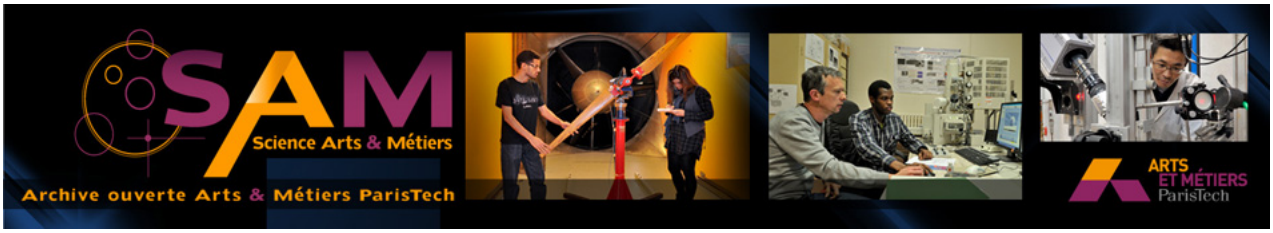
Proper Generalized Decomposition method for incompressible Navier-Stokes equations with a spectral discretization

Antoine Dumon, Cyrille Allery, Amine Ammar



Proper Generalized Decomposition method for incompressible Navier–Stokes equations with a spectral discretization

A. Dumon [a,*], C. Allery [a], A. Ammar [b]

[a] LaSIE FRE-CNRS 3474, Pôle Sciences et Technologie, Avenue Michel Crepeau, 17042 La Rochelle Cedex 1, France
[b] Arts et Metiers ParisTech, 2 Bvd du Ronceray, 49035 Angers Cedex 01, France

ARTICLE INFO

Keywords:
Reduced order model
Proper Generalized Decomposition
Incompressible flow
Spectral discretization

ABSTRACT

Proper Generalized Decomposition (PGD) is a method which consists in looking for the solution to a problem in a separate form. This approach has been increasingly used over the last few years to solve mathematical problems. The originality of this work consists in the association of PGD with a spectral collocation method to solve transfer equations as well as Navier–Stokes equations. In the first stage, the PGD method and its association with spectral discretization is detailed. This approach was tested for several problems: the Poisson equation, the Darcy problem, Navier–Stokes equations (the Taylor Green problem and the lid-driven cavity). In the Navier–Stokes problems, the coupling between velocity and pressure was performed using a fractional step scheme and a $\mathbb{P}_{\mathbf{N}}-\mathbb{P}_{\mathbf{N-2}}$ discretization. For all problems considered, the results from PGD simulations were compared with those obtained by a standard solver and/or with the results found in the literature. The simulations performed showed that PGD is as accurate as standard solvers. PGD preserves the spectral behavior of the errors in velocity and pressure when the time step or the space step decreases. Moreover, for a given number of discretization nodes, PGD is faster than the standard solvers.

1. Introduction

Over the last few decades, many approaches have been used to solve transfer equations. Some problems are related to very large systems that cannot be easily solved numerically. Therefore, traditional methods require very long computation times and a large storage capacity. One way to reduce computation time and memory storage when solving partial differential equations (PDE) consists in using reduced order models (ROM). ROM have been used extensively in recent decades. They consist in approximating the solution f of the PDE as follows

$$f(\mathbf{x},t) \approx f_m(\mathbf{x},t) = \sum_{k=1}^{m} a_k(t)\Phi_k(\mathbf{x}), \tag{1}$$

where $\mathbf{x}$ is the spatial coordinates vector and $\Phi_k(\mathbf{x})$ is a low-dimensional reduced basis. m is the reduced-basis size which is usually much smaller than the full grid size of the discretized solution. The coefficients $a_k(t)$ are solutions for a very low-order system obtained by a Galerkin projection of the initial equation of the problem over this basis. The difference with ROM lies in the way in which the reduced basis is computed.

* Corresponding author.
E-mail addresses: adumon01@univ-lr.fr (A. Dumon), cyrille.allery@univ-lr.fr (C. Allery), Amine.Ammar@ensam.eu (A. Ammar).

http://dx.doi.org/10.1016/j.amc.2013.02.022

Various reduced bases can be used to reduce the order of models, including the Lagrange, Hermite, Taylor or POD basis. In fluid mechanics, the most popular reduced-order modeling technique is POD (Proper Orthogonal Decomposition). This method has been used to predict the dynamics of a flow [1–3], to control flows [4–7], or to compute particle dispersion in a flow [8,9]. Note that an "a posteriori" error estimation is available for the reduced basis solution of parametrized linear parabolic partial differential equations [10], and of parametrized Stokes [11] and Navier–Stokes [12–14] equations. To build these bases, these approaches often require some snapshots of the flow, which mat require significant computation time. In order to circumvent this drawback, "a priori" model reduction techniques were developed. These methods consist in constructing a reduced basis without "a priori" knowledge of the solution.

The first is the A Priori model Reduction (APR) which has been the subject of several developments [15–18]. In this approach, the basis is adaptively improved and expanded with the residuals of the full discretized model. The incremental process is carried out by taking into account the whole time interval of the equation resolution.

The second, which was applied here, is Proper Generalized Decomposition (PGD). PGD consists in looking for the solution $f(x_1,\ldots,x_N)$ of a problem in the following separated form:

$$f(x_1,\ldots,x_N)\approx\sum_{i=1}^{Q}\prod_{k=1}^{N}F_{ki}(x_k)=\sum_{i=1}^{Q}F_{1i}(x_1)F_{2i}(x_2)\cdots F_{Ni}(x_N)$$

(x_i can be any scalar or vector variable 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 the unknowns involved in the solution is $Q\times N\times M$ instead of the M^N degrees of freedom involved in standard techniques. In most cases, when the field is sufficiently regular, the number of terms Q in the finite sum is usually quite small (a few dozen) and in all cases the approximation converges towards the solution of the full grid description (see [19–21]). The method was originally proposed in the nineties by Ladeveze et al. as one of the main ingredients of the LATIN method (see for example [22,23]). In this framework, the approach is based on a space–time representation and is called "radial approximation". This space–time representation was also used by Nouy [24,25] to solve stochastic problems (in this context, the method is called "Generalized Spectral Decomposition") A few years ago, Ammar et al. extended PGD to multidimensional decomposition to solve models of polymeric systems [26–28]. Since this work, the PGD was also applied to solve quantum chemistry problems [29], 3D elastic problems [30], fluid problems [31,32] and structural optimization problems [33]. Finally, a review of the method can be found in [34,35]. PGD is usually restricted to a rectangular domain for a best performance of the tensorial decomposition. However it can be used in a non rectangular domain [36]. In this case the non rectangular domain is immersed in a rectangular one and the initial operator is associated with a conjugate operator describing the boundaries.

In order to treat transient problems, PGD could be applied using one of the three following decompositions.

- The first consists in a time space decomposition: $f(t,\mathbf{x})=\sum_{i=1}^{Q}F_t^i(t)F_{\mathbf{x}}^i(\mathbf{x})$. We are directly looking for a time/space solution. The main drawback encountered here is that a full grid description is required to define the functions over the physical space.
- To circumvent this difficulty, the second possibility consists in writing a full decomposition involving the two dimensions of the physical space: $f(t,x,y)=\sum_{i=1}^{Q}F_t^i(t)F_x^i(x)F_y^i(y)$. In practice, making such a decomposition leads to a significant increase in the number of terms required in the sum.
- The third possibility, which was used in this work, consists in keeping the incremental approach and performing the separation over the physical space at each time step: $f^t(x,y)=\sum_{i=1}^{Q}F_x^i(x)F_y^i(y)$.

The PGD approach is usually associated with a finite-difference, finite-element or finite volume discretization. In some situations, such as turbulence flows, these dicretization methods are not accurate enough to simulate the relevant physical phenomena. This drawback cannot be easily solved using a mesh refinement due to the memory storage limit. To avoid this, it is possible to use spectral methods. Therefore, in this paper, we coupled the PGD method with a spectral discretization to solve the Navier–Stokes equations. The aim was to show numerically that the PGD method, while being faster than the standard method, does not affect the accuracy of the solution and preserves the benefits of spectral discretization. It should be noted that Nouy [37] has already used a spectral discretization (Spectral Element Method) with PGD to solve an advection–diffusion problem. The originality of this work is the extension of the approach to solve the Navier–Stokes equations, which leads to some difficulties. In fact, the major difficulty in solving incompressible flows is that the velocity and the pressure are coupled with the incompressibility constraints. This is achieved by using a fractional step scheme. A second difficulty is related to the choice of the approximation spaces associated with velocity and pressure. These spaces must satisfy the inf-sup condition [38]. One possibility consists in approximating the pressure with polynomials of a degree lower than those approximating the velocity (with a difference equal to two between the two degrees). This method is called the $\mathbb{P}_{\mathbf{N}}-\mathbb{P}_{\mathbf{N-2}}$ formulation [39–41].

It should be remembered that the originality of this work is the extension of the PGD approach to solve the Navier–Stokes equations. In fact, in this work, the spectral formalism associated with the PGD method is detailed and the method is used to solve the Navier–Stokes equations through a fractional step scheme combined with the $\mathbb{P}_{\mathbf{N}}-\mathbb{P}_{\mathbf{N-2}}$ formulation.

This paper is organized as follows. First, the PGD method will be presented in a general framework. Secondly, spectral discretization will be detailed and its association with PGD performed on a Poisson equation. Then, the discretization of

the Navier–Stokes equations with a $\mathbb{P}_{\mathbf{N}}-\mathbb{P}_{\mathbf{N-2}}$ method associated with a fractional step scheme and its PGD formulation is described. Finally, the results on the test cases of the Darcy and Taylor–Green problems and the 2D lid-driven cavity in stationary case are detailed.

2. PGD and spectral formulation

2.1. Description of PGD

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

$$\text{Find } U(x,y) \text{ as} \begin{cases} \mathcal{L}(U) = \mathcal{G} & \text{in}\Omega \\ +\text{Boundary Conditions} \end{cases}, \tag{2}$$

where $\mathcal{L}$ is a linear 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:

$$U(x,y) \approx U_m(x,y) = \sum_{i=1}^{m} \alpha^i F^i(x) G^i(y), \tag{3}$$

where $U_m(x,y)$ is the approximation of the solution of order m. The coefficients $\alpha^i \in \mathbb{R}$ are the weighting factors. At each iteration, the solution is enriched with an additional term $\alpha^{m+1} F^{m+1}(x) G^{m+1}(y)$. In the following, an iteration of the PGD algorithm will consist in constructing the solution of order $m+1$ from the solution of order m by applying the following three steps:

1. *Enrichment step*: At the m stage, the solution approximation of order $m-1$ is taken as known. In this step we look for the new product of functions $F^m(x)G^m(y)$ such as:

$$U_m(x,y) = \sum_{i=1}^{m-1} \alpha^i F^i(x) G^i(y) + F^m(x) G^m(y). \tag{4}$$

Introducing this expression into (2), results in:

$$\mathcal{L}(U_{m-1} + F^m G^m) = \mathcal{G} + Res^m, \tag{5}$$

where Res^m, defined by

$$Res^m = \mathcal{L}(U_m) - \mathcal{G} \tag{6}$$

is a residual, due to the fact that Eq. (4) is an approximation of the solution. To determine F^m and G^m, Eq. (5) is projected onto each of the unknowns F^m and G^m. Furthermore, the residual must be orthogonal to the F^m and G^m functions. Thus, we obtain the following applications:

- The application $S_m : X \to Y$ that associates a function dependent on $x, F^m \in X$, to a function dependent on $y, G^m \in Y$ which is given by:

$$\langle \mathcal{L}\left(F^m(x)G^m(y)\right), F^m \rangle_{L^2(X)} = \langle \mathcal{G} - \mathcal{L}(U_{m-1}(x,y)), F^m \rangle_{L^2(X)}. \tag{7}$$

- The application $T_m : Y \to X$ that associates a function dependent on $y, G^m \in Y$, to a function dependent on $x, F^m \in X$ which is given by:

$$\langle \mathcal{L}\left(F^m(x)G^m(y)\right), G^m \rangle_{L^2(Y)} = \langle \mathcal{G} - \mathcal{L}(U_{m-1}(x,y)), G^m \rangle_{L^2(Y)} \tag{8}$$

where $\langle \cdot, \cdot \rangle_{L^2(X)}$ and $\langle \cdot, \cdot \rangle_{L^2(Y)}$ are the scalar products on L^2, in the x and y directions.

In order to obtain the new functions F^m and G^m, Eq. (7) and (8) must be solved simultaneously. This was achieved using the standard fixed point algorithm detailed in Table 1. There are some restrictions for the convergence of the fixed point method. If the operators are symmetrical, convergence is guaranteed. If not, a preconditioned symmetrical problem is built up by multiplying the initial operator (and the second member) by its transposed form. The initialisation of the fixed point algorithm is also an important factor for the convergence of the method. In this study, the functions were randomly initialized and satisfied the boundary conditions.

2. *Projection step:* In order to increase the accuracy of the decomposition, the m α^i coefficients are now sought in such a way that the residual Res^m is orthogonal to each product of functions $F^i G^i$. The α^i coefficients are then solutions of the following systems of equations:

$$\left\langle \mathcal{L}\left(\sum_{i=1}^{m} \alpha^i F^i(x) G^i(y)\right), F^k G^k \right\rangle_{L^2(\Omega)} = \langle \mathcal{G}, F^k G^k \rangle_{L^2(\Omega)} \text{ for } 1 \leqslant k \leqslant m \tag{9}$$

Table 1
Fixed point algorithm (η is fixed by the user).

1. Initialization of $G^{(0)}$ (randomly)
2. **for** $k = 1, k_{max}$ **do**
3. Computation of $F^{(k)} = T_m(G^{(k-1)})$
4. Computation of $G^{(k)} = S_m(F^{(k)})$
5. Check convergence of $(F^{(k)}G^{(k)})$
   $(||F^{(k)} - F^{(k-1)}||_{L^2(X)} \cdot ||G^{(k)} - G^{(k-1)}||_{L^2(Y)} \leq \eta)$
6. **end for**
7. Let $F^m = F^{(k)}$ and $G^m = G^{(k)}$

We denote $W : (X)^m \times (Y)^m \to \mathbb{R}^m$as the application that associates functions $\mathbb{F}^m = \{F^i\}_{i=1}^m$ and $\mathbb{G}^m = \{G^i\}_{i=1}^m$ to the vector $\Lambda_m = \{\alpha_i\}_{i=1}^m \in \mathbb{R}^m$. Its expression is given by (9).

3. *Checking the convergence step:*
   If the L^2 norm of the residual defined by Eq. (6) is lower than the coefficient ϵ set by the user, the PGD algorithm is converged. Otherwise, one more iteration is needed, and the enrichment and projection steps are repeated taking $m = m + 1$ until convergence is obtained.

The PGD algorithm[1] is summarized in Table 2. In this work, the PGD algorithm was achieved by performing a discretization of the three previous steps on Tchebychev–Gauss–Lobato collocation points, which is detailed in the following section.

2.2. Spectral discretization

We defined the resolution domain $\Omega =]-1,1[\times]-1,1[$. It is noteworthy that a simple affine transformation can deal with any rectangular 2D domain $]a,b[\times]a',b'[$. The domain Ω is discretized using the Tchebychev–Gauss–Lobato grid defined as:

$$\begin{cases} x_i = \cos(i\frac{\pi}{N_x}) \text{ for } i = 0,\ldots,N_x \\ y_j = \cos(j\frac{\pi}{N_y}) \text{ for } j = 0,\ldots,N_y \end{cases} \tag{10}$$

where N_x and N_y are the maximum indexes of collocation points in the x and y direction.

The PGD approximation of the solution of order m is given by:

$$U_m(x,y) = \sum_{k=1}^{m} \alpha^i F^k(x) G^k(y) \tag{11}$$

where $F^k(x)$ (respectively $G^k(y)$) are approximated by polynomials of order N_x on the (N_x+1) nodes x_i (respectively, polynomials of order N_y on the (N_y+1) nodes y_j) of the grid. Then,

$$F^k(x) = \sum_{l=0}^{N_x} F^k(x_l) h_l(x) \quad \text{and} \quad G^k(y) = \sum_{l=0}^{N_y} G^k(y_l) f_l(y) \tag{12}$$

$h_l(x)$ (respectively $f_l(y)$) is the Lagrange interpolation polynomial of degree N_x (resp. N_y) associated with the collocation point x_l (resp. y_l). This is given by:

$$h_l(x) = \prod_{j=0; j\neq l}^{N_x} \frac{x - x_j}{x_l - x_j} = \frac{(-1)^{l+1}(1-x^2)T\prime_{N_x}(x)}{\overline{c_l} N_x^2 (x - x_l)} \tag{13}$$

where $T_{N_x}(x) = \cos(N_x \arccos(x))$ is the Tchebychev polynomial of degree N_x and $\overline{c_l} = \frac{1}{2}$ if $l = 0$ or $l = N_x$, and $\overline{c_l} = 1$ otherwise. $f_l(y)$ is given by the same expression by replacing x by y.

The differentiation of $U_m(x,y)$ in x-direction at any collocation point (x_i, y_j) can be written

$$\frac{\partial U_m}{\partial x}(x_i, y_j) = \sum_{k=1}^{m} \alpha^k \left(\sum_{l=0}^{N_x} \mathcal{D}_{il}^{N_x} F^k(x_l) \right) G^k(y_j) \tag{14}$$

The components $\mathcal{D}_{ij}^{N_x}$ of the $\mathcal{D}^{N_x}$ matrix correspond to $h_j\prime(x_i)$ and are given by:

[1] This formulation of the PGD is called "progressive PGD with projection". Other formulations exist (for more details, see [37]).

Table 2
PGD algorithm (ϵ is a parameter fixed by the user).

1. **for** $m = 1,\ m_{max}$ **do**
2. Perform step 1. to 7. of fixed point algorithm (see Table 1)
3. Let $\mathbb{F}^m = \{\mathbb{F}^{m-1}, F^m\}$ and $\mathbb{G}^m = \{\mathbb{G}^{m-1}, G^m\}$
4. Computation of $\Lambda_m = \{\alpha^i\}_{i=1}^m = W(\mathbb{F}^m, \mathbb{G}^m)$
5. Let $U_m = \sum_{i=1}^m \alpha^i F^i(x) G^i(y)$ and check convergence for U ($||Res^m||_{L^2(\Omega)} \leq \epsilon$)
6. **end for**

$$\begin{aligned}
\mathcal{D}_{ij}^{N_x} &= -\frac{\overline{c_i}}{2\overline{c_j}} \frac{(-1)^{i+j}}{\sin(\frac{\pi(i+j)}{2N_x}) \sin(\frac{\pi(i-j)}{2N_x})}, \quad 0 \leqslant i,\ j \leqslant N_x,\ i \neq j \\
\mathcal{D}_{ii}^{N_x} &= -\frac{\cos\frac{\pi j}{N_x}}{2\sin^2\frac{\pi i}{N_x}}, \quad 1 \leqslant i \leqslant N_x - 1 \\
\mathcal{D}_{00}^{N_x} &= -\mathcal{D}_{N_x,N_x}^{N_x} = \frac{2N_x^2 + 1}{6}
\end{aligned} \tag{15}$$

The second derivative of U_m is performed by

$$\frac{\partial^2 U_m}{\partial x^2}(x_i, y_j) = \sum_{k=1}^{m} \alpha^k \left(\sum_{l=0}^{N_x} \mathcal{D}_{il}^{N_x,2} F^k(x_l) \right) G^k(y_j) \tag{16}$$

where $\mathcal{D}^{N_x,2} = (\mathcal{D}^{N_x})^2$.

The differentiation of $U_m(x, y)$ in y-direction is obtained in the same way.

3. Application: 2D Poisson problem

3.1. The PGD problem

The following section provides an illustration of PGD associated with a spectral discretization on the 2D Poisson problem. We looked for $U(x, y) \in \Omega$ such that,

$$\begin{cases} \nabla^2 U(x,y) = g_1(x,y) & \text{in } \Omega =]0,1[\times]0,1[\\ U(x,y) = g_2(x,y) & \text{on } \partial\Omega \end{cases} \tag{17}$$

where $\partial\Omega$ is the boundary of the domain Ω.

After discretization by a spectral formulation, Eq. (17) could be written for $i = 0, \ldots, N_x$ and $j = 0, \ldots, N_y$:

$$\sum_{l=1}^{N_x-1} \mathcal{D}_{il}^{N_x,2} U(x_l, y_j) + \sum_{l=1}^{N_y-1} \mathcal{D}_{il}^{N_y,2} U(x_i, y_l) = SM_{ij} \tag{18}$$

where

$$SM_{ij} = g_1(x_i, y_j) - \mathcal{D}_{i0}^{N_x,2} g_2(x_0, y_j) - \mathcal{D}_{iN_x}^{N_x,2} g_2(x_{N_x}, y_j) - \mathcal{D}_{j0}^{N_y,2} g_2(x_i, y_0) - \mathcal{D}_{jN_y}^{N_y,2} g_2(x_i, y_{N_y}) \tag{19}$$

Using the PGD method, the solution was sought in the separated form described in Eq. (3) at the collocation points (x_i, y_j).

In this problem, the three steps of the PGD method described in Section (2.1) are written:

1. *Enrichment step*: The new terms $F^m(x_i)$ and $G^m(y_j)$ (for $1 \leqslant i \leqslant N_x - 1$ and $1 \leqslant j \leqslant N_y - 1$) are the solutions to the non linear problem defined by:

$$\begin{aligned}
\langle \mathcal{A}_{ij}(F^m, G^m), F^m \rangle_{L^2(X)} &= \langle SM_{ij}^m, F^m \rangle_{L^2(X)} \\
\langle \mathcal{A}_{ij}(F^m, G^m), G^m \rangle_{L^2(Y)} &= \langle SM_{ij}^m, G^m \rangle_{L^2(Y)}
\end{aligned} \tag{20}$$

where

$$\mathcal{A}_{ij}(F^m, G^m) = \left(\sum_{l=1}^{N_x-1} \mathcal{D}_{il}^{N_x,2} F^m(x_l) \right) G^m(y_j) + F^m(x_i) \left(\sum_{l=1}^{N_y-1} \mathcal{D}_{jl}^{N_y,2} G^m(y_l) \right) \tag{21}$$

This system was solved using a fixed point method.

2. *Projection step*: The m first functions F^k and G^k, and the m weighting factors α^k are given by solving the following linear system:

$$\mathbb{H}\boldsymbol{\alpha} = \mathbf{J} \quad \text{with} \quad {}^t\boldsymbol{\alpha} = \{\alpha_1, \ldots, \alpha_m\} \tag{22}$$

The components of $\mathbb{H}$ and $\mathbf{J}$ are defined (for $1 \leqslant k,\ q \leqslant m$ and for $i = 0, \ldots, N_x$ and $j = 0, \ldots, N_y$) by:

$$\mathbb{H}_{kq} = \sum_{i=1}^{N_x-1}\sum_{j=1}^{N_y-1} \mathcal{A}_{ij}(F^k, G^k)F^q(x_i)G^q(y_j) \tag{23}$$

and

$$J_q = \sum_{i=1}^{N_x-1}\sum_{j=1}^{N_y-1} SM_{ij}F^q(x_i)G^q(y_j) \tag{24}$$

3. *Checking the convergence*: If the L^2 norm of the residual defined (for $i = 0, \ldots, N_x$ and $j = 0, \ldots, N_y$) by

$$\sum_{k=1}^{m} \alpha^k \mathcal{A}_{ij}(F^k, G^k) - SM_{ij} \tag{25}$$

is lower than the coefficient ϵ set by the user, the algorithm is converged. Otherwise, one more iteration at least is needed, and the enrichment and projection steps are repeated taking $m = m + 1$ until convergence is achieved.

3.2. Numerical example

Here, we will consider two different test cases with a given exact solution.

Problem 1. Find $U(x, y) \in \Omega =]0, 1[\times]0, 1[$ solution of the problem (17) such that

$$g_1(x, y) = -8\pi^2 \cos(2\pi x)\sin(2\pi y) \quad \text{and} \quad \begin{cases} g_2(x, 0) = g_2(x, 1) = 0 \\ g_2(0, y) = g_2(1, y) = sin(2\pi y) \end{cases}$$

The analytical solution is then given by: $U_{ana}(x, y) = \cos(2\pi x)\sin(2\pi y)$.

Problem 2. Find $U(x, y) \in \Omega =]0, 1[\times]0, 1[$ solution of the problem (17) such that

$$g_1(x, y)(x, y) = 12(x^2y^4 + x^4y^2) - 8\pi^2 \cos(2\pi x)\sin(2\pi y) \quad \text{and} \quad \begin{cases} g_2(x, 0) = 0 & g_2(0, y) = sin(2\pi y) \\ g_2(x, 1) = x^4 & g_2(1, y) = y^4 + sin(2\pi y) \end{cases}$$

The analytical solution is then given by: $U_{ana}(x, y) = x^4y^4 + \cos(2\pi x)\sin(2\pi y)$.

These problems were solved with the PGD algorithm detailed previously as well as with the full grid spectral standard solver in order to compare accuracy and performance. The number of nodes in each direction is denoted by N. The convergence coefficient of PGD algorithm ϵ was taken to be equal to 10^{-13} and the convergence parameter for the fixed point algorithm η iwas set at 10^{-5} with a maximum iteration value of $k_{max} = 50$. The error between the analytical solution and the computed solution obtained by PGD or by the full grid solver, plotted on Fig. 1, was defined by

$$E = \frac{||U - U_{ana}||_{L^2(\Omega)}}{||U_{ana}||_{L^2(\Omega)}} \tag{26}$$

We observed that the PGD solver was as accurate as the full grid solver and that the machine error was reached for $N = 20$. In addition, the error had an exponential convergence depending on the number of nodes, which is typical in spectral discretization.

For Problem 1, PGD required only one function ($m = 1$) to converge, while sixty functions ($m = 60$) were required for Problem 2. In both cases, PGD was much faster than the full grid solver. For Problem 1, with a mesh size of 64×64, PGD was around thousand times faster than the full grid solver. This is because the error of the full grid solver is reached with only one PGD function product. For Problem 2, although the PGD method required sixty function products, the gain in computation time was also significant. Indeed, for a mesh size of 64×64, PGD was around one hundred times faster than the full grid solver. For PGD, the cost of preparing the decomposition and computing the bases (steps 1–3) is included in the comparison of the computational costs.

In order to study the influence of the various PGD parameters (ϵ, η, k_{max}) on the solution obtained, the following error was introduced:

$$E_m = \frac{||U_m - U_F||_{L^2(\Omega)}}{||U_F||_{L^2(\Omega)}} \tag{27}$$

where U_F denotes the reference solution computed with the full grid solver, and U_m is the PGD solution obtained with m products of functions (see Eq. (3)). Only Problem 2 was considered.[2]

[2] This study was not possible for Problem 1 because only one couple of functions provides the solution.

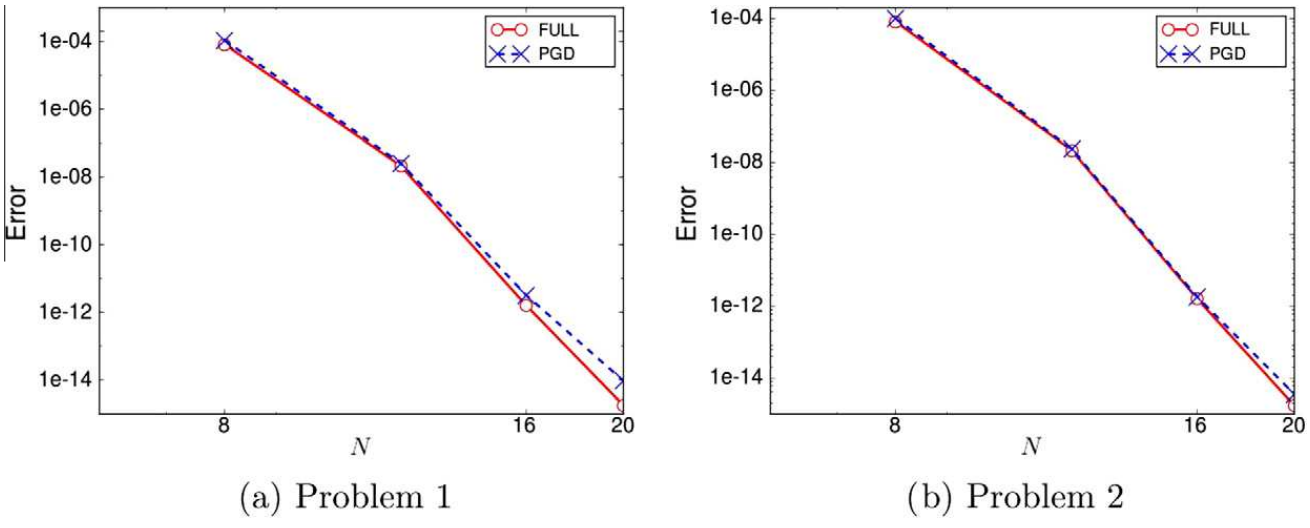


Fig. 1. Error between the computed and the exact solution depending on the number of nodes N in each direction.

For each enrichment m and for various values of k_{max}, the number of iterations required for the convergence of the fixed point algorithm is illustrated in Fig. (2a). It should be remembered that the parameter k_{max} corresponds to the maximum number of iterations set by the user in the fixed point algorithm. This figure shows that for a $k_{max} = 2, 5, 10$, the fixed point algorithm never converged to the prescribed tolerance $\eta = 10^{-15}$. To obtain this level of accuracy, it is generally necessary to perform at least twenty iterations (see results for $k_{max} = 50$). Fig. (2b) shows the change in error E_m for each PGD enrichment m. To obtain an error lower than 10^{-13}, the PGD method required around seventy functions for $k_{max} = 5, 10, 50$, compared with one hundred and ten for $k_{max} = 2$. Thus, even when $k_{max} = 50$ and the fixed point algorithm converges, the number of PGD functions required to obtain a given accuracy is very close to that required when $k_{max} = 10$. To conclude, even when $k_{max} = 10$ and the fixed point algorithm does not converge, this is still sufficient to obtain an accurate solution quickly.

The influence of parameter η, which corresponds to the accuracy of the fixed point algorithm, on PGD convergences, is illustrated in Fig. (3a). It should be noted that for values smaller than 10^{-4}, parameter η does not affect PGD convergence. Finally, Fig. (3b) shows that when the number of discretization nodes increases, the number of functions needed for PGD convergence increases too.

4. Navier–Stokes problem

4.1. $\mathbb{P}_{\mathbf{N}}$—$\mathbb{P}_{\mathbf{N-2}}$ method associated with a fractional step scheme

4.1.1. A third-order projection scheme

We considered the dimensionless unsteady Navier–Stokes equations for Newtonian incompressible fluids in a primal velocity–pressure formulation. For any open domain $\Omega \in \mathbb{R}^2$ and a positive T, we looked for the velocity field $\mathbf{u}$ and the pressure field p solutions of

$$\begin{cases} \frac{\partial \mathbf{u}}{\partial t} - \frac{1}{Re}\Delta\mathbf{u} + \mathbf{u}.\nabla\mathbf{u} + \nabla p = \mathbf{f} \text{ in } \Omega\times]0,T] \\ \nabla.\mathbf{u} = 0 \\ \mathbf{u}(t=0) = \mathbf{u}_0 \\ \mathbf{u} = \mathbf{g} \text{ on } \partial\Omega \end{cases} \tag{28}$$

where $\mathbf{f}, \mathbf{u}_0$ and $\mathbf{g}$ are given vectors and Re is the Reynolds number.

The fractional step schemes involve computing the velocity and the pressure fields separately, through the computation of an intermediate velocity, which is then projected onto the subspace of divergence-free functions (see [42–44]). In this work, an Adams–Bashforth/third-order backward Euler projection scheme introduced by Botella in [41] was used. At the $n+1$ time step, we were looking for $\mathbf{u}^{n+1}$ and p^{n+1}. The steps which compose this algorithm are detailed in the following:

(1) The estimated velocity $\tilde{\mathbf{u}}^{n+1}$ is the solution to the correction step defined by:

$$\begin{cases} \sigma\tilde{\mathbf{u}}^{n+1} - \frac{1}{Re}\nabla^2\tilde{\mathbf{u}}^{n+1} = \tilde{\mathbf{f}}^{n+1} \\ \tilde{\mathbf{u}}^{n+1} = \mathbf{g}^{n+1} \text{ on } \partial\Omega \end{cases} \tag{29}$$

where $\sigma = \frac{11}{6\Delta t}$ and Δt is the time step. Vector $\tilde{\mathbf{f}}^{n+1}$ is given by:

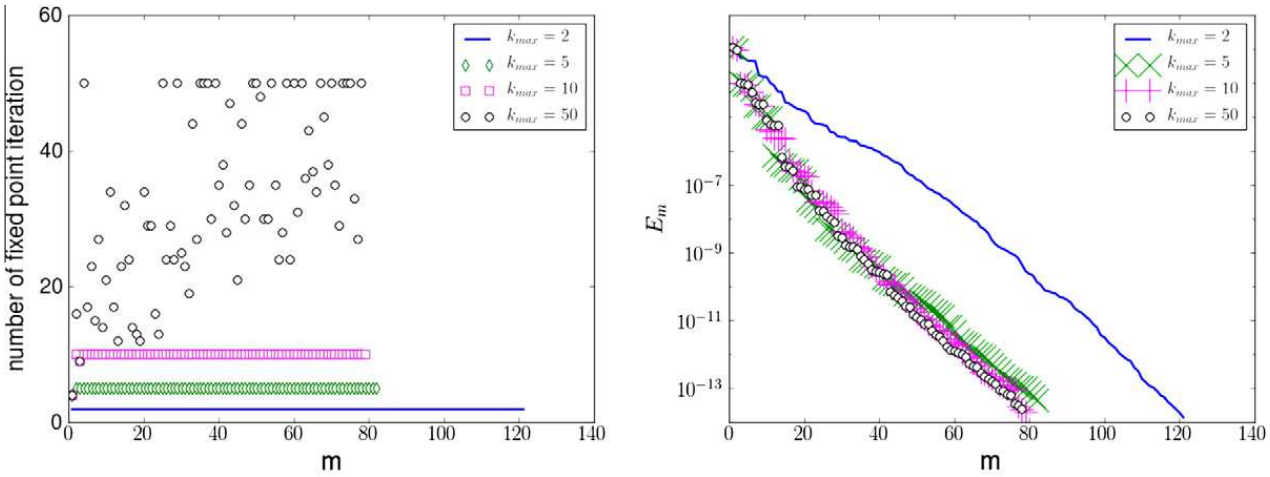


Fig. 2. Convergence of PGD for Problem 2 for $N = 26$ and $\eta = 10^{-15}$.

$$\tilde{\mathbf{f}}^{n+1} = \mathbf{f}^{n+1} + \frac{1}{6\Delta t}(18\mathbf{u}^n - 9\mathbf{u}^{n-1} + 2\mathbf{u}^{n-2}) - \nabla(2p^n - p^{n-1}) - 3[(\mathbf{u}^n.\nabla)\mathbf{u}^n] - 3[(\mathbf{u}^{n-1}.\nabla)\mathbf{u}^{n-1}] + [(\mathbf{u}^{n-2}.\nabla)\mathbf{u}^{n-2}] \tag{30}$$

(2) Pressure p^{n+1} is the solution to the projection step:

$$\begin{cases} \sigma\mathbf{u}^{n+1} + \nabla p^{n+1} = \hat{\mathbf{f}}^{n+1} \\ \nabla.\mathbf{u}^{n+1} = 0 \\ \mathbf{u}^{n+1}.\mathbf{n} = \mathbf{g}^{n+1}.\mathbf{n} \text{ on } \partial\Omega \end{cases} \tag{31}$$

where $\hat{\mathbf{f}}^{n+1} = \sigma\tilde{\mathbf{u}}^{n+1} + \nabla(2p^n - p^{n-1})$

(3) Velocity $\mathbf{u}^{n+1}$ is updated with:

$$\mathbf{u}^{n+1} = \tilde{\mathbf{u}}^{n+1} - \frac{t}{1}\sigma\nabla(p^{n+1} - 2p^n + p^{n-1}) \tag{32}$$

4.1.2. $\mathbb{P}_{\mathbf{N}}$—$\mathbb{P}_{\mathbf{N-2}}$ *method*

In this work, the velocity and the pressure were discretized over one collocation grid only. The discrete velocity and pressure spaces used in the resolution of the Darcy problem must fulfil the inf-sup condition, otherwise the pressure is polluted by spurious modes (four modes for a 2D problem) [39]. In order to avoid these spurious modes, we used a $\mathbb{P}_{\mathbf{N}}$—$\mathbb{P}_{\mathbf{N-2}}$ scheme. This scheme approximates the velocity field by local polynomials of degree N in each space variable, and approximates the pressure by polynomials of degree $N-2$.

In this collocation method, the values of the velocity involved in the solution are the values at the $(N_x+1)\times(N_y+1)$ nodes of Ω, while there are only values of the pressure at the $(N_x-1)\times(N_y-1)$ inner nodes. A derivative operator of the velocity $\mathcal{D}^{N_x}$ (respectively $\mathcal{D}^{N_y}$) was defined in Section 2.2. The entries for the pressure derivative matrix in the x-direction, called $\widetilde{\mathcal{D}}^{N_x}$, are given by:

$$\begin{aligned} \widetilde{\mathcal{D}}^{N_x}_{ij} &= -\frac{1}{2}\frac{\sin^2\frac{\pi j}{N_x}}{\sin^2\frac{\pi i}{N_x}}\frac{(-1)^{i+j}}{\sin(\frac{\pi(i+j)}{2N_x})\sin(\frac{\pi(i-j)}{2N_x})} \quad 1 \leqslant i,\ j \leqslant N_x - 1,\ i \neq j \\ \widetilde{\mathcal{D}}^{N_x}_{ii} &= \frac{3\cos\frac{\pi j}{N_x}}{2\sin^2\frac{\pi i}{N_x}} \qquad\qquad\qquad\qquad 1 \leqslant i = j \leqslant N_x - 1 \end{aligned} \tag{33}$$

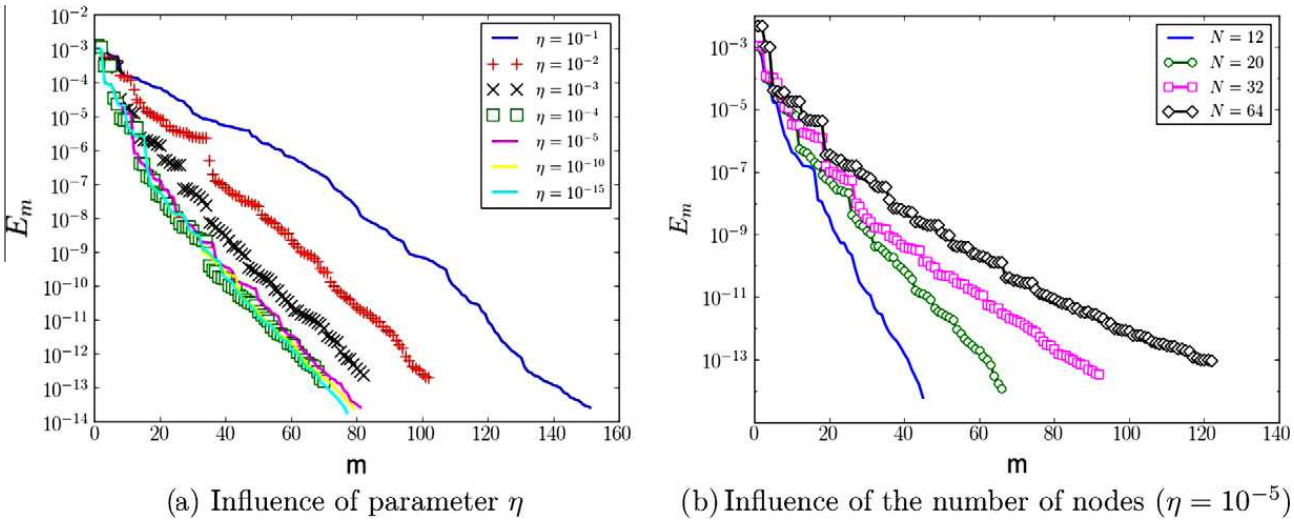
The entries for the pressure derivative matrix in the y-direction, called $\widetilde{\mathcal{D}}^{N_y}$, are defined in the same way.

The prediction step (29) constitutes a Helmholtz problem for the estimated velocity $\tilde{\mathbf{u}}^{n+1}$ with Dirichlet boundary conditions which could be solved using standard methods. The projection step corresponds to a Darcy problem which is solved using an Uzawa algorithm.

4.1.3. Uzawa algorithm

In the following u denotes the velocity component in the x-direction and v the velocity component in the y-direction. Similarly, $\hat{f}_x^{n+1}$ is the component of $\hat{\mathbf{f}}^{n+1}$ in the x-direction and $\hat{f}_y^{n+1}$ in the y-direction. The Darcy problem (31), written on the inner collocation nodes, is given by the following matrix formulation:

$$\sigma U + \widetilde{\mathcal{D}}^{N_x} P = \widehat{F}_x \tag{34}$$



(a) Influence of parameter η (b) Influence of the number of nodes ($\eta = 10^{-5}$)

Fig. 3. Convergence of PGD for Problem 2 for $k_{max} = 10$.

$$\sigma V + P^t \widetilde{\mathcal{D}}^{N_y} = \widehat{F}_y \tag{35}$$

$$\mathcal{D}^{N_x} U + V^t \mathcal{D}^{N_y} = S \tag{36}$$

The components of the previous matrices were performed for $1 \leqslant i \leqslant N_x - 1$ and $1 \leqslant j \leqslant N_y - 1$ by:

$$U_{ij} = \left[u^{n+1}(x_i, y_j)\right], \quad V_{ij} = \left[v^{n+1}(x_i, y_j)\right], \quad P_{ij} = \left[p^{n+1}(x_i, y_j)\right],$$
$$\widehat{F}_{x_{ij}} = \left[\hat{f}_x^{n+1}(x_i, y_j)\right], \quad \widehat{F}_{y_{ij}} = \left[\hat{f}_y^{n+1}(x_i, y_j)\right]$$

and

$$S_{ij} = -\mathcal{D}_{i0}^{N_x} u(x_0, y_j) - \mathcal{D}_{iN_x}^{N_x} u(x_{N_x}, y_j) - \mathcal{D}_{j0}^{N_y} v(x_i, y_0) - \mathcal{D}_{jN_y}^{N_y} v(x_i, y_{N_y}). \tag{37}$$

Substituting Eqs. (34) and (35) into (36), the following matrix equation for pressure is obtained:

$$\left(\mathcal{D}^{N_x} \widetilde{\mathcal{D}}^{N_x}\right) P + P^t \left(\mathcal{D}^{N_y} \widetilde{\mathcal{D}}^{N_y}\right) = G \tag{38}$$

with

$$G = \mathcal{D}^{N_x} \widehat{F}_x + \widehat{F}_y^t \mathcal{D}^{N_y} - \sigma S$$

The operators $\mathcal{D}^{N_x} \widetilde{\mathcal{D}}^{N_x}$ and $\mathcal{D}^{N_y} \widetilde{\mathcal{D}}^{N_y}$ are called Uzawa operators. The pressure is then obtained by solving Eq. (38).

4.1.4. PGD for Navier–Stokes equations

The problem is solved using the PGD algorithm. Assuming we are at $n+1$ time step, $p^n, p^{n-1}, \mathbf{u}^n, \mathbf{u}^{n-1}$ and $\mathbf{u}^{n-2}$ are known and p^{n+1} and $\mathbf{u}^{n+1}$ are then calculated. For that purpose, the following steps were carried out:

(1) The x-component $\tilde{u}^{n+1}$ of the estimated velocity $\tilde{\mathbf{u}}^{n+1}$ is sought in the form

$$\tilde{u}^{n+1}(x, y) = \sum_{k=1}^{m_{\tilde{u}}} \alpha_{\tilde{u}}^k F_{\tilde{u}}^k(x) G_{\tilde{u}}^k(y) \tag{39}$$

The PGD strategy is similar what was presented in Section 3.1 with

$$\mathcal{A}_{ij}(F_{\tilde{u}}^k, G_{\tilde{u}}^k) = \sigma F_{\tilde{u}}^k(x_i) G_{\tilde{u}}^k(y_j) - \frac{1}{Re}\left[\left(\sum_{l=1}^{N_x-1} \mathcal{D}_{il}^{N_x,2} F_{\tilde{u}}^k(x_l)\right) G_{\tilde{u}}^k(y_j) + F_{\tilde{u}}^k(x_i)\left(\sum_{l=1}^{N_y-1} \mathcal{D}_{jl}^{N_y,2} G_{\tilde{u}}^k(y_l)\right)\right] \tag{40}$$

and

$$SM_{ij} = \tilde{f}_x^{n+1}(x_i, y_j) - \mathcal{D}_{i0}^{N_x,2} g_x^{n+1}(x_0, y_j) - \mathcal{D}_{iN_x}^{N_x,2} g_x^{n+1}(x_{N_x}, y_j) - \mathcal{D}_{j0}^{N_y,2} g_x^{n+1}(x_i, y_0) - \mathcal{D}_{jN_y}^{N_y,2} g_x^{n+1}(x_i, y_{N_y}) \tag{41}$$

where g_x is the x-component of the vector $\mathbf{g}$ defined in problem (28) and $\tilde{f}_x^{n+1}$ is the x-component of the source term vector $\tilde{\mathbf{f}}^{n+1}$.

$\tilde{v}^{n+1}$, the y-component of the estimated velocity $\tilde{\mathbf{u}}^{n+1}$, is computed in the same way such that:

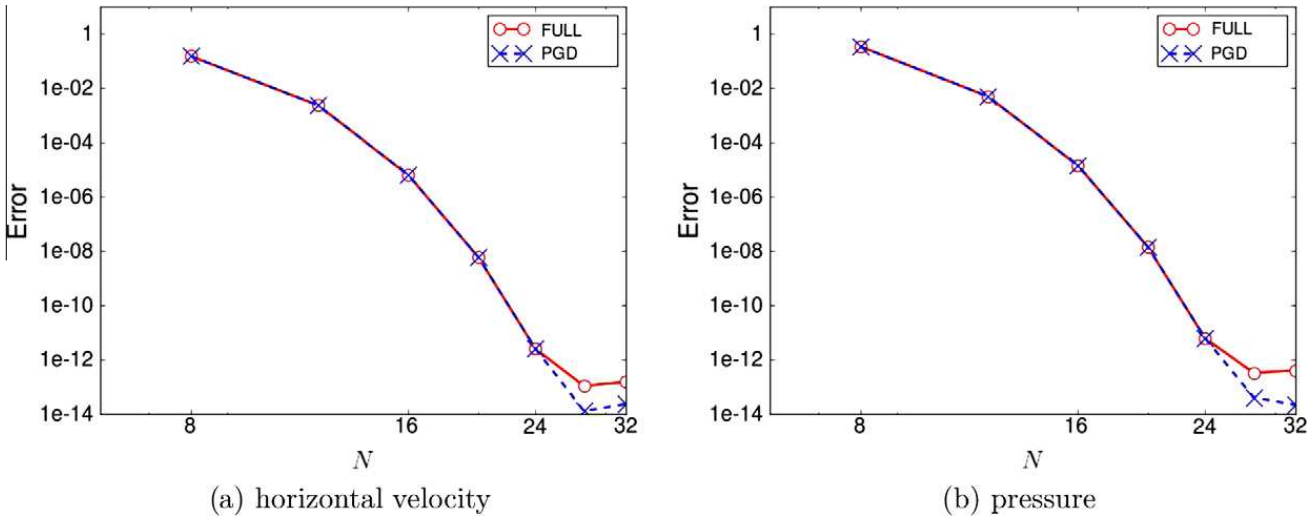


Fig. 4. Relative error for the horizontal velocity (left) and for the pressure (right) with the number of nodes N in each direction.

$$\tilde{v}^{n+1}(x,y) = \sum_{k=1}^{m_{\tilde{v}}} \alpha_{\tilde{v}}^k F_{\tilde{v}}^k(x) G_{\tilde{v}}^k(y) \tag{42}$$

(2) The pressure p^{n+1} was calculated in the form

$$p^{n+1} = \sum_{k=1}^{m_p} \alpha_p^k F_p^k(x) G_p^k(y) \tag{43}$$

In this case, Eq. (38) was solved in a similar manner to that shown in Section 3.1 with the pressure operators defined in section 4.1.3.

(3) Velocity $\mathbf{u}^{n+1}$ was updated using Eq. (32).

4.2. Applications

We decided to solve Navier–Stokes equations using PGD associated with a $\mathbb{P}_{\mathbf{N}}-\mathbb{P}_{\mathbf{N-2}}$ spectral discretization. In order to validate the approach, we first considered a Darcy problem, which corresponds to the projection step of the fractional step scheme. Then, two different Taylor Green problems were examined to assess the accuracy of the method when the time step and the grid size vary. Finally, the method was applied to solve the flow in a 2D lid-driven cavity.

4.2.1. Darcy problem

In this section the following problem is solved:

$$\begin{cases} \mathbf{u} + \nabla p = \mathbf{f} \text{ in } \Omega \\ \nabla.\mathbf{u} = 0 \text{ in } \Omega \\ \mathbf{u}.\mathbf{n} = \mathbf{g}.\mathbf{n} \text{ on } \partial\Omega \end{cases} \tag{44}$$

where $\mathbf{u} = (u, v)$ is the velocity and p is the pressure. This problem corresponds to the projection step (31). We chose the case studied by Botella [40] which admits the following analytical solutions:

$$u_{ana}(x,y) = -4\pi \sin(4\pi x)\cos(4\pi y),$$
$$v_{ana}(x,y) = 4\pi \cos(4\pi x)\sin(4\pi y),$$
$$p_{ana}(x,y) = \cos(4\pi x)\cos(4\pi y)$$

By carrying these solutions into problem (44), the source terms and boundary conditions are well defined. An arbitrary pressure constant was chosen in order to make the calculated pressure at the central point of the mesh equal to the exact one.

The problem was solved using PGD and with a full grid spectral standard solver. The convergence coefficient of the PGD algorithm was taken to be equal to $\epsilon_p = 10^{-8}$. Simulations were performed for a mesh characterized by N collocation points in each direction and ranging from 8 to 64. The relative error of the analytical solution, defined by Eq. (26), was plotted as a function of the number of nodes in each direction N (see Fig. 4).

This figure shows that the PGD solver is as accurate as the full grid solver. Moreover, from 32 nodes in each direction, the error is no longer significant. We observed an exponential convergence with the number of nodes. These results are similar

Table 3
Error of the stationary solution of the Taylor–Green problem for $Re = 100$.

N	E_u		E_v		E_p	
	Standard	PGD	Standard	PGD	Standard	PGD
8	$9,60.10^{-7}$	$9,60.10^{-7}$	$9,60.10^{-7}$	$9,60.10^{-7}$	$8,63.10^{-4}$	$8,63.10^{-4}$
12	$2,02.10^{-11}$	$2,02.10^{-11}$	$2,02.10^{-11}$	$2,02.10^{-11}$	$4,37.10^{-7}$	$4,37.10^{-7}$
16	$1,33.10^{-15}$	$1,73.10^{-15}$	$1,33.10^{-15}$	$1,73.10^{-15}$	$6,41.10^{-11}$	$6,42.10^{-11}$
32	$7,29.10^{-19}$	$6,49.10^{-19}$	$7,29.10^{-19}$	$6,49.10^{-19}$	$1,53.10^{-15}$	$1,41.10^{-15}$

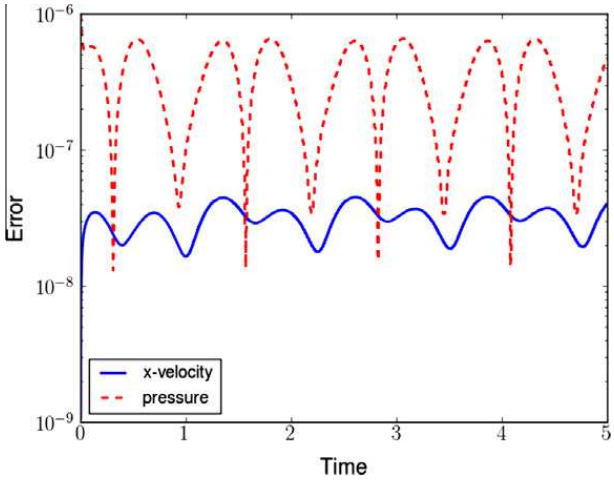


Fig. 5. Change with time of the error on the velocity and the pressure with $\Delta t = 10^{-3}$ and $N = 16$ (PGD solver).

to those obtained by Botella in [40]. The error associated with the vertical velocity has not been drawn because it is similar to that obtained with the horizontal velocity.

The CPU time required to solve this problem by PGD was much lower than the CPU time necessary with a full grid solver. In this case, with $N = 64$, the PGD solver was 100 times faster than the standard solver. This result is due to the fact that PGD only requires one couple of functions to reconstruct the solution properly.

4.2.2. Taylor–Green problem

The Taylor Green problem corresponds to the Navier–Stokes problem where an analytical solution is provided. We considered the case studied by Botella in [41]. This consists in solving Navier–Stokes equations in a domain $\Omega =]-1,1[\times]-1,1[$ and on the time interval $]0,T]$, where the solutions are given by:

$$u_{ana}(x,y,t) = \cos(\gamma t)\sin\left(\frac{\pi x}{2}\right)\cos\left(\frac{\pi y}{2}\right),$$
$$v_{ana}(x,y,t) = -\cos(\gamma t)\cos\left(\frac{\pi x}{2}\right)\sin\left(\frac{\pi y}{2}\right),$$
$$p_{ana}(x,y,t) = \frac{1}{4}\cos^2(\gamma t)[\cos(\pi x)+\cos(\pi y)] + 10(x+y)\cos(\gamma t)$$

The boundary conditions and the source term $\mathbf{f}$ were defined in order to match the analytical solution. Reynolds number was set at $Re = 100$. To initialize the time integration scheme, the fields $\mathbf{u}$ and p at $-2\Delta t, -\Delta t$ and 0 were taken to be equal to the exact solution. The coefficients needed to check PGD convergence were set at $\epsilon_{\tilde{u}} = 10^{-13}$ for intermediate velocities and $\epsilon_p = 10^{-8}$ for pressure. In both cases, the fixed point criterion was set at $\eta = 10^{-5}$.

To check the spatial accuracy of PGD, we first studied the stationary Navier–Stokes equation by taking $\gamma = 0$. The discrete errors $E_\Phi = ||\Phi - \Phi_{ana}||$ on the collocation inner points for fields $\Phi = u, v$ or p are displayed in Table 3. These results show that PGD is as accurate as the full grid standard solver. In this case too, PGD requires only one couple of functions to reconstruct the solution exactly.

The study of temporal accuracy was performed on the unsteady, time periodic analytical solutions with $\gamma = 5$. The time step was set to $\Delta t = 10^{-3}$ and a grid with 16 collocation points in each direction was considered. Fig. 5 shows the changeover time of errors E_u and E_p obtained by PGD. We observed no amplification of the errors with time, which suggests the numerical solutions are stable. It is noteworthy that these results are close to those obtained by Botella and confirm the ability of PGD to treat Navier–Stokes equations with a spectral discretization. Errors obtained with the full grid solver have not been plotted because they are exactly the same as those obtained by PGD.

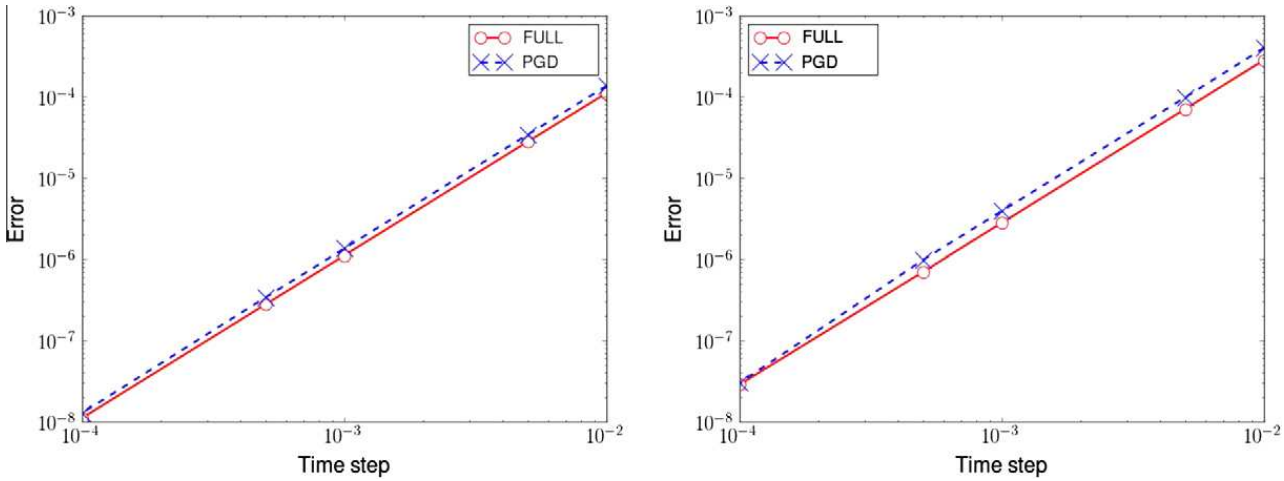


Fig. 6. Velocity error (right) and pressure error (left) depending on the time step with $N = 16$ at $T = 1$ seconds.

Another test consists in studying the relative error with respect to the analytical solution, defined by Eq. (26), according to the time step. Fig. 6 shows that the behaviour of the PGD errors for velocity and pressure are very close to those obtained by a full grid solver. Moreover, the errors obtained by PGD have an exponential decay with the time step, which is characteristic of spectral methods.

As in Section 3.2, the influence of the various parameters of the fixed point algorithm (η, k_{max}) on the accuracy of the solution was studied. As a first step, we were interested in the influence of the $k^{\tilde{u}}_{max}$ parameter on the prediction of the intermediate velocity. Figs. (7a) and (7b) show that, in this case, the convergence of the fixed point algorithm does not affect the prediction of the velocity. Indeed, the algorithm did not converge, except for $k^{\tilde{u}}_{max} = 50$, and the error E_m was the same for each $k^{\tilde{u}}_{max}$. Note that the results obtained for the intermediate y-velocity are similar to those obtained here. Fig. (8a) shows that the parameter $\eta_{\tilde{u}}$ does not influence the prediction of the velocity. Consequently, in the following, the parameters of the fixed point algorithm required to compute the intermediate velocities was set at: $\eta_{\tilde{u}} = 10^{-5}$ and $k^{\tilde{u}}_{max} = 10$.

As a second step, we were interested in the influence of the k^{p}_{max} parameter for the prediction of the pressure. Fig. (7c) shows that independently from the choice of the k^{p}_{max} parameter, the fixed point algorithm did not converge. In this case, the number of iterations in the fixed point algorithm affects the behaviour of the error (see Fig. (7d)). We observed that the smaller the k^{p}_{max}, the higher the number of PGD functions necessary for the convergence of the PGD algorithm. To obtain an error of $E_m = 10^{-13}$, the choice of a $k^{p}_{max} = 2$ led to $m = 80$, the choice of a $k^{p}_{max} = 10$ led to $m = 32$ and the choice of a $k^{p}_{max} = 50$ led to $m = 20$. Since the purpose of this work was to save computational time, the choice of $k^{p}_{max} = 50$ is not convenient because the time spent in the fixed point algorithm is too long. The best compromise between CPU time saving in the fixed point algorithm and the desired number of PGD function products for convergence seems to correspond to a value of k^{p}_{max} equal to 10. Fig. (8a) shows that the value of η_p does not affect the accuracy of the pressure prediction. Consequently, in the section below, for the lid driven cavity, the parameters were set to: $\eta_p = 10^{-5}$ and $k^{p}_{max} = 10$.

The comparison of PGD CPU time and standard solver CPU time is illustrated in Fig. 9. It is clear that beyond a mesh size of 26×26, PGD was faster than the standard method. In fact, for a mesh size of 48×48, PGD was ten times faster than the standard method.

4.2.3. Lid-driven cavity flow problem

We will now discuss the case of a flow in a square lid-driven cavity, as illustrated in Fig. 10. The two velocity components vanish on the boundary, except on the north face where the x-velocity is equal to U_0. The source term f is taken equal to zero.

Simulations were performed for $Re = \frac{U_0 \times d}{\nu}$ equal to 100 with a time step equal to $\Delta t = 10^{-3}$ and a non regular grid with 20 nodes in each direction. For this Reynolds number, simulations converged towards stationary solutions. So the following convergence criteria were defined:

$$\frac{t}{||\Phi^k - \Phi^{k-1}||}||\Phi^k|| \leqslant \gamma_u \quad \text{with} \quad \Phi = u, v \tag{45}$$

$$\frac{t}{||\nabla p^k - \nabla p^{k-1}||}||\nabla p^k|| \leqslant \gamma_p \tag{46}$$

γ_u (resp. γ_p) was taken as being equal to 10^{-10} (resp. 10^{-5}). Finally, the simulations were performed with a full grid solver and with the PGD method. Coefficients needed to check the convergence of PGD algorithm were set at $\epsilon_{\tilde{u}} = 10^{-10}$ for intermediate velocities and $\epsilon_p = 10^{-5}$ for pressure. Concerning the fixed point parameters, based on the results obtained in the previous section, we chose $\eta = 10^{-5}$ and $k_{max} = 10$ to compute velocities and pressure.

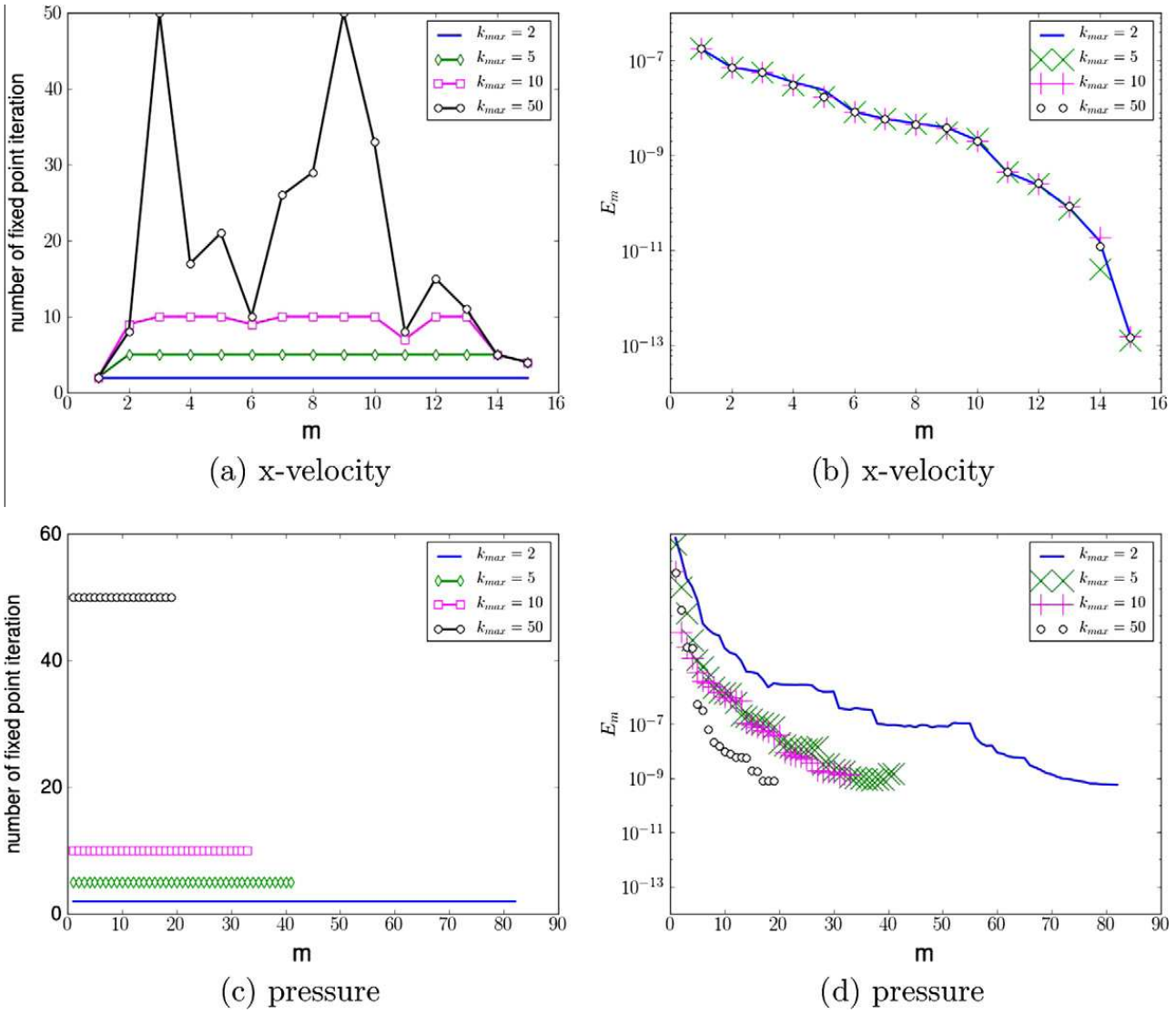


Fig. 7. Convergence of PGD for the Taylor–Green problem at $t = 1$ s with $N = 16$. (a and b) Intermediate x-velocity for $\eta_{\tilde{u}} = 10^{-15}$, (c and d) pressure with $\eta_p = 10^{-15}$.

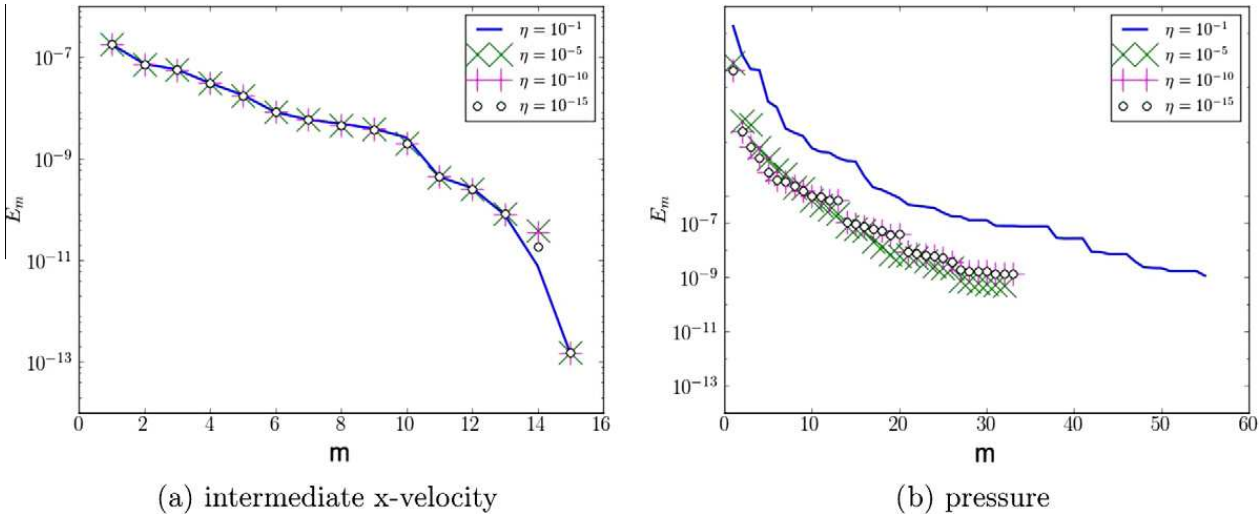


Fig. 8. Convergence of PGD solutions at $t = 1$ s for the Taylor–Green problem with $k_{max} = 10$ and $N = 16$.

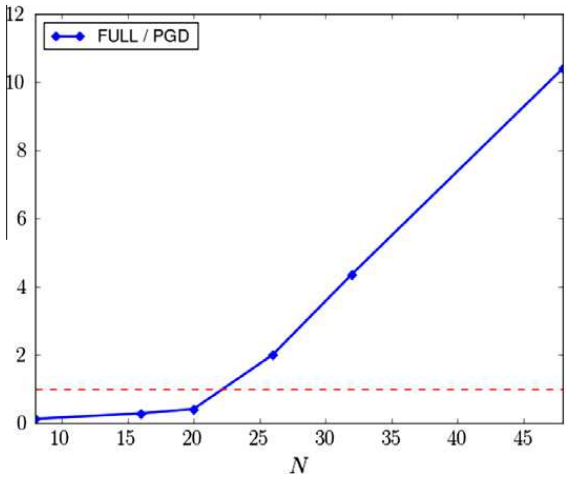


Fig. 9. Comparison between full model CPU time and PGD model CPU time.

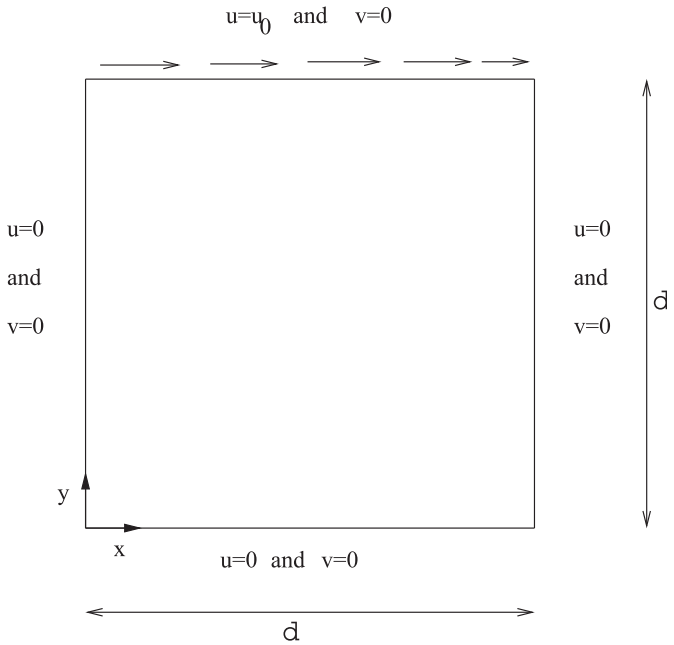


Fig. 10. Geometry of a lid-driven cavity.

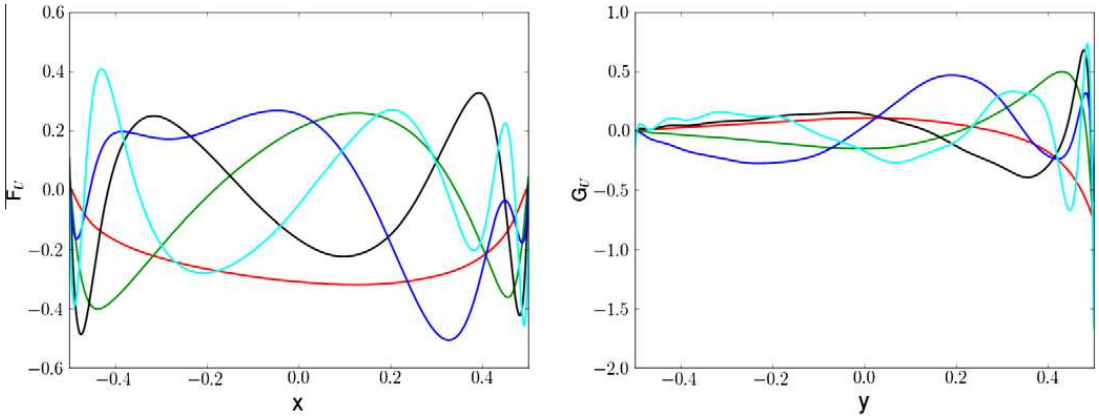


Fig. 11. First functions of the tensor product computed with PGD that represent the *x*-velocity field.

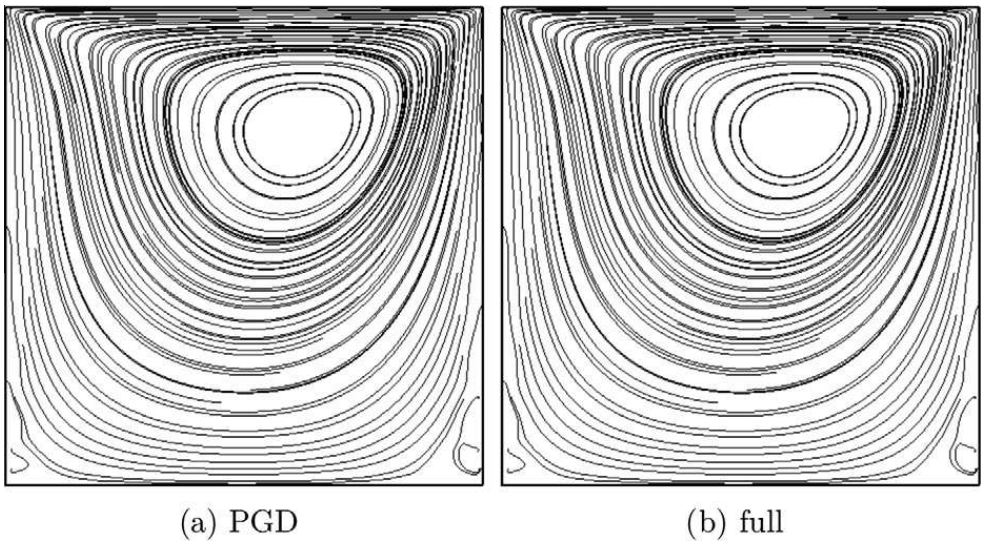


Fig. 12. Streamlines computed by PGD and with the full grid solver with $N = 20$.

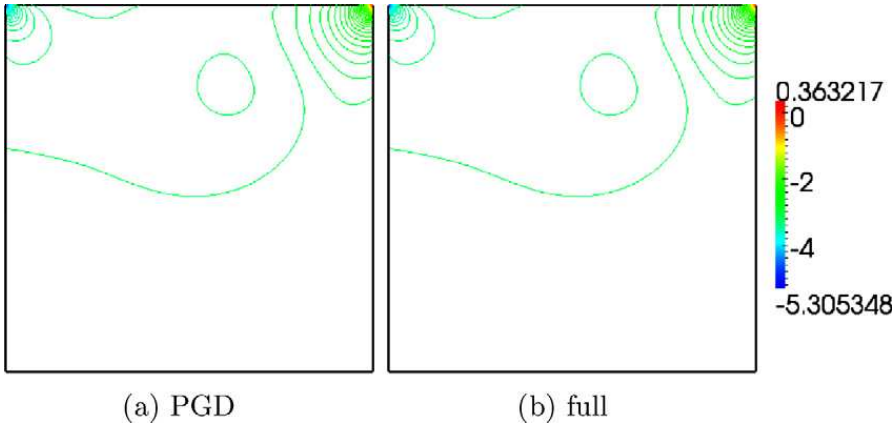


Fig. 13. Pressure fields computed by PGD and by the full grid solver with $N = 20$.

Table 4
Extreme values of velocities through the centerline of the cavity, $Re = 100$.

Source	Method and grid	u_{min}	y_{min}	v_{max}	x_{max}	v_{min}	x_{min}
Ghia et al. [45]	Finite difference, $N = 129$	−0.21090	−0.0469	0.17527	−0.2656	−0.24533	0.3047
Deng et al. [46]	Finite Volume	−0.21405	–	0.17896	–	−0.25399	–
Zhang et al. [47]	Pseudospectral, $N = 48$	−0.2140482	−0.0419	0.1795763	−0.2630	−0.2538134	0.3104
Present	Pseudospectral, $N = 20$	−0.2141353	−0.0419	0.1797115	−0.2630	−0.2541496	0.3104

The first PGD functions F_u^i and G_u^i for the x-velocity are shown in Fig. 11. The streamlines and the isocontours of pressure obtained by PGD and by the full grid solver are shown on Figs. 12 and 13. The PGD results are in agreement with those of the standard solver.

Table 4 shows the extreme velocities along the centerlines and their corresponding positions. The minimum horizontal velocity on the vertical centerline is denoted as u_{min} and its location as y_{min}. The maximum and minimum vertical velocities on the horizontal centerline are denoted by v_{max} and v_{min}, respectively, and their locations by x_{max} and x_{min}. These results show that PGD associated with a spectral discretization is able to provide a highly accurate solution.

The change in the error defined by Eq. (27) in relation to the number of PGD functions is plotted in Fig. 14. We observed that, for the velocity in the lid-driven cavity, sixty function products were necessary to reconstruct the full grid solution. For the pressure, approximately eighty products were necessary to achieve convergence.

The change in the ratio between the CPU time of the full solver and the PGD solver with the number of nodes is shown in Fig. 15. At 25 nodes, and above, the PGD method was faster than the standard solver. Moreover, with a 36×36 grid, PGD was four times faster than the standard solver.

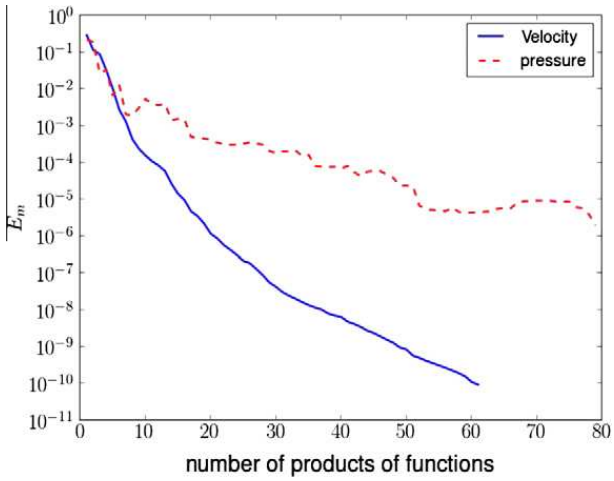


Fig. 14. Convergence of PGD for horizontal velocity and for pressure in the case of the lid driven cavity problem with $N = 20$.

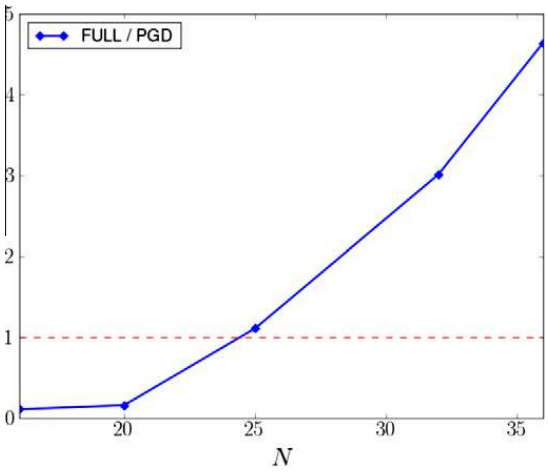


Fig. 15. Ratio between full model CPU time and PGD model CPU time.

We will now discuss the optimality of the PGD method. For this purpose, the number of PGD functions required to obtain a solution was analyzed. A SVD (Singular Value Decomposition) [48,49] was applied. If the PGD method is optimal, the number of PGD functions should be almost the same as the number of functions obtained by the SVD. To achieve this, the change in the norm of the residual of each equation solved depending on the number of PGD functions or SVD functions was considered. The residual is defined as:

$$Res_w^N = \mathcal{L}_w\left[\sum_{i=1}^{N_w} F_w^i G_w^i\right] - SM_w \tag{47}$$

where w corresponds to the different unknowns of the problem, namely the horizontal velocity and the pressure. The results obtained for vertical velocity were very close to those obtained for horizontal velocity, which is why this velocity will not be considered in the following. $\mathcal{L}_w$ (respectively SM_w) is the operator (resp. the second member) associated with the equation which was used to calculate the variable w. Functions F_w^i, G_w^i are the PGD functions or the SVD functions of the unknown w at the last time step. Below, FULL-SVD corresponds to the results obtained with a SVD on the full grid spectral standard solver, and PGD-SVD refers to the results obtained with a SVD on the PGD solution.

The norm of the residual associated with the equation satisfied by the velocity depending on the number of retained functions $N_{\vec{u}}$ is plotted on Fig. 16 (left). The residual associated with the horizontal velocity obtained by PGD, PGD-SVD and FULL-SVD gave the same curve of residuals up to a value of 10^{-5}. This proves that, for this variable, PGD gives the same optimal representation as could be provided by a SVD of the full solution. From this value, the PGD curve decreased more slowly to a norm of the residual close to zero in approximately sixty functions, while SVDs suddenly decreased to zero in twenty functions. The same results are plotted for pressure in Fig. 16 (right). Here, it is more difficult to reach a conclusion. In fact, both SVDs converged to a value of the residual norm that was less than 10^{-5} in twenty function products, while PGD needed more

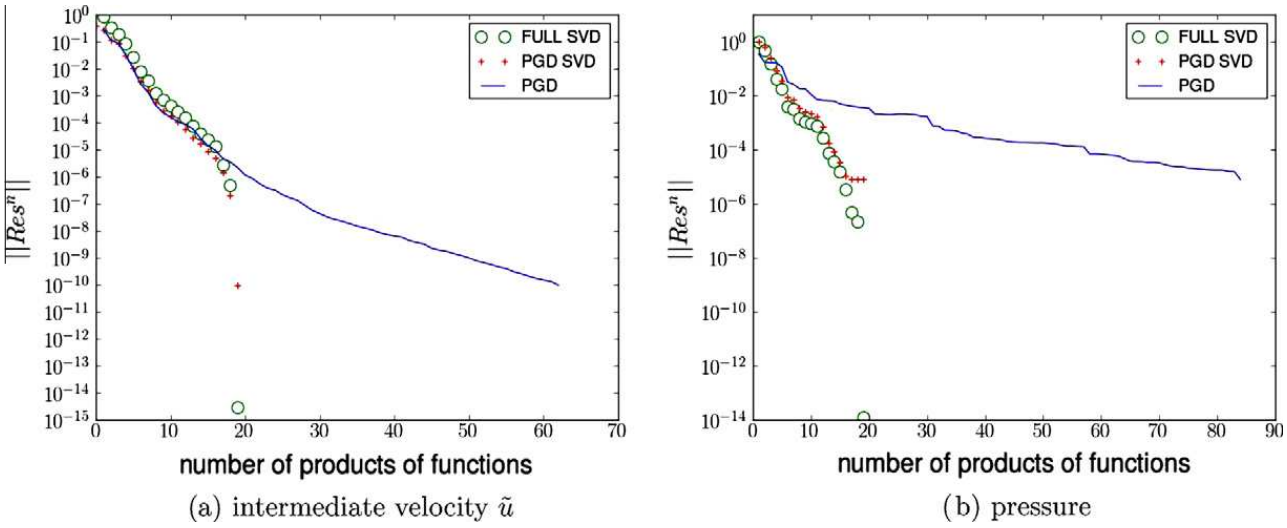


Fig. 16. Norm of the residual according to the number of functions for each PGD and FULL SVD decomposition for intermediate horizontal velocity and for the pressure at $Re = 100$ with $N = 20$.

than seventy products. It is clear that the convergence in pressure is slower than the convergence in velocity. PGD is not optimal in this case either.

5. Conclusions

In this work, the PGD method was coupled with a spectral discretization to solve various transfer equations. For all the 2D cases considered (the stationary diffusion equation, the Darcy problem and the Navier–Stokes equations), the PGD results were as accurate as those obtained with a full model. Moreover, above a given number of nodes, PGD was faster than the full solver. For the 2D stationary diffusion equation, using PGD was about 100 times faster on a 64×64 mesh size. For Navier Stokes equations, it was around five or ten times faster, depending on the problem considered. The study of the influence of the fixed point algorithm on the accuracy of the solution showed that the convergence of the fixed point algorithm does not affect the solution significantly.

Increasing the Reynolds number will be the subject of further studies. This work is also the first step toward dealing with a 3D situation. Indeed it would be interesting to include the time coordinate in the separated representation of the PGD solution or to consider 3D spatial problems. It might be hoped that the time saved will be greater than in the 2D case.

References

[1] P. Holmes, J. Lumley, G. Berkooz, Turbulence, Coherent Structures, Dynamical Systems and Symmetry, Cambridge University, Cambridge, 1996.
[2] C. Allery, S. Guerin, A. Hamdouni, A. Sakout, Experimental and numerical POD study of the Coanda effect used to reduce self-sustained tones, Mechanics Research Communications 31 (1) (2004) 105–120.
[3] C. Leblond, C. Allery, C. Inard, An optimal projection method for the reduced-order modeling of incompressible flows, Computer Methods in Applied Mechanics and Engineering 200 (33–36) (2011) 2507–2527.
[4] H. Park, W. Lee, An efficient method of solving the Navier–Stokes equations for flow controls, Phys. Rep 287 (1997) 337–384.
[5] J.A. Atwell, B.B. King, Proper orthogonal decomposition for reduced basis feedback controllers for parabolic equations, Mathematical and Computer Modelling 33 (1–3) (2001) 1–19.
[6] C. Prud'homme, D. Rovas, K. Veroy, Y. Maday, A.T. Patera, G. Turinici, Reliable real-time solution of parametrized partial differential equations: reduced-basis output bound methods, Journal of Fluids Engineering 124 (1) (2002) 70–80.
[7] M. Krasnyk, M. Mangold, A. Kienle, Reduction procedure for parametrized fluid dynamics problems based on proper orthogonal decomposition and calibration, Chemical Engineering Science 65 (23) (2010) 6238–6246.
[8] C. Allery, C. Béghein, A. Hamdouni, Applying proper orthogonal decomposition to the computation of particle dispersion in a two-dimensional ventilated cavity, Communications in Nonlinear Science and Numerical Simulation 10 (8) (2005) 907–920.
[9] C. Allery, C. Béghein, A. Hamdouni, On investigation of particle dispersion by a POD approach, International Applied Mechanics 44 (2008) 110–119.
[10] K. Urban, A.T. Patera, A new error bound for reduced basis approximation of parabolic partial differential equations, Comptes Rendus Mathematique 350 (3–4) (2012) 203–207.
[11] G. Rozza, K. Veroy, On the stability of the reduced basis method for Stokes equations in parametrized domains, Computer Methods in Applied Mechanics and Engineering 196 (7) (2007) 1244–1260.
[12] K. Veroy, A.T. Patera, Certified real-time solution of the parametrized steady incompressible Navier–Stokes equations: rigorous reduced-basis a posteriori error bounds, International Journal for Numerical Methods in Fluids 47 (8–9) (2005) 773–788.
[13] A. Quarteroni, G. Rozza, Numerical solution of parametrized Navier–Stokes equations by reduced basis methods, Numerical Methods for Partial Differential Equations 23 (4) (2007) 923–948.
[14] S. Deparis, G. Rozza, Reduced basis method for multi-parameter-dependent steady Navier–Stokes equations: applications to natural convection in a cavity, Journal of Computational Physics 228 (12) (2009) 4359–4378.
[15] A. Ammar, D. Ryckelynck, F. Chinesta, R. Keunings, On the reduction of kinetic theory models related to finitely extensible dumbbells, Journal of Non-Newtonian Fluid Mechanics 134 (2006) 136–147.

[16] D. Ryckelynck, D. Missoum Benziane, Multi-level A Priori Hyper-Reduction of mechanical models involving internal variables, Computer Methods in Applied Mechanics and Engineering 199 (17–20) (2010) 1134–1142.
[17] C. Allery, A. Hamdouni, D. Ryckelynck, N. Verdon, A priori reduction method for solving the two-dimensional Burgers' equations, Applied Mathematics and Computation 217 (15) (2011) 6671–6679.
[18] N. Verdon, C. Allery, C. Beghein, A. Hamdouni, D. Ryckelynck, Reduced-Order Modelling for solving linear equations and non-linear equations, International Journal for Numerical Methods in Biomedical Engineering 27 (1) (2011) 43–58.
[19] A. Ammar, F. Chinesta, P. Diez, A. Huerta, An error estimator for separated representations of highly multidimensional models, Computer Methods in Applied Mechanics and Engineering 199 (2010) 1872–1880.
[20] A. Ammar, F. Chinesta, A. Falco, On the convergence of a greedy rank-one update algorithm for a class of linear systems, Archives of Computational Methods in Engineering 17 (4) (2010) 473–486.
[21] P. Ladeveze, L. Chamoin, On the verification of model reduction methods based on the proper generalized decomposition, Computer Methods in Applied Mechanics and Engineering 200 (23–24) (2011) 2032–2047.
[22] P. Ladeveze, Non linear Computational Structural Mechanics: New Approaches and Non-incremental Methods of Calculation, Mechanical Engineering Series, Springer, 2009.
[23] P. Ladeveze, J.-C. Passieux, D. Neron, The LATIN multiscale computational method and the Proper Generalized Decomposition, Computer Methods in Applied Mechanics and Engineering 199 (21–22) (2010) 1287–1296.
[24] A. Nouy, A generalized spectral decomposition technique to solve a class of linear stochastic partial differential equations, Computer Methods in Applied Mechanics and Engineering 196 (45–48) (2007) 4521–4537.
[25] A. Nouy, O.P. Le Maitre, Generalized spectral decomposition for stochastic nonlinear problems, Journal of Computational Physics 228 (1) (2009) 202–235.
[26] A. Ammar, B. Mokdad, F. Chinesta, R. Keunings, 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 139 (3) (2006) 153–176.
[27] A. Ammar, B. Mokdad, F. Chinesta, R. Keunings, 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 Non-Newtonian Fluid Mechanics 144 (2–3) (2007) 98–121.
[28] B. Mokdad, E. Pruliére, A. Ammar, F. Chinesta, On the simulation of kinetic theory models of complex fluids using the Fokker–Planck approach, Applied Rheology 2 (2007) 1–14.
[29] F. Chinesta, A. Ammar, P. Joyot, 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 6/3 (2008) 191–213.
[30] B. Bognet, F. Bordeu, F. Chinesta, A. Leygue, A. Poitou, Advanced simulation of models defined in plate geometries: 3D solutions with 2D computational complexity, Computer Methods in Applied Mechanics and Engineering 201–204 (2012) 1–12.
[31] A. Dumon, C. Allery, A. Ammar, Proper general decomposition (PGD) for the resolution of Navier–Stokes equations, Journal of Computational Physics 230 (4) (2011) 1387–1407.
[32] A. Dumon, C. Allery, A. Ammar, Proper Generalized Decomposition method for incompressible flows in stream-vorticity formulation, European Journal of Computational Mechanics 19 (5–7) (2010) 591–617.
[33] A. Leygue, E. Verron, A first step towards the use of proper general decomposition method for structural optimization, Archives of Computational Methods in Engineering 17 (4) (2010) 465–472.
[34] F. Chinesta, A. Ammar, E. Cueto, Recent advances and new challenges in the use of the proper generalized decomposition for solving multidimensional models, Archives of Computational Methods in Engineering 201–204 (4) (2010) 327–350.
[35] F. Chinesta, P. Ladeveze, E. Cueto, A short review on model order reduction based on proper generalized decomposition, Journal of Fluids Engineering 18 (4) (2011) 395–404.
[36] D. Gonzalez, A. Ammar, F. Chinesta, E. Cueto, Recent advances in the use of separated representations, International Journal of Numerical Methods in Engineering 81 (5) (2010) 637–659.
[37] A. Nouy, A priori model reduction through Proper Generalized Decomposition for solving time-dependent partial differential equations, Computer Methods in Applied Mechanics and Engineering 199 (23–24) (2010) 1603–1626.
[38] F. Brezzi, On the existences, uniquness and approximation of saddle-point problems arising from lagrangian multipliers, R.A.I.R.O., R2 (1998) 129–151.
[39] M. Azaiez, C. Bernardi, M. Grundmann, Spectral methods applied to porous media equations, East–West Journal of Numerical Mathematics 2 (2) (1994) 91–105.
[40] O. Botella, Résolution des équations de Navier-Stokes par des schémas de projection Tchebychev, Rapport de Recherche INRIA, 1996.
[41] O. Botella, On the solution of the Navier–Stokes equations using Chebyshev projection schemes with third-order accuracy in time, Computers and Fluids 26 (2) (1997) 107–116.
[42] J. Van-Kan, A second order accurate pressure-correction scheme for viscous incompressible flow, SIAM – Journal on Scientific and Statistical Computing 7 (1986) 271–288.
[43] J.L. Guermond, P. Minev, Jie Shen, An overview of projection methods for incompressible flows, Computer Methods in Applied Mechanics and Engineering 195 (44–47) (2006) 6011–6045.
[44] A. Quarteroni, F. Saleri, A. Veneziani, Factorization methods for the numerical approximation of Navier–Stokes equations, Computer Methods in Applied Mechanics and Engineering 188 (1–3) (2000) 505–526.
[45] U. Ghia, K.N. Ghia, C.T. Shin, High-re solutions for incompressible flow using the Navier–Stokes equations and a multigrid method, Journal of Computational Physics 48 (1) (1982) 387–411.
[46] G. B Deng, J. Piquet, P. Queutey, M. Visonneau, Incompressible flow calculations with a consistent physical interpolation finite volume approach, Computers and Fluids 23 (8) (1994) 1029–1047.
[47] W. Zhang, C.H. Zhang, G. Xi, An explicit Chebyshev pseudospectral multigrid method for incompressible Navier–Stokes equations, Computers and Fluids 39 (1) (2010) 178–188.
[48] Q. Liang, P.H. Taylor, A.G.L. Borthwick, Particle mixing and reactive front motion in unsteady open shallow flow – Modelled using singular value decomposition, Computers and Fluids 36 (2) (2007) 248–258.
[49] Z. Luo, X. Yang, Y. Zhou, A reduced finite difference scheme based on singular value decomposition and proper orthogonal decomposition for Burgers equation, Journal of Computational and Applied Mathematics 29 (1) (2009) 97–107.