Proper Generalized Decomposition method for incompressible flows in stream-vorticity formulation

Antoine Dumon* — Cyrille Allery* — Amine Ammar**

** LEPTIAB*
Pôle Sciences et Technologie
Avenue Michel Crepeau
F-17042 La Rochelle cedex 1
{adumon01 ; cyrille.allery}@univ-lr.fr

*** Arts et Metiers ParisTech*
2 Bd du Ronceray
F-49035 Angers cedex 1
Amine.Ammar@ensam.eu

ABSTRACT. *In this work, the Proper Generalized Decomposition (PGD) method will be considered in order to solve Navier-stokes equations with a stream-vorticity formulation by looking for the solution as a sum of tensor product functions. In the first stage, PGD will be applied to a model equation in order to test the capacity of the method to treat some time-dependent problem. Then, we will solve the Navier-Stokes problem in the case of the lid-driven cavity for different Reynolds numbers (Re = 100, 1000 and 10000). Finally, the PGD method will be compared to the standard resolution technique, both in terms of CPU time and accuracy.*

RÉSUMÉ. *L'objectif de ce travail est d'appliquer la méthode Proper Generalized Decomposition (PGD) pour résoudre les équations de Navier-Stokes en formulation ligne de courant-vorticité. Par cette technique la solution est recherchée comme une somme de produits tensoriels de chacune des variables du problème (espace, temps…). Afin de tester les capacités de la méthode pour la résolution de problème instationnaire, la PGD sera tout d'abord appliquée à l'équation de diffusion instationnaire. Dans un second temps elle sera appliquée pour simuler l'écoulement dans une cavité entraînée. Les résultats obtenus seront comparés à ceux obtenus par une méthode de résolution standard aussi bien en termes de précision que de temps de simulations.*

KEYWORDS: *reduced order model, lid-driven cavity, separation of variables.*

MOTS-CLES : *réduction de modèles, cavité entraînée, séparation de variable.*

DOI:10.3166/EJCM.19.591-617

1. Introduction

Fluid-structure interactions play an important role in many applications which present a coupling between movements of a structure in a fluids flow. Different methods to treat these problems has been extensively investigated : see (Dowell *et al.*, 2001) for a comprehensive review. The numerical resolution of such a problem could combine the resolution of the fluid and the structural equations, which requires an important CPU time and stockage capacity. With traditional methods of resolution, effectuate optimization or parametric analyses is not possible. That's the reason why some methods called Reduced Order Models (ROM), which deals with reducing this cost, appear these least years. The most well-known ROM method used is the POD (Proper Orthogonal Decomposition). For example, in order to modelize flow around an oscillating cylinder, Liberge et al.(Liberge *et al.*, 2010) compute the POD modes for a global velocity field (fluid and solid), and then construct a low-order dynamical system obtained by using a multiphase method similar to the fictitious domain method. This multiphase method extends the Navier-Stokes equations to the solid domain by using a penalisation method and a Lagrangian multiplier. Plazcek et al. in (Placzek *et al.*, 2008) study with the hybrid POD method cases of introduction of structural damping and a nonlinear force applied at the free end of rod. Lieu et al. in (Lieu *et al.*, 2006) apply POD to model a complete F-16 fighter configuration, in order to assess its potential for the solution of realistic aeroelastic problems.

The main drawback of this previous technique is the need of a snapshots set of the solution to construct the reduced-basis. The computing time requested for the calculation of these snapshots could be very important. Consequently some methods called "*a priori*" model reduction techniques have been developed. They consist in building a reduced basis without an "*a priori*" knowledge of the solution. The A Priori model Reduction (APR) (Ryckelynck, 2002)(Ryckelynck, 2005)(Ryckelynck *et al.*, 2005)(Ammar *et al.*, 2006b)(Verdon *et al.*, 2009) has been subject of several developments. Thanks to this approach, the basis is adaptatively improved and expanded with the residuals of the full discretized model. The incremental process is done by taking into account the whole time interval where the reduced equation is solved.

An other a priori method, which will be applied in the following part is the PGD (Proper Generalized decomposition). The PGD consists in seeking the separated representation of the solution of a partial diferential equation. Concretely, the separated representation of a function $f(x_1, \ldots, x_N)$ could be write:

$$\sum_{i=1}^{Q} \prod_{k=1}^{N} F_{ki}(x_k)$$

(x_i can be any scalar or vector variables involving space, time or any other parameter of the problem). Thus, if M degrees of freedom are used to discretize each variable, the total number of unknowns involved in the solution is $Q \times N \times M$ instead of the M^N degrees of freedom involved in mesh based discretization techniques. In most cases, when the field is sufficiently regular, the number of terms Q in the finite sum

is generally quite small (a few dozen) and in all cases the approximation converges towards the solution of the full grid description (see (Ammar *et al.*, 2010; Ammar *et al.*, n.d.)).

This technique has been proposed a few years ago by Ammar *et al.*, (2006a, 2007) in the context of the multi-bead-spring FENE models of polymeric systems. The technique has been applied on other more complex models based on the reptation theory of polymeric liquids in (Mokdad *et al.*, 2007). This technique was also used in quantum chemistry problems (Chinesta *et al.*, 2008a), and in materials homogenization in (Chinesta *et al.*, 2008b). This method has already been applied in a stochastic framework (Nouy, 2007; Nouy *et al.*, 2009). In the context of LATIN method, Ladeveze (Ladeveze, 1999; Ladeveze *et al.*, 2010) use a method called 'radial approximation'. This approach could be seen as a variant of the PGD with a space-time separation.

In fluide structure interaction, for strongly coupled problem, the more expensive step is the resolution of the fluid problem. Then, as a first stage, in order to test the capacity of PGD, we will apply PGD to solve the Navier-Stokes equations only. The paper is organized as follows. Firstly, the PGD method will be presented in continuous form and in algebraic form. Secondly, PGD will be applied to solve the unsteady 2D diffusion equation. This example enables us to test two approaches of the PGD. The first one consists in separating the solution on the spaces and time variabes, the second one is a separation only on the spaces variables. Then the discretization of the Navier-Stokes equations in streamline-vorticity formulation and its PGD formulation has been detailed. Finally, results on the test-case of the 2D lid-driven cavity in stationary and unstationary case will be illustrated. For the Navier-Stokes equations only a separation on the spaces variables will be applied.

2. Description of the PGD

In this part, PGD will be describe in continuous form and in algebraic form.

2.1. *Continuous formulation*

2.1.1. *Preliminaries*

For the sake of clarity and without losing its general scope, PGD will be examined in the case of a 2D space decomposition. The problem is expressed as follows :

$$\text{Find } U(x,y) \text{ as } \begin{cases} \mathcal{L}(U) = \mathcal{G} & \text{in } \Omega \\ +\text{Boundary Conditions} \end{cases} \qquad [1]$$

where $\mathcal{L}$ is a linear[1] differential operator and $\mathcal{G}$ is the second member.

PGD, which is an iterative method, consists in finding an approximation of the solution $U(x,y) \in \Omega = X \times Y \subset \mathbb{R}^2$ with $x \in X \subset \mathbb{R}$ and $y \in Y \subset \mathbb{R}$ as:

1. If the operator is not linear, it is necessary to linearize it.

$$U(x,y) \approx U_m(x,y) = \sum_{i=1}^{m} \alpha^i F^i(x) G^i(y) \qquad [2]$$

where $U_m(x,y)$ is the approximation of the solution of order m. At each iteration, the solution is enriched with an additional term $\alpha^{m+1} F^{m+1}(x) G^{m+1}(y)$. PGD is an iterative procedure which should be decomposed in three steps. During the first step, "called the enrichment step", the F^{m+1} and G^{m+1} functions are obtained by solving a small size non-linear problem. Then, for the second step, called the "projection step", in order to improve the quality of the reconstruction, the $m+1$ α^i coefficients are determined by solving a linear system of size $(m+1)$. Finally, the "check convergence step" consists in the computing of the norm of the residual in order to decide if the solution need more enrichment or not. In the following these three steps will be described in details.

2.1.2. *Enrichment step*

At the $m+1$ stage, the solution approximation of order m is supposed to be known. In this step we search to compute the functions $F^{m+1}(x)$ and $G^{m+1}(y)$. We search $U_m(x,y)$ as

$$U_m(x,y) = \sum_{i=1}^{m} \alpha^i F^i(x) G^i(y) + F^{m+1}(x) G^{m+1}(y) \qquad [3]$$

Introducing Equation [3] into Problem [1], it gives :

$$\mathcal{L}(\sum_{i=1}^{m} \alpha^i F^i(x) G^i(y) + F^{m+1}(x) G^{m+1}(y)) = \mathcal{G} + Res^{m+1} \qquad [4]$$

where Res^{m+1} is a residual whose appears because Equation [3] is an approximation of the solution. Equation [4] is then projected onto each of the unknowns F^m and G^m and the residual Res^{m+1} is forced to be orthogonal to each of these functions. It gives the two following problems :

$$< \mathcal{L}(\sum_{i=1}^{m} \alpha^i F^i(x) G^i(y) + F^{m+1}(x) G^{m+1}(y)), F^{m+1} >_{L^2(X)} = < \mathcal{G}, F^{m+1} >_{L^2(X)} \qquad [5]$$

$$< \mathcal{L}(\sum_{i=1}^{m} \alpha^i F^i(x) G^i(y) + F^{m+1}(x) G^{m+1}(y)), G^{m+1} >_{L^2(Y)} = < \mathcal{G}, G^{m+1} >_{L^2(Y)} \qquad [6]$$

Equations [5] and [6] are solved using the fixed point method. After convergence of the fixed point, the first $m+1$ functions F^i and G^i are now known.

2.1.3. *Projection step*

In order to increase the accuracy of the decomposition, the α^i coefficients are now searched in such a way that the residual is orthogonal to each of the $m+1$ products of the F^iG^i functions. At this step, we search $U_{m+1}(x,y)$ as,

$$U_{m+1}(x,y) = \sum_{i=1}^{m+1} \alpha^i F^i(x) G^i(y) \qquad [7]$$

Solution [7] is introduced into Equation [1] :

$$\mathcal{L}(\sum_{i=1}^{m+1} \alpha^i F^i(x) G^i(y)) = \mathcal{G} + Res^{m+1} \qquad [8]$$

The α^i coefficients are then computed by projecting the above equation according to the F^iG^i :

$$< \mathcal{L}(\sum_{i=1}^{m+1} \alpha^i F^i(x) G^i(y)), F^k G^k >_{L^2(\Omega)} = < \mathcal{G}, F^k G^k >_{L^2(\Omega)} \qquad \text{for } 1 \leqslant k \leqslant m+1 \qquad [9]$$

Equation [9] could be solved using a classical solver of linear problems. At this point we know the approximation of the solution of order $m+1$, $U_{m+1}(x,y)$.

2.1.4. *Check convergence step*

At this step the residual is computed in the following way :

$$Res^{m+1} = \mathcal{L}(\sum_{i=1}^{m+1} \alpha^i F^i(x) G^i(y)) - \mathcal{G} \qquad [10]$$

If the L^2 norm of this residual is lower than a coefficient ϵ set by the user, the PGD algorithm was converged. Else, one more iteration at least is needed, and the enrichment and projection steps are repeated taking $m = m+1$ until convergence.

2.2. *Algebraic formulation*

2.2.1. *Preliminaries*

After discretisation by finite element, finite volume or other technique, Problem [1] can be written in a discrete form :

$$\mathcal{L}_h(\mathbf{U}_h) = \mathcal{G}_h \quad [11]$$

with

$$\mathcal{L}_h = \sum_{j=1}^{n_\mathcal{L}} \mathbb{A}_x^j \otimes \mathbb{A}_y^j, \quad \mathcal{G}_h = \sum_{j=1}^{n_\mathcal{G}} \mathbf{f}_x^j \otimes \mathbf{f}_y^j, \quad \mathbf{U}_h = \sum_{i=1}^{N} \alpha^i \mathbf{F}^i \otimes \mathbf{G}^i \quad [12]$$

The operator $\mathcal{L}$ is discretized as a tensor product of operators $\mathbb{A}_x^j$ and $\mathbb{A}_y^j$ in the direction x and y respectively. The discretized operator $\mathbb{A}_x^j$ (resp. $\mathbb{A}_y^j$) is a square matrix whose size is N_x (resp N_y) where N_x(resp N_y) is the number of discretisation nodes in the direction x (resp y). The second term was decomposed as products of the sum of vectors $\mathbf{f}_x^j$ and $\mathbf{f}_y^j$ of size N_x and N_y. Finally the unknown U_h is calculated as a product sum of vectors $\mathbf{F}^i$ and $\mathbf{G}^i$ of size N_x and N_y using a weight coefficient α^i. $n_\mathcal{L}$ (resp. $n_\mathcal{G}$) represents the number of tensor products required to represent the separated form of the initial operator $\mathcal{L}$ (resp. the second member $\mathcal{G}$).

Taking into account the property of the tensor product, Equation [11] can be written as :

$$\sum_{k=1}^{n_\mathcal{L}} \sum_{i=1}^{N} \alpha^i \left(\mathbb{A}_x^k \mathbf{F}^i \otimes \mathbb{A}_y^k \mathbf{G}^i \right) = \sum_{j=1}^{n_\mathcal{G}} \mathbf{f}_x^j \otimes \mathbf{f}_y^j \quad [13]$$

The three steps of enrichment, projection and checking convergence will now be described with these notations.

2.2.2. *The enrichment step*

We suppose to be at iteration (m+1) At this stage the unknown U_h is search in tensorial form as follows :

$$\mathbf{U}_h = \sum_{i=1}^{m} \alpha^i \mathbf{F}^i \otimes \mathbf{G}^i + \mathbf{R} \otimes \mathbf{S} \quad [14]$$

where $\mathbf{R}$ and $\mathbf{S}$ are unknowns and where the solution at the previous iteration $U_m = \sum_{i=1}^{m} \alpha^i \mathbf{F}^i \otimes \mathbf{G}^i$ is known. By introducing this new approximation of the solution into Equation [13], we have to solve :

$$\sum_{k=1}^{n_\mathcal{L}} \left(\mathbb{A}_x^k \mathbf{R} \otimes \mathbb{A}_y^k \mathbf{S} \right) = \mathcal{G}_h - \sum_{k=1}^{n_\mathcal{L}} \sum_{i=1}^{m} \alpha^i \left(\mathbb{A}_x^k \mathbf{F}^i \otimes \mathbb{A}_y^k \mathbf{G}^i \right) \quad [15]$$

This non-linear system is solved within a fixed-point strategy . In order to compute $\mathbf{R}$ we choose to fix $\mathbf{S}$ and we project Equation [15] onto the vector $\mathbf{S}$. This gives the following problem, corresponding to Equation [6] in continuous form:

$$\sum_{k=1}^{n_{\mathcal{L}}} \gamma_k^1 \mathbb{A}_x^k \mathbf{R} = \sum_{j=1}^{n_{\mathcal{G}}} \gamma_j^2 \mathbf{f}_x^j - \sum_{k=1}^{n_{\mathcal{L}}} \sum_{i=1}^{m} \gamma_{i,k}^3 \alpha^i \mathbb{A}_x^k \mathbf{F}^i \qquad [16]$$

with

$$\gamma_k^1 = {}^t\mathbf{S}\mathbb{A}_y^k\mathbf{S} \in \mathbb{R}, \qquad \gamma_j^2 = {}^t\mathbf{S}\,\mathbf{f}_y^j \in \mathbb{R}, \qquad \gamma_{i,k}^3 = {}^t\mathbf{S}\mathbb{A}_y^k\mathbf{G}^i \in \mathbb{R} \qquad [17]$$

Similarly, in order to compute $\mathbf{S}$ we set $\mathbf{R}$ at the value just computed in Equation [16] and we project Equation [15] onto the vector $\mathbf{R}$. This gives the following problem, corresponding to Equation [5] in continuous form :

$$\sum_{k=1}^{n_{\mathcal{L}}} \beta_k^1 \mathbb{A}_y^k \mathbf{S} = \sum_{j=1}^{n_{\mathcal{G}}} \beta_j^2 \mathbf{f}_y^j - \sum_{k=1}^{n_{\mathcal{L}}} \sum_{i=1}^{m} \beta_{i,k}^3 \alpha^i \mathbb{A}_y^k \mathbf{G}^i \qquad [18]$$

with

$$\beta_k^1 = {}^t\mathbf{R}\mathbb{A}_x^k\mathbf{R} \in \mathbb{R}, \qquad \beta_j^2 = {}^t\mathbf{R}\,\mathbf{f}_x^j \in \mathbb{R}, \qquad \beta_{i,k}^3 = {}^t\mathbf{R}\mathbb{A}_x^k\mathbf{F}^i \in \mathbb{R} \qquad [19]$$

Problems [16] and [18] are solved iteratively. The fixed-point procedure stops when the k^{th} iteration satisfies:

$$||(\mathbf{R} \otimes \mathbf{S})_k - (\mathbf{R} \otimes \mathbf{S})_{k-1}|| \leq \epsilon \qquad [20]$$

where $||\cdot||$ is the L^2 norm and ϵ is a parameter chosen by the user. The new $\mathbf{F}^{m+1}$ and $\mathbf{G}^{m+1}$ are then given by the next normalization:

$$\mathbf{F}^{m+1} = \frac{\mathbf{R}}{||\mathbf{R}||} \qquad \mathbf{G}^{m+1} = \frac{\mathbf{S}}{||\mathbf{S}||} \qquad [21]$$

2.2.3. *Projection step*

The $m+1$ functions $\mathbf{F}^i$ and $\mathbf{G}^i$ are known, α^i $(1 \leq i \leq m+1)$ has to be computed. For this purpose, we project Equation [13] onto the products $\mathbf{F}^j\mathbf{G}^j$ $(1 \leq j \leq m+1)$. Thus the following linear problem, whose size is $(m+1)$, corresponding to Equation [3] in continuous form, is obtained

$$\mathbb{H}\boldsymbol{\alpha} = \mathbf{J} \quad \text{with} \quad {}^t\boldsymbol{\alpha} = \{\alpha_1, \ldots, \alpha_{N+1}\} \qquad [22]$$

where the components of $\mathbb{H}$ and $\mathbf{J}$ are defined by:

$$\mathbb{H}_{ij} = \sum_{k=1}^{n_{\mathcal{L}}} {}^t\mathbf{F}^j \mathbb{A}_x^k \mathbf{F}^i . {}^t\mathbf{G}^j \mathbb{A}_y^k \mathbf{G}^i \quad \text{and} \quad J_j = \sum_{k=1}^{n_{\mathcal{G}}} {}^t\mathbf{F}^j \mathbf{f}_x^k . {}^t\mathbf{G}^j \mathbf{f}_y^k \qquad [23]$$

2.2.4. *Check convergence*

In order to estimate the convergence of the algorithm, a computation is performed of the residual Res of Equation [1] defined by:

$$\mathbf{Res} = \sum_{k=1}^{n_{\mathcal{L}}} \sum_{i=1}^{m+1} \alpha^i \left(\left[\mathbb{A}_x^k\right] \mathbf{F}^i \otimes \left[\mathbb{A}_y^k\right] \mathbf{G}^i \right) - \sum_{k=1}^{n_{\mathcal{G}}} \mathbf{f}_x^k \otimes \mathbf{f}_y^k \qquad [24]$$

When the $L^2(\Omega)$ norm of this residual becomes lower than a coefficient ϵ set by the user, the algorithm is considered to be at convergence, and the solution of the problem is expressed as:

$$U_h = \sum_{i=1}^{m+1} \alpha^i \mathbf{F}^i \otimes \mathbf{G}^i \qquad [25]$$

3. PGD applied to the unsteady 2D diffusion equation

3.1. *Methodology*

In this section we will study the capacity of the PGD to solve an unsteady diffusion equation. The problem to solve is :

$$\text{Find } \Theta\text{(x,y,t) as } \begin{cases} \dfrac{\partial \Theta(x,y,t)}{\partial t} - \nu \Delta \Theta(x,y,t) = f(x,y,t) \quad \text{in } \Omega \times I \\ \Theta_{|\Gamma} = g \\ \Theta(x,y,0) = \Theta^0 \end{cases} \qquad [26]$$

where $\Omega = X \times Y \in \mathbb{R}^2$ is a spatial domain, and $I =]0, T[$ is a time interval.

We apply two different formulations of PGD : PGD with space-time decomposition and PGD with space decomposition only.

3.1.1. *PGD with space-time decomposition*

Let the second term be :

$$f(x,y,t) = \sum_{j=1}^{n_f} f_x^j(x) f_y^j(y) f_t^j(t) \qquad [27]$$

The first formulation, which will be noted PGD(XYT), consists in seeking the solution as a function of the space variables and the time. Then, the solution is searched as :

$$\Theta(x,y,t) \approx \Theta_N(x,y,t) = \sum_{i=1}^{N} \alpha^i F^i(x) G^i(y) H^i(t) \qquad [28]$$

Injecting this formulation in Problem [26] and using a finite volume formulation, Problem [26] could be written for the control volume Ω_{lp} and the time q ($1 \leq l \leq N_x$, $1 \leq p \leq N_y$ and $1 \leq q \leq N_t$)

$$\sum_{j=1}^{n_f} \int_{X_l} f_x^j dx \int_{Y_p} f_y^j dy \int_{T_q} f_t^j dt = \sum_{i=1}^{N} \alpha^i \left(\int_{X_l} F^i dx \int_{Y_p} G^i dy \int_{T_q} \frac{dH^i}{dt} dt \right.$$
$$\left. -\nu \left[\int_{X_l} \frac{d^2F^i}{dx^2} dx \int_{Y_p} G^i dy \int_{T_q} H^i dt + \int_{X_l} F^i dx \int_{Y_p} \frac{d^2G^i}{dy^2} dy \int_{T_q} H^i dt dt \right] \right) \qquad [29]$$

This equation should be written in the same way as in the PGD algebraic formulation (see Equation [13]).

$$\sum_{k=1}^{3} \sum_{i=1}^{N} \alpha^i \left(A_x^k \mathbf{F}^i \otimes A_y^k \mathbf{G}^i \otimes A_t^k \mathbf{H}^i \right) = - \int_{\Omega \times I} \sum_{j=1}^{n_f} \mathbf{f}_x^j \otimes \mathbf{f}_y^j \otimes \mathbf{f}_t^j d\Omega \qquad [30]$$

where A_x^k (respectively A_y^k, A_t^k) is a square matrix whose size is N_x (resp. N_y, N_t). These matrices are defined by

$$A_x^1 = diag(\Delta_{x_1}, \cdots, \Delta_{x_{N_x}}) \qquad A_y^1 = diag(\Delta_{y_1}, \cdots, \Delta_{y_{N_y}})$$

$$A_t^1 = \begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 \\ -1 & 1 & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & -1 & 1 \end{pmatrix}$$

$$A_x^2 = \begin{pmatrix} a_1 & c_1 & 0 & \cdots & \cdots & \cdots & 0 \\ b_2 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & b_i & a_i & c_i & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & c_{N_x-1} \\ 0 & \cdots & \cdots & \cdots & 0 & b_{N_x} & a_{N_x} \end{pmatrix}$$

where

$$a_i = -\frac{1}{x_{i+1} - x_i} - \frac{1}{x_i - x_{i-1}}, \quad b_i = \frac{1}{x_i - x_{i-1}}, \quad c_i = \frac{1}{x_{i+1} - x_i}$$

$$A_y^2 = -\nu A_y^1 \qquad A_t^2 = diag(\Delta_{t_1}, \cdots, \Delta_{t_{N_t}})$$

$$A_x^3 = -\nu A_x^1 \qquad A_t^3 = A_t^2$$

$$A_y^3 = \begin{pmatrix} d_1 & f_1 & 0 & \cdots & \cdots & \cdots & 0 \\ e_2 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & e_j & d_j & f_j & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & f_{N_y-1} \\ 0 & \cdots & \cdots & \cdots & 0 & e_{N_y} & d_{N_y} \end{pmatrix}$$

where

$$d_j = -\frac{1}{y_{j+1} - y_j} - \frac{1}{y_j - y_{j-1}}, \quad e_j = \frac{1}{y_j - y_{j-1}}, \quad f_j = \frac{1}{y_{j+1} - y_j}$$

where (x_i, y_j) is the coordinate of the center of the control volume Ω_{ij}. Δ_{x_i} (resp. Δ_{y_j}) is the horizontal (resp. vertical) lenght of the control volume Ω_{ij} and $\Delta_{t_k} = \frac{t_{k+1}-t_{k-1}}{2}$. We define $x_0 = x_{CL_W}$ (resp. $x_{N_{x+1}} = x_{CL_E}$) the x-coordinate of the west (resp. of the east) boundary of the domain, and $y_0 = y_{CL_S}$ (resp. $y_{N_{y+1}} = y_{CL_N}$) the y-coordinate of the south (resp. of the north) boundary of the domain.

3.1.2. *PGD with space decomposition*

In this case a temporal discretisation has to be done using a Cranck-nicholson scheme. Then, knowing Θ^n at the time $t^n = n * \delta t$, we search Θ^{n+1} as :

$$\frac{\Theta^{n+1}}{\delta t} - \frac{\nu}{2}\Delta\Theta^{n+1} = \frac{\Theta^n}{\delta t} + \frac{\nu}{2}\Delta\Theta^n + \frac{1}{2}(f^n + f^{n+1}) \quad [31]$$

Here the PGD is only applied to the space variables, Θ^{n+1} is seeking as :

$$\Theta(x, y, t^{n+1}) \approx \Theta_N^{n+1}(x, y) = \sum_{i=1}^{N} \alpha^i F^i(x) G^i(y) \quad [32]$$

Let the source be :

$$(f^n + f^{n+1}) = \sum_{j=1}^{n_f} f_x^j(x) f_y^j(y) \quad [33]$$

Injecting this formulation, which we called PGD(XY) in the following, in Equation [31] and using a finite volume method, we obtain for the volume control Ω_{lp} :

$$\begin{aligned} \sum_{j=1}^{n_f} \int_{X_l} f_x^j dx \int_{Y_p} f_y^j dy = \sum_{i=1}^{N} \alpha^i \left(\frac{1}{\delta_t} \int_{X_l} F^i dx \int_{Y_p} G^i dy \right. \\ \left. - \frac{\nu}{2} \left[\int_{X_l} \frac{d^2 F^i}{dx^2} dx \int_{Y_p} G^i dy + \int_{X_l} F^i dx \int_{Y_p} \frac{d^2 G^i}{dy^2} dy \right] \right) \end{aligned} \quad [34]$$

or

$$\sum_{k=1}^{3} \sum_{i=1}^{N} \alpha^i \left(A_x^k \mathbf{F}^i \otimes A_y^k \mathbf{G}^i \right) = - \int_{\Omega \times I} \sum_{j=1}^{n_f} \mathbf{f}_x^j \otimes \mathbf{f}_y^j d\Omega \quad [35]$$

where A_x^k (respectively A_y^k) is a square matrix whose size is N_x (resp. N_y). Using the same notations that in (3.1.1), these matrices are defined by

$$A_x^1 = \frac{1}{\delta t} diag(\Delta_{x_1}, \cdots, \Delta_{x_{N_x}}) \qquad A_y^1 = diag(\Delta_{y_1}, \cdots, \Delta_{y_{N_y}})$$

$$A_x^2 = \begin{pmatrix} a_1 & c_1 & 0 & \cdots & \cdots & \cdots & 0 \\ b_2 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & b_i & a_i & c_i & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & c_{N_x-1} \\ 0 & \cdots & \cdots & \cdots & 0 & b_{N_x} & a_{N_x} \end{pmatrix} \qquad A_y^2 = -\frac{\nu}{2} A_y^1$$

$$A_x^3 = -\frac{\nu}{2} A_x^1; \qquad A_y^3 = \begin{pmatrix} d_1 & f_1 & 0 & \cdots & \cdots & \cdots & 0 \\ e_2 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & e_j & d_j & f_j & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & f_{N_y-1} \\ 0 & \cdots & \cdots & \cdots & 0 & e_{N_y} & d_{N_y} \end{pmatrix}$$

These two representations of PGD aim at illustrate the different way to use the method, but none the resolution wich was detailed previously.

3.2. *Results*

The two PGD methods have been tested on a case where an analytical solution is known. In the following, Problem [26] will be solved with $\Omega =]-1;1[\times]-1;1[$, and a source term f :

$$f(x,y,t) = \frac{x^4 y^4}{12} - x^2 y^4 t - x^4 y^2 t + 4x^2 t - 4y^2 t \qquad [36]$$

In this case the problem has the following analytical solution :

$$\Theta_{ana}(x,y,t) = \frac{x^4y^4}{12}t + 2x^2t^2 - 2y^2t^2 \quad [37]$$

The boundary conditions are chosen verifying this analytical solution. The solution computed by the various solvers will be compared with the analytical solution Θ_{ana}. The PGD's solvers will be compared to the standard solver (bi-conjugate gradient). Thus, a relative error could be defined as follows :

$$\epsilon_{rel} = \frac{||\Theta_{solve} - \Theta_{ana}||_{L^2}}{||\Theta_{ana}||_{L^2}} \quad [38]$$

where Θ_{solve} is the solution computed by one of the solvers used (PGD(XYT), PGD(XY) or standard).

We will study the effect of the space-step size with a constant time-step size, and the effect of the time-step size with a constant space-step in each direction. These two test are done with a constant time interval I. Finally, we will study the effect of the time-interval lenght with a constant time-step a constant space-step. In the following, we choose the convergence criteria define in Equation 2.1.4 equal to 10^{-6} for each PGD methods.

3.2.1. *Effect of the space-step size*

For this test, the time step is set to $\delta t = 10^{-3}$, the time interval is set to $I =]0,1[$ and Problem [26] is solved with different space steps size.

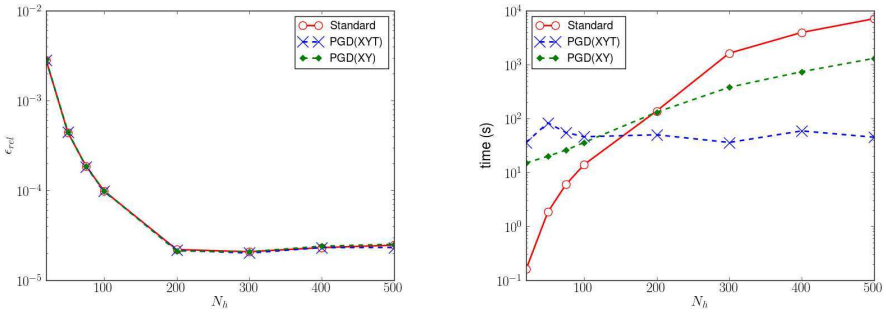


Figure 1. *Relative error (a) and computational duration (b) with the number of nodes of spatial discretisation for standard,* $PGD(XY)$ *and* $PGD(XYT)$ *solvers with* $I_t =]0,1[$ *and* $\delta t = 10^{-3}$

Figure 1a shows that appearance of the relative error with the number of nodes in each spatial direction is similar for each solvers. The computational duration with

the number of nodes is plotted on Figure 1b. We could notice that, from a certain number of nodes in each direction, PGD's methods become faster than the standard solver. Then, from 200 nodes in each direction, PGD(XY) solver is faster than the standard one, for $N_h = 500$ the computational duration is divided by seven. The figure also shows that from 150 nodes in each direction, PGD(XYT) method is faster than the standard one. In fact, for $N_h = 500$, computational duration is divided into one hundred and fifty with PGD(XYT) solver. This important time saving is due to the fact that the number of functions needed to approximate the solution decrease when the number of nodes increases (see Figure 2).

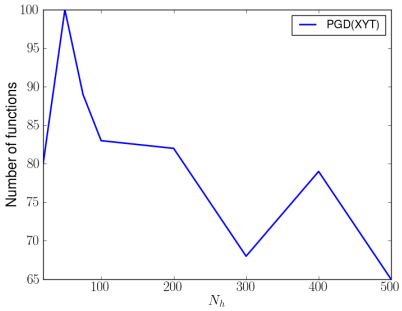


Figure 2. *Number of functions for PGD(XYT) solver with the number of nodes of spatial discretisation with* $I_t =]0, 1[$ *and* $\delta t = 10^{-3}$

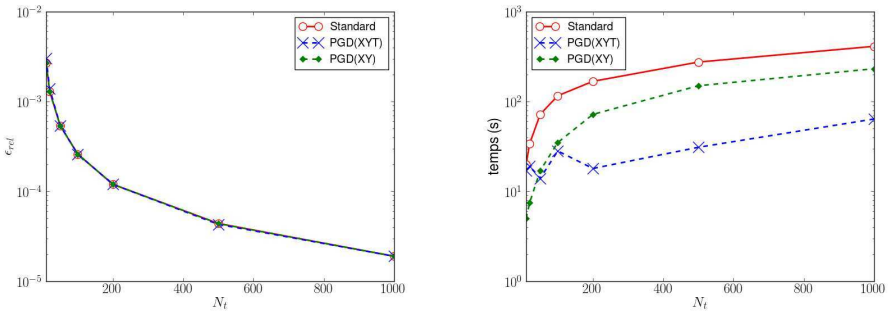
3.2.2. *Effect of the time-step size*

In this section, the number of nodes in each direction will be set to $N_h = 250$ and the time interval will be set to $I =]0, 1[$. Problem [26] will be solved with a time step varying between $\delta t = 10^{-3}$ (1000 nodes in the time direction) to $\delta t = 10^{-1}$ (10 nodes in the time direction).

Figure 3a shows that the relative error with the number of nodes in each spatial direction is similar for each solvers. As shown in Figure 3b PGD(XYT) is faster than the standard solver and faster than PGD(XY). In fact, it is seven times faster with $N_t = 1000$ than the standard solver and four times faster than the PGD(XY) solver. According to this figure, the behaviour of the PGD(XY) solver and of the standard one is very similar.

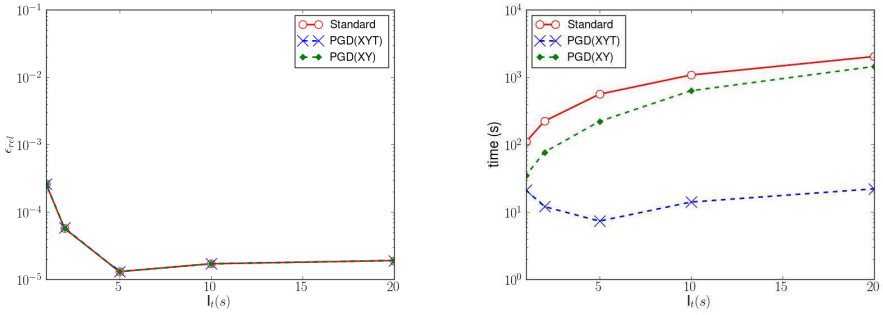
3.2.3. *Effect of the time-interval lenght*

In this part, the aim is to test the influence of the time interval lenght on the performance of each method. Then, time-step is set to $\delta t = 10^{-2}$ and the number of nodes in each space direction is set to $N_h = 250$. The time interval will be taken from $I =]0; 1[$ to $I =]0; 20[$.



(a) (b)

Figure 3. *Relative error (a) and computational duration (b) with the number of nodes of spatial discretisation for standard,* $PGD(XY)$ *and* $PGD(XYT)$ *solvers with* $I_t =]0, 1[$ *and* $N_h = 250$



(a) (b)

Figure 4. *Relative error (a) and computational duration (b) with the the time interval lenght for the solvers standard,* $PGD(XY)$ *and* $PGD(XYT)$ *with* $\delta t = 10^{-2}$ *and* $N_h = 250$

Figure 4a shows the evolution of the relative error with the time interval lenght. As for the previous section, errors are the same for the three solvers.

Concerning the computational duration, Figure 4b shows that the ratio between PGD(XY) and standard solver is the same for each time-interval. As in the previous test, PGD(XY) and standard solvers seem to have the same behaviour with the time interval lenght. PGD(XYT) is faster than the other solver. In fact, for a time interval equal to $I =]0, 20[$, PGD(XYT) is sixty five times faster than PGD(XY) and an hundred and fifty times faster than the standard solver.

4. Navier-Stokes equations

It exists many ways to solve incompressible Navier-Stokes equations. One of them consists in treating the problem with its primitive variable (velocity and pressure). But in this formulation, there is a limitation, termed the inf-sup (or LBB) condition, in approximating the velocity and the pressure. If this constraint is not respected, the numerical instabilities or the spurious pressure values are inevitable. That is the reason why it is necessary to discretize unknowns on a staggered-grid (see (Kress *et al.*, 2003)(Piller *et al.*, 2004)). Another way to solve incompressible 2D Navier-Stokes equations is to transform the equations into a fourth-order nonlinear partial differential equation with the biharmonic operator as a principal part where the unknown is the stream function ψ which exists thanks to the incompressibility constraint. This equation could be separated by introducing the vorticity ω, into systems of two second-order differential partial equations of which unknowns are the vorticity and the stream functions. This formulations studied in many papers, here we cite ((Tezduyar *et al.*, 1990)(Tokunaga *et al.*, 1994)(Chudanov *et al.*, 1995)(Ramsak *et al.*, 2005)(Kim *et al.*, 2007)).

4.1. *Stream function-vorticity formulation*

4.1.1. *Governing equations*

Let Ω be a fluid domain in $\mathbb{R}^2$, the primitive variable formulation for the unsteady incompressible Navier-Stokes flow could be written

$$\begin{cases} \dfrac{\partial \mathbf{u}}{\partial t} - \nu\Delta\mathbf{u} + \mathbf{u}.\nabla\mathbf{u} = -\dfrac{1}{\rho}\nabla p + \mathbf{f} \text{ in } \Omega\times]0,T[\\ \nabla.\mathbf{u} = 0 \\ \mathbf{u}(t=0) = \mathbf{u}_0 \\ \mathbf{u} = \mathbf{g}_D \text{ on } \partial\Omega \end{cases} \quad [39]$$

where $\mathbf{u}$ and p are respectively the velocity and the pressure fields, and $\mathbf{f}$ is a known body force. ν is the kinematic viscosity and ρ is the constant fluid density.

The incompressibility conditions of Problem [39] implies the existence of a stream function ψ satisfying :

$$\mathbf{u} = (u_1, u_2) = (\frac{\partial\psi}{\partial y}, -\frac{\partial\psi}{\partial x}) \quad [40]$$

Taking the 2D curling operator ($\nabla\times$) of both sides of velocity equations of Problem [39], and using the following relation :

$$\nabla \times \mathbf{u} = -\Delta\psi.\overrightarrow{\mathbf{z}} \quad [41]$$

and

$$\nabla \times (\mathbf{u}.\nabla)\mathbf{u} = (\nabla.\mathbf{u})(\nabla \times \mathbf{u}) + (\mathbf{u}.\nabla)(\nabla \times \mathbf{u}) \quad [42]$$

the following equation is obtained

$$\frac{\partial \Delta\psi}{\partial t} - \nu\Delta^2\psi + (\mathbf{u}.\nabla)\Delta\psi = -\nabla \times \mathbf{f} \quad [43]$$

Generally this fourth-order partial differential equation is difficult to solve directly. One way to give this problem tractable is to apply a separation of this equation. It is possible to introduce the definition of vorticity in fluids dynamics, and the Equation [43] with his boundary conditions become:

$$\begin{cases} \Delta\psi = -\omega & \text{dans } \Omega \\ \dfrac{\partial\psi}{\partial n} = G & \text{sur } \partial\Omega \end{cases} \quad [44]$$

and

$$\begin{cases} \dfrac{\partial\omega}{\partial t} - \nu\Delta\omega + (\mathbf{u}.\nabla)\omega = \nabla \times \mathbf{f} & \text{dans } \Omega \\ \omega = \dfrac{\partial u_2}{\partial x} - \dfrac{\partial u_1}{\partial y} & \text{sur } \partial\Omega \end{cases} \quad [45]$$

where the vorticity $\omega = \omega.\overrightarrow{\mathbf{z}}$ is given by $\nabla \times \mathbf{u}$ and G is given by the boundary value of ψ corresponding to the boundary velocity. For further details see (Chudanov *et al.*, 1995)(Ramsak *et al.*, 2005)(Kim *et al.*, 2007).

Solving the Navier-Stokes equations in "stream-line vorticity" formulation consists in solving successively Problems [45] and [44].

4.1.2. *Discretisation*

Here we consider that the known body force **f** is equal to zero. Let us first consider the Equation [45] giving the vorticity. This equation can be written as:

$$\frac{1}{\delta t}\omega^{n+1} - \frac{\nu}{2}\Delta\omega^{n+1} = B \quad [46]$$

where the diffusive term has been discretized using a Cranck-nicholson scheme, and the convective term has been discretized using an Adams-Bashforth scheme.

$$B = \frac{1}{\delta t}\omega^n + \frac{\nu}{2}\Delta\omega^n - \frac{1}{2}(3\mathbf{u}^n.\nabla\omega^n - \mathbf{u}^{n-1}.\nabla\omega^{n-1}) \quad [47]$$

Equations [44] and [45] will then be spatially discretized with the finite volume method with a second order scheme.

4.1.3. *PGD associated to the streamline-vorticity formulation*

The Proper Generalized Decomposition will be used for the calculation of vorticity ω (Equation [45]) and for the the calculation of the streamline ψ (Equation [44]). In order to use the PGD algorithm detailed in Section 2.2, the Equations [45] and [44] have to be written in a discrete form as in Equation [12].

We look for ω^{n+1} as:

$$\omega^{n+1} = \sum_{k=1}^{m} \alpha^k F^k(x) G^k(y) \qquad [48]$$

Within the finite volume framework, Equation [46] has to be integrated on each control volume Ω_{lp} giving the following equation:

$$\frac{1}{\delta t} \int_{\Omega_{lp}} \omega^{n+1} d\Omega - \frac{\nu}{2} \int_{\Omega_{lp}} \Delta\omega^{n+1} d\Omega = \int_{\Omega_{lp}} B d\Omega \qquad [49]$$

Introducing Equation [48] in Equation [49], the left hand-side could be rewritten as:

$$\sum_{k=1}^{m} \left(\frac{\alpha^k}{\delta t} \int_{X_l} F^k dx \int_{Y_p} G^k dy - \frac{\nu}{2} \int_{X_l} \frac{d^2 F^k}{dx^2} dx \int_{Y_p} G^k dy + \int_{X_l} F^k dx \int_{Y_p} \frac{d^2 G^k}{dy^2} dy \right) \qquad [50]$$

where X_l (respectively Y_p) is the definition interval of the control volume Ω_{lp} in the direction x (resp. y).

Then Equation [49] could be written:

$$\mathcal{A}\omega^{n+1} = \mathcal{B} \qquad [51]$$

where,

$$\mathcal{A}\omega^{n+1} = \sum_{q=1}^{3} \sum_{k=1}^{m} \alpha^k \; A_x^q F^k \otimes A_y^q G^k \qquad [52]$$

The operators A_x^q and A_y^q are exactly the same matrices as these defined in Section 3.1.2.

Now, we have precised the tensorial operators needed to solve the streamline (Equation [44]). We thus look for ψ as:

$$\psi^{n+1} = \sum_{k=1}^{N_\psi} \alpha_\psi^k F_\psi^k \otimes G_\psi^k \qquad [53]$$

After discretization with finite volume method and after taking the previous form of $\tilde{p}$ in Equation [44] into account , we obtain :

$$\sum_{k=1}^{N_\psi} \alpha_\psi^k \left(\int_{X_l} \frac{d^2 F_\psi^k}{dx^2} dx \int_{Y_p} G_\psi^k dy + \int_{X_l} F_\psi^k dx \int_{Y_p} \frac{d^2 G_\psi^k}{dy^2} dy \right) = \int_{X_l} \int_{Y_p} D dx dy \qquad [54]$$

on the volume Ω_{lp}, which could be written

$$\mathcal{C}\psi^{n+1} = \mathcal{D} \qquad [55]$$

with,

$$\mathcal{C}\psi^{n+1} = \sum_{q=1}^{2} \sum_{k=1}^{N_\psi} \alpha_\psi^k \, \mathbb{C}_x^q \mathbf{F}_\psi^k \otimes \mathbb{C}_y^q \mathbf{G}_\psi^k \qquad [56]$$

where, using the notations of Section 3.1.1,

$$A_x^1 = \begin{pmatrix} a_1 & c_1 & 0 & \cdots & \cdots & \cdots & 0 \\ b_2 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & b_i & a_i & c_i & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & c_{N_x-1} \\ 0 & \cdots & \cdots & \cdots & 0 & b_{N_x} & a_{N_x} \end{pmatrix}; \qquad A_y^1 = diag(\Delta_{y_1}, \cdots, \Delta_{y_{N_y}})$$

$$A_x^2 = diag(\Delta_{x_1}, \cdots, \Delta_{x_{N_x}}); \qquad A_y^2 = \begin{pmatrix} d_1 & f_1 & 0 & \cdots & \cdots & \cdots & 0 \\ e_2 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & e_j & d_j & f_j & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & f_{N_y-1} \\ 0 & \cdots & \cdots & \cdots & 0 & e_{N_y} & d_{N_y} \end{pmatrix}$$

As for vorticity, this tensorial form is easily solved within the General PGD framework.

From the streamline we can easily define the velocity $\mathbf{u}^{n+1} = (u_1^{n+1}, u_2^{n+1})$ in tensorial form :

$$\begin{aligned} u_1^{n+1} &= \frac{\partial}{\partial y}(\sum_{k=1}^{N_\psi} \alpha_\psi^k \mathbf{F}_\psi^k \otimes \mathbf{G}_\psi^k) \\ u_2^{n+1} &= -\frac{\partial}{\partial x}(\sum_{k=1}^{N_\psi} \alpha_\psi^k \mathbf{F}_\psi^k \otimes \mathbf{G}_\psi^k) \end{aligned} \qquad [57]$$

4.2. *Results*

4.2.1. *The steady lid-driven cavity*

Let us consider the square domain $\Omega =]0;1[\times]0;1[$ for the resolution of the Navier-Stokes equations with Dirichlet boundary conditions, as illustrated in Figure 5. Here the two velocities components vanish on the boundary, except on the north face where the x-velocity is equal to U_0.The simulations were made with $\delta t = 10^{-3}$, for two Reynolds numbers[2] ($Re = 100$ and $Re = 1000$) and with the source term f (see Equation [39]) equal to zero.

Since we were looking for a stationary flow, we defined the following convergence criteria:

$$\frac{||u_i^k - u_i^{k-1}||}{||u_i^k||} \leq \epsilon_u \qquad \text{with} \quad i = 1, 2 \qquad [58]$$

2. $Re = \dfrac{U_0 \times d}{\nu}$ where d is the width of the cavity.

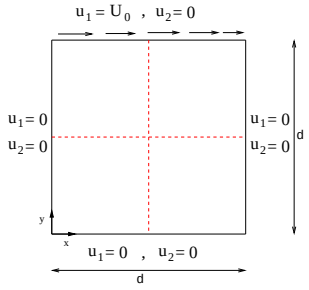
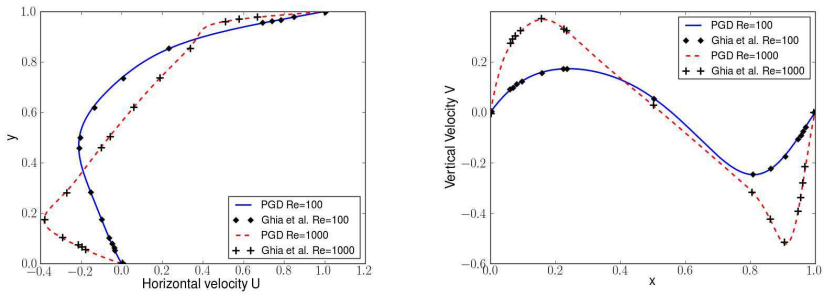


Figure 5. *Geometry of lid-driven cavity*

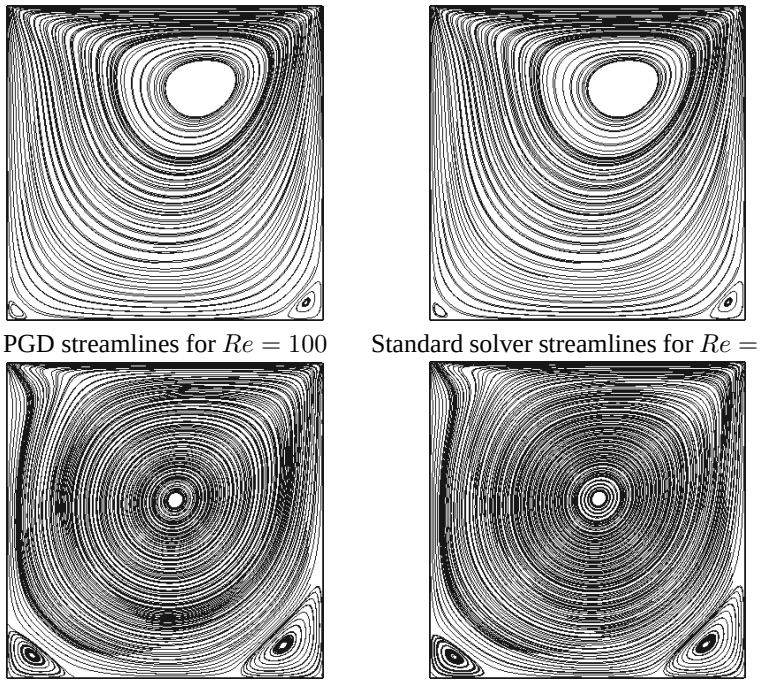
In the following, ϵ_u is set to 1.10^{-10}.Figure 7 denotes the comparison between the streamlines computed from the PGD method and the standard method for $Re = 100$ and $Re = 1000$. This figure shows that streamlines obtained by PGD are very close to those obtained with the standard solver. Figure 6 illustrates the comparison between the x-velocity at $x = 0.5$ and the y-velocity at $y = 0.5$ (see the dashed line on Figure 5 computed with the PGD solver, and the results obtained by Ghia. and al. (see (Ghia *et al.*, 1982)) for the two Reynolds numbers considered. The results drawn in this figure were obtained with the same number of nodes in each direction ($N_h = 200$). The velocity profiles obtained with the PGD method clearly correspond to those obtained by the standard method and Ghiaś results.



(a) (b)

Figure 6. *Comparison of the x-velocity at x=0.5 (a) and y-velocity at y=0.5 (b) with Ghia. and al. results for* $Re = 100$ *and* $Re = 1000$

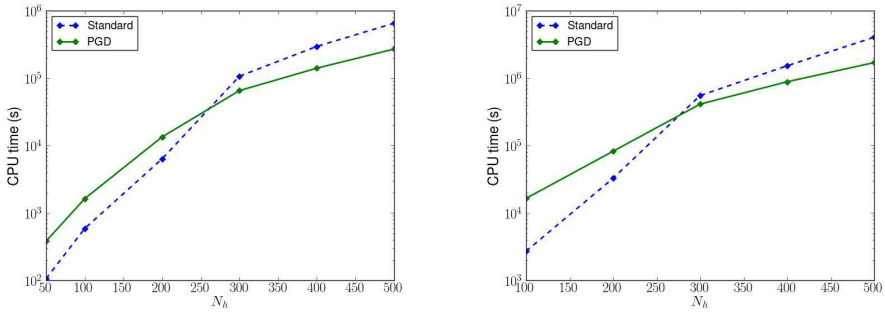
It is interesting to compare the CPU time of the PGD solver and standard solver with the numbers of nodes (Figure 8) for each Reynolds number. It can be seen that beyond a mesh size of 300×300, PGD becomes faster than the standard method. In fact, for a mesh size of 500×500, the CPU time was twice lower with the PGD solver for $Re = 100$. For $Re = 1000$, with the same grid (500×500), PGD was four times faster than the standard solver.



PGD streamlines for $Re = 100$ Standard solver streamlines for $Re = 100$

PGD streamlines for $Re = 1000$ Standard solver streamlines for $Re = 1000$

Figure 7. *Streamlines computed with PGD and computed from the standard solver with* $N_h = 200$ *for* $Re = 100$ *and* $Re = 1000$



(a) (b)

Figure 8. *Comparison of CPU time between the PGD solver and standard solver for the resolution in lid-driven cavity for* $Re = 100$ *(a) and* $Re = 1000$ *(b)*

4.2.2. *The unsteady lid-driven cavity*

Here we will consider the same lid-driven cavity as in the previous case, with the same boundary conditions. We will study the case of a post-critical Reynolds number where the flow is unsteady. We chose to fix this Reynolds number at 10000. The simulations were made with $\delta t = 10^{-3}$ and with a 250×250 mesh grid.

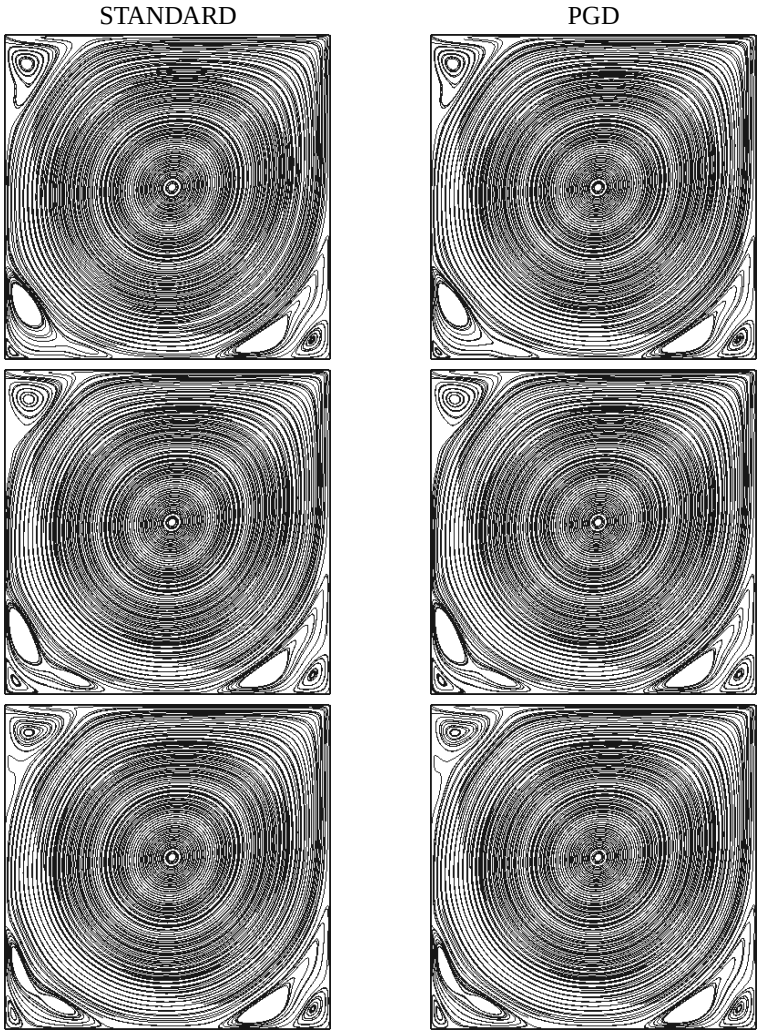


Figure 9. *Change in the stream function during one main period for* $Re = 10000$ *on a* 250×250 *grid. From top to bottom for each solver (standard and PGD) times* $t = 0$, $t = 0.3$ *and* $t = 0.6$ *are represented*

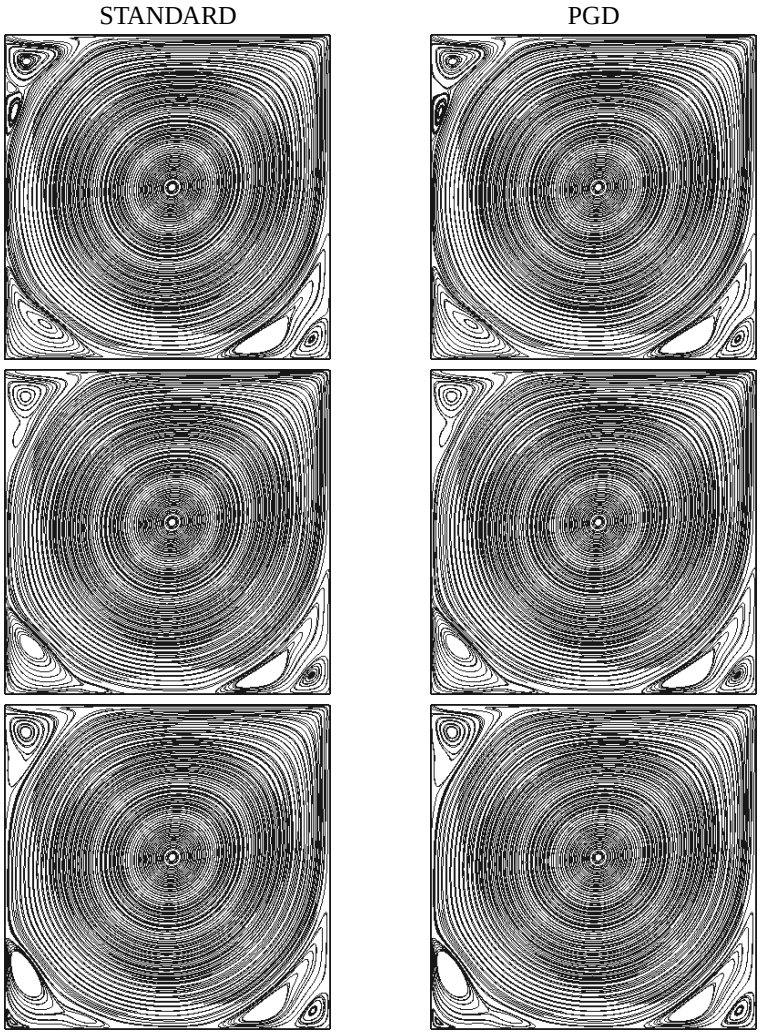


Figure 10. *Change in the stream function during one main period for* $Re = 10000$ *on a* 250×250 *grid. From top to bottom for each solver (standard and PGD) times* $t = 0.9$, $t = 1.2$ *and* $t = 1.51$ *are represented*

The results obtained using the standard model and with PGD are similar to those obtained by Bruneau and Saad in (Bruneau *et al.*, 2006). In fact with our simulations we find a period of $1.51s$, whereas Bruneau et al. have found a period of $1.64s$.

The streamlines are also shown on Figures 9 and 10 at different time points of the period and for every single solver studied. It is also worth noting that there is a very good correspondance between the two models and the results obtained in (Bruneau *et al.*, 2006).

The results of this part demonstrate that the computation of the lid-driven cavity with $Re = 10000$ was performed efficiently with PGD. In fact, we were able to reproduce some results from the literature, like the behaviour of the vortex during one main period.

5. Conclusion

In this paper we have applied the PGD method to solve the Navier-Stokes equations in stream-vorticity formulation for the flow in a 2D lid-driven cavity. In the steady cases ($Re = 100$ and $Re = 1000$), PGD results are in line with the ones obtained by the standard solver and with the results issue from the literature. Furthermore, from 300 nodes in each direction, time-saving has been succeded. In the unsteady case ($Re = 10000$), PGD method gives a period similar to the one obtained in the litterature and the comportement of the streamline during one period is similar to the one obtained by the standard solver. As in the exemple of the unsteady diffusion equation, we have seen that the PGD with time variable in the decomposition is very efficient in time saving, the next development is to use this approach to solve Navier-Stokes equations. Moreover, the introduction of a solid in the fluid problem is subject to further development.

6. References

Ammar A., Chinesta F., Diez P., Huerta A., An Error Estimator for Separated Representations of Highly Multidimensional Models, *Computer Methods in Applied Mechanics and Engineering*, vol. 199, p. 1872-1880, 2010.

Ammar A., Chinesta F., Falco A.*Archives of Computational Methods in Engineering, In press*, n.d.

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*, vol. 139, n° 3, p. 153 - 176, 2006a.

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 modelling of complex fluids: Part II: Transient simulation using space-time separated representations , *Journal of Non-Newtonian Fluid Mechanics*, vol. 144, n° 2-3, p. 98 - 121, 2007.

Ammar A., Ryckelynck D., Chinesta F., Keunings R., On the reduction of kinetic theory models related to finitely extensible dumbbells, *J. Non-Newtonian Fluid Mech.*, vol. 134, p. 136-147, 2006b.

Arnold D., Liu X., Local errors estimates for finite elemnt discretizations of the Stokes equations , *Math. Model. Numer. Anal.*, vol. 29, p. 367 - 389, 1995.

Bruneau C.-H., Saad M., The 2D lid-driven cavity problem revisited, *Computers and Fluids*, vol. 35, n° 3, p. 326 - 348, 2006.

Chinesta F., Ammar A., Joyot P., The nanometric and micrometric scales of the structure and mechanics of materials revisited: An introduction to he challenges of fully deterministic numerical descriptions , *International Journal for Multiscale Computational Engineering*, vol. 6/3, p. 191-213, 2008a.

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*, vol. 197, n° 5, p. 400 - 413, 2008b. Enriched Simulation Methods and Related Topics.

Chudanov V. V., Popkov A. G., Churbanov A. G., Vabishchevich P. N., Makarov M. M., Operator-splitting schemes for the stream function-vorticity formulation, *Computers and Fluids*, vol. 24, n° 7, p. 771 - 786, 1995.

Dowell E., HAll K., Â«Modelling of fluid-structure interactionÂ» *Annal Review of Fluids Mechanics*, vol. 33, p. 445-490, 2001.

Ghia U., Ghia K., Shin C., High-re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method, *J. of Computational physics*, vol. 48, p. 387 - 411, 1982.

Kim Y., Kim D. W., Jun S., Lee J. H., Meshfree point collocation method for the stream-vorticity formulation of 2D incompressible Navier-Stokes equations , *Computer Methods in Applied Mechanics and Engineering*, vol. 196, n° 33-34, p. 3095 - 3109, 2007.

Kress W., Nilsson J., Boundary conditions and estimates for the linearized Navier-Stokes equations on staggered grids , *Computers and Fluids*, vol. 32, n° 8, p. 1093 - 1112, 2003.

Ladeveze P., Non linear computational structural mechanics : new approaches and non-incremental methods of calculation , *Mechanical engineering series, Springer;*, 1999.

Ladeveze P., Passieux J.-C., Neron D., The LATIN multiscale computational method and the Proper Generalized Decomposition , *Computer Methods in Applied Mechanics and Engineering*, vol. 199, n° 21-22, p. 1287 - 1296, 2010. Multiscale Models and Mathematical Aspects in Solid and Fluid Mechanics.

Li J., Sun W. W., Spectral method of decoupling the vorticity and stream function for the incompressible fluid flows, *Computers and Mathematics with Applications*, vol. 37, n° 1, p. 35 - 40, 1999.

Liberge E., Hamdouni A., Reduced order modelling method via proper orthogonal decomposition (POD) for flow around an oscillating cylinder, *Journal of Fluids and Structures*, vol. 26, n° 2, p. 292 - 311, 2010.

Lieu T., Farhat C., Lesoinne M., Reduced-order fluid/structure modeling of a complete aircraft configuration , *Computer Methods in Applied Mechanics and Engineering*, vol. 195, n° 41-43, p. 5730 - 5742, 2006. John H. Argyris Memorial Issue. Part II.

Mokdad B., Pruliere E., Ammar A., Chinesta F., On the simulation of kinetic theory models of complex fluids using the FokkerâPlanck approach, *Appl. Rheol.*, vol. 2, p. 1 - 14, 2007.

Nouy A., A generalized spectral decomposition technique to solve a class of linear stochastic partial differential equations , *Computer Methods in Applied Mechanics and Engineering*, vol. 196, n° 45-48, p. 4521 - 4537, 2007.

Nouy A., Maitre O. P. L., Generalized spectral decomposition for stochastic nonlinear problems , *Journal of Computational Physics*, vol. 228, n° 1, p. 202 - 235, 2009.

Piller M., Stalio E., Finite-volume compact schemes on staggered grids , *Journal of Computational Physics*, vol. 197, n° 1, p. 299 - 340, 2004.

Placzek A., Tran D.-M., Ohayon R., Hybrid proper orthogonal decomposition formulation for linear structural dynamics , *Journal of Sound and Vibration*, vol. 318, n° 4-5, p. 943 - 964, 2008.

Ramsak M., Skerget L., Hribersek M., Zunic Z., A multidomain boundary element method for unsteady laminar flow using stream function-vorticity equations , *Engineering Analysis with Boundary Elements*, vol. 29, n° 1, p. 1 - 14, 2005.

Ryckelynck D., Reduction a priori de modeles thermomecaniques, *Comptes Rendus Mecanique*, vol. 330, n° 7, p. 499-505, 2002.

Ryckelynck D., A priori hyperreduction method: an adaptive approach, *Journal of Computational Physics*, vol. 202, p. 346-366, 2005.

Ryckelynck D., Hermanns L., Chinesta F., Alarcon E., An efficient a priori model reduction for boundary element models , *Engineering Analysis with Boundary Elements 29*, vol. 29, p. 796-801, 2005.

Taylor C., Hood P., A numerical solution of the Navier-Stokes equations using the finite element technique , *Computers and Fluids*, vol. 1, p. 73 - 100, 1973.

Tezduyar T., Liou J., Ganjoo D., Incompressible flow computations based on the vorticity-stream function and velocity-pressure formulations , *Computers and Structures*, vol. 35, n° 4, p. 445 - 472, 1990. Special Issue: Frontiers in Computational Mechanics.

Tokunaga H., Tanaka T., Ichinose K., Satofuka N., Numerical solutions of incompressible flows in multiply connected domains by the vorticity stream function formulation , *Computers and Fluids*, vol. 23, n° 2, p. 241 - 249, 1994.

Verdon N., Allery C., Beghein C., Hamdouni A., Ryckelynck D., Reduced-Order Modelling for solving linear equations and non-linear equations , *Communications in Numerical Methods in Engineering*, 2009.