Résolution de systèmes linéaires issus de la discrétisation d'une équation de Navier-Stokes

Dany Mezher, Bernard Philippe

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Résolution de systèmes linéaires issus de la discrétisation d'une équation de Navier-Stokes

Dany Mezher , Bernard Philippe

N°3777

Octobre 1999

THÈME 4

ISSN 0249-6399 ISRN INRIA/RR--3794--FR+ENG

Résolution de systèmes linéaires issus de la discrétisation d'une équation de Navier-Stokes

Dany Mezher* , Bernard Philippe

Thème 4 — Simulation et optimisation
de systèmes complexes
Projet Aladin

Rapport de recherche n° 3777 — Octobre 1999 — 25 pages

Résumé : On étudie dans ce rapport le système linéaire issu de la modélisation d'un phénomène d'eutrophisation de bassin. On y compare des méthodes de relaxation par bloc et des méthodes de type gradient conjugué. Les préconditionnements introduits sont des décompositions incomplètes de Choleski.

Mots-clé : Jacobi par bloc, Gauss-Seidel par bloc, Gradient Conjugué, Résidus Conjugués, Gradient Bi-Conjugué, factorisation de Choleski incomplète, système linéaire, équation de Navier-Stokes.

(Abstract: pto)

Le travail a été réalisé dans le cadre du projet Européen ESIMEAU (Programme INCO-DC, No. 961785).

* ESIB, Campus des Sciences et Technologies, Mar Roukoz, Liban.

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex (France)
Téléphone : 02 99 84 71 00 - International : +33 2 99 84 71 00
Télécopie : 02 99 84 71 71 - International : +33 2 99 84 71 71

Resolution of Linear Systems obtained from discretized Navier Stokes Equations

Abstract: We study the resolution of a linear system which is obtained from the model of eutrophication of a basin. Block iterative methods such as Jacobi and Gauss-Seidel are compared to Conjugate Gradient like methods. The considered preconditioners are based on Incomplete Cholesky Factorisations.

Key-words: Block Jacobi, Block Gauss-Seidel, Conjugate Gradient, Conjugate Residuals, Bi-Conjugate Gradient, Incomplete Cholesky Factorisation, linear systems, Navier-Stokes.

1 Position du problème

1.1 L'eutrophisation

L'eutrophisation est définie comme l'enrichissement de l'eau en matières nutritives (principalement l'azote et le phosphate) qui sont à la base de la chaîne alimentaire et qui stimulent la croissance des organismes vivants. C'est un phénomène naturel de vieillissement des lacs qui s'étendait jadis sur des millénaires. Le phénomène se trouve actuellement accéléré à cause du rejet des déchets urbains et industriels dans les rivières. Il a de graves conséquences aux plans écologiques, économiques et sanitaires. Les techniques de restauration de l'eau sont nombreuses et dans cette application on s'intéresse à l'oxygénation qui consiste à injecter de l'air au fond du lac pour entraîner l'eau vers le haut afin de l'oxygéner par contact avec les eaux de surface.

1.2 Modélisation

Dans cette section on s'intéresse à la modélisation des mouvements de l'eau. Comme l'air occupe un volume négligeable par rapport au volume de l'eau, on suppose un écoulement monophasique. Cette approche ne tient pas compte de la présence de l'air dans l'eau mais elle tient compte de la quantité de mouvement qu'elle apporte sous la forme de conditions aux limites particulières. Dans ce cas, on résout les équations de conservation de masse, de quantité de mouvement et d'énergie dans un domaine homogène. Dans le cadre de ce travail, on se limite au modèle incompressible s'appuyant sur les équations de Navier-Stokes décrivant le mouvement d'un fluide incompressible

$$\begin{cases} \frac{\partial U}{\partial t} + (U.\nabla)U - \nu\triangle U + \nabla P = F, & \forall x \in \Omega, t > 0 \\ \nabla.U = 0, & \forall x \in \Omega, t > 0 \end{cases} \tag{1}$$

où U est la vitesse d'écoulement, P la pression, F est la force par unité de masse et ν la viscosité cinématique du fluide.

La formulation (1), dite vitesse-pression, permet l'utilisation de conditions aux limites prenant bien en compte la situation physique. Pour mieux contrôler la dynamique du fluide dans le domaine Ω, on utilise souvent la formulation fonction courant-vorticité. On obtient cette formulation à partir du système (1), en passant en notations $\psi - \omega$ avec le changement de variables :

$$\omega = \nabla \times U \text{ et } U = \nabla \times \psi$$

où la fonction courant ψ et la vorticité ω sont solutions de:

$$\begin{cases} \frac{\partial \omega}{\partial t} + (U.\nabla)\omega - \nu\triangle\omega = \nabla \times F, & \forall x \in \Omega, t > 0 \\ \omega + \triangle\psi = 0, & \forall x \in \Omega, t > 0 \end{cases}$$

Une étude complète de cette formulation et de ses propriétés numériques est exposée dans [6] (pages 352-357) et dans [7, 8, 9, 10, 11]. Pour résoudre le système d'équations 1.2, on doit introduire les conditions aux limites. Puisque la vorticité est inconnue sur les bords, nous

utilisons des conditions de Dirichlet explicites et des conditions de Neuman implicites sur ψ sans imposer des conditions sur ω. Ces conditions traduisent des conditions aux limites sur la vitesse.

2 Le système issu de la discrétisation

A partir de cette formulation, la discrétisation de l'espace est obtenue par éléments finis du type P1 [6, 13] ; la méthode des caractéristiques est utilisée pour traiter l'évolution en temps [14].

Sans intégration des conditions aux limites de Dirichlet, le système linéaire obtenu a une matrice de la forme

$$\begin{bmatrix} A & B \\ -B^t & -\lambda B \end{bmatrix} \tag{2}$$

où A et B sont des matrices carrées de même dimension définies comme suit:

$$A = [a_{i,j}]$$

avec

$$a_{i,j} = \int_\Omega \phi_i \phi_j dS + \sum_{\eta_{\alpha\beta}} |\eta_{\alpha\beta}| \int_{\eta_{\alpha\beta}} \partial^n \phi_i \partial^n \phi_j dl$$

où les fonctions ϕ_i sont les fonctions tests, $\eta_{\alpha\beta}$ est le coté commun aux deux triangles T_α et T_β et $\partial^n \phi_i$ est la dérivée normale de ϕ_i définie par:

$$\partial^n \phi = \nabla \phi_{|T_\alpha} . \vec{n}_{T_\alpha} + \nabla \phi_{|T_\beta} . \vec{n}_{T_\beta}$$

avec $\vec{n}_{T_\alpha}$ vecteur normal sortant du triangle T_α.

$$B = [b_{i,j}] \text{ (matrice de rigidité)}$$

$$\text{avec } b_{i,j} = \int_\Omega \nabla \phi_i \nabla \phi_j dS$$

La constante λ est définie par:

$$\lambda = \frac{1}{\nu \Delta t}$$

où Δt est le pas de discrétisation en temps.

2.1 Propriétés des matrices A et B

Les matrices précédentes ont les propriétés suivantes:

- A est une matrice symétrique définie positive.
- B est une matrice symétrique.

- Les valeurs propres de B sont négatives ou nulles.
- B est singulière (cette singularité est éliminée en introduisant les conditions aux limites de Dirichlet).

L'introduction des conditions aux limites de type Dirichlet modifie le système; on se ramène au problème de résolution d'un système de la forme :

$$\begin{bmatrix} A & C \\ -C^t & -\lambda B' \end{bmatrix}$$

où C est obtenue à partir de B en annulant les colonnes correspondantes aux points de la frontière correspondant à une condition aux limites de type Dirichlet et où B' est ensuite obtenue à partir de C en remplaçant les lignes l_i correspondantes aux conditions aux limites par $-e_i^t$ (e_i désigne le $i$ème vecteur canonique) ce qui rend définie négative la matrice obtenue.

Lemme 2.1 *La matrice $B'^{-1}C^t$ est celle d'un projecteur.*

Preuve. On suppose qu'on a ordonné les inconnues de manière que sont d'abord énumérées les inconnues ne correspondant pas aux conditions de Dirichlet puis celles de Dirichlet alors:

$$B = \begin{bmatrix} B_1 & B_2 \\ B_2^t & B_3 \end{bmatrix} \qquad \text{et} \qquad C = \begin{bmatrix} B_1 & 0 \\ B_2^t & 0 \end{bmatrix}$$

$$B' = \begin{bmatrix} B_1 & 0 \\ 0 & -I \end{bmatrix}$$

d'où

$$B'^{-1}C^t = \begin{bmatrix} I & 0 \\ -B_2^t & 0 \end{bmatrix}$$

ce qui prouve que $B'^{-1}C^t$ est un projecteur de noyau engendré par les vecteurs de la base canonique $[e_i]$, définis par $i \in E$ ensemble des inconnues qui correspondent aux conditions aux limites du type Dirichlet. □

Dans la suite, la matrice B' sera simplement notée B. Après introduction des conditions aux limites de Dirichlet, la matrice introduite en (2) est donc :

$$A_g = \begin{bmatrix} A & C \\ -C^t & -\lambda B \end{bmatrix}. \tag{3}$$

2.2 Expérimentations

Nous allons dans la suite étudier différentes méthodes itératives de résolution en analysant leur comportement. On comparera leur efficacité sur un cas test défini par :

- une coupe verticale d'un bassin cubique maillée suivant une grille 2-D régulière

- chaque carré élémentaire est divisé en deux triangles
- des conditions aux limites du type Dirichlet explicite sur ψ
- des conditions aux limites du type Neuman implicite sur ψ
- pas de condition sur ω

Trois tests correspondent à trois niveaux de maillages. Les caractéristiques des matrices obtenues sont rassemblées dans le tableau 1.

Propriété	Test T_1	Test T_2	Test T_3
Grille	21×21	41×41	81×81
Matrices A, B et C	441×441	1681×1681	6561×6561
$N_z(A)$	5,241	20,881	83,361
$N_z(B)$	1,809	9,747	39,879
$N_z(C)$	1,805	9,849	40,018
Dimension A_g	882×882	3261×3261	13122×13122
$N_z(A_g)$	10,660	50,326	203,276
λ	250,000	250,000	250,000
$\|A\|_F$	2.9717×10^2	6.0295×10^3	1.2146×10^4
$\|B\|_F$	8.4994×10^1	1.7137×10^2	3.4930×10^2
$\|C\|_F$	8.4970×10^1	1.7115×10^2	3.4925×10^2

TAB. 1 – *Caractéristiques des matrices de test*

3 Les méthodes de relaxations par bloc

A partir du système linéaire

$$A_g x = b \tag{4}$$

on définit une méthode de relaxation par une décomposition de la matrice A_g en deux matrices M et N telles que

$$A_g = M - N$$

et le procédé itératif associé par:

$$Mx^{(k+1)} = Nx^{(k)} + b.$$

En notant x_{ex} la solution du système, on en déduit que :

$$x^{(k+1)} - x_{ex} = (M^{-1}N)(x^{(k)} - x_{ex})$$

ce qui prouve que le taux de convergence d'un tel procédé est caractérisé par le rayon spectral de la matrice $M^{-1}N$.

Dans la suite, on partagera les vecteurs du système (4) en deux sous vecteurs de taille moitié :

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad et \quad b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

avec x_1, x_2, b_1 et b_2 de même dimension.

3.1 Méthode de Jacobi par bloc

Pour cette méthode, les matrices M et N sont :

$$M = \begin{bmatrix} A & 0 \\ 0 & -\lambda.B \end{bmatrix} \quad N = \begin{bmatrix} 0 & -C \\ C^t & 0 \end{bmatrix}$$

et les itérations sont définies par:

$$\left\{ \begin{array}{lcl} x_1^{(k+1)} & = & A^{-1}(b_1 - Cx_2^{(k)}) \\ x_2^{(k+1)} & = & -\frac{1}{\lambda}B^{-1}(C^t x_1^{(k)} + b_2) \end{array} \right. \tag{5}$$

et la matrice de réduction d'erreur est :

$$M^{-1}N = \begin{bmatrix} 0 & -A^{-1}C \\ -\frac{1}{\lambda}B^{-1}C^t & 0 \end{bmatrix}$$

Proposition 3.1 *Le taux asymptotique de convergence de l'itération de Jacobi par bloc (5) est égal à :*

$$\Gamma = \frac{1}{\sqrt{\lambda}}\sqrt{\rho(A^{-1}CB^{-1}C^t)}.$$

Preuve. La convergence de deux itérations est caractérisée par le rayon spectral de la matrice

$$(M^{-1}N)^2 = \frac{1}{\lambda}\begin{bmatrix} A^{-1}CB^{-1}C^t & 0 \\ 0 & B^{-1}C^tA^{-1}C \end{bmatrix}$$

Comme les deux sens de multiplication des matrices $A^{-1}C$ et $B^{-1}C^t$ donnent des produits de même rayon spectral, le taux de convergence de deux itérations est donné par:

$$\Gamma_{2iter.} = \frac{1}{\lambda}\max(\rho(A^{-1}CB^{-1}C^t),\rho(B^{-1}C^tA^{-1}C)) = \frac{1}{\lambda}\rho(A^{-1}CB^{-1}C^t)$$

et donc rapporté à une itération

$$\Gamma_{1iter} = \frac{1}{\sqrt{\lambda}}\sqrt{\rho(A^{-1}CB^{-1}C^t)}$$

□

L'algorithme de la méthode est décrit dans la figure 1.

iter=0
tant que (iter $\leq$ MaxIter)
Résoudre $Ax_1^{(k+1)} = b_1 - Cx_2^{(k)}$
Résoudre $-\lambda Bx_2^{(k+1)} = C^t x_1^k + b2$
$r_1 = ||Ax_1^{(k+1)} + Cx_2^{(k+1)} - b1||$
$r_2 = || - C^t x_1^{(k+1)} - \lambda Bx_2^{(k+1)} - b2||$
$res = \sqrt{r_1^2 + r_2^2}$
$Nx = \sqrt{||x_1^{(k+1)}||^2 + ||x_2^{(k+1)}||^2}$
si $res/Nx < \epsilon$ **sortir**
iter=iter+1
fin tantque

FIG. 1 – *Algorithme de Jacobi*

Un avantage de cette méthode est de permettre un calcul indépendant de $x_1^{(k+1)}$ et $x_2^{(k+1)}$, ce qui permet de résoudre en parallèle les systèmes en A et en B.

3.2 Méthode de Gauss-Seidel par Bloc

Deux possibilités de subdivision sont possibles; la première consiste à prendre une matrice M supérieure par bloc et N strictement inférieure par bloc, alors que la deuxième méthode consiste à prendre M inférieure par bloc et N strictement supérieure par bloc. Nous appelerons la première méthode *méthode de Gauss-Seidel supérieure* tandis que la seconde sera dite *inférieure*. C'est en fait cette dernière qui est la forme habituelle de Gauss-Seidel.

3.2.1 M est triangulaire supérieure par bloc

Dans ce cas, on prend M et N comme suit:

$$M = \begin{bmatrix} A & C \\ 0 & -\lambda.B \end{bmatrix} \quad \text{et} \quad N = \begin{bmatrix} 0 & 0 \\ C^t & 0 \end{bmatrix}$$

Les itérations sont alors définies par:

$$\left\{ \begin{array}{lcl} x_1^{(k+1)} & = & A^{-1}(b_1 - Cx_2^{(k+1)}) \\ x_2^{(k+1)} & = & -\frac{1}{\lambda}B^{-1}(C^t x_1^{(k)} + b_2) \end{array} \right. \tag{6}$$

et la matrice de réduction d'erreur est :

$$M^{-1}N = -\frac{1}{\lambda}\begin{bmatrix} -A^{-1}CB^{-1}C^t & 0 \\ B^{-1}C^t & 0 \end{bmatrix}$$

Proposition 3.2 *Le taux asymptotique de convergence de l'itération de Gauss-Seidel par bloc (6) est égal à :*

$$\Gamma = \frac{1}{\lambda}\rho(A^{-1}CB^{-1}C^t).$$

Preuve. Evidente. □

Cela prouve qu'un pas de Gauss-Seidel par bloc a un taux de convergence équivalent à celui de deux pas de Jacobi par bloc, alors que le coût des pas des deux méthodes sont de même ordre (multiplication par C et C^t et résolution de deux systèmes de matrices A et B).

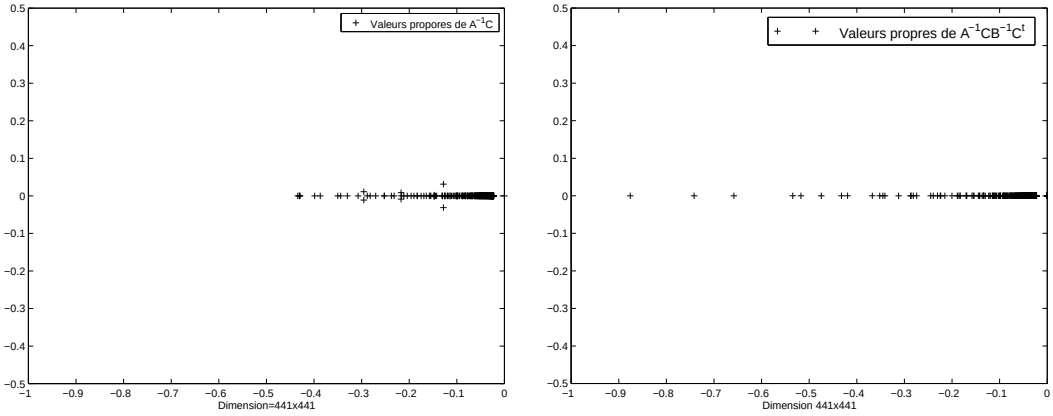

FIG. 2 – *Spectres de $A^{-1}C$ et $A^{-1}CB^{-1}C^t$, test T_1.*

L'algorithme de Gauss-Seidel supérieur par bloc est décrit dans la figure 3.

iter=0
tant que (iter $\leq$ MaxIter)
Résoudre $-\lambda B x_2^{(k+1)} = C^t x_1^{(k)} + b_2$
Résoudre $A x_1^{(k+1)} = b_1 - C x_2^{(k+1)}$
$r_1 = ||A x_1^{(k+1)} + C x_2^{(k+1)} - b1||$
$r_2 = || - C^t x_1^{(k+1)} - \lambda B x_2^{(k+1)} - b2||$
$res = \sqrt{r_1^2 + r_2^2}$
$Nx = \sqrt{||x_1^{(k+1)}||^2 + ||x_2^{(k+1)}||^2}$
si $res/Nx < \epsilon$ **sortir**
iter=iter+1
fin tantque

FIG. 3 – *Algorithme de Gauss-Seidel avec M triangulaire supérieure par bloc*

3.2.2 M est triangulaire inférieure par bloc

Dans ce cas,

$$M = \begin{bmatrix} A & 0 \\ -C^t & -\lambda.B \end{bmatrix} \quad \text{et} \quad N = \begin{bmatrix} 0 & -C \\ 0 & 0 \end{bmatrix}$$

Les itérations sont alors définies par:

$$\begin{cases} x_1^{(k+1)} &= A^{-1}(b_1 - C x_2^{(k)}) \\ x_2^{(k+1)} &= -\frac{1}{\lambda} B^{-1}(C^t x_1^{(k+1)} + b_2) \end{cases}$$

et la convergence est caractérisée par le rayon spectral de

$$M^{-1}N = \begin{bmatrix} 0 & -A^{-1}C \\ 0 & \frac{1}{\lambda} B^{-1} C^t A^{-1} C \end{bmatrix}$$

d'où la valeur du taux

$$\Gamma = \frac{1}{\lambda} \rho(B^{-1} C^t A^{-1} C)$$

Les deux méthodes de Gauss-Seidel sont donc équivalentes. Dans la suite, nous ne retenons que la méthode supérieure. La figure 2 représente le spectre des matrices $A^{-1}C$ et $A^{-1}CB^{-1}C^t$ dans le cas du test T_1. On peut y remarquer l'effet de la multiplication de la matrice $A^{-1}C$ par le projecteur $B^{-1}C^t$. En réduisant le pas de temps Δt, les valeurs de λ seraient plus élevées et donc le taux de convergence meilleur.

3.3 Décomposition SOR par bloc

Dans ce cas, M et N sont définies à partir d'un paramètre $\omega \in [0,2]$ comme suit:

$$M = \begin{bmatrix} A & 0 \\ -\omega C^t & -\lambda B \end{bmatrix} \quad \text{et} \quad N = \begin{bmatrix} (1-\omega)A & -\omega C \\ 0 & -\lambda(1-\omega)B \end{bmatrix}$$

Le schéma itératif est donné par:

$$\left\{ \begin{array}{rcl} Ax_1^{(k+1)} & = & (1-\omega)Ax_1^{(k)} - \omega C x_2^{(k)} + \omega b_1 \\ -\lambda B x_2^{(k+1)} & = & \omega C^t x_1^{(k+1)} - \lambda(1-\omega)Bx_2^{(k)} + \omega b_2 \end{array} \right.$$

La convergence est caractérisée par le rayon spectral de

$$M^{-1}N = \left[\begin{array}{cc} (1-\omega)I & -\omega A^{-1}C \\ \frac{\omega(1-\omega)}{-\lambda} B^{-1}C^t & (1-\omega)I + \frac{\omega^2}{\lambda} B^{-1}C^t A^{-1}C \end{array} \right]$$

A la figure 4, on a représenté les variations du rayon spectral de $M^{-1}N$ en fonction de ω variant de 0 à 2 dans le cas de test T_1.

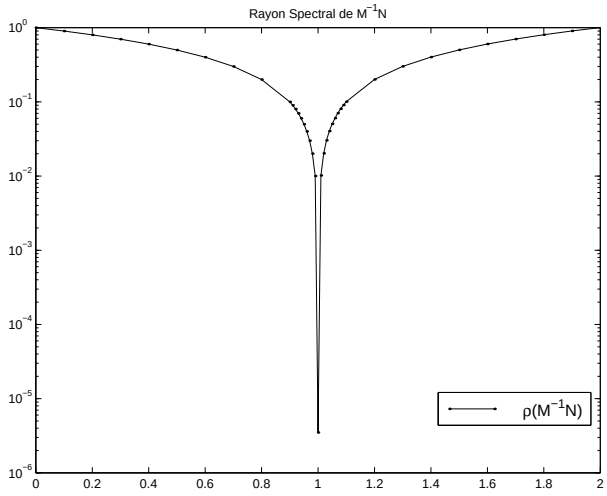

FIG. 4 – *Rayon spectral de $M^{-1}N$ en fonction de ω, test T_1*

On voit clairement que le minimum correspond au cas $\omega = 1$. Dans ce cas, la méthode SOR est réduite à la méthode Gauss-Seidel par bloc (inférieure).

3.4 Résolutions des systèmes en A et B

Chaque itération des méthodes précédentes, conduit à résoudre deux systèmes linéaires en A et B.

Les matrices A et B étant symétriques définies, positive pour A et négative pour B, nous utiliserons la méthode du *Gradient Conjugué* qui est considérée comme la meilleure méthode itérative sous ces hypothèses. L'algorithme du Gradient Conjugué est donné dans la figure 5.

$r_0 = b - Ax_0$,$p_0 = r_0$
Pour $j = 0,1,...,$jusqu'à la convergence **faire**
$\alpha_j = (r_j,r_j)/(Ap_j,p_j)$
$x_{j+1} = x_j + \alpha_j p_j$
$r_{j+1} = r_j - \alpha Ap_j$
$\beta_j = (r_{j+1},r_{j+1})/(r_j,r_j)$
$p_{j+1} = r_{j+1} + \beta_j p_j$
Fin faire

FIG. 5 – *Algorithme du Gradient Conjugué*

4 Résultats numériques pour les méthodes de relaxation

4.1 Sans préconditionnement

Dans cette partie, nous illustrons les résultats numériques obtenus pour les cas tests du tableau 1.

Test	Itération	$Ax_1^{(k+1)} = b_1 - Cx_2^{(k)}$	$-\lambda Bx_2^{(k+1)} = C^t x_1^{(k)} + b_2$	$\frac{\|r\|}{\|b\|}$
T_1	1	289	55	5.4639×10^{-6}
	2	306	27	1.1744×10^{-6}
	3	191	28	8.2910×10^{-9}
Total		786	110	
T_2	1	768	86	1.8894×10^{-5}
	2	548	68	8.9048×10^{-6}
	3	326	49	8.3170×10^{-9}
Total		1642	203	
T_3	1	1525	164	3.4768×10^{-5}
	2	1472	145	2.0215×10^{-5}
	3	718	103	9.6016×10^{-9}
Total		3715	412	

TAB. 2 – *Convergence de la méthode de Jacobi par bloc (sans préconditionnement)*

Les tableaux 2 et 3 donnent les performances de la méthode de Jacobi par bloc et Gauss Seidel par bloc respectivement, ainsi que le nombre d'itérations du Gradient Conjugué pour la résolution des deux systèmes en A et B.

Test	Itération	$Ax_1^{(k+1)} = b_1 - Cx_2^{(k+1)}$	$-\lambda Bx_2^{(k+1)} = C^t x_1^{(k)} + b_2$	$\frac{\|\|r\|\|}{\|\|b\|\|}$
T_1	1	173	55	9.6208×10^{-9}
T_2	1	752	86	8.8286×10^{-6}
	2	307	86	9.3618×10^{-9}
Total		1059	172	
T_3	1	1529	164	1.9585×10^{-5}
	2	645	140	9.3686×10^{-9}
Total		2174	304	

TAB. 3 – *Convergence de la méthode de Gauss-Seidel par bloc (sans préconditionnement)*

Nous constatons ce qui avait été prédit à savoir que la méthode de Jacobi présente une convergence plus lente que la méthode de Gauss-Seidel. Donc bien qu'elle soit *plus parallélisable*, nous allons opter pour cette dernière.

Nous remarquons aussi que la méthode du Gradient Conjugué nécessite un grand nombre d'itérations pour converger d'où la nécessité de recourir à un préconditionnement des deux systèmes.

4.2 Avec préconditionnement

Dans cette partie, nous décrivons les préconditionneurs choisis ainsi que les résultats numériques obtenus. Plusieurs préconditionneurs ont été testés (diagonal, décomposition incomplète en LU avec et sans remplissage, décomposition incomplète de Choleski avec et sans remplissage,...) Les meilleurs résultats sont donnés ici. Ils ont été obtenus pour la factorisation incomplète de Choleski [2] :

Cas de la matrice A: la décomposition incomplète de Choleski sans remplissage de la matrice A translatée a donné des résultats prometteurs. En pratique, la décomposition de toute matrice de la forme $A+\alpha I$ (où I est la matrice identité) donne de bon résultats pour $\alpha > 9$. Sans décalage, on peut aboutir à un échec de la factorisation.

Cas de la matrice B: la décomposition incomplète de Choleski sans remplissage de B a donné des résultats acceptables, mais le bénéfice est moins sensible car la convergence était déjà meilleure pour ce système que pour le précédent.

L'efficacité des préconditionnements des Gradients Conjugués pour une valeur $\alpha = 10$ est décrite dans les courbes des figures 6 et 7 et les tableaux 4 et 5.

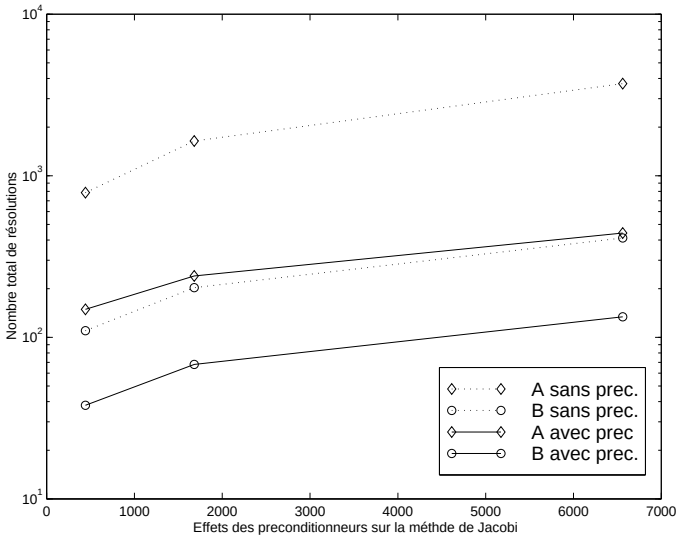

FIG. 6 – *Nombre total d'itérations de Gradient Conjugué pour la méthode de Jacobi par bloc (Tests T_1, T_2 et T_3).*

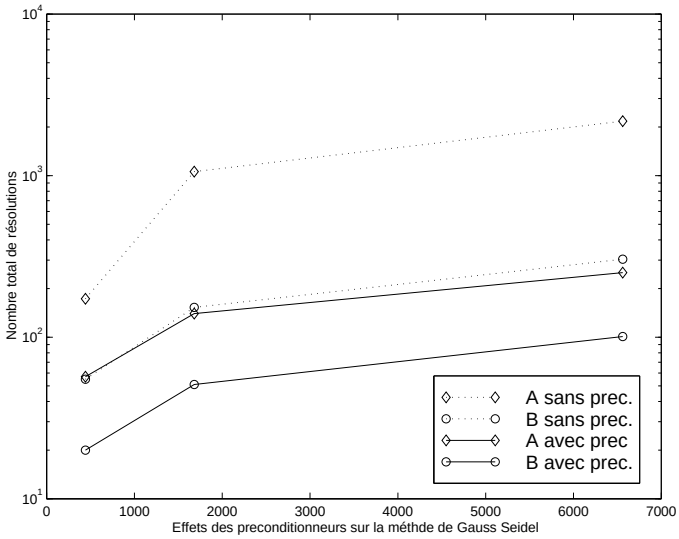

FIG. 7 – *Nombre total d'itérations de Gradient Conjugué pour la méthode de Gauss-Seidel par bloc (Tests T_1, T_2 et T_3).*

Test	Itération	$Ax_1^{(k+1)} = b_1 - Cx_2^{(k)}$	$-\lambda Bx_2^{(k+1)} = C^t x_1^{(k)} + b_2$	$\frac{\|\|r\|\|}{\|\|b\|\|}$
T_1	1	54	20	5.4639×10^{-6}
	2	63	9	1.1744×10^{-6}
	3	32	9	7.0566×10^{-9}
Total		149	38	
T_2	1	99	33	1.8894×10^{-5}
	2	98	19	8.9045×10^{-6}
	3	43	16	8.6234×10^{-9}
Total		240	58	
T_3	1	181	65	3.4768×10^{-5}
	2	177	37	2.0215×10^{-5}
	3	84	32	8.4543×10^{-9}
Total		442	134	

TAB. 4 – *Résidus de la méthode de Jacobi par bloc avec préconditionnement du Gradient Conjugué*

Test	Itération	$Ax_1^{(k+1)} = b_1 - Cx_2^{(k+1)}$	$-\lambda Bx_2^{(k+1)} = C^t x_1^{(k)} + b_2$	$\frac{\|\|r\|\|}{\|\|b\|\|}$
T_1	1	57	20	5.0215×10^{-9}
T_2	1	98	33	8.8286×10^{-6}
	2	42	18	7.9537×10^{-9}
Total		140	51	
T_3	1	176	65	1.9585×10^{-5}
	2	75	36	9.5223×10^{-9}
Total		251	101	

TAB. 5 – *Résidus de la méthode de Gauss-Seidel par bloc avec préconditionnement du Gradient Conjugué*

4.3 Diminution du coût de la résolution pour la méthode de Gauss-Seidel

Nous venons de voir que la méthode de Gauss-Seidel est celle que l'on doit considérer. On peut encore en diminuer le coût en remarquant que puisque la convergence se produit en deux itérations, il semble suffisant à la première itération de résoudre les systèmes en A et en B à une précision inférieure. Si τ représente la tolérance finale demandée, nous proposons d'imposer un seuil de convergence en $\sqrt{\tau}$ à la première itération et en τ aux suivantes. Pour montrer le gain obtenu, on applique cette technique sur les trois cas tests, avec $\tau = 10^{-8}$. Les résultats sont donnés dans le tableau 6 : on y indique le nouveau nombre d'itérations avec entre parenthèses l'ancien nombre qui était indiqué dans le tableau 5. On constate, qu'en dehors du cas test T_1 où la stratégie impose une deuxième itération qui n'existait pas avant, l'amélioration est nette. Nous l'adoptons donc car pour les problèmes de grande taille il est peu probable que la convergence soit obtenue en une itération.

Tests	A	B
T_1	72 (57)	25 (20)
T_2	104 (140)	41 (51)
T_3	186 (251)	70 (101)

TAB. 6 – *Diminution des coûts en itérations dans la méthode de Gauss-Seidel*

5 Méthodes de projection

Nous considérons maintenant l'approche qui consiste à résoudre le système

$$A_g x = b$$

par une méthode itérative de type gradient conjugué sur la matrice entière, donc de dimension double des matrices précédemment manipulées. Dans cette partie, nous mesurons les performances de trois méthodes appliquées au grand système : la méthode du gradient conjugué, des résidus conjugués [1], du gradient biconjugué. Pour une présentation détaillée des algorithmes, le lecteur peut consulter [1, 5, 12]. Les algorithmes sont présentés sous leur version préconditionnée dans les figures 8, 9 et 10

Le gradient conjugué, peut ne pas converger car la matrice est non définie. La méthode des résidus conjugués est assurée de converger mais malheureusement cette méthode est généralement plus lente que la méthode du gradient conjugué quand celui-ci converge. Le gradient bi-conjugué peut échouer et il entraîne deux multiplications de matrices par itération.

1. Pour appliquer la méthode du gradient conjugué ou celle des résidus conjugués, on multiple le bloc inférieur du système par -1 et on se ramène alors à la résolution d'un système symétrique, mais non défini.

Calculer $r_0 = b - Ax_0$, $z_0 = M^{-1}r_0$ et $p_0 = z_0$
Pour j=0,1,...
  $\alpha_j = (r_j,z_j)/(Ap_j,p_j)$
  $x_{j+1} = x_j + \alpha_j p_j$
  $r_{j+1} = r_j - \alpha_j Ap_j$
  $z_{j+1} = M^{-1}r_{j+1}$
  $\beta_j = (r_{j+1},z_{j+1})/(r_j,z_j)$
  $p_{j+1} = z_{j+1} + \beta_j p_j$
Fin Pour

FIG. 8 – *Algorithme du Gradient Conjugué préconditionné ([12] page 247).*

Calculer $r_0 = b - Ax_0$, choisir r_0^* (par exemple $r_0^* = r_0$)
Pour i=1,2,... jusqu'à la convergence
  Résoudre $Mz_{i-1} = r_{i-1}$
  Résoudre $M^t z_{i-1}^* = r_{i-1}^*$
  $\rho_{i-1} = x_{i-1}^t r_{i-1}^*$
  Si $\rho_{i-1} = 0$, Echec
  Si $i = 1$
    $p_i = z_{i-1}$
    $p_i^* = z_{i-1}^*$
  sinon
    $\beta_{i-1} = \rho_{i-1}/\rho_{i-2}$
    $p_i = z_{i-1} + \beta_{i-1}p_{i-1}$
    $p_i^* = z_{i-1}^* + \beta_{i-1}p_{i-1}^*$
  FinSi
  $q_i = Ap_i$
  $q_i^* = A^t p_i^*$
  $\alpha_i = \rho_{i-1}/{p_i^*}^t q_i$
  $x_i = x_{i-1} + \alpha_i p_i$
  $r_i = r_{i-1} - \alpha_i q_i$
  $r_i^* = r_{i-1}^* - \alpha_i q_i^*$
Fin Pour

FIG. 9 – *Algorithme de BICG préconditionné ([1] page 22)*

Sans préconditionnement, ces méthodes ne donnent pas des résultats satisfaisant comme le montrent les figures 11, 12 et 13 pour le test T_3.

Le préconditionnement par les décompositions incomplètes sans remplissage des matrices:

Calculer $r_0 = b - Ax_0$,$p_0 = r_0$
Pour j=1,2,... jusqu'à la convergence
  $\alpha = (r_j, Ar_j)/(Ap_j, Ap_j)$
  $x_{j+1} = x_j + \alpha p_j$
  $r_{j+1} = r_j - \alpha A p_j$
  $\beta_j = (r_{j+1}, Ar_{j+1})/(r_j, Ar_j)$
  $p_{j+1} = r_j + 1 - \beta p_j$
  Calculer $Ap_{j+1} = Ar_{j+1} + \beta_j Ap_j$
Fin Pour

FIG. 10 – *Méthode des résidus conjugués non préconditionnée ([12] page 183)*

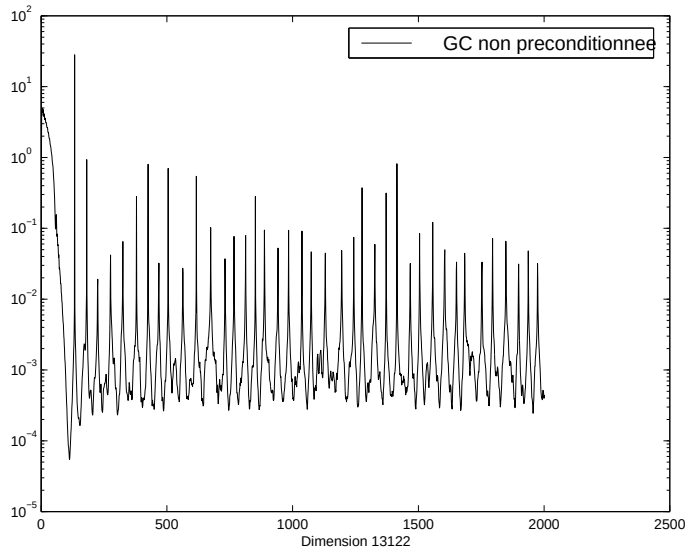

FIG. 11 – $\frac{|r|}{|b|}$ *pour GC non préconditionné, test* T_3.

$$A'_{GC} = A'_{CR} = \begin{bmatrix} A + \alpha I & C \\ C^t & \lambda.B \end{bmatrix} \qquad A'_{BICG} = \begin{bmatrix} A + \alpha I & C \\ -C^t & -\lambda.B \end{bmatrix}$$

ou

$$A''_{GC} = A''_{CR} = \begin{bmatrix} A + \alpha I & 0 \\ 0 & \lambda.B \end{bmatrix} \qquad A''_{BICG} = \begin{bmatrix} A + \alpha I & 0 \\ 0 & -\lambda.B \end{bmatrix}$$

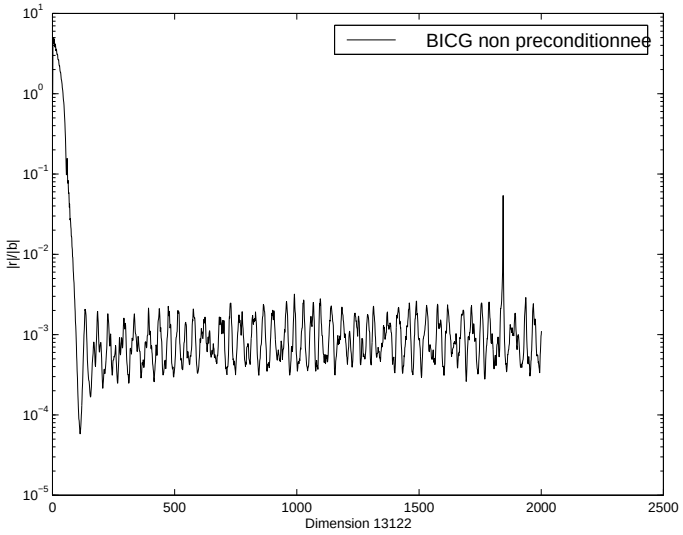

FIG. 12 – $\frac{|r|}{|b|}$ *pour BICG non préconditionné, test* T_3.

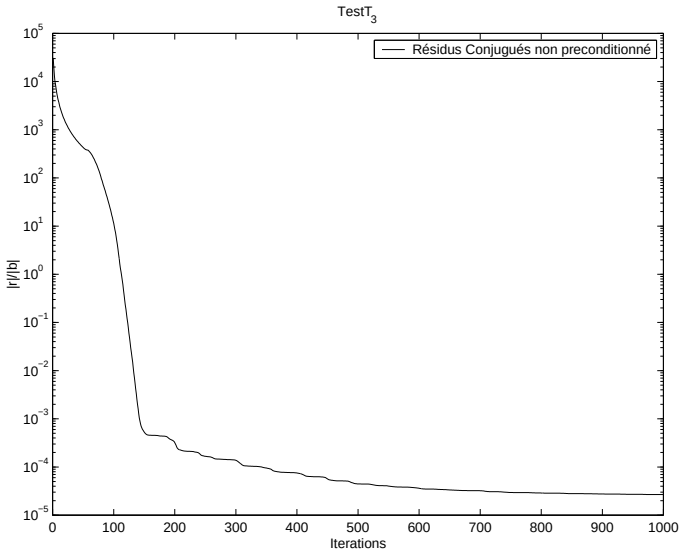

FIG. 13 – $\frac{|r|}{|b|}$ *pour RC non préconditionné, test* T_3.

accélère la convergence. Les essais des méthodes avec les deux types de préconditionnements sont rapportés dans les courbes des figures 14, 15, 16, 17 et 18. Comme la convergence est comparable entre les deux préconditionnements, nous optons pour les décompositions incomplètes de A''_{GC}, A''_{CR} et A''_{BICG} puisque la construction de ces précondutionnements est moins coûteuse. Ce choix correspond bien aux préconditionnements étudiés par Elman dans [3, 4]. Une étude de la sensibilité des valeurs propres de ces préconditionnements en fonction de la viscosité et du maillage est faite dans [4].

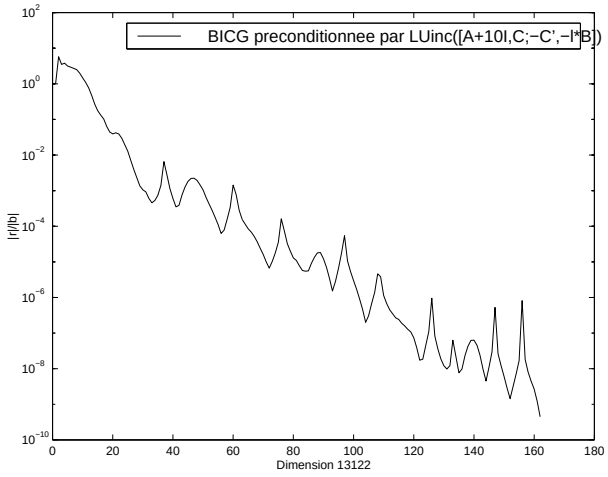

FIG. 14 – $\frac{|r|}{|b|}$ *pour le BICG préconditionné par la décomposition de* A'_{BICG}*, test* T_3.

6 Comparaison de l'efficacité des méthodes

Une étude comparative des coûts de la méthode de Gauss-Seidel par Bloc, Gradient Conjugué, Résidus conjugués et Gradient Bi-Conjugué peut se faire pour sélectionner la meilleure méthode pour la résolution du système (3).

6.1 Complexités des procédures mises en œuvre dans les méthodes

Gradient biConjugué : Une itération de BICG comporte :

- la résolution de deux systèmes creux pour le préconditionnement.
- 5 opérations SAXPY [2]
- 2 opérations SDOT [3]

2. opération du type $y = a \times x + y$ où x et y sont des vecteurs et a est un réel.
3. Produit scalaire.

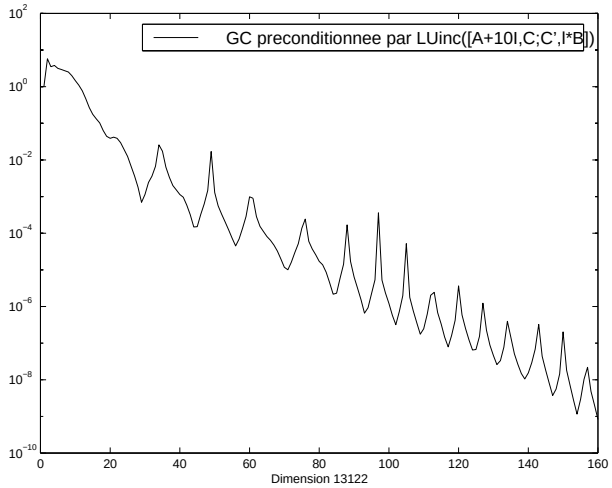

FIG. 15 – $\frac{|r|}{|b|}$ *pour le GC préconditionné par la décomposition de* A'_{GC}*, test* T_3*.*

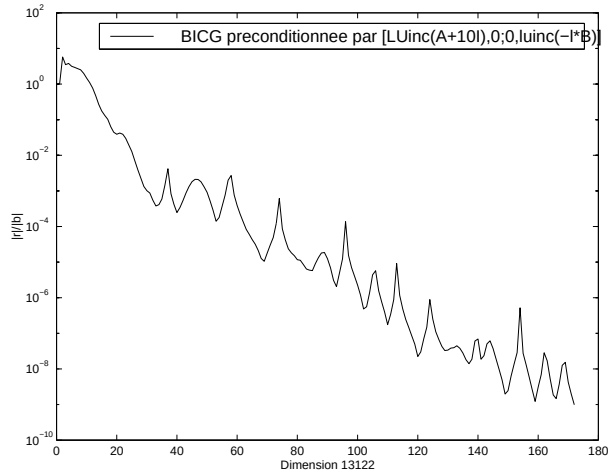

FIG. 16 – $\frac{|r|}{|b|}$ *pour BICG préconditionné par la décomposition de* A''_{BICG}*, test* T_3*.*

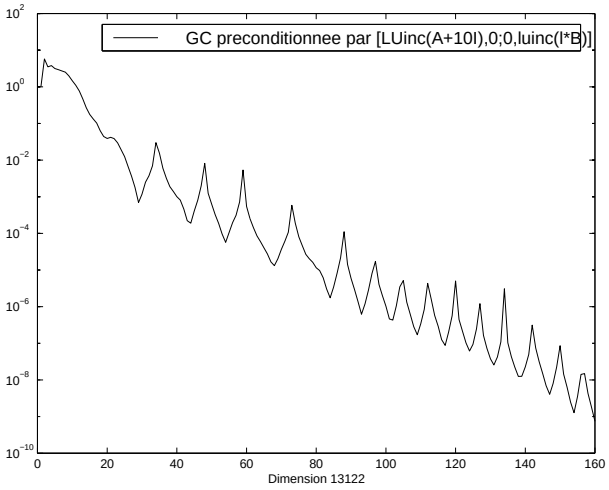

FIG. 17 – $\frac{|r|}{|b|}$ *pour le GC préconditionné par la décomposition de* A''_{GC}*, test* T_3*.*

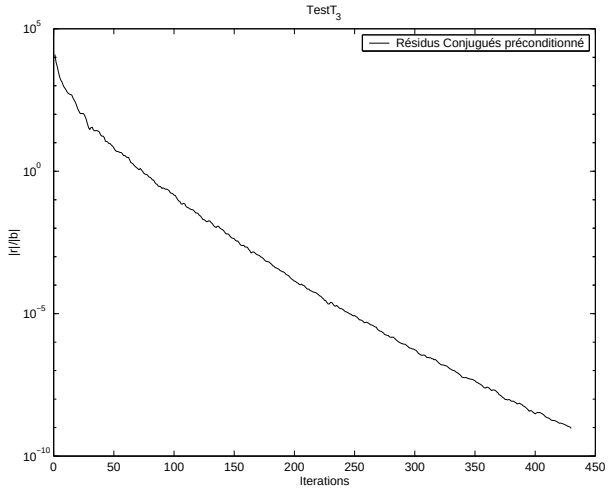

FIG. 18 – $\frac{|r|}{|b|}$ *pour la methode CR préconditionné par la décomposition de* A''_{CR}*, test* T_3*.*

– 2 multiplications avec la matrice creuse du système (dont une avec sa transposée).

Gradient Conjugué: Chaque itération comporte:

– la résolution d'un système creux pour le préconditionnement.
– 3 opérations SAXPY.
– 2 opérations SDOT.
– 1 multiplication avec la matrice creuse du système.

Résidus Conjugués: Chaque itération comporte:

– la résolution d'un système creux pour le préconditionnement.
– 4 opérations SAXPY.
– 2 opérations SDOT.
– 1 multiplication avec la matrice creuse du système.

Gauss Seidel par bloc : Dans chaque itération de la méthode Gauss Seidel par bloc, nous sommes amenés à résoudre par la méthode du gradient conjugué deux systèmes en A et B et à multiplier par les matrices C et C^t.

6.2 Coût total et comparaison des méthodes

Les préconditionnements choisis se basent sur une décomposition incomplète (LU ou Choleski) sans remplissage. Nous sommes donc amenés à résoudre des systèmes de la forme:

$$LUz = r$$

avec L triangulaire inférieure à diagonale unitaire et U triangulaire supérieure. On peut approcher le nombre d'opérations nécessaires pour une descente - remontée du système à $2N_z$ opérations, où N_z est le nombre d'éléments non nuls dans la matrice préconditionnée.

En supposant que les blocs A, B, C sont d'ordre n et respectivement de nombres d'éléments non nuls n_z^A, n_z^B, n_z^C, les complexités pour les méthodes sont données par les formules suivantes (I, I_A et I_B y représentent respectivement le nombre d'itérations de la méthode, le nombre total d'itérations pour la résolution des systèmes en A et le même nombre pour les systèmes en B) :

Gauss-Seidel par bloc (GSB): $4n_z^C I + (4n_z^A + 10n)I_A + (4n_z^B + 10n)I_B$.

Bi-gradient conjugué préconditionné (BICG) : $[8(n_z^A + n_z^B + n_z^C) + 28n]I$.

Gradient conjugué préconditionné (CG) : $[4(n_z^A + n_z^B + n_z^C) + 20n]I$.

Résidus conjugués préconditionné (CR) : $(24n + 4n_z^A + 4n_z^B + 4n_z^c)I$

(On a choisi comme indiqué précédemment le préconditionnement bloc diagonal pour les méthodes BICG, CG et CR.)

On applique maintenant ces formules au cas du test T_3 (système d'ordre $2n = 13122$). Les constantes sont donc : $n = 6561$, $n_z^A = 83361$, $n_z^B = 39879$, $n_z^C = 40018$, ce qui aboutit au résultat suivant :

Méthode	I	I_A	I_B	Nombre d'opérations (en millions)
GSB	2	186	70	90.3
BICG	173	-	-	257.8
CG	160	-	-	125.4
CR	430	-	-	348.5

Les méthodes BICG et CR sont nettement plus coûteuses que les deux autres. Cela s'explique par le fait que BICG ne tire pas bénéfice de la symétrie du système et que la convergence de la méthode des résidus conjugués est plus lente car le produit scalaire sur laquelle est basée la méthode est défini par le carré de la matrice au lieu de la matrice comme dans le cas du gradient conjugué. Les méthodes GSB et CG ont des performances plus proches avec cependant un net avantage à la première. Comme d'autre-part, nous avons vu que la méthode de Gauss-Seidel est assurée de converger rapidement dès que le pas de temps est assez petit et que par contre la méthode GC peut échouer puisque le système n'est pas défini, nous conseillons l'emploi de la première méthode.

7 Conclusion

Dans ce rapport, on a recherché un algorithme efficace pour résoudre un système linéaire issu de la discrétisation d'une équation de Navier-Stokes. Le système est partagé en quatre blocs carrés de même dimension. On y a mené une étude complète de la convergence des méthodes de Jacobi et Gauss-Seidel par blocs. Il apparaît que la convergence de l'itération de type Gauss-Seidel est asymptotiquement deux fois meilleure que sa correspondante de type Jacobi. Pour résoudre les systèmes linéaires de dimension moitié que ces résolutions entraînent, on montre qu'un préconditionnement défini par une factorisation incomplète de Choleski est efficace. La comparaison des algorithmes obtenus à ceux du gradient conjugué, des résidus conjugués ou du bi-gradient conjugué, algorithmes appliqués à la matrice entière avec un préconditionnement comparable à celui des méthodes de relaxations sur un problème test, est ensuite menée afin de déterminer une procédure efficace de résolution.

En conclusion, la méthode la plus rapide est la méthode de Gauss-Seidel combinée avec l'application du gradient conjugué préconditionné. La méthode du gradient conjugué préconditionné par un préconditionnement diagonal par blocs a aussi donné de bons résultats, mais cette approche a l'inconvénient de pouvoir échouer car on l'applique à un système non défini. La méthode du bi-gradient conjuqué préconditionné a donné de moins bons résultats puisqu'elle entraîne deux multiplications de matrices par itération au lieu d'une pour les autres méthodes. De même la méthode des résidus conjugués est coûteuse car sa convergence, bien que garantie, est plus lente que celle des autres méthodes.

Références

[1] R. Barrett, M. Berry, T. Chan, J.Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine et H. van der Vorst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*

[2] J. J. Dongarra, I. S. Duff, D. C. Sorensen, H. A. van der Vorst, *Numerical Linear Algebra for High-Performance Computers*, Software-Environnements-Tools, SIAM, 1998.

[3] H. Elman, D. Silvester et A. Wathen, Iterative Methods for Problems in Computational Fluid Dynamics, Report CS-TR-3675, UMIACS-TR-96-58, 1996.

[4] H. Elman, Perturbation of Eigenvalues of Preconditioned Navier-Stokes Operators, Report CS-TR-3559, UMIACS-TR-95-110, 1996.

[5] G. H. Golub et C. F. Van Loan - *Matrix Computations*, John Hopkins, Baltimore, 1st edition, 1983.

[6] K. Huebner *The Finite Element Method for Engineers*, Wiley-Interscience Publication, 1974.

[7] M. Amara, A. Chatti, F. Dabaghi *An optimal finite element for plane crack propagation problems*, Rapport de recherche INRIA RR-3379, Mars 1998.

[8] M. Amara, M. Benyounes, C. Bernardi *Error indicators for Navier-Stokes equations in stream function and vorticity formulation*, Numerisch Mathematik, 1998, 80,pp. 181-206.

[9] M. Amara, H. Barucq, M. Duloue *Une formulation mixte convergente pour le probleme de Stokes tridimensionnel*, Comptes Rendus de l'Académie des Sciences, t 328, Série I, 1999.

[10] M. Amara, C. Bernardi *Convergence of a finite element discretizatrion of the Navier Stokes equations in vorticity and stream function formulation*, M2AN Mathematical Modeling and Numerical Analysis.

[11] M. Amara, F. Dabaghi, *An optimal C^0 finite element method for the 2D biharmonic problem*, Inria Report, RR 3068, 1996.

[12] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, Boston, 1996.

[13] O. Pironneau, *Méthodes des éléments finis pour les fluides*, Masson, 1988.

[14] K. Boukir, Y Maday, B. Metivet et E. Razafindrakoto, *A high-order characteristics/finite element method for the incompressible Navier-Stokes equations*, International journal for numerical methods in fluids; ISSN 0271-2091; Coden IJNFDW; 1997; VOL. 25; NO. 12; PP. 1421-1454

[15] E.Durand, *Solutions Numériques des Equations Algébriques*, Tome II, Systèmes de plusieurs équations; Valeurs propres des matrices, Masson et Cie, 1961.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399