbol{u}_1^i \otimes \cdots \otimes \boldsymbol{u}_D^i + \boldsymbol{R}_1 \otimes \cdots \otimes \boldsymbol{R}_D \tag{23}$$

where $\boldsymbol{R}_i$, $i = 1, \cdots, D$, are the unknown enrichment vectors. We assume the following form of the test field:

$$\mathcal{U}^* = \boldsymbol{R}_1^* \otimes \boldsymbol{R}_2 \otimes \cdots \otimes \boldsymbol{R}_D + \ldots + \boldsymbol{R}_1 \otimes \cdots \otimes \boldsymbol{R}_{D-1} \otimes \boldsymbol{R}_D^* \tag{24}$$

Introducing the enriched approximation into the weak form, the following discrete form results:

$$\sum_{i=1}^{n_A}\sum_{j=1}^{n}(\boldsymbol{R}_1^*)^T\boldsymbol{A}_1^i\boldsymbol{u}_1^j\times\cdots\times(\boldsymbol{R}_D)^T\boldsymbol{A}_D^i\boldsymbol{u}_D^j+\cdots+$$
$$+\sum_{i=1}^{n_A}\sum_{j=1}^{n}(\boldsymbol{R}_1)^T\boldsymbol{A}_1^i\boldsymbol{u}_1^j\times\cdots\times(\boldsymbol{R}_D^*)^T\boldsymbol{A}_D^i\boldsymbol{u}_D^j+$$
$$+\sum_{i=1}^{n_A}(\boldsymbol{R}_1^*)^T\boldsymbol{A}_1^i\boldsymbol{R}_1\times\cdots\times(\boldsymbol{R}_D)^T\boldsymbol{A}_D^i\boldsymbol{R}_D+\cdots+$$
$$+\sum_{i=1}^{n_A}(\boldsymbol{R}_1)^T\boldsymbol{A}_1^i\boldsymbol{R}_1\times\cdots\times(\boldsymbol{R}_D^*)^T\boldsymbol{A}_D^i\boldsymbol{R}_D=$$
$$=\sum_{i=1}^{n_B}\left((\boldsymbol{R}_1^*)^T\boldsymbol{B}_1^i\times\cdots\times(\boldsymbol{R}_D)^T\boldsymbol{B}_D^i+\cdots+(\boldsymbol{R}_1)^T\boldsymbol{B}_1^i\times\cdots\times(\boldsymbol{R}_D^*)^T\boldsymbol{B}_D^i\right)\tag{25}$$

For alleviating the notation, we define:

$$\sum_{i=1}^{n_C}\boldsymbol{C}_1^i\otimes\cdots\otimes\boldsymbol{C}_D^i=\sum_{i=1}^{n_B}\boldsymbol{B}_1^i\otimes\cdots\otimes\boldsymbol{B}_D^i-\sum_{i=1}^{n_A}\sum_{j=1}^{n}\boldsymbol{A}_1^i\boldsymbol{u}_1^j\otimes\cdots\otimes\boldsymbol{A}_D^i\boldsymbol{u}_D^j\tag{26}$$

where $n_C = n_B + n_A \times n$. This sum only contains known fields. Thus, Equation (25) can be written as:

$$\sum_{i=1}^{n_A}(\boldsymbol{R}_1^*)^T\boldsymbol{A}_1^i\boldsymbol{R}_1\times\cdots\times(\boldsymbol{R}_D)^T\boldsymbol{A}_D^i\boldsymbol{R}_D+\cdots+$$
$$+\sum_{i=1}^{n_A}(\boldsymbol{R}_1)^T\boldsymbol{A}_1^i\boldsymbol{R}_1\times\cdots\times(\boldsymbol{R}_D^*)^T\boldsymbol{A}_D^i\boldsymbol{R}_D=$$
$$=\sum_{i=1}^{n_C}\left((\boldsymbol{R}_1^*)^T\boldsymbol{C}_1^i\times\cdots\times(\boldsymbol{R}_D)^T\boldsymbol{C}_D^i+\cdots+(\boldsymbol{R}_1)^T\boldsymbol{C}_1^i\times\cdots\times(\boldsymbol{R}_D^*)^T\boldsymbol{C}_D^i\right)\tag{27}$$

This problem is strongly nonlinear. To solve it, a method of alternated directions can be applied. The idea is, starting with the trial vectors $\boldsymbol{R}_i^{(0)}$, $i = 1,\cdots,D$ or assuming known these vectors at iteration $r-1$, $\boldsymbol{R}_i^{(r-1)}$, $i = 1,\cdots,D$, update them using an appropriate strategy. The simplest alternatives consist of:

- Update vectors $\boldsymbol{R}_i^{(r)}$, $\forall i$, from $\boldsymbol{R}_1^{(r-1)},\cdots,\boldsymbol{R}_{i-1}^{(r-1)},\boldsymbol{R}_{i+1}^{(r-1)},\cdots,\boldsymbol{R}_D^{(r-1)}$.
- Update vectors $\boldsymbol{R}_i^{(r)}$, $\forall i$, from $\boldsymbol{R}_1^{(r)},\cdots,\boldsymbol{R}_{i-1}^{(r)},\boldsymbol{R}_{i+1}^{(r-1)},\cdots,\boldsymbol{R}_D^{(r-1)}$.

The last strategy converges faster, but the advantage of the first one is the possibility of updating each vector simultaneously making use of a parallel computing platform. The fixed point of this iteration algorithm allows defining the enrichment vectors $\boldsymbol{u}_i^{n+1} = R_i, i = 1,\cdots,D$.

When we look for vector $\boldsymbol{R}_k$, assuming known all the others $\boldsymbol{R}_i, i \neq k$, the test field reduces to:

$$\mathcal{U}^{*T} = \boldsymbol{R}_1\otimes\cdots\otimes\boldsymbol{R}_{k-1}\otimes\boldsymbol{R}_k^*\otimes\boldsymbol{R}_{k+1}\cdots\otimes\boldsymbol{R}_D\tag{28}$$

The resulting discrete weak form writes:

$$\sum_{i=1}^{n_A}\left(\boldsymbol{R}_1^T\boldsymbol{A}_1^i\boldsymbol{R}_1\times\cdots\times\boldsymbol{R}_k^{*T}\boldsymbol{A}_k^i\boldsymbol{R}_k\times\cdots\times\boldsymbol{R}_D^T\boldsymbol{A}_D^i\boldsymbol{R}_D\right)=$$
$$=\sum_{i=1}^{n_C}\boldsymbol{R}_1^T\boldsymbol{C}_1^i\times\cdots\times\boldsymbol{R}_k^{*T}\boldsymbol{C}_k^i\times\cdots\times\boldsymbol{R}_D^T\boldsymbol{C}_D^i\tag{29}$$

Making use of the arbitrariness of $\boldsymbol{R}_K^*$, the following linear system can be easily obtained:

$$\left(\sum_{i=1}^{n_A}\left(\prod_{j=1,j\neq k}^{D}\boldsymbol{R}_j^T\boldsymbol{A}_j^i\boldsymbol{R}_j\right)\boldsymbol{A}_k^i\right)\boldsymbol{R}_k=\sum_{i=1}^{n_C}\left(\prod_{j=1,j\neq k}^{D}\boldsymbol{R}_j^T\boldsymbol{C}_j^i\right)\boldsymbol{C}_k^i \tag{30}$$

which can be easily solved.

4. INTRODUCING SOLUTION CONSTRAINTS

In numerical homogenization techniques, a classical procedure consists of considering different time scales as different dimensions of the problem. In principle, this seems to be specially well suited for a PGD treatment, especially if the number of scales is high. But the question is, now if the PGD can effectively treat the same problem, is there no neat separation of scales? Converting the time axis in a multidimensional space forces us to enforce the continuity on the resulting domain boundary. This section develops the PGD formulation in the presence of constraints and their tensor form, which are particularly convenient for code implementation.

Using the previous notation, we assume that the problem to be solved writes:

$$\mathcal{A}\mathcal{U}=\mathcal{B} \tag{31}$$

with

$$\mathcal{A}=\sum_{j=1}^{n_A}\boldsymbol{A}_1^j\otimes\boldsymbol{A}_2^j\otimes\ldots\otimes\boldsymbol{A}_D^j \tag{32}$$

$$\mathcal{B}=\sum_{j=1}^{n_B}\boldsymbol{B}_1^j\otimes\boldsymbol{B}_2^j\otimes\ldots\otimes\boldsymbol{B}_D^j \tag{33}$$

being the solution tensor form:

$$\mathcal{U}\approx\sum_{j=1}^{n_u}\boldsymbol{u}_1^j\otimes\boldsymbol{u}_2^j\otimes\ldots\otimes\boldsymbol{u}_D^j \tag{34}$$

The sizes of the different vectors are $r_1, r_2, .., r_D$, respectively, whereas the size of matrix $\boldsymbol{A}_i^j$ is $r_i\times r_i, i=1,\cdots,D$.

We assume in this section that the solution is searched under the constraints given by:

$$\mathcal{H}\mathcal{U}=\mathcal{J} \tag{35}$$

with

$$\mathcal{H}=\sum_{j=1}^{n_H}\boldsymbol{H}_1^j\otimes\boldsymbol{H}_2^j\otimes\ldots\otimes\boldsymbol{H}_D^j \tag{36}$$

$$\mathcal{J}=\sum_{j=1}^{n_J}\boldsymbol{J}_1^j\otimes\boldsymbol{J}_2^j\otimes\ldots\otimes\boldsymbol{J}_D^j \tag{37}$$

whose particular form will be illustrated later. In the last expression, the sizes of vectors $\boldsymbol{J}_i^j$ are assumed to be identical, s, being the sizes of matrices $\boldsymbol{H}_i^j$ are $s\times r_i$ with $i=1,\cdots,D$.

4.1. Penalty formulation

We introduce the functional to be minimized as

$$E = \|\mathcal{A}\mathcal{U} - \mathcal{B}\|_2 + \alpha \|\mathcal{H}\mathcal{U} - \mathcal{J}\|_2 \tag{38}$$

with α being the penalty parameter.

The previous equation can be rewritten as:

$$E = (\mathcal{A}\mathcal{U} - \mathcal{B})^T \cdot (\mathcal{A}\mathcal{U} - \mathcal{B}) + \alpha (\mathcal{H}\mathcal{U} - \mathcal{J})^T \cdot (\mathcal{H}\mathcal{U} - \mathcal{J}) \tag{39}$$

whose minimization with respect to the unknown field leads to:

$$\frac{dE}{d\mathcal{U}} = 0 \Rightarrow \left(\mathcal{A}^T\mathcal{A}\mathcal{U} - \mathcal{A}^T\mathcal{B}\right) + \alpha \left(\mathcal{H}^T\mathcal{H}\mathcal{U} - \mathcal{H}^T\mathcal{J}\right) = 0 \tag{40}$$

Finally, the system to be solved writes:

$$\left(\mathcal{A}^T\mathcal{A} + \alpha\mathcal{H}^T\mathcal{H}\right)\mathcal{U} = \mathcal{A}^T\mathcal{B} + \alpha\mathcal{H}^T\mathcal{J}. \tag{41}$$

Remark 3

This strategy deserves the following comments:

- The number of operators involved by $\left(\mathcal{A}^T\mathcal{A} + \alpha\mathcal{H}^T\mathcal{H}\right)$ results in $n_A^2 + n_H^2$.
- All the terms have dimensions $(r_1, r_1) \otimes \ldots \otimes (r_D, r_D)$.
- The best choice of the penalty parameter is far to be simple.
- The resulting system is symmetric, making possible the use of solution strategies like the one described in Section 2.

4.2. Lagrange multipliers

By introducing the Lagrange multiplier λ the system to be solved writes:

$$\underbrace{\begin{pmatrix} \mathcal{A} & \mathcal{H}^T \\ \mathcal{H} & 0 \end{pmatrix}}_{\mathcal{K}} \begin{pmatrix} \mathcal{U} \\ \lambda \end{pmatrix} = \underbrace{\begin{pmatrix} \mathcal{B} \\ \mathcal{J} \end{pmatrix}}_{\mathcal{F}} \tag{42}$$

where the size of $\mathcal{K}$ is $(r_1 + s, r_1 + s) \otimes \cdots \otimes (r_D + s, r_D + s)$ that can be written as:

$$\begin{aligned} \mathcal{K} = &\sum_{j=1}^{n_A} \begin{pmatrix} \boldsymbol{A}_1^j & \mathbf{0}_{(r_1,s)} \\ \mathbf{0}_{(s,r_1)} & \mathbf{0}_{(s,s)} \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} \boldsymbol{A}_D^j & \mathbf{0}_{(r_D,s)} \\ \mathbf{0}_{(s,r_D)} & \mathbf{0}_{(s,s)} \end{pmatrix} + \\ &\sum_{j=1}^{n_H} \begin{pmatrix} \mathbf{0}_{(r_1,r_1)} & \boldsymbol{H}_1^{jT} \\ \mathbf{0}_{(s,r_1)} & \mathbf{0}_{(s,s)} \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} \mathbf{0}_{(r_D,r_D)} & \boldsymbol{H}_D^{jT} \\ \mathbf{0}_{(s,r_D)} & \mathbf{0}_{(s,s)} \end{pmatrix} + \\ &\sum_{j=1}^{n_H} \begin{pmatrix} \mathbf{0}_{(r_1,r_1)} & \mathbf{0}_{(r_D,s)} \\ \boldsymbol{H}_1^j & \mathbf{0}_{(s,s)} \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} \mathbf{0}_{(r_D,r_D)} & \mathbf{0}_{(r_D,s)} \\ \boldsymbol{H}_D^j & \mathbf{0}_{(s,s)} \end{pmatrix} \end{aligned} \tag{43}$$

expression that involves $n_A + 2n_H$ terms.

Concerning the right hand member, we have

$$\mathcal{F} = \sum_{j=1}^{n_B} \begin{pmatrix} \boldsymbol{B}_1^j \\ \mathbf{0}_{(s,1)} \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} \boldsymbol{B}_D^j \\ \mathbf{0}_{(s,1)} \end{pmatrix} + \sum_{j=1}^{n_J} \begin{pmatrix} \mathbf{0}_{(r_1,1)} \\ \boldsymbol{J}_1^j \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} \mathbf{0}_{(r_D,1)} \\ \boldsymbol{J}_D^j \end{pmatrix}. \tag{44}$$

The searched solution writes in the present case:

$$\overline{\mathcal{U}} = \begin{pmatrix} \mathcal{U} \\ \lambda \end{pmatrix} = \sum_{j=1}^{n_F} \boldsymbol{F}_1^j \otimes \boldsymbol{F}_2^j \otimes \cdots \otimes \boldsymbol{F}_D^j \tag{45}$$

which involves a tensor product of vectors of size $(r_1+s,1)\otimes\cdots\otimes(r_D+s,1)$ from which we can easily extract the problem solution $\mathcal{U}$ from

$$\mathcal{U}=\sum_{j=1}^{n_u}\boldsymbol{u}_1^j\otimes\cdots\otimes\boldsymbol{u}_D^j \tag{46}$$

where vector $\boldsymbol{u}_i$ contains the first r_i entries of vector $\boldsymbol{F}_i$. Representation (46) defines a non-optimal decomposition that could be reduced by applying a multidimensional singular value decomposition [10].

Remark 4
This strategy deserves the following comments:

- The main drawback of this approach lies in the fact that the resulting system is nonsymmetric. It could be symmetrized by minimizing the L^2 norm of the residual:

  $$E=\left\|\,\mathcal{K}\overline{\mathcal{U}}-\mathcal{F}\right\|_2 \tag{47}$$

  whose minimization leads to:

  $$\frac{dE}{d\overline{\mathcal{U}}}=0 \tag{48}$$

  that is equivalent to:

  $$\mathcal{K}^T\mathcal{K}\overline{\mathcal{U}}=\mathcal{K}^T\mathcal{F} \tag{49}$$

- The number of terms involved in $\mathcal{K}^T\mathcal{K}$ is $(n_A+2n_H)^2$.
- The main advantage of this approach is the absence of any adjustable parameter.

4.3. Illustrating the prescription of constraints through a simple example

In this section, we are defining the constraints tensor form related to a quite simple example that will be used later for separating the time axis.

The generic tensor form related to the solution constraints writes, using the notation just introduced, as

$$\mathcal{H}\,\mathcal{U}=\mathcal{J} \tag{50}$$

where vectors $\boldsymbol{J}_i^j$ have the same size s and matrices $\boldsymbol{H}_i^j$ are of size $s\times r_i$ with $i=1,\cdots,D$.

We assume the situation depicted in Figure 1 that represents a 2D model defined in a square domain whose axes x_1 and x_2 were discretized by using $three$ and $four$ nodes, respectively, that is, $r_1=3$ and $r_2=4$.

We would like to prescribe the following constraints in the discrete model:

- Prescribe the value u_d at the node symbolized by a square in Figure 1.
- Prescribe the same value of the solution at nodes symbolized in Figure 1 by a triangle.
- Prescribe the same value of the solution at nodes symbolized in Figure 1 by a star.

These three conditions write:

$$\left\{\begin{pmatrix}1&0&0\end{pmatrix}\otimes\begin{pmatrix}1&0&0&0\end{pmatrix}\right\}\mathcal{U}=u_d\otimes 1 \tag{51}$$

$$\left\{\begin{pmatrix}1&0&0\end{pmatrix}\otimes\begin{pmatrix}0&0&0&1\end{pmatrix}+\begin{pmatrix}0&1&0\end{pmatrix}\otimes\begin{pmatrix}-1&0&0&0\end{pmatrix}\right\}\mathcal{U}=0\otimes 0 \tag{52}$$

$$\left\{\begin{pmatrix}0&1&0\end{pmatrix}\otimes\begin{pmatrix}0&0&0&1\end{pmatrix}+\begin{pmatrix}0&0&1\end{pmatrix}\otimes\begin{pmatrix}-1&0&0&0\end{pmatrix}\right\}\mathcal{U}=0\otimes 0 \tag{53}$$

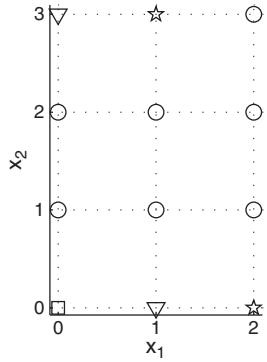

Figure 1. 2D problem with constraints.

whose matrix form reads as:

$$\begin{aligned}\mathcal{H} = &\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} + \\ &\begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} + \\ &\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \otimes \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{pmatrix}\end{aligned} \tag{54}$$

The right-hand member results in:

$$\mathcal{J} = \begin{pmatrix} u_d \\ 0 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{55}$$

5. TIME-MULTISCALE PGD FORMULATION WITHOUT SEPARATION OF SCALES

In this section, we are applying the procedure described in the previous section for separating the time axis, firstly, from 1D in nature into a 2D domain, and then to more complex scenarios.

5.1. Solving an ordinary differential equation by increasing the temporal dimensionality

For this purpose, we should define the structure of the discrete problem expressed by:

$$\mathcal{A}\mathcal{U} = \mathcal{B} \tag{56}$$

related to the transient ODE:

$$\frac{du}{dt} = f(t) \tag{57}$$

that we would integrate in the whole time interval using $r_1 \cdot (r_2 - 1)$ time steps. When the number of time steps is excessive, we could divide the whole time interval $[0, t_{\max}]$ into r_1 intervals $\left[0, \frac{t_{\max}}{r_1}\right] \cdots \left[\frac{t_{\max}(r_1-1)}{r_1}, t_{\max}\right]$. It is important to note that the end of each one of these intervals corresponds to the beginning of the next one. Thus, we define two dimensions, the first one discrete

$$x_1 = 0, 1, \cdots, (r_1 - 1) \tag{58}$$

and the second one continuous

$$x_2 = \left[0, \frac{t_{\max}}{r_1}\right]. \tag{59}$$

Now, the transient equation can be rewritten in the 2D time space as:

$$\frac{\partial}{\partial x_2} u(x_1, x_2) = f(x_1, x_2) \tag{60}$$

The discrete model writes:

$$\mathcal{A} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}_{(r_1, r_1)} \otimes \frac{1}{\Delta t} \begin{pmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{pmatrix}_{(r_2, r_2)} \tag{61}$$

with

$$\Delta t = \frac{t_{\max}}{r_1 \cdot (r_2 - 1)} \tag{62}$$

and

$$\mathcal{B} = \text{svd}\,(f(x_1, x_2)) \tag{63}$$

were the singular value decomposition is applied after discretizing the continuous coordinate x_2. Obviously, the separation of $f(x_1, x_2)$ is a delicate point. There are different scenarios: (1) proceed to a continuous "singular value decomposition" separation when it is possible as described in [21]; (2) use the coarsest discrete description and then project on the finest mesh; and (3) in some cases, the source term comes from the solution of other equation (in coupled models), and in that case, it has the appropriate separated form.

In what follows, we consider the source term of the transient model given by

$$f(t) = 2t\cos^2(\omega t) - 2t^2\omega\cos(\omega t)\sin(\omega t) \tag{64}$$

with $\omega = 10$ and $t_{max} = 5$. The solution is performed by assuming $r_1 = 10$ (that constitutes the first, coarse, time coordinate that will be designated by t) intervals, each one involving $r_2 = 41$ time steps (that constitute the second time coordinate, the finest one, designated by τ).

The solution is then searched in the separated form:

$$u(t, \tau) \approx \sum_{i=1}^{i=N} F_1^i(t) \cdot F_2^i(\tau) \tag{65}$$

by enforcing both the verification of the equation as well as the continuity between the end of one interval and the beginning of the next one. This enforcement, either by penalization or by Lagrange multipliers, was deeply addressed in the previous section.

Note that, in the present approach, no neat separation of scales is assumed, in deep contrast with the time homogenization approach in [13]. In that work that assumes an additive (instead of multiplicative) enrichment of the macro time scale, the response fields are assumed to be periodic with respect to the fine scale, which is not the case here. Note also that in [17] a POD-like approach is employed, in which an averaged fine-scale time evolution is employed, leading to discontinuous time evolutions. This makes it necessary to employ discontinuous Galerkin approaches in time. In the approach here presented, the continuity between time scales is imposed explicitly, as commented before.

Figure 2 depicts the different functions $F_1^i(t)$ and $F_2^i(\tau)$ involved in the solution computed by applying the penalty enforcement of the continuity conditions. Figure 3 shows the computed solution $u(t, \tau)$. Similar results were obtained when applying the Lagrange multiplier strategy.

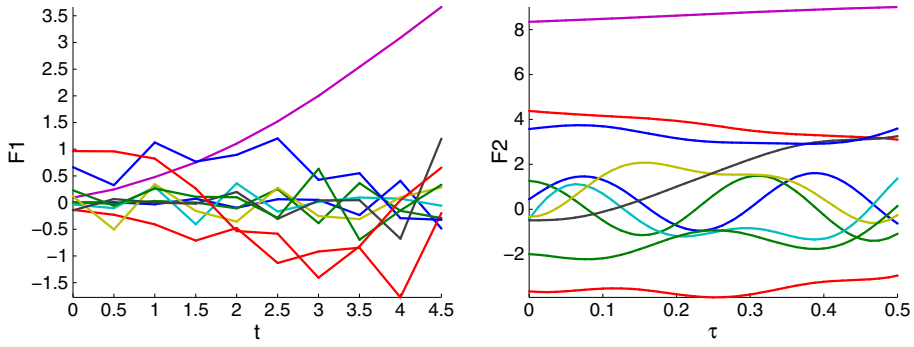

Figure 2. Functions involved in the separated representation using a penalty technique for enforcing the continuity constraints: $F_1^i(t)$ (left) and $F_2^i(\tau)$ (right).

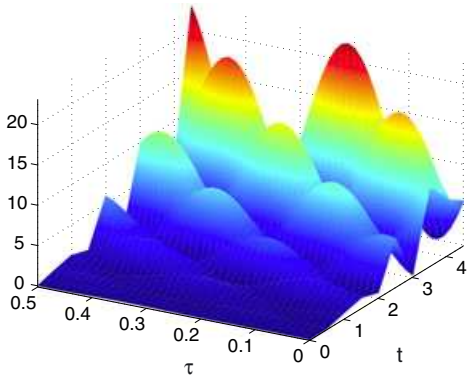

Figure 3. Separated representation solution $u(t,\tau)$ computed using a penalty technique for enforcing the continuity constraints.

Finally, both 2D solutions (the one computed by applying the penalty strategy and the one obtained by using Lagrange multipliers) can be pushed down for defining the associated 1D solutions.

When the continuity is enforced by using a penalty technique, the computed solution for different number of terms in the finite sum (65) is depicted in Figure 4. Figure 5 depicts similar results when the continuity is enforced by using lagrange multipliers. We can notice in these figures that by using 10 terms in the sum (65), the separated representation solutions agree in minute to the reference ones computed by using a standard fully incremental integration.

Figure 6 compares the convergence obtained by using both the Lagrange multipliers and the penalty strategies. We can notice that, as expected, the convergence is better when applying Lagrange multipliers. On the other hand Figure 7 compare the computing time needed for constructing both separated representations, and proves that Lagrange multipliers is more expensive from the computational time viewpoint.

We can compare the complexities related to a fully incremental integration and the one related to the construction of separated representation. In the fully incremental integration, we should compute the solution at each one of the $r_1 \cdot (r_2 - 1)$ time steps that quantify the solution complexity. When a separated representation is built, one should compute $N \cdot m$ ODE integrations (m being the iterations involved in the nonlinear solver used for computing each functional product in Equation (65)) in the intervals involving $r_2 - 1$ time steps. Thus, the complexity results $N \cdot m \cdot (r_2 - 1)$. By comparing both complexities, we can conclude that if $N \cdot m < r_1$ the time axis separation could be an appealing solution for speeding up the integration of transient models. In general, N is of the order of few tens and m is lower than 10. This implies that r_1 should be greater than 100 to induce CPU time savings. However, in this paper, we are more interested in analyzing the possible multidimensional description of the time axis than in proving its numerical advantages.

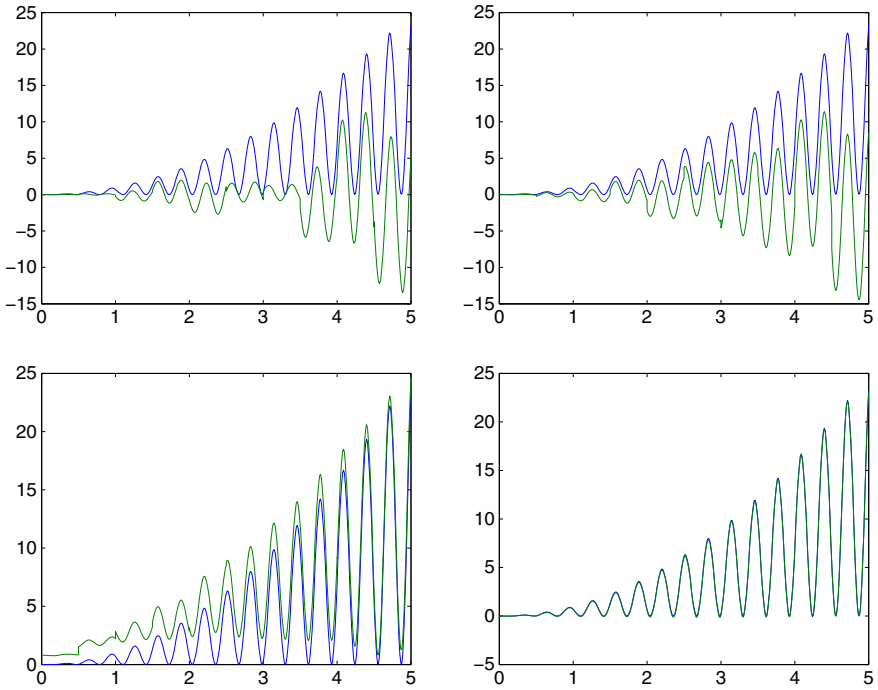

Figure 4. Comparison of the separated solution and the fully incremental one computed for different number of terms N in the separated representation: (top-left) $N = 1$, (top-right) $N = 2$, (bottom-left) $N = 3$, and (bottom-right) $N = 10$, when a penalty strategy was applied. The green curve represents the approximated solution, whereas the blue one is the reference, exact, solution.

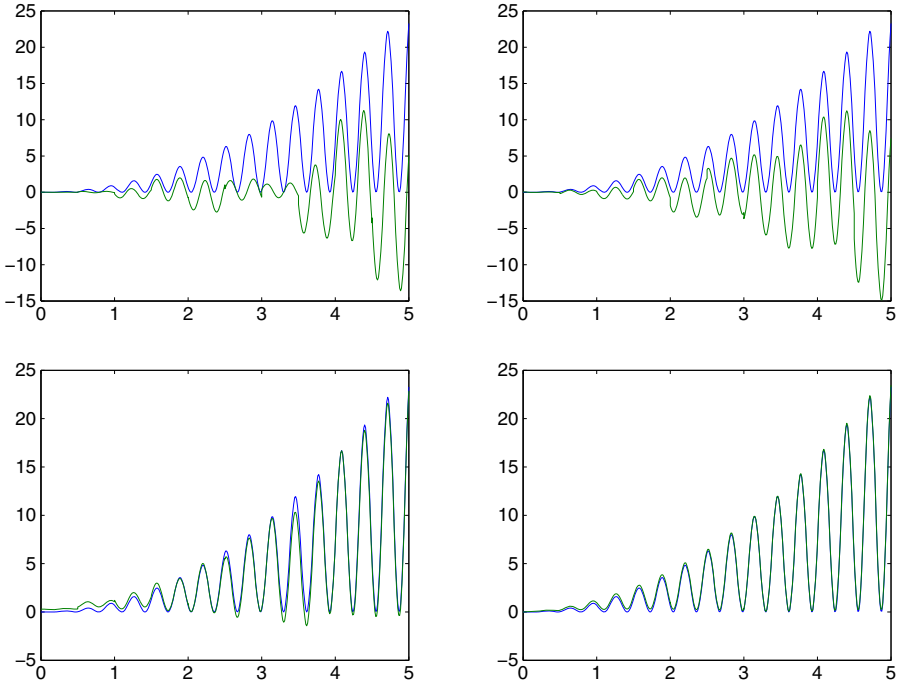

Figure 5. Comparison of the separated solution and the fully incremental one computed for different number of terms N in the separated representation: (top-left) $N = 1$, (top-right) $N = 2$, (bottom-left) $N = 3$, and (bottom-right) $N = 10$, when a Lagrange multiplier was introduced. The green curve represents the approximated solution, whereas the blue one is the reference, exact, solution.

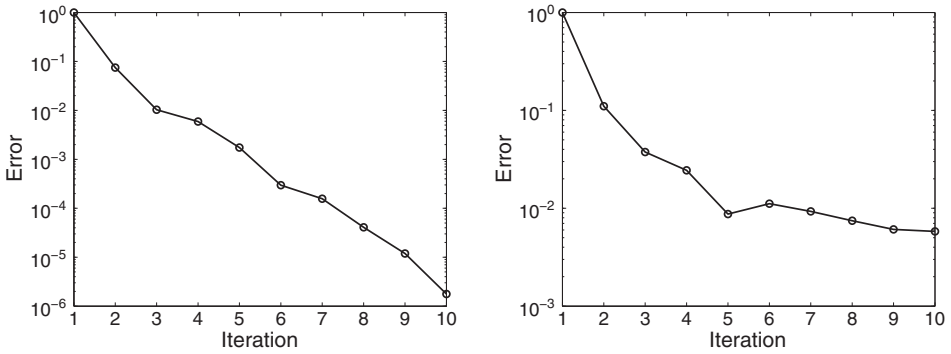

Figure 6. Error versus the number of iterations (number of terms in the finite sums decomposition) when using Lagrange multipliers (left) and penalty (right) strategies for enforcing the solution continuity.

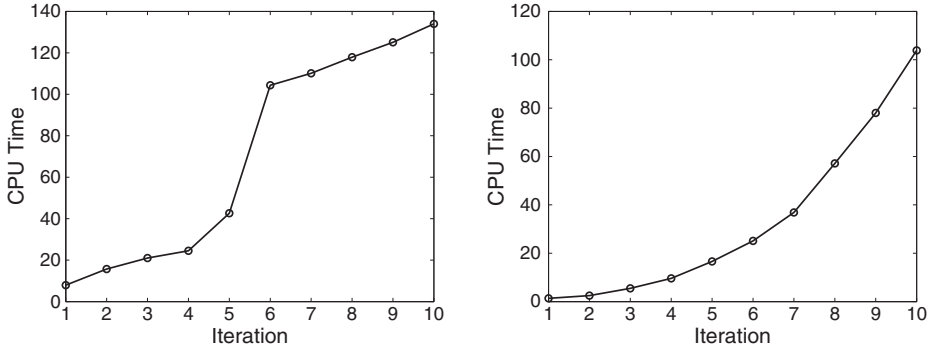

Figure 7. CPU time versus the number of iterations (number of terms in the finite sums decomposition) when using Lagrange multipliers (left) and penalty (right) strategies for enforcing the solution continuity.

5.2. Solving time-multiscale PDE by increasing the temporal dimensionality

Until now, our attention focused in the solution of time-multiscale ODE. In the present section, we are considering models involving time and space coordinates (PDEs) and exploring the applicability of a decomposition of the time axis in a higher dimensional time space.

For this purpose, we consider the transient PDE equation

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + f(x,t), \quad (x,t) \in (0,\pi) \times (0, t_{max}] \tag{66}$$

where

$$f(x,t) = \left(2t\cos^2(\omega t) - 2t^2\omega\cos(\omega t)\sin(\omega t) + t^2\cos^2(\omega t)\right)\cdot\sin(x) \tag{67}$$

with $u(x=0,t) = u(x=\pi,t) = 0$, and $u(x,t=0) = 0$.

The exact solution of Equation (66) writes:

$$u^{ex}(x,t) = t^2\cos^2(\omega t)\sin(x). \tag{68}$$

For its numerical solution, we make a partition of the whole time interval $[0, t_{\max}]$ into r_1 intervals $\left[0, \frac{t_{\max}}{r_1}\right]\cdots\left[\frac{t_{\max}(r_1-1)}{r_1}, t_{\max}\right]$. In what follows, we consider $t_{\max} = 5$, $r_1 = 10$, and $\omega = 10$.

Thus, we define two time dimensions, the first one discrete

$$x_1 = 0, 1, \cdots, (r_1 - 1) \tag{69}$$

and the second one continuous

$$x_2 = \left[0, \frac{t_{\max}}{r_1}\right]. \tag{70}$$

Moreover, the space dimension x_3, in the present case and without loss of generality, 1D, is discretized by using r_3 nodes.

Now, the transient equation can be rewritten in the resulting 3D space as:

$$\frac{\partial u(x_1, x_2, x_3)}{\partial x_2} = \frac{\partial^2 u(x_1, x_2, x_3)}{\partial x_3^2} + f(x_1, x_2, x_3) \tag{71}$$

For the solution, we consider the Lagrange multiplier strategy for enforcing the continuity with respect to the time coordinate that considers a Lagrange multiplier for each node of the spatial discretization.

Figures 8 and 9 depict the time and space functions respectively involved in the separated representation. Figure 10 shows the reconstructed space–time solution post-treated from the 3D separated representation, where the two time coordinates were pushed down for defining a single 1D time axis. This solution is compared with the exact solution, and, as expected, they are in perfect agreement. In order to appreciate the solution features, Figure 11 depicts another view of the computed solution in order to emphasize the presence of fast oscillations.

6. AN ALTERNATIVE FORMULATION

When the number of dimensions increases, so does the complexity associated with the imposition of time continuity, as discussed in Section 4.3. In this section, an alternative method is developed that has rendered excellent results when applied to ODEs and PDEs.

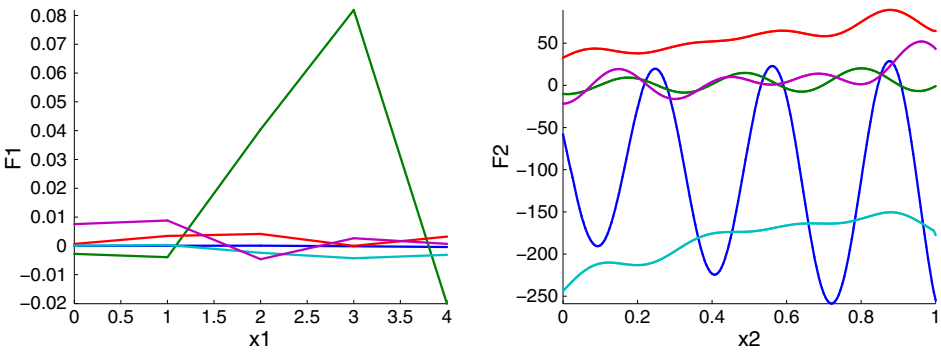

Figure 8. Functions of time involved in the separated representation using Lagrange multipliers for enforcing the continuity constraints: $F_1^i(x_1)$ (left) and $F_2^i(x_2)$ (right).

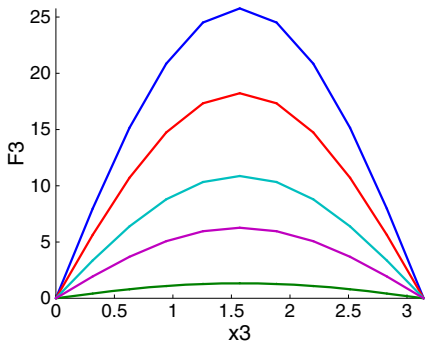

Figure 9. Functions of space $F_3^i(x_3)$ involved in the separated representation using Lagrange multipliers for enforcing the continuity constraints.

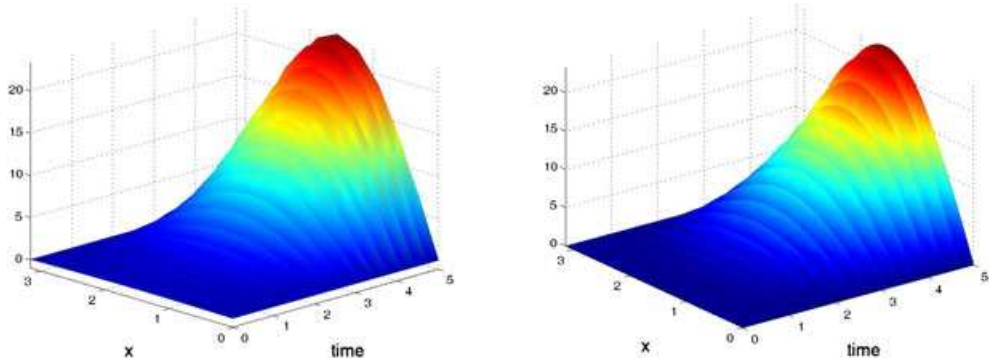

Figure 10. Space–time reconstructed solution $u(x,t)$ computed from the separated representation whose involved functions were depicted in Figures 8 and 9 (left) and exact solution (right).

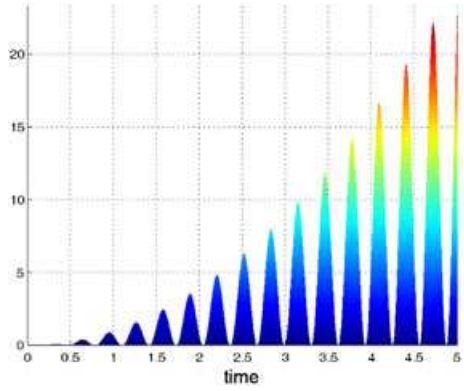

Figure 11. Another view of the space–time reconstructed solution $u(x,t)$ depicted in Figure 10 (left).

6.1. Application to ODEs

To illustrate the method, we consider firstly the ODE already considered in Section 5.1

$$\frac{\partial u}{\partial t} = f(t) = 2t\cos^2(\omega t) - 2t^2\omega\cos(\omega t)\sin(\omega t) \tag{72}$$

with $u(t=0) = 0$.

Because of the linearity of Equation (72), an alternative method to find a numerical solution to this problem consists in writing

$$u(t) = u_{\text{gen, hom}}(t) + u_{\text{part, compl}}(t) \tag{73}$$

that is, express the solution of Equation (72) as the sum of the general solution of the homogeneous equation plus a particular solution of the complete equation. By performing the time-axis decomposition, it results in

$$u(t,\tau) = u_{\text{gen, hom}}(t,\tau) + u_{\text{part, compl}}(t,\tau). \tag{74}$$

Concerning the particular solution, it is sought in the separated form:

$$u_{\text{part, compl}}(t,\tau) \approx \sum_{j=1}^{Q} F_t^j(t)\cdot F_\tau^j(\tau) \tag{75}$$

Because we are looking for a particular solution, we could enforce it to vanish on the boundary $\tau = 0$, that is, $u_{\text{part, compl}}(t,\tau=0) = 0$. This condition can be obtained by enforcing $F_\tau^j(\tau=0) = 0$, $j = 1,\ldots,Q$.

The second part of the solution derives from the homogeneous equation whose separated form reads:

$$u_{\text{hom}}(t,\tau) \approx \sum_{j=1}^{Q'} H_t^j(t) \cdot H_\tau^j(\tau) \tag{76}$$

that is integrated by enforcing the boundary condition $u_{\text{hom}}(t, \tau = 0) = 1$. For this purpose, we prescribe $H_t^1(t) = 1$ and $H_\tau^1(\tau = 0) = 1$ and $H_\tau^j(\tau = 0) = 0$, $j = 2, \cdots, Q'$.

Thus, the general solution of the homogeneous equation reads:

$$u_{\text{gen, hom}}(t_i, \tau) = \alpha_i \cdot u_{\text{hom}}(t_i, \tau) \tag{77}$$

The coefficients α_i are determined by enforcing the solution time-continuity, by solving sequentially

$$\begin{cases} \alpha_1 = u(t=0) = 0 \\ u_{\text{part, compl}}(t_i, \Delta\tau) + \alpha_i \cdot u_{\text{hom}}(t_i, \Delta\tau) = \alpha_{i+1}, \;\; i = 1, \ldots, r_1 - 1 \end{cases} \tag{78}$$

where $\Delta\tau = t_{max}/r_1$.

By using this strategy, the solution of the problem is obtained in only one iteration. The result agrees in minute with the reference one as shown in Figure 12.

Because of the simplicity of the operators involved in this description, with respect to the Lagrange multiplier strategy discussed previously, the computing time savings in the example just treated is of two orders of magnitude.

Figure 13 compares the evolution of the error with the number of terms considered in the separated representation of the general solution of the homogenous equation Q' and the particular solution of the complete equation Q.

For a given accuracy, the number of terms is much lower than that of the one needed when applying the strategies described in Section 5. That is, for the same number of functions in the separated representation of the solution, the one related to the technique described in this section is better than that of the ones computed by using the strategies described in Section 5.

This strategy is generalizable to transient PDEs.

6.2. Application to PDEs

For illustrating the application to discretize PDE, we consider the model

$$\frac{\partial u}{\partial t} - \Delta u = f(\mathbf{x}, t) \tag{79}$$

with, for the sake of simplicity, homogeneous initial and boundary conditions. We consider the model defined in a $2D$ space domain Ω, $\mathbf{x} = (x, y) \in \Omega$, with $t \in (0, t_{max}]$.

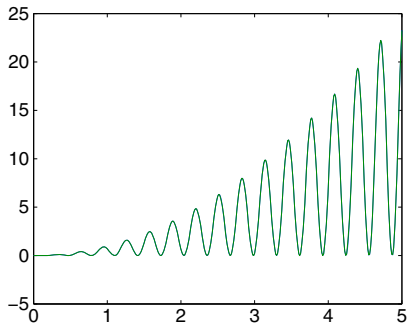

Figure 12. Result for the ODE obtained with the technique explained in Section 6.1.

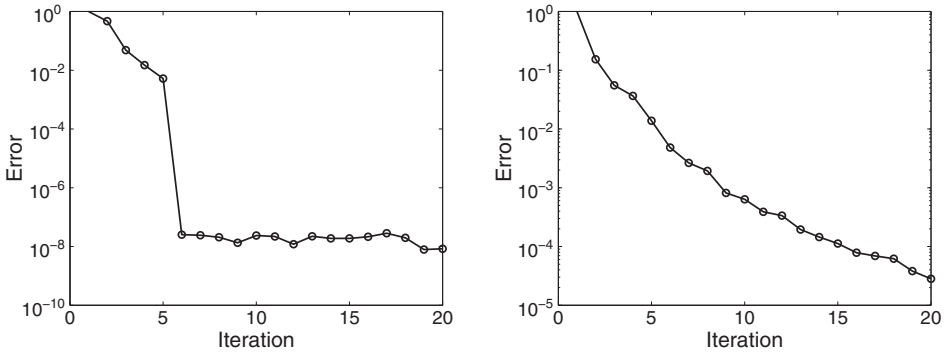

Figure 13. Error versus the number of iterations (number of terms in the separated representation) in the particular solution of the complete equation Q (left) and in the general solution of the homogeneous equation Q' (right).

We compute the particular solution of the complete equation in the separated form

$$u_{\text{part, compl}}(\mathbf{x}, t, \tau) \approx \sum_{j=1}^{Q} F_{\mathbf{x}}^{j}(\mathbf{x}) \cdot F_{t}^{j}(t) \cdot F_{\tau}^{j}(\tau) \tag{80}$$

with $u_{\text{part, compl}}(\mathbf{x}, t, \tau = 0) = 0$.

The general solution of the homogeneous equation needs solving

$$\frac{\partial u}{\partial t} - \Delta u = 0 \tag{81}$$

for any initial condition.

If we are considering a mesh composed of M nodes for approximating the field u in the space domain Ω, we could solve the following M homogeneous problems

$$\left.\begin{array}{l} \frac{\partial u_{\text{hom}}^{i}}{\partial t} - \Delta u_{\text{hom}}^{i} = 0 \\ u_{\text{hom}}^{i}(\mathbf{x}, t, \tau = 0) = N_i(\mathbf{x}) \end{array}\right\} \quad i = 1, \ldots, M \tag{82}$$

with $N_i(\mathbf{x})$ the elements of a discrete space approximation basis. In the context of finite element techniques, $N_i(\mathbf{x})$ represent the standard shape functions verifying the Kronecker's delta property, that is, $N_i(\mathbf{x}_k) = \delta_{ik}$.

The preceding solutions are sought in the separated form:

$$u_{\text{hom}}^{i}(\mathbf{x}, t, \tau) \approx \sum_{j=1}^{Q'} F_{\mathbf{x},i}^{j}(\mathbf{x}) \cdot F_{t,i}^{j}(t) \cdot F_{\tau,i}^{j}(\tau), \quad i = 1, \ldots, M \tag{83}$$

With all these solutions $u_{\text{hom}}^{i}(\mathbf{x}, t, \tau)$, $i = 1, \cdots, M$, known, the general solution of the homogeneous equation reads:

$$u_{\text{gen,hom}}(\mathbf{x}, t_i, \tau) = \sum_{j=1}^{j=M} \alpha_j^i \cdot u_{\text{hom}}^{j}(\mathbf{x}, t_i, \tau) \tag{84}$$

where the coefficients α_j^i are computed by enforcing the solution time-continuity. Considering a simple nodal collocation for enforcing the time continuity, we can write, for $i = 1, \cdots, r_1 - 1$:

$$u_{\text{part, compl}}(\mathbf{x}_k, t_i, \tau = \Delta\tau) + \sum_{j=1}^{j=M} \alpha_j^i \cdot u_{\text{hom}}^j(\mathbf{x}_k, t_i, \tau = \Delta\tau) =$$

$$= \sum_{j=1}^{j=M} \alpha_j^{i+1} \cdot N_j(\mathbf{x}_k) = \alpha_k^{i+1}, \quad k = 1, \cdots, M \tag{85}$$

where the fact that $N_j(\mathbf{x}_k) = \delta_{jk}$ has been used.

Coefficients α_k^1 are computed from the initial condition $u(\mathbf{x}, t = 0) = u_g(\mathbf{x})$:

$$u_g(\mathbf{x}_k) = u_{\text{part, compl}}(\mathbf{x}_k, t_1, \tau = 0) + \sum_{j=1}^{j=M} \alpha_j^1 \cdot u_{\text{hom}}^j(\mathbf{x}_k, t_1, \tau = 0) =$$

$$= \sum_{j=1}^{j=M} \alpha_j^1 \cdot \ N_j(\mathbf{x}_k) = \alpha_k^1, \quad k = 1, \cdots, M \tag{86}$$

This technique could be applied in the nonlinear case by performing a linearization prior to proceeding to the solution. However, the main drawback of such an approach lies in the necessity of solving M homogeneous equations. In some applications, M can be extremely high. Obviously, if M is of the same order of magnitude than that of r_1, the solution procedure has the same complexity than that of a standard incremental solution. For circumventing this difficulty, we propose considering an adaptive reduced basis in order to enforce the initial condition as well as the solution continuity in time.

Thus, we consider two different space meshes: one, fine enough, for approximating the space functions involved in the separated representation $F_{\mathbf{x},i}^j(\mathbf{x})$, and $F_{\mathbf{x}}^j(\mathbf{x})$; and the second one, $N_i(\mathbf{x})$, coarser, for enforcing the time-continuity. Because both associated approximation spaces are different by enforcing the time-continuity at the M nodes related to the coarsest mesh, we do not ensure the perfect continuity. The resulting gap could be used to define a criterion for refining the coarsest mesh, the one used to enforce the continuity. However, if one considers standard polynomial approximation bases, as soon as a refinement is performed, the whole solution process must be repeated again. For alleviating the solution process, one could consider a hierarchical basis (very well known in the finite element framework) because in that case, the addition of a new approximation function allows to keep all the work already done. Thus, if we consider the hierarchical basis $H_i(\mathbf{x})$ instead of the classical one $N_i(\mathbf{x})$, with $i = 1, \ldots, M$, if we enrich the discrete approximation space, consisting now in M' approximation functions $H_i'(\mathbf{x})$, $M' > M$, the M first approximation functions in the refined basis remains unchanged, that is, $H_i'(\mathbf{x}) = H_i(\mathbf{x})$, $i = 1, \ldots M$. For standard finite element bases, as soon as we change the space dimension, the approximation functions change. Thus, when we proceed to refine a hierarchical approximation basis, the work already done is reused.

When using a hierarchical approximation basis, the solution procedure is slightly modified. If we are considering a discrete hierarchical basis of size M, we should solve the following M homogeneous problems:

$$\left.\begin{array}{l} \frac{\partial u_{\text{hom}}^i}{\partial t} - \Delta u_{\text{hom}}^i = 0 \\ u_{\text{hom}}^i(\mathbf{x}, t, \tau = 0) = H_i(\mathbf{x}) \end{array}\right\} \quad i = 1, \ldots, M \tag{87}$$

The preceding solutions are sought in the separated form:

$$u_{\text{hom}}^i(\mathbf{x}, t, \tau) \approx \sum_{j=1}^{Q'} F_{\mathbf{x},i}^j(\mathbf{x}) \cdot F_{t,i}^j(t) \cdot F_{\tau,i}^j(\tau) \tag{88}$$

where functions $F_{\mathbf{x},i}^j$ are approximated using a fine enough approximation basis.

With all these solutions $u^i_{\text{hom}}(\mathbf{x}, t, \tau)$, $i = 1, \cdots, M$, known, the general solution of the homogeneous equation reads:

$$u_{\text{gen,hom}}(\mathbf{x}, t_i, \tau) = \sum_{j=1}^{j=M} \alpha^i_j \cdot u^j_{\text{hom}}(\mathbf{x}, t_i, \tau) \tag{89}$$

where the coefficients α^i_j are computed by enforcing the solution continuity in time. Considering a simple nodal collocation at M locations $\mathbf{x}_k$ for enforcing the time-continuity, we can write, for $i = 1, \cdots, r_1 - 1$:

$$\begin{aligned} u_{\text{part, compl}}(\mathbf{x}_k, t_i, \tau = \Delta\tau) + \sum_{j=1}^{j=M} \alpha^i_j \cdot u^j_{\text{hom}}(\mathbf{x}_k, t_i, \tau = \Delta\tau) = \\ = \sum_{j=1}^{j=M} \alpha^{i+1}_j \cdot H_j(\mathbf{x}_k), \quad k = 1, \cdots, M \end{aligned} \tag{90}$$

Coefficients α^1_k are computed from the initial condition $u(\mathbf{x}, t = 0) = u_g(\mathbf{x})$:

$$u_g(\mathbf{x}_k) = u_{\text{part, compl}}(\mathbf{x}_k, t_1, \tau = 0) + \sum_{j=1}^{j=M} \alpha^1_j \cdot u^j_{\text{hom}}(\mathbf{x}_k, t_1, \tau = 0) = \sum_{j=1}^{j=M} \alpha^1_j \cdot \ H_j(\mathbf{x}_k) \tag{91}$$

for $k = 1, \ldots, M$.

Obviously, by enforcing the solution continuity in time at positions $\mathbf{x}_k$, $k = 1, \cdots, M$, we do not ensure the continuity in time everywhere. Thus, the resulting gap can be used as a criterion for enriching the coarse approximation space whose size becomes $M' > M$. However, because its hierarchical nature of the approximation functions, all the already computed solutions $u^i_{\text{hom}}(\mathbf{x}, t, \tau)$, $i = 1, \ldots, M$, remain unchanged, and then, we only need to compute the $M' - M$ new homogeneous problems

$$\left.\begin{array}{l} \frac{\partial u^i_{\text{hom}}}{\partial t} - \Delta u^i_{\text{hom}} = 0 \\ u^i_{\text{hom}}(\mathbf{x}, t, \tau = 0) = H_i(\mathbf{x}) \end{array}\right\} \quad i = M + 1, \ldots, M' \tag{92}$$

For illustrating the procedure, we are considering the linear parabolic PDE

$$\frac{\partial u}{\partial t} - \Delta u = (x^2 - y^2) \cdot \sin(\omega \cdot t) \tag{93}$$

with homogeneous initial and boundary conditions. We consider $\mathbf{x} \in \Omega$ with Ω depicted in Figure 14, $t \in (0, 1]$, and $\omega = 45$.

The hierarchical approximation basis consists of the Szabo and Cernevali basis involving vertex, edge, and face shape functions [22]. The hierarchical basis is associated to the coarse mesh depicted in Figure 15. Figure 16 superpose the fine mesh used for solving the different equations, and the corse mesh used for defining the hierarchical basis employed for enforcing the continuity in time.

The zero level of the hierarchical basis consists on the standard vertex linear finite element shape functions. The first level consists of edge functions. In Figure 17, we depict the ones related to one of the triangles of the mesh shown in Figure 15.

The second level consists of other edge shape function as well as a face function defined on each triangle. Figure 18 depicts those functions for one of the triangles involved in the hierarchical mesh.

Because of the mesh used in our calculations (Figure 15) and the assumed homogeneous boundary conditions, we are not considering the vertex shape functions because all of them are located on the domain boundary $\partial\Omega$ in which the solution vanishes. By the same reasons, we are not considering all the edge shape functions of any level related to edges located on the domain boundary $\partial\Omega$. Thus, we are only retaining the edge shape functions associated with internal edges as well as all the face shape functions.

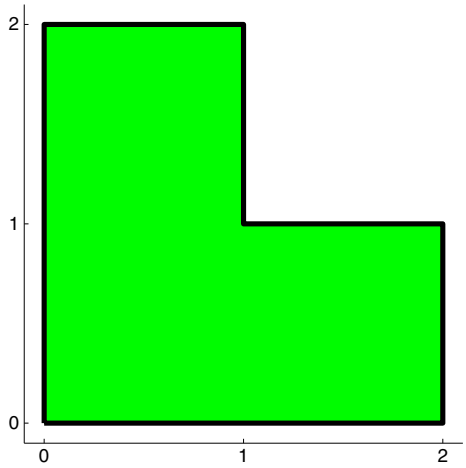

Figure 14. Geometry of the domain considered for the solution of transient problem (93).

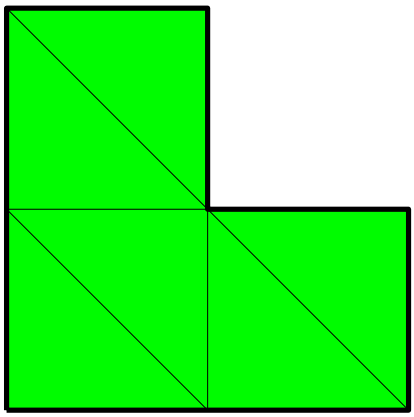

Figure 15. Mesh associated to the Szabo and Cernevali hierarchical basis.

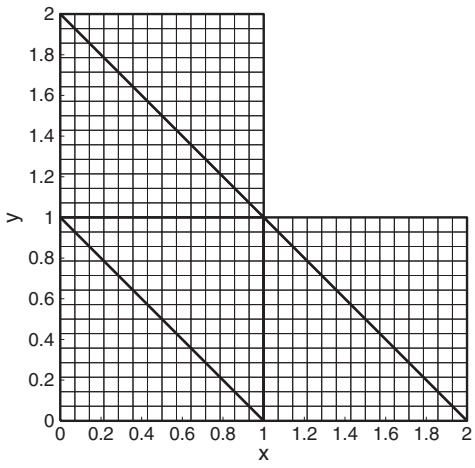

Figure 16. Fine mesh used for solving the equations superposed and coarse mesh employed for defining the hierarchical basis.

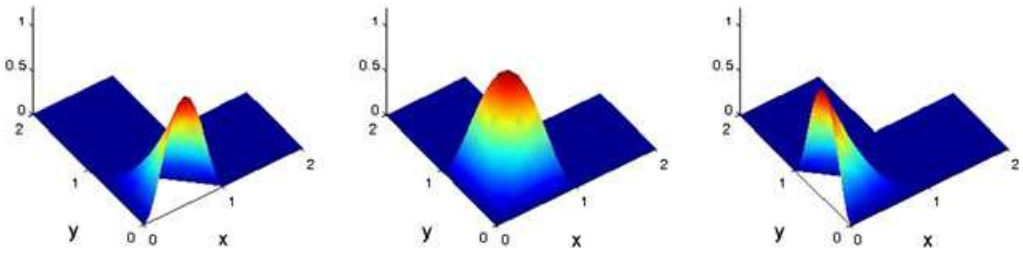

Figure 17. Edge shape functions related to one of the triangles depicted in Figure 15.

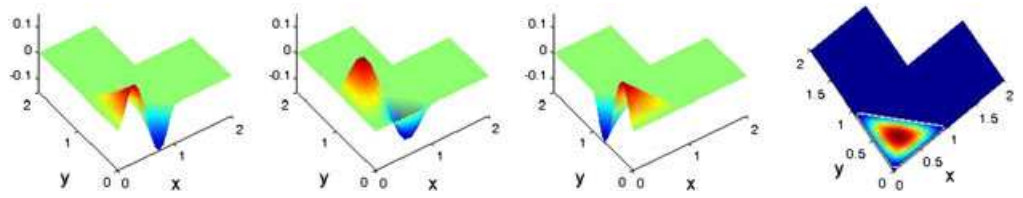

Figure 18. Second level edge shape functions and face shape functions related to one of the triangles depicted in Figure 15.

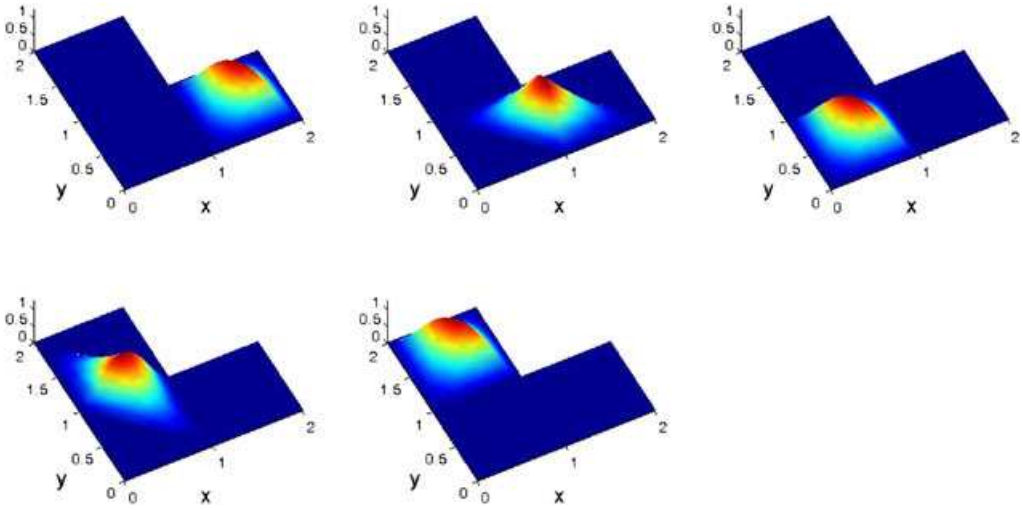

Figure 19. Active approximation functions at the first level of the hierarchical approximation related to the mesh depicted in Figure 15.

At the first level, we have $five$ edge shape functions related to the five internal mesh edges, all of them depicted in Figure 19.

At the second level, we should consider the next five edge shape functions (related to the five internal mesh edges) as well as the six face shape functions, all of them depicted in Figure 20.

Figure 21 depicts the separated representation of the solution u^1_{hom} of the homogeneous problem:

$$\begin{cases} \frac{\partial u^1_{\text{hom}}}{\partial t} - \Delta u^1_{\text{hom}} = 0 \\ u^1_{\text{hom}}(\mathbf{x}, t, \tau = 0) = H_1(\mathbf{x}) \end{cases} \tag{94}$$

where $H_1(\mathbf{x})$ is the first level edge shape function depicted in Figure 19 at the top and left.

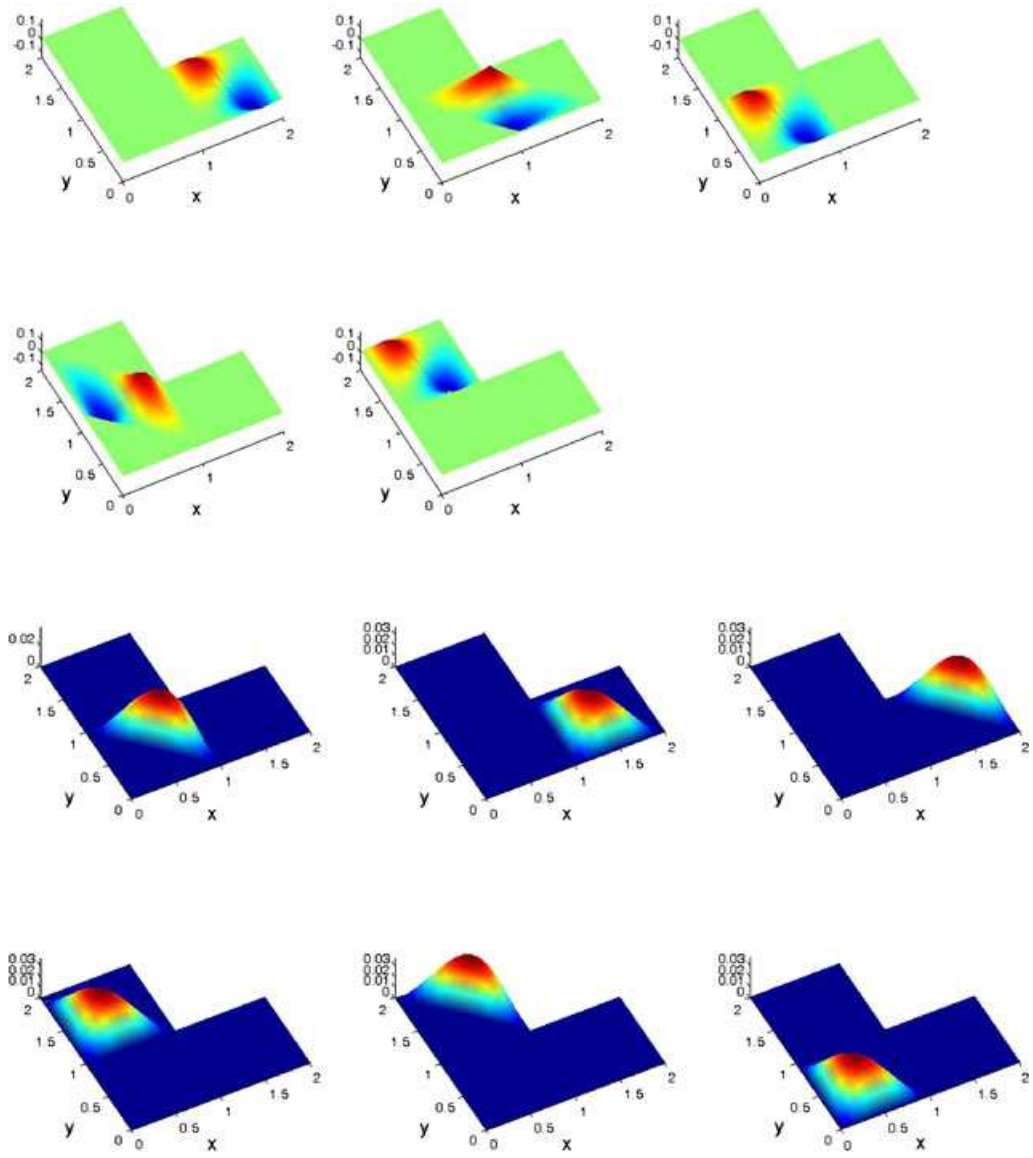

Figure 20. Active approximation functions at the second level of the hierarchical approximation related to the mesh depicted in Figure 15: (top) the five second level edge shape functions, and (down) the six first face shape functions.

The separated representation, for the desired precision, involves 15 functional products, that is,

$$u^1_{\text{hom}}(\mathbf{x}, t, \tau) \approx \sum_{i=1}^{15} F^i_{\mathbf{x},1}(\mathbf{x}) \cdot F^i_{t,1}(t) \cdot F^i_{\tau,1}(\tau) \tag{95}$$

whose four most significative functions are depicted in Figure 21. We depict, in this figure, the first four space functions $F^i_{\mathbf{x},1}(\mathbf{x})$, $i = 1, \cdots, 4$, and the first time functions $F^i_{t,1}(t)$ and $F^i_{\tau,1}$, $i = 1, \cdots, 4$.

The reconstructed solution $u(\mathbf{x}, t = 1)$ is depicted in Figure 22 when a hierarchical approximation basis consisting of the first two levels (involving 16 approximation functions: $five$ first-level edge shape functions, $five$ second-level edge shape functions, and finally six second-level face shape functions) was applied.

Obviously, the difference between the computed solution and the reference one obtained by using a standard incremental strategy in the whole time interval, decreases as the size of the hierarchical

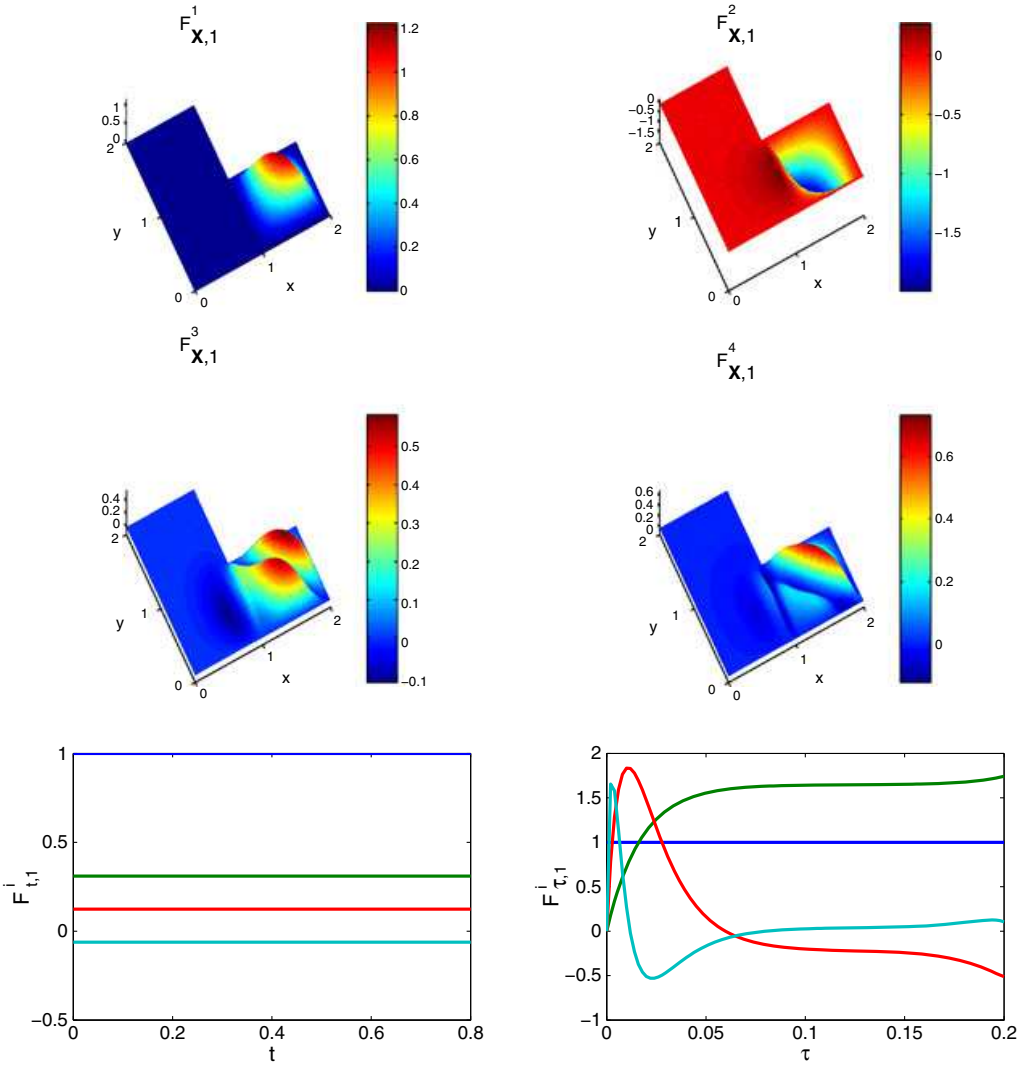

Figure 21. Main functions involved in the separated representation of $u^1_{\text{hom}}(\mathbf{x}, t, \tau)$: (top) the first four space functions $F^i_{\mathbf{x},1}(\mathbf{x})$, $i = 1, \cdots, 4$; (down) the first time functions $F^i_{t,1}(t)$ (left) and $F^i_{\tau,1}$ (right), $i = 1, \cdots, 4$.

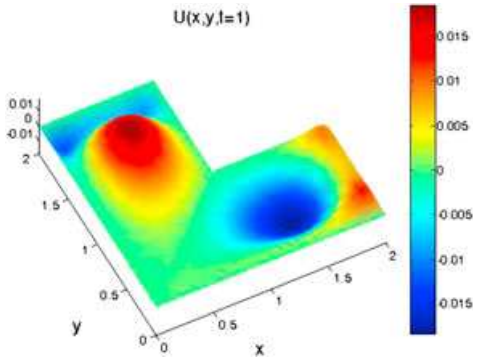

Figure 22. Reconstructed solution $u(\mathbf{x}, t = 1)$ obtained from $u(\mathbf{x}, t, \tau)$.

basis increases. Thus, when we use for enforcing the time-continuity only the first-level approximation functions ($five$ shape functions) the error is 0.1 (10%) whereas by using the shape functions

of the two first levels (16 shape functions), the error reduced to 0.03 (3%). These errors are defined from the L2 norm of the difference between the computed solution and the reference one (the one computed by using a standard incremental strategy) divided by the L2 norm of the reference solution.

7. CONCLUSION

In this paper, we proved that models involving different non-separable time scales can be reformulated by introducing different time coordinates allowing to express the time dependence of the solution from a multidimensional time approximation. Event if, at present, it is too early for concluding on the computing time and the procedure performances, the possibility of performing such decomposition is, in our opinion, quite interesting, and it could be at the origin of many developments for reducing the computing time of complex multiscale thermomechanical models.

ACKNOWLEDGEMENT

Contract/grant sponsor: Spanish Ministry of Education and Science; contract/grant number: CICYT-DPI2008-00918

REFERENCES

1. Ladeveze P. *Nonlinear Computational Structural Mechanics*. Springer: NY, 1999.
2. Ladeveze P, Nouy A. On a multiscale computational strategy with time and space homogenization for structural mechanics. *Computer Methods in Applied Mechanics and Engineering* 2003; **192**:3061–3087.
3. Nouy A, Ladeveze P. Multiscale computational strategy with time and space homogenization: a radial-type approximation technique for solving microproblems. *International Journal of Multiscale Computational Engineering* 2004; **170**(2):557–574.
4. Ladeveze P, Passieux J-C, Neron D. The LATIN multiscale computational method and the proper generalized decomposition. *Computer Methods in Applied Mechanics and Engineering* 2010; **199**(21-22):1287–1296.
5. Ammar A, Normandin M, Daim F, Gonzalez D, Cueto E, Chinesta F. Non-incremental strategies based on separated representations: applications in computational rheology. *Communications in Mathematical Sciences* 2010; **8**(3):671–695.
6. Ammar A, Mokdad B, Chinesta F, Keunings R. A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids. *Journal of Non-Newtonian Fluid Mechanics* 2006; **139**:153–176.
7. Ammar A, Mokdad B, Chinesta F, Keunings R. A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids. Part II: transient simulation using space–time separated representations. *Journal of NonNewtonian Fluid Mechanics* 2007; **144**:98–121.
8. Nouy A. A generalized spectral decomposition technique to solve a class of linear stochastic partial differential equations. *Computer Methods in Applied Mechanics and Engineering* 2007; **196**:4521–4537.
9. Chinesta F, Ammar A, Lemarchand F, Beauchene P, Boust F. Alleviating mesh constraints: model reduction, parallel time integration and high resolution homogenization. *Computer Methods in Applied Mechanics and Engineering* 2008; **197**(5):400–413.
10. Chinesta F, Ammar A, Cueto E. Recent advances and new challenges in the use of the proper generalized decomposition for solving multidimensional models. *Archives of Computational Methods in Engineering* 2010; **17**(4):327–350.
11. Ammar A, Chinesta F, Joyot P. The nanometric and micrometric scales of the structure and mechanics of materials revisited: an introduction to the challenges of fully deterministic numerical descriptions. *International Journal for Multiscale Computational Engineering* 2008; **6**(3):191–213.
12. Chinesta F, Ammar A, Cueto E. Proper generalized decomposition of multiscale models. *International Journal of Numerical Methods in Engineering* 2010; **83**(8–9):1114–1132.
13. Yu Q, Fish J. Temporal homogenization of viscoelastic and viscoplastic solids subjected to locally periodic loading. *Computational Mechanics* 2002; **29**:199–211.
14. Frantziskonis G, Muralidharan K, Deymier P, Simunovic S, Nukala P, Pannala S. Time-parallel multiscale/multiphysics framework. *Journal of Computational Physics* 2009; **228**(21):8085–8092.
15. Baffico L, Bernard S, Maday Y, Turinici G, Zérah G. Parallel-in-time molecular-dynamics simulations. *Physical Review E* 2002; **66**(5):057701.
16. Bottasso CL. Multiscale temporal integration. *Computer Methods in Applied Mechanics and Engineering* 2002; **191**(25–26):2815–2830.
17. Neron D, Dureisseix D. A computational strategy for poroelastic problems with a time interface between coupled physics. *International Journal for Numerical Methods in Engineering* 2008; **73**(6):783–804.

18. Neron D, Dureisseix D. A computational strategy for thermo-poroelastic structures with a time–space interface coupling. *International Journal for Numerical Methods in Engineering* 2008; **75**(9):1053–1084.
19. Pruliere E, Chinesta F, Ammar A. On the deterministic solution of multidimensional parametric models using the proper generalized decomposition. *Mathematics and Computers in Simulation* 2010; **81**:791–810.
20. Gonzalez D, Ammar A, Chinesta F, Cueto E. Recent advances on the use of separated representations. *International Journal for Numerical Methods in Engineering* 2010; **81**(5):637–659.
21. Beylkin G, Mohlenkamp M. Algorithms for numerical analysis in high dimensions. *SIAM Journal on Scientific Computing* 2005; **26**(6):2133–2159.
22. Szabo B, Babuska I. *Introduction to Finite Element Analysis*. John Wiley and Sons: New York, 1989.