

# ECINADS-D2.2a: Algorithmes de Schwarz multi-niveau pour des applications au modèle de Navier-Stokes compressible

Carine Moussaed<sup>a</sup>, Hubert Alcin<sup>b</sup>, Bruno Koobus<sup>a</sup>, Alain Dervieux<sup>b</sup>

<sup>a</sup> *I3M, Université Montpellier 2, Place Eugène Bataillon,  
34095 Montpellier cedex, France*

<sup>b</sup> *INRIA, 2004 Route des Lucioles, 06902 Sophia-Antipolis, France*

## 1 Résumé

Dans ce travail, on étudie l'application d'algorithmes de Schwarz multi-niveau pour la résolution de systèmes linéaires issus de l'implicitation des équations de Navier-Stokes compressibles équipées de modèles de turbulence de type Simulation des Grandes Echelles (LES) et discrétisées sur des maillages non structurés. L'objectif est de construire un algorithme scalable et plus efficace que l'algorithme de Schwarz additif restreint qui est l'algorithme par défaut utilisé dans le logiciel de calcul d'écoulements fluides AIRONUM par les équipes de l'université Montpellier 2, de l'INRIA Sophia-Antipolis et de l'université de Pise. Le domaine de calcul est décomposé en sous-domaines, le solveur itératif de base est GMRES et la résolution des problèmes locaux dans chacun des sous-domaines repose sur la factorisation incomplète de type ILU. Une grille grossière est introduite pour accélérer la convergence du solveur linéaire.

Les méthodes de Déflation (DM) et de Domaine de Décomposition par Balancing (BDD) sont utilisées pour introduire la "correction grille grossière" comme préconditionneur. Les deux options sont comparées du point de vue efficacité. Les bases grossières sont construites avec des fonctions caractéristiques associées aux sous-domaines. Les performances en termes de scalabilité et d'efficacité de l'algorithme de Schwarz deux-niveau sont évaluées sur des calculs d'écoulements turbulents pour un nombre de cœurs variant de 64 à 2048. Une méthode de Schwarz trois-niveau est aussi présentée dans cette partie.

Cette étude a été réalisée dans le cadre de l'ANR ECINADS (*Écoulements instationnaires turbulents et adjoints par Simulation Numérique de Haute Performance*). Ces travaux ont d'autre part bénéficié d'un accès aux moyens de calcul du CINES au travers de l'allocation de ressources 2013- [c20132a5067] attribuée par GENCI (Grand Equipement National de Calcul Intensif).

## 2 Introduction

Les systèmes linéaires sont au cœur de nombreux problèmes du calcul scientifique. Dans le domaine de la CFD (Computational Fluid Dynamics) en particulier, on peut être amené à résoudre des systèmes linéaires creux de grande dimension n’ayant pas de structure particulière. Ces systèmes sont souvent très difficiles à résoudre et ils peuvent engendrer des coûts de calcul très importants, d’autant plus que les écoulements considérés sont compressibles, instationnaires, tridimensionnelles et discrétisés sur des maillages non structurés. Dans ce travail, nous proposons justement d’évaluer une stratégie d’algorithmes itératifs qui se veut efficace pour le modèle des équations de Navier-Stokes compressibles dans un contexte de discrétisation en maillages non structurés et dans un cadre d’utilisation d’ordinateurs parallèle à mémoire distribuée.

Pour la résolution des grands systèmes linéaires issus de la CFD, des méthodes itératives de type Krylov sont généralement utilisées. Afin d’accélérer la convergence de ces méthodes, le solution est de préconditionner le système linéaire, c’est à dire de remplacer le système  $Ax = b$  par un système, supposé plus facile à résoudre, de la forme  $M^{-1}Ax = M^{-1}b$  (préconditionnement à gauche) ou  $AM^{-1}y = b$  avec  $x = M^{-1}y$  (préconditionnement à droite). Le preconditionneur  $M^{-1}$  doit être peu coûteux quand on l’applique à un vecteur, et d’autre part il doit être suffisamment “proche” de l’inverse de  $A$ . L’une des grandes difficultés des méthodes de Krylov est la construction d’un bon preconditionneur  $M^{-1}$  pour la matrice  $A$ . Pour une revue des techniques de preconditionnement pour les grands systèmes linéaires, on peut citer par exemple le travail de M. Benzi [1]. Dans un contexte d’ordinateur séquentiel, les factorisations incomplètes du type ILU ont montré de bonnes propriétés d’efficacité et de robustesse [2, 3].

Dans un cadre d’ordinateur parallèle à mémoire distribuée, la mise en œuvre d’un bon preconditionneur est plus délicate et doit satisfaire à diverses contraintes dont sa capacité à être parallélisé lorsqu’on l’applique à un vecteur et sa capacité à “approcher” l’inverse de la matrice du système linéaire. Les preconditionneurs issus des techniques de décomposition de domaines pour la résolution des EDP (Equations aux Dérivées Partielles) sont celles qui permettent le mieux de réaliser ces contraintes. On montre par exemple dans [3] que ces preconditionneurs sont plus robustes que les factorisations incomplètes parallèles.

Les techniques de décomposition de domaine consistent à diviser le domaine d’étude du problème en sous-domaines où les solutions du problème seront plus facile à obtenir. Dans ce travail, les techniques de décomposition de domaine sont utilisées comme preconditionneur des méthodes de Krylov. L’efficacité des méthodes de décomposition de domaine peut dépendre du nombre de cœurs de calcul, ou de manière équivalente du nombre de sous-domaines. La question qui se pose est la mesure de cette dépendance. Au début du développement de l’informatique vectorielle et parallèle, la notion émergente était le speedup, obtenu par comparaison avec une machine scalaire, aussi appelée scalabilité forte ou scalabilité à taille fixe. Selon G. Amdahl l’accélération d’un programme à l’aide

de plusieurs cœurs de calcul est limitée par le temps nécessaire pour exécuter la fraction séquentielle du programme. Dans [4], la scalabilité forte est mesurée par comparaison entre le temps parallèle divisé par le temps le plus rapide obtenu pour l'exécution en série de l'algorithme. J.L. Gustafson a souligné en 1988 que les utilisateurs ne sont souvent pas intéressés à résoudre un problème fixe dans l'intervalle de temps le plus court possible, comme la loi d'Amdahl le décrit, mais plutôt dans la résolution du problème le plus grand possible dans un laps de temps raisonnable. Un programme passe à l'échelle ou est dit faiblement scalable si son comportement ne se dégrade pas lorsque la quantité de données à traiter augmente. La scalabilité faible ou à travail fixe par cœur de calcul mesure la façon dont le temps de résolution varie avec le nombre de cœurs de calcul pour une taille de problème par cœur fixe.

Deux familles d'algorithmes de décomposition de domaine sont généralement distinguées. Dans les méthodes de Schur l'ensemble des problèmes locaux permet de réduire les inconnues aux degrés de liberté d'interface. Les méthodes de Schur s'applique de manière très efficace aux systèmes elliptiques tels que ceux découlant de la Mécanique des Structures et sont donc également appliqués au modèle de Navier-Stokes incompressible. La deuxième famille, adoptée dans ce travail, concerne les méthodes de Schwarz additive (AS) où l'ensemble des problèmes locaux préconditionnent la résolution du problème global, formée d'une boucle mettant à jour la totalité des degrés de liberté. Les conditions aux limites pour chaque problème dans un sous-domaine sont récupérées dans les domaines voisins. Cela suppose un certain recouvrement, qui, lorsqu'il est réduit à zéro ramène l'algorithme à un simple bloc-Jacobi. L'intérêt de la méthode AS est qu'elle est parallèle en ce sens que la construction de la solution globale s'effectue de façon simultanée à partir des solutions obtenues dans les sous-domaines, contrairement à la variante multiplicative de l'algorithme de Schwarz qui est de nature séquentielle. Le solveur itératif global choisi est généralement une itération de type Krylov et est souvent désigné comme Newton-Krylov-Schwarz [5]. En contraste avec le cas incompressible, les écoulements compressibles sont quasi-exclusivement résolus par des méthodes de Schwarz. Plus précisément, l'algorithme de Schwarz additif restreint (RAS), initialement introduit pour les modèles elliptiques [21] et qui sera présenté ultérieurement dans ce manuscrit, a été étendu avec succès aux modèles d'écoulements compressibles [19, 22]. Les algorithmes de Schwarz et RAS se combinent bien avec différents preconditionneurs locaux. Dans [22] un preconditionneur local par factorisation incomplète ILU est utilisé. Dans [6], un Gauss-Seidel colorié est utilisé.

Cependant, les méthodes de Schwarz sont sujettes à un manque de scalabilité, ce qui se traduit par une dégradation de la vitesse de convergence quand le nombre de sous-domaines augmente. Il a été démontré par Brenner [8] que cet algorithme n'est pas scalable, à moins qu'une extension appelée grille grossière ne soit ajoutée. Dans [8], la "correction grille grossière" est calculée sur un maillage plus grossier, le maillage initial étant une division du maillage grossier (les deux maillages sont emboîtés). Une approche similaire a été appliquée dans [9] pour la CFD compressible. L'avantage de cette approche

emboîtée est de produire une solution du niveau grossier qui converge vers la solution en continu. Cependant l’option maillage grossier n’est pas pratique dans de nombreux cas, en particulier pour les maillages non structurés quelconques. En conséquence, de nombreux travaux ont ensuite exploré la possibilité de construire une base du niveau grossier à l’aide d’autres principes, puis d’en déduire algébriquement un système grossier. Une base du niveau grossier très simple est l’ensemble des fonctions caractéristiques des sous-domaines, qui est l’option choisie dans ce travail.

Les systèmes grossiers produisent des “corrections grille grossière” qui sont maintenant fréquemment introduits comme préconditionneurs de l’iteration globale, essentiellement par les méthodes de Déflation (DM) ou de Domaine de Décomposition par Balancing (BDD). Ces méthodes ont été respectivement introduites par Nicolaidis [10] et Mandel [11]. Dans [12] elles sont comparées et il est démontré qu’elles peuvent avoir des propriétés de convergence très similaires. Des bases grossières très simplifiées peut être utilisées, comme les fonctions caractéristiques des sous-domaines. En particulier, les méthodes DM et BDD permettent l’usage de bases grossières sans tenir compte de la convergence de la solution grille grossière vers la solution en continu. De nombreux travaux ont été réalisés sur ces techniques de préconditionnement dans le cadre d’EDP elliptiques et de matrices symétriques définies positives, voir par exemple [12, 13, 14, 15, 16, 17]. Nous proposons dans cette étude d’évaluer les performances en termes de scalabilité et d’efficacité des méthodes DM et BDD dans le cadre du modèle de Navier-Stokes compressible tridimensionnel et dans un contexte de discrétisation sur maillages non structurés.

Dans ce manuscrit, après avoir brièvement rappelé le cadre numérique de travail, nous décrivons dans la section 4 la méthode de Schwarz additif restreint telle qu’elle est mise en œuvre dans notre logiciel de calcul AIRONUM. Dans la section 4, nous présentons l’algorithme de Schwarz deux-niveau que nous avons développé et implémenté dans notre solveur d’écoulements compressibles. Deux options sont présentées qui utilisent les méthodes DM et BDD comme préconditionneur. La section 6 discute d’un troisième niveau de préconditionnement. La section suivante présente des applications dans un contexte de simulation d’écoulement compressible turbulent afin d’évaluer les performances de l’algorithme de Schwarz deux-niveau. Le manuscrit se termine en section 8 par une conclusion et des perspectives.

### 3 Schéma numérique

Dans cette étude, les méthodes DM et BDD ont été étendues à un code de calcul d’écoulements turbulents tridimensionnels compressibles en maillages non structurés. L’approximation spatiale repose sur une méthode mixte éléments finis/volumes finis dans laquelle un schéma décentré de dissipation d’ordre élevé basé sur des dérivées sixième est utilisé. Ce schéma est particulièrement bien adapté aux calculs de type LES de par le fait

que les effets de sa dissipation numérique se concentre plus principalement sur les échelles spatiales de plus hautes fréquences.

Dans l’option considérée dans ce travail, l’avancement en temps est implicite, basé sur un schéma aux différences finies du second ordre rétrograde que l’on peut écrire sous forme concise

$$F(W^{n+1}, W^n, W^{n-1}) = 0 \quad (1)$$

où  $W$  désigne la discrétisation à cinq composantes des inconnues  $(\rho, \rho \mathbf{u}, \rho E)$ ,  $\rho$  étant la masse volumique,  $\mathbf{u}$  le vecteur vitesse, et  $\rho E$  l’énergie totale par unité de volume,  $n + 1$ ,  $n$  et  $n - 1$  faisant référence aux temps discrets  $t^{n+1}$ ,  $t^n$  et  $t^{n-1}$ .

Ce système non-linéaire doit être résolu à chaque pas de temps afin d’obtenir  $W^{n+1}$ . Il est résolu par la méthode du défaut corrigé (quasi-Newton) qui s’écrit

$$A(W^{(\alpha+1)} - W^{(\alpha)}) = -F(W^{(\alpha)}, W^n, W^{n-1}) \quad (2)$$

dans lequel une matrice Jacobienne  $A$  simplifiée à l’ordre un spatial est utilisée, et où  $W^{(\alpha)}$  désigne l’itéré de la méthode du défaut corrigé. Le calcul de la matrice Jacobienne à l’ordre deux peut en effet s’avérer être une tâche très difficile, et celui à l’ordre un a de plus l’avantage de produire une matrice compacte où les coefficients non diagonaux ne concernent que les voisins directs.

Puisque le système (1) possède cinq champs inconnus,  $A$  est définie comme une matrice creuse bloc  $5 \times 5$ , de taille globale  $5 * ns \times 5 * ns$  où  $ns$  est le nombre de nœuds du maillage. Le Jacobien est construit à partir de la somme des discrétisations au premier ordre spatial des flux d’Euler linéarisés et d’une linéarisation des flux visqueux précise au second ordre. Typiquement, 2 iterations de la méthode du défaut corrigé sont appliquées à chaque pas de temps, chacune nécessitant la résolution d’un système linéaire. La performance de cet algorithme a été étudiée dans [7]. Il est à noter que la partie d’une simulation la plus consommatrice en temps CPU est la résolution des systèmes linéaires creux définis dans (2). Nous allons donc maintenant présenter les différents algorithmes de Schwarz utilisés dans ce travail pour résoudre ces systèmes linéaires.

## 4 Algorithmes de Schwarz additifs

Dans cette section, nous décrivons l’algorithme de Schwarz additif et sa variante restreinte que l’on utilise dans notre logiciel de calcul AIRONUM pour préconditionner l’algorithme de GMRES afin de résoudre le système linéaire (2).

Nous procédons dans un premier temps à une mise à l’échelle en transformant le système linéaire (2) à l’aide de la matrice diagonale par blocs  $D = BlockDiag(A)$ , constituée des blocs  $5 \times 5$  diagonaux extraits de  $A$ , comme suit:

$$D^{-1} A \delta W = D^{-1} f \quad (3)$$

où  $\delta W$  est l'incrément  $W^{(\alpha+1)} - W^{(\alpha)}$ .

On se propose dans cette étude de préconditionner à droite le système linéaire (3) afin de se ramener à la résolution d'un système linéaire, supposé plus facile :

$$(D^{-1}A)M^{-1}v = D^{-1}f,$$

avec  $\delta W = M^{-1}v$ ,  $M$  étant la matrice de préconditionnement qui est une matrice d'ordre  $5 * ns$  inversible. Un avantage de préconditionner à droite est que l'on conserve la même expression du résidu  $D^{-1}A\delta W - D^{-1}f$  que pour le système non-préconditionné.

Si l'on suppose que le domaine  $\Omega$  représentant l'ensemble des  $ns$  sommets  $x_k$  est décomposé en  $N$  sous-domaines  $\Omega_i$ , on peut alors définir les opérateurs de restriction :

$$R_i = \text{Diag}(B_k) \text{ pour } i = 1 \cdots N \text{ et } k = 1 \cdots ns$$

$$\text{où } B_k = I_{5 \times 5} \text{ si le sommet } x_k \in \Omega_i, \text{ et } B_k = 0_{5 \times 5} \text{ sinon}$$

à partir desquels on peut définir des systèmes locaux associés aux matrices :

$$(D^{-1}A)_i = R_i(D^{-1}A)R_i.$$

L'algorithme de Schwarz additif s'écrit alors en termes de préconditionnement de la manière suivante:

$$M^{-1} = M_{AS}^{-1} = \sum_{i=1}^N R_i(D^{-1}A)_i^{-1}R_i \quad (4)$$

où :

$$(D^{-1}A)_i^{-1}u = R_i((D^{-1}A)_i|_{\Omega_i})^{-1}u|_{\Omega_i}.$$

Dans l'algorithme de Schwarz additif précédent, le problème de Dirichlet local est résolu exactement sur chaque sous-domaine, ce qui peut s'avérer coûteux. Une alternative est d'utiliser un algorithme de Schwarz additif-ILU où cette solution exacte est remplacée par la solution approchée moins coûteuse obtenue sur chaque sous-domaine par une factorisation incomplète ILU [18] :

$$M^{-1} = M_{ASILU}^{-1} = \sum_{i=1}^2 R_i \text{ILU}((D^{-1}A)_i)^{-1}R_i. \quad (5)$$

Nous allons maintenant décrire l'option de décomposition du domaine  $\Omega$  choisie, ainsi que la variante de l'algorithme de Schwarz additif utilisé dans le logiciel AIRONUM.

La manière dont un algorithme de Schwarz additif va converger dépend fortement de la façon dont la décomposition de domaine est définie. Du point de vue complexité de l'algorithme, il est important que le recouvrement soit le plus petit possible. Mais pour

un problème donné, l'épaisseur de recouvrement peut avoir une influence importante sur la convergence itérative. D'autre part, dans l'analyse théorique de l'algorithme de Schwarz additif, la scalabilité n'est établie que si l'épaisseur de recouvrement ne diminue pas lorsque le nombre de nœuds augmente [8].

L'option de décomposition choisie est celle, souvent désignée comme décomposition minimale, introduite dans [19, 21]. Les différents sous-domaines se recouvrent sur une rangée de sommets comme montré sur la figure 1.

Sur chaque sommet de  $\Omega_i \cap \Omega_j$ , les préconditionneurs locaux produisent un correcteur. Dans l'algorithme de Schwarz additif restreint de [21, 22], l'ensemble des sommets communs à plusieurs  $\Omega_i$  est à nouveau partitionné :

$$\bigcup_{1 \leq i \leq N, 1 \leq j \leq N} (\Omega_i \cap \Omega_j) = \bigcup_{1 \leq i \leq N} \hat{\Omega}_i$$

où les  $\hat{\Omega}_i$  sont disjoints, et  $\hat{\Omega}_i \subset \Omega_i$ . Seul le préconditionneur local  $((D^{-1}A)_i)^{-1}$  du sous-domaine  $\Omega_i$  met à jour tous les nœuds de  $\hat{\Omega}_i$ . En d'autres termes, si on définit :

$$\Omega_{i,0} = \{x_j \in \Omega_i, \forall k \neq i, x_j \notin \Omega_k\} \cup \hat{\Omega}_i$$

alors :

$$(R_i^0 u)_j = u_j \text{ si } u_j \text{ est rattaché à un sommet de } \Omega_{i,0}, \quad 0 \text{ sinon.} \quad (6)$$

Le préconditionneur de Schwarz additif restreint s'écrit alors, dans le cas d'une résolution exacte des problèmes locaux :

$$M^{-1} = M_{RAS}^{-1} = \sum_{i=1}^N R_i^0 (D^{-1}A)_i^{-1} R_i.$$

et dans le cas d'une solution approchée des problèmes locaux par factorisation ILU :

$$M^{-1} = M_{RASILU}^{-1} = \sum_{i=1}^N R_i^0 ILU((D^{-1}A)_i)^{-1} R_i.$$

Grâce au remplacement d'un des opérateurs de restriction  $R_i$  par sa version restreinte  $R_i^0$ , le produit des résidus par le préconditionneur peut être calculé localement sans aucune communication supplémentaire. Il a été également observé que l'algorithme de Schwarz additif restreint a généralement un meilleur conditionnement et une meilleure convergence que l'algorithme de Schwarz additif. Le lecteur peut aussi consulter une analyse proposée dans [23].

Le préconditionneur  $M^{-1}$  peut être introduit dans une boucle de résolution quasi-Newton. Dans ce travail, le préconditionnement est introduit dans l'algorithme de GMRES [20] qui est une méthode itérative très souvent utilisée en mécanique des fluides numérique où les matrices ne sont pas symétriques.

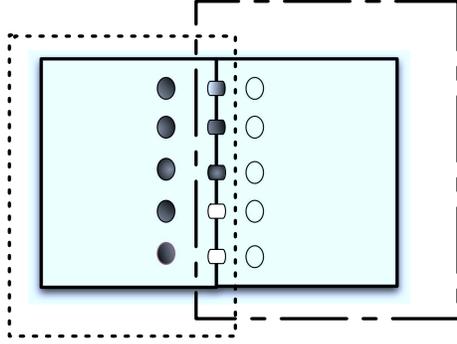


Figure 1: *Décomposition de domaine minimale [21]: les matrices des solveurs locaux s’appliquent sur des ensembles de sommets délimités par les pointillés sur la figure. Dans le cas de RAS, la mise à jour dans le domaine 1 est restreinte aux sommets noirs, la mise à jour dans le domaine 2 est restreinte aux sommets blancs. Le domaine 1 est délimité par des pointillés (...), le domaine 2 est délimité par des tirets (- -).*

## 5 Algorithme de Schwarz deux-niveau

Nous présentons dans cette section l’algorithme de Schwarz deux-niveau avec “correction grille grossière” par les méthodes DM et BDD qui préconditionne l’algorithme de GMRES.

Dans notre approche deux-niveau, la grille grossière est définie à partir des fonctions de base caractéristiques  $\Phi_i$ ,  $i = 1 \cdots N$ , associés aux sous-domaines  $\Omega_i$  :

$$\phi_i(x_j) = 1 \text{ si } x_j \in \Omega_i, 0 \text{ sinon.}$$

Nous montrerons dans la partie *Applications* la bonne propriété de consistance de la grille grossière pour une base de fonctions caractéristiques.

Décrivons maintenant comment DM et BDD sont introduits dans la solution de (3). A noter que les préconditionneurs correspondant sont utilisés comme préconditionneurs à droite, le residu restant donc inchangé.

Commençons par définir les opérateurs de base associés à la partition de  $\Omega$  qui vont nous permettre ensuite de préciser notre méthode deux-niveau basée sur DM ou BDD :

- Notons  $Z$  l’opérateur de prolongement représenté sous forme matricielle par :

$$Z_{ij} = \Phi_j(x_i) \text{ avec } i = 1 \cdots ns \text{ et } j = 1 \cdots N$$

où  $ns$  et  $N$  sont respectivement le nombre de nœuds  $x_i$  et le nombre de sous-domaines  $\Omega_j$  du domaine  $\Omega$ .

- Notons  $E$  la matrice grille grossière définie par :

$$E = Z^t(D^{-1}A)Z .$$

- Notons  $P$  et  $Q$  les opérateurs de projections définis par :

$$\begin{aligned} P &= I - (D^{-1}A)ZE^{-1}Z^t \\ Q &= I - ZE^{-1}Z^t(D^{-1}A) \end{aligned}$$

On peut alors définir le préconditionnement de Schwarz additif restreint avec DM et BDD par :

- Préconditionnement de Schwarz additif restreint-ILU + DM :  
Avec ce préconditionnement, la méthode de résolution itérative, GMRES dans notre cas, résout le système préconditionné suivant :

$$(D^{-1}A)QM_{RASILU}^{-1}v = PD^{-1}f$$

de sorte que la solution  $\delta W$  de (3) est finalement donnée par

$$\begin{aligned} \delta W &= ZE^{-1}Z^tD^{-1}f + Q\delta\tilde{W}. \\ \text{où } \delta\tilde{W} &= (M_{RASILU}^{-1})v \end{aligned}$$

- Préconditionnement de Schwarz additif restreint-ILU + BDD :  
Ce préconditionnement est défini par :

$$P_B = ZE^{-1}Z^t + QM_{RASILU}^{-1}P$$

de sorte que la méthode itérative résout

$$(D^{-1}A)P_Bv = D^{-1}f .$$

La solution  $\delta W$  de (3) est alors obtenue par  $\delta W = P_Bv$ .

## 6 Algorithme de Schwarz trois-niveau

### 6.1 Motivation

Avec un solveur non exacte (utilisation d'un algorithme de type ILU par exemple), la convergence de la méthode deux-niveau peut se dégrader lorsque le nombre de nœuds par sous-domaine devient grand. C'est une configuration souvent rencontrée dans les

simulations parallèles d'écoulement de fluide, qui est l'objectif de nos développements de solveurs linéaires, pour lesquelles le nombre de nœuds par sous-domaines est très vite de l'ordre de plusieurs dizaines de milliers. Pour remédier à ce problème, une solution consiste à ajouter une grille intermédiaire construite à partir des sous-domaines. L'idée est de partitionner chaque sous-domaine, avec un nombre de partitions de quelques dizaines, et d'utiliser BDD ou DM sur la grille grossière intermédiaire. Nous décrivons ci-après cet algorithme 3 niveaux dans le cas où la méthode BDD est mise en œuvre sur la grille grossière et le préconditionneur de Schwarz additif restreint est utilisé.

Il est à noter que les performances de cet algorithme trois-niveau seront évaluées dans une étude ultérieure, la phase d'implémentation dans le logiciel AIRONUM n'étant pas encore finalisée.

## 6.2 Algorithme

Dans un premier temps, commençons par définir les opérateurs de base associés à chaque sous-domaine  $\Omega_i$  partitionnant le domaine initial  $\Omega$  et qui vont nous permettre ensuite de préciser la méthode de Schwarz trois-niveau :

- Partitionnement des sous-domaines  $\Omega_i$

$$\Omega_i = \bigcup_{k=1}^{N_i} \Omega_{ik} \text{ avec } \bigcap_{k=1}^{N_i} \Omega_{ik} = \emptyset$$

où  $N_i$  représente le nombre de sous-partitions  $\Omega_{ik}$  de  $\Omega_i$ .

- Notons  $\Phi_{ik}$  les fonctions de base associées à chaque sous-partition  $\Omega_{ik}$ . Dans le cas des fonctions caractéristiques, on a alors pour  $x_j \in \Omega_i$ ,  $\Phi_{ik}(x_j) = 1$  si  $x_j \in \Omega_{ik}$ , 0 sinon.
- Notons  $Z_i$  l'opérateur de prolongement associé au sous-domaine  $\Omega_i$  : cet opérateur est défini par  $(Z_i)_{jk} = \Phi_{ik}(x_j)$  où  $i = 1 \cdots N$ ,  $j = 1 \cdots NS_i$ , et  $k = 1 \cdots N_i$ , avec  $N$  le nombre de sous-domaines partitionnant  $\Omega$  et  $NS_i$  le nombre de nœuds par sous-domaines  $\Omega_i$ .
- Notons  $E_i$  la matrice grossière intermédiaire associée à chaque sous-domaine  $\Omega_i$  définie par :  

$$E_i = Z_i^T (D^{-1}A)|_{\Omega_i} Z_i .$$
- Notons  $P_i$  et  $Q_i$  les opérateurs de projection associés au sous-domaine  $\Omega_i$  définis par:  

$$P_i = I|_{\Omega_i} - (D^{-1}A)|_{\Omega_i} Z_i E_i^{-1} Z_i^T \quad \text{et} \quad Q_i = I|_{\Omega_i} - Z_i E_i^{-1} Z_i^T (D^{-1}A)|_{\Omega_i} .$$

L'algorithme de Schwarz additif restreint trois-niveau avec factorisation ILU et la méthode BDD pour la grille grossière (DM aurait pu être aussi considérée) consiste alors, pour trouver une approximation de la solution du système linéaire (3), à résoudre par la méthode itérative choisie, GMRES par exemple, le système linéaire preconditionné à droite :

$$(D^{-1}A)P_B v = D^{-1}f .$$

Dans cette expression, le préconditionneur  $P_B$  est défini par

$$P_B = ZE^{-1}Z^T + QM_{RASILU-3l}^{-1}P$$

avec  $M_{RASILU-3l}^{-1} = \sum_{i=1}^N R_i^0 (D^{-1}A)_I^{-1} R_i$  où  $(D^{-1}A)_i^{-1}$  est remplacé par une inversion approximative associant la grille grossière intermédiaire et BDD :

$$(D^{-1}A)_i^{-1} \approx Z_i E_i^{-1} Z_i^T + Q_i ILU((D^{-1}A)_i)^{-1} P_i .$$

La solution  $\delta W$  de (3) est alors obtenue par  $\delta W = P_B v$ .

On rappelle que dans le cas de Schwarz additif restreint deux-niveau avec factorisation ILU,  $(D^{-1}A)_i^{-1}$  est remplacé par l'inversion approximative  $ILU((D^{-1}A)_i)^{-1}$ .

## 7 Applications

Afin d'évaluer les performances de l'algorithme de Schwarz additif restreint deux-niveau avec factorisation ILU, nous considérons un écoulement turbulent 3D compressible autour d'un cylindre à section circulaire (figure 2). Le nombre de Reynolds est de 20.000, le nombre de Mach est fixé à 0.1, et le calcul est réalisé en appliquant une approche de simulation des grandes structures en formulation variationnelle multi-échelle [24] avec comme modèle de sous-maille celui de Smagorinsky. Deux maillages sont considérés dans cette étude de performance : un premier maillage composé de 9.4 millions d'éléments avec un partitionnement allant de 64 à 1024 sous-domaines, et un second maillage constitué de 23.3 millions d'éléments avec une décomposition variant de 128 à 2048 sous-domaines. Dans nos simulations parallèles, chaque sous-domaine est attribué à un cœur de calcul.

### 7.1 Maillage du cylindre circulaire à 9.3M éléments

Dans un premier temps, nous vérifions que la base grossière caractéristique *i.e.* construite à partir des fonctions caractéristiques des sous-domaines est un choix raisonnable. Dans ce but, la consistance de la grille grossière caractéristique est évaluée en introduisant une solution manufacturée  $\phi$  dans le système linéarisé. La fonction  $\phi$  est quadratique sur

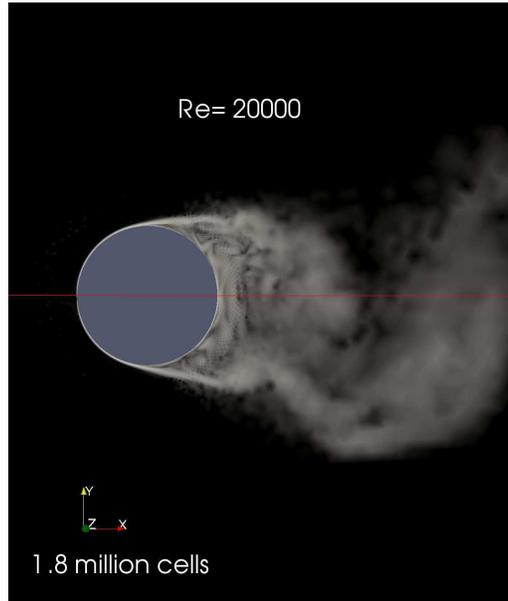


Figure 2: *Simulation des grandes structures par l'approche variationnelle multi-échelle de l'écoulement turbulent autour d'un cylindre circulaire, à Reynolds 20000, et avec un maillage de 9.4 millions d'éléments; champ de vorticit .*

chacune de ses cinq composantes. Le second membre du syst me lin aire est choisi de telle sorte que sa solution est exactement  $\phi$ .

Le syst me grossier est d fini   partir des 64 sous-domaines de la partition. La solution du syst me grossier pour le second membre choisi est montr e en figure 3. Comme l'illustre cette figure, elle est en bon accord avec la solution manufactur e quadratique.

Nous allons maintenant  valuer les performances de l'algorithme original de Schwarz additif restreint et de sa version deux-niveau. Nous utiliserons pour cela la notion de *scalabilit  forte*, qui dans un cas id al dit que le temps de calcul est divis  par deux avec deux fois plus de processeurs. Nous appellerons *facteur de scalabilit *, le rapport  $n_2/n_1$  entre le nombre  $n_2$  d'it rations pour converger sur  $2N$  sous-domaines et le nombre  $n_1$  d'it rations pour converger du m me facteur sur  $N$  sous-domaines.

**Performance de l'algorithme original de Schwarz additif restreint.** La scalabilit  forte de la version originale de Schwarz additif restreint est d'abord test e. Ce type d'algorithme est connu comme n' tant pas trop loin de la scalabilit , voir une discussion sur ce point dans [25]. Dans la figure 4 et le tableau 1, nous observons que pour un CFL de 100 la convergence peut se d grader quand le nombre de processeurs augmente avec un nombre d'it rations 37% plus grand lorsque le nombre de sous-domaine double, une perte de scalabilit  qui est mesur e dans le tableau 2. Cependant, ce chiffre est bien meilleur pour un CFL de 1000, comme on peut le constater dans le tableau 3. Pour 1024 pro-

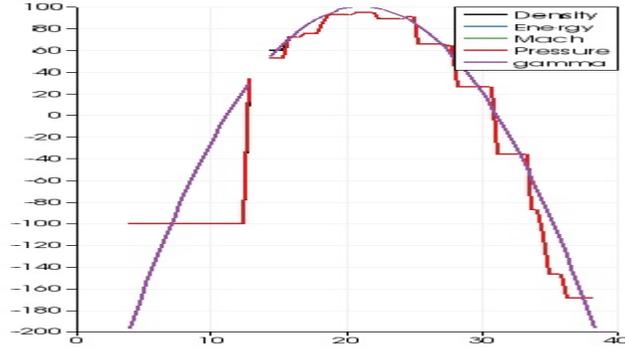


Figure 3: *Evaluation de la consistance de la grille grossière pour une base de fonctions caractéristiques : comparaison d’une solution manufacturée quadratique avec son approximation sur la grille grossière. Coupe horizontales de la solution fine (courbe lisse) et de la solution grossière (en escalier).*

cesseurs, toutefois, la figure 5 montre que la convergence est a) plus lente et b) présente une pente qui varie entre le début des itérations et leur fin. En particulier, le facteur de convergence pour le mode le plus lent de l’erreur itérative correspond à 130 itérations pour une réduction du résidu d’une décade, comme montré en figure 5.

Table 1: *Simulation des Grandes Structures turbulentes autour d’un cylindre: CFL=100, maillage à 9.4 millions d’éléments. Residu  $\leq 10^{-8}$ . Scalabilité de 64 cœurs à 1024 cœurs.*

Type de préconditionneur	CFL	64 procs # it.	128 procs # it.	256 procs # it.	1024 procs # it.
RAS	100	138	190	230	222
Déflation-RAS	100	83	88	87	70
Balancing-RAS	100	80	81	78	

**Performance de la version deux-niveau.** La pente de la courbe de convergence est plus proche d’être constante, et à un CFL de 1000 sur 1024 cœurs, le facteur de convergence du mode le plus lent de l’erreur correspond à 26 itérations pour une réduction du résidu d’une décade. Ce chiffre est 5 fois meilleur que pour l’algorithme de Schwarz additif restreint original. De plus, les figures 4 et les tableaux 1,2 et 3, montrent que la scalabilité est meilleure que l’unité.

Avec les équations de Navier-Stokes, le système grossier comporte cinq fois plus d’inconnues que de sous-domaines. Pour 1024 sous-domaines, on est donc amené à résoudre exactement un problème à 5120 inconnues.

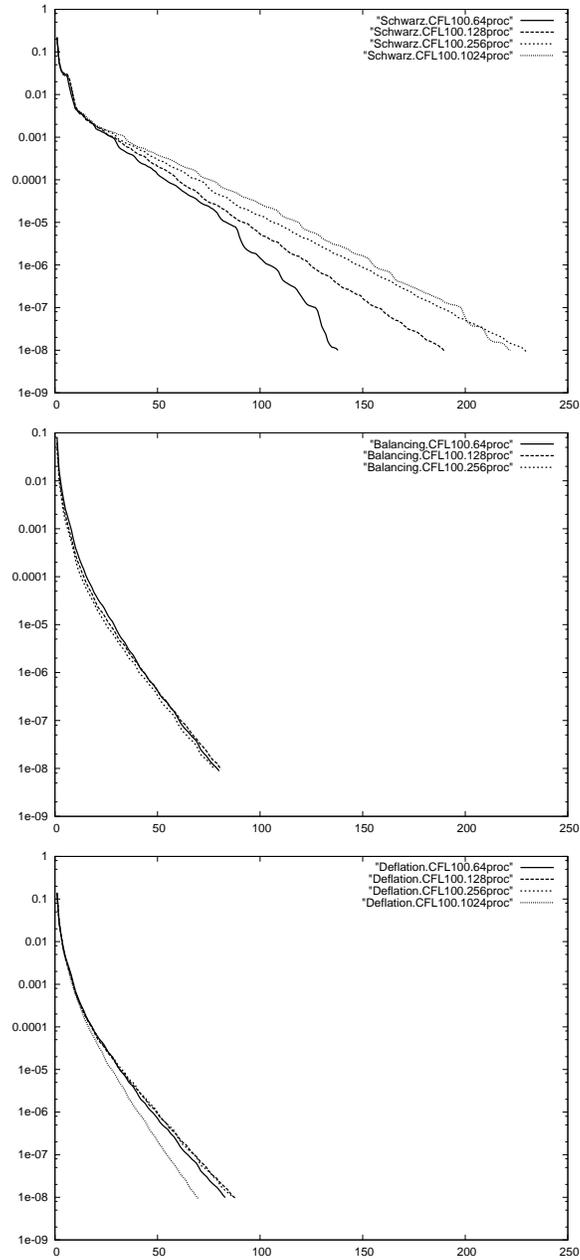


Figure 4: *Simulation des Grandes Structures turbulentes autour d'un cylindre, maillage à 9.4 millions d'éléments: convergence un pas de temps (CFL=100) de RAS, RAS avec Déflation, RAS avec Balancing. Residus en fonction des itérations.*

Table 2: *Simulation des Grandes Structures turbulentes autour d'un cylindre: CFL=100, maillage à 9.4 millions d'éléments. Residu  $\leq 10^{-8}$ . Une mesure du facteur de scalabilité.*

Type de préconditionneur	CFL	128 procs # it.	256 procs # it.	Sca. Factor
RAS	100	190	230	1.21
Déflation-RAS	100	88	87	.988
Balancing-RAS	100	81	78	.963

Table 3: *Simulation des Grandes Structures turbulentes autour d'un cylindre: CFL=1000, maillage à 9.4 millions d'éléments. Residu  $\leq 10^{-8}$ . Une mesure du facteur de scalabilité.*

Type de préconditionneur	CFL	64 procs # it.	1024 procs # it.	Sca. Factor
RAS	1000	473	693	1.10
Déflation-RAS	1000	272	163	0.88

Afin d'obtenir un algorithme deux-niveau efficace en termes de CPU, nous utilisons la bibliothèque MUMPS [26, 27] pour résoudre le système algébrique grossier. MUMPS contient un puissant solveur direct multi-frontal parallèle comportant une phase de factorisation et une phase de résolution. Des détails sur cette bibliothèque sont donnés sur le site web de MUMPS [27].

Pour des raisons pratiques de comparaison de temps CPU, une simulation instationnaire de l'écoulement turbulent autour d'un cylindre circulaire est réalisée en utilisant 1024 cœurs sur la plateforme de calcul parallèle SGI Altix ICE 8200 du CINES (France). Le nombre de CFL est fixé à 1000, calculé sur les contraintes eulérienne et visqueuses portant sur le pas de temps. A chaque pas de temps, on effectue deux itérations de la méthode du défaut-corrigé au cours desquelles le système linéaire est résolu jusqu'à ce que le résidu soit diminué d'un ordre de grandeur pour le mode le plus lent de l'erreur (26 itérations dans le cas Déflation, 125 dans le cas RAS). D'autre part, la matrice grille grossière est gelée dans le préconditionneur pendant 10 pas de temps consécutifs.

Dans le tableau 4, nous donnons les temps CPU correspondant à 10 itérations en temps et à des nombres différents de cœurs attribués à MUMPS au travers l'utilisation d'un communicateur MPI spécifique. Les valeurs indiquées correspondent aux temps CPU cumulés de résolution des systèmes linéaires, ces temps comprenant l'assemblage de la matrice grille grossière dans le cas de l'algorithme de Schwarz additif restreint avec déflation.

Par rapport à la formulation originale de Schwarz additif restreint, une meilleure performance de l'algorithme deux-niveau par déflation est observée lorsque les phases MUMPS

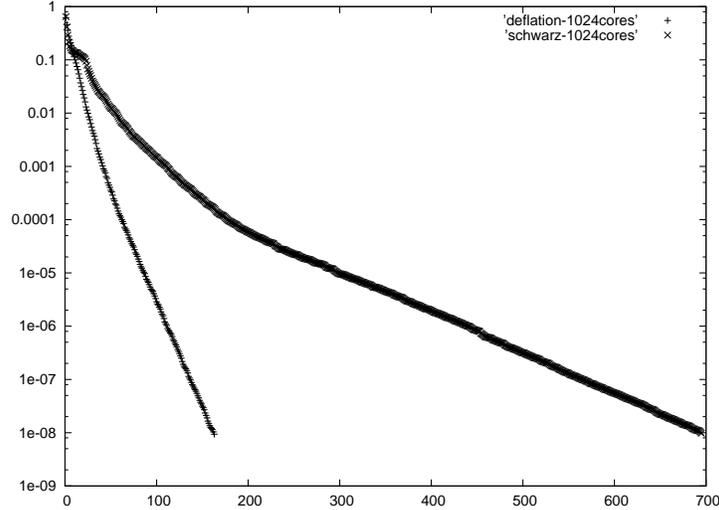


Figure 5: *Simulation des Grandes Structures turbulentes autour d'un cylindre, maillage à 9.4 millions d'éléments: convergence durant un pas de temps (CFL=1000) de RAS (700 it.) et de RAS avec Déflation (160 it.). Résidu en fonction des itérations. 1024 processors.*

ne sont pas exécutées sur l'intégralité des 1024 cœurs utilisés pour la simulation. Les mesures de CPU montrent que les meilleurs choix sont entre 5 et 20 cœurs. Le coût le plus bas du préconditionneur deux-niveaux est obtenu avec 10 cœurs avec un temps CPU global environ deux fois plus faible que pour la méthode de Schwarz additif restreint originale. Au contraire, lorsque le même communicateur MPI est utilisé pour MUMPS et le solveur de l'écoulement (la façon la plus directe de mettre en œuvre MUMPS), *ie* lorsque 1024 cœurs sont aussi dédiés à MUMPS, les performances se dégradent fortement produisant un temps CPU bien supérieur à celui du solveur linéaire d'origine. En effet, avec un trop grand nombre de cœurs pour résoudre le système grille grossière à 5120 inconnues, les temps de communication de MUMPS explosent et détériorent l'efficacité globale.

Dans les temps CPU donnés dans le tableau 4, l'assemblage de la matrice grille grossière représente 4.50 *sec.* seulement. Ceci a été rendu possible en effectuant un important travail d'optimisation en termes d'implémentation. Dans la première version du logiciel, l'assemblage de la matrice grille grossière  $E = Z^t(D^{-1}A)Z$  était réalisé en calculant, à chaque itération d'une boucle sur un indice  $j = 1 \dots 5 \times N$  où  $N$  est le nombre de sous-domaines, le produit matrice-vecteur  $Ee_j$  où  $e_j$  est le  $j$ -ième vecteur de la base canonique. Chaque évaluation de  $Ee_j$ , *ie.* la colonne  $j$  de  $E$ , était réalisée à l'aide d'appels à trois routines de base respectivement dédiées à la multiplication d'un vecteur générique par les matrices  $Z$ ,  $D^{-1}A$  et  $Z^t$ . Cela s'est traduit par une énorme quantité d'opérations arithmétiques et surtout de communications puisque la multiplication d'un vecteur par la matrice  $A$  exige des communications entre sous-domaines voisins, et le produit d'un vecteur par la matrice de restriction  $Z^t$  génère l'évaluation de sommes globales MPI. Au

Table 4: *Simulation des Grandes Structures turbulentes autour d'un cylindre, utilisant 1024 cœurs: temps CPU pour 10 pas de temps implicites. Maillage à 9.4 millions d'éléments.*

Type de préconditionneur	Temps CPU	MUMPS factorisation	MUMPS phase de résolution
RAS	41.95 sec.		
Déflation-RAS, MUMPS/2 cœurs	26.32 sec.	0.93 sec.	5.54 sec.
Déflation-RAS, MUMPS/10 cœurs	23.72 sec.	0.93 sec.	3.06 sec.
Deflation-RAS, MUMPS/100 cœurs	32.53 sec.	0.83 sec.	11.42 sec.
Déflation-RAS, MUMPS/1024 cœurs	249.62 sec.	2.11 sec.	226.70 sec.
Balancing-RAS, MUMPS/10 cœurs	43.94 sec.	0.93 sec.	8.10 sec.

total, on a mesuré un temps CPU supérieur à 60 *sec.* pour l'assemblage de la matrice grille grossière. Afin de réduire considérablement ce coût, les options suivantes de mise en œuvre ont été considérées :

- dans le calcul de  $D^{-1}A(Ze_j)$  pour  $j = 1 \dots 5 \times N$ , seules les composantes non nulles du vecteur  $Ze_j$  sont prises en compte, et les sous-domaines qui ne contribuent pas à ce produit matrice-vecteur pour l'indice  $j$  sont écartés de ce calcul,
- toutes les communications entre sous-domaines voisins intervenant dans le calcul du produit matrice-vecteur  $D^{-1}A(Ze_j)$ , qui usuellement demanderaient des MPI Send/Receive, sont maintenant prises en compte dans les sommes globales MPI utilisées pour calculer  $Z^t(D^{-1}AZe_j)$ ,
- au lieu d'exécuter 5 sommes globales MPI (correspondant aux 5 inconnues par sous-domaine) sur des vecteurs à  $N$  composantes dans chaque boucle  $j = 1 \dots 5 \times N$ , ces sommes globales sont extraites de la boucle et seulement 5 sommes globales MPI de  $5 \times N \times N$  composantes sont exécutées. Cette option demande bien sûr plus d'espace mémoire mais celui-ci n'est alloué que temporairement, et de plus la plupart des plateformes parallèles sont maintenant équipées de cœurs disposant d'une mémoire de plusieurs gigabytes. Notons que ces 5 sommes globales MPI sont les seules communications réalisées dans la version optimisée de l'assemblage de la matrice grille grossière.

Dans la dernière ligne du tableau 4, nous présentons la meilleure performance obtenue

avec la version BDD pour ce banc d'essai. Bien que la convergence soit un peu meilleure que pour DM, l'efficacité est décevante, pas supérieure à la version originale, en raison du coût de construction du préconditionneur BDD.

## 7.2 Maillage du cylindre circulaire à 23.3M éléments

Dans ce paragraphe, le même calcul turbulent que dans la section précédente est considéré, *i.e.* même modèle numérique, même modèle de turbulence et même configuration d'écoulement de cylindre, mais cette fois le maillage est plus fin et contient 23.3 millions d'éléments. Le nombre de CFL est fixé à 1000, et le nombre de cœurs pour le calcul de l'écoulement varie entre 128 et 2048.

Suite à l'étude précédente, nous avons choisi de sélectionner l'algorithme deux-niveau par déflation et nous avons attribué 10 cœurs au solveur MUMPS afin d'évaluer et de comparer les performances réalisées sur ce maillage par la méthode de Schwarz additif restreint originale et sa variante deux-niveau.

**Performance de l'algorithme original de Schwarz additif restreint.** La scalabilité forte de la version originale de Schwarz additif restreint est évalué dans un premier temps. Dans la figure 6 et le tableau 5, nous observons que la convergence se dégrade quand le nombre de processeurs augmente. Un nombre d'itérations 39% plus grand peut être atteint lorsque le nombre de sous-domaines double. Cette perte de scalabilité est mesurée dans le tableau 6. Sur la figure 6, on peut aussi observer que la convergence présente une pente qui varie entre le début des itérations et leur fin. En particulier, dans le cas du calcul sur 2048 cœurs, le facteur de convergence pour le mode le plus lent de l'erreur itérative correspond à 441 itérations pour une réduction du résidu d'une décade.

**Performance de la version deux-niveau.** Sur la figure 6, on peut observer que la pente de la courbe de convergence pour la méthode deux-niveau par déflation est plus proche d'être constante, et sur 2048 cœurs, le facteur de convergence du mode le plus lent de l'erreur correspond à 36 itérations pour une réduction du résidu d'une décade. Ce chiffre est 12 fois meilleur que pour l'algorithme de Schwarz additif restreint original. De plus, les tableaux 5 et 6 montrent que le nombre d'itérations de Schwarz-déflation est entre 5 et 9 fois plus faible comparativement à l'algorithme original de Schwarz additif restreint et que la scalabilité est meilleure que l'unité pour le préconditionneur deux-niveau. Sur le tableau 7, on peut noter que pour 2048 cœurs le coût en termes de temps CPU total (assemblage de la matrice grille grossière inclus pour l'option déflation) est 2.5 fois plus faible avec l'algorithme deux-niveau, un résultat aussi observé pour les autres nombres de cœurs. Il est aussi à remarquer que sur le temps CPU requis par le préconditionneur par déflation sur 2048 cœurs, 20% concerne l'assemblage de la matrice grille grossière, les

temps CPU utilisés pour les phases de factorisation et de résolution du solveur MUMPS étant limités respectivement à 1% et 2% du temps CPU total. Il est à noter que ce temps CPU d’assemblage de la matrice grille grossière se réduit à 5% du temps CPU total dans le cas de 128 sous-domaines pour lequel le système grossier comporte 640 inconnues au lieu de 10240 dans le cas 2048 cœurs. On peut aussi remarquer, qu’en raison du coût du préconditionneur de déflation, une itération Schwarz-déflation est 3 fois plus chère qu’une itération Schwarz sur cet exemple d’application.

Table 5: *Simulation des Grandes Structures turbulentes autour d’un cylindre: maillage à 23.3 millions d’éléments. Residu  $\leq 10^{-8}$ . Scalabilité de 128 cœurs à 2048 cœurs*

Type de préconditionneur	128 procs # it	256 procs # it.	512 procs # it.	1024 procs # it.	2048 procs # it.
RAS	1006	1059	1352	1359	1891
Déflation-RAS	234	266	237	231	200

Table 6: *Simulation des Grandes Structures turbulentes autour d’un cylindre: maillage à 23.3 millions d’éléments. Residu  $\leq 10^{-8}$ . Une mesure du facteur de scalabilité.*

Type de préconditionneur	128 procs # it.	2048 procs # it.	Sca. Factor
RAS	1006	1891	1.17
Déflation-RAS	234	200	0.96

Table 7: *Simulation des Grandes Structures turbulentes autour d’un cylindre, utilisant 2048 cœurs : temps CPU. Residu  $\leq 10^{-8}$ . Maillage à 23.3 millions d’éléments.*

Type de préconditionneur	Temps CPU	MUMPS factorisation	MUMPS phase de résolution	Assemblage matrice grille grossière
RAS	260.70 sec.			
Déflation-RAS, MUMPS/10 cœurs	109.22 sec.	1.16 sec.	2.29 sec.	22.00 sec.

## 8 Conclusion et perspectives

Dans ce travail, nous avons présenté pour le modèle de Navier-Stokes compressible une stratégie de préconditionnement de la méthode de GMRES qui combine l’algorithme de Schwarz additif restreint avec la méthode de Déflation ou celle de Domaine de Décomposition

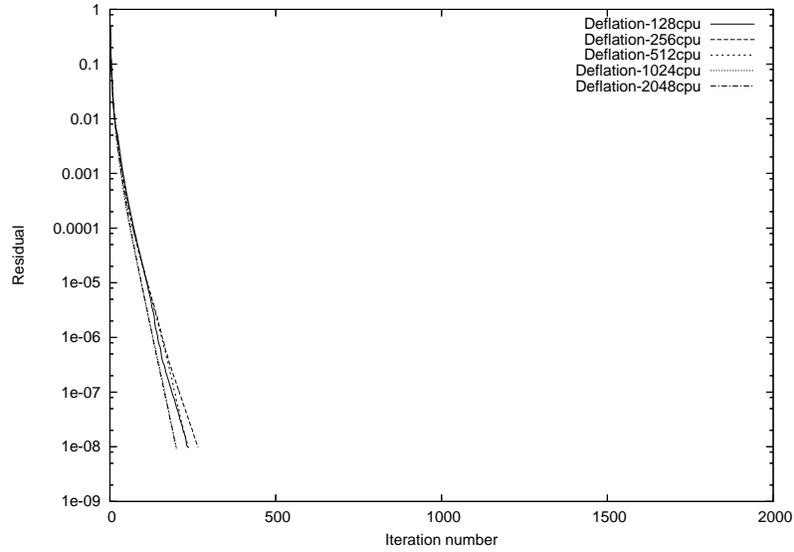
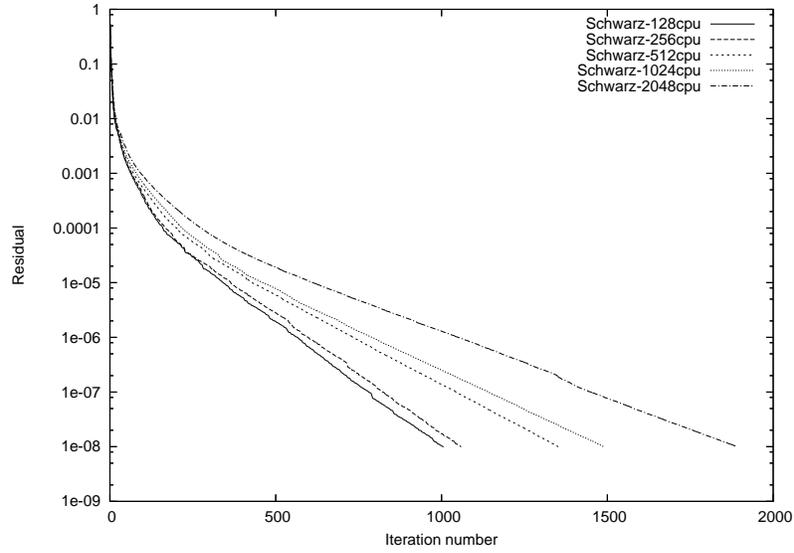


Figure 6: *Simulation des Grandes Structures turbulentes autour d'un cylindre, maillage à 23.3 millions d'éléments : convergence du solveur linéaire sur un pas de temps ( $CFL=1000$ ) avec RAS (en haut), et RAS-Déflation (en bas). Residus en fonction des itérations.*

par Balancing. Ces techniques de préconditionnement introduisent une “correction grille grossière” permettant d’améliorer la scalabilité de l’algorithme de Schwarz de base. Nous avons aussi présenté un algorithme de Schwarz trois-niveau qui doit permettre d’améliorer la convergence de l’algorithme deux-niveau (qui utilise un solveur local non exacte de type ILU) dans le cas où le nombre de nœuds par sous-domaine devient grand.

Des applications de l’algorithme de Schwarz deux-niveau à un écoulement compressible instationnaire ont montré les bonnes propriétés de convergence et de scalabilité de cette technique de préconditionnement. Le facteur de scalabilité de la méthode deux-niveau s’est révélé inférieur à 1, alors que pour l’algorithme de Schwarz additif restreint la convergence s’est dégradée avec l’augmentation du nombre de cœurs. Le facteur d’amélioration de la convergence apportée par la méthode deux-niveau s’est d’autre part avéré important avec un facteur pouvant atteindre 9 pour un nombre de cœurs de calcul de 2048. Bien que la version BDD soit de convergence légèrement plus rapide, l’option la plus efficace en termes de temps CPU est la méthode DM (un facteur 2 a été observé en faveur de DM), le préconditionneur BDD étant d’un coût de construction supérieur. Par rapport à l’algorithme de Schwarz additif restreint original, on peut aussi remarquer dans les cas test considérés que le coût CPU de la méthode deux-niveau avec déflation est jusqu’à 2.5 fois plus faible pour la résolution des systèmes linéaires résultant de l’implication des équations de Navier-Stokes compressibles. Ce résultat a pu être obtenu grâce à un certain nombre d’optimisations réalisées dans l’implémentation de l’assemblage de la matrice grille grossière qui peut sinon s’avérer très (voir prohibitivement) coûteux lorsqu’un grand nombre de cœurs de calcul est utilisé, ainsi qu’à l’utilisation d’une bibliothèque parallèle de résolution directe (MUMPS, [27]) pour une résolution efficace des systèmes algébriques grossiers. Il est aussi à remarquer que l’option la plus efficace n’a pas été d’attribuer le même nombre de cœurs au solveur fluide et à la bibliothèque parallèle de résolution directe, qui est l’option la plus naturelle et la plus facile à mettre en œuvre, mais d’attribuer seulement une dizaine de cœurs à cette bibliothèque afin de limiter le coût des communications qui augmente très vite avec le nombre de cœurs de calcul.

Enfin, en termes de perspectives, nous projetons d’évaluer les performances de l’algorithme trois-niveau pour le même modèle de Navier-Stokes compressible, et d’estimer ainsi l’apport que peut procurer cet algorithme par rapport à la variante deux-niveau, en particulier dans les configurations où le nombre de nœuds par sous-domaine est important.

## References

- [1] M. Benzi, *Preconditioning Techniques for Large Linear Systems: A Survey*, Journal of Computational Physics, 182, 418-477 (2002).
- [2] Y. Saad, *Iterative methods for sparse linear systems*, Society for Industrial and Applied Mathematics, SIAM, 2003.

- [3] H.A. Van Der Vorst, *Iterative Krylov Methods for Large Linear Systems*, Cambridge University Press, 2003.
- [4] J.L. Hennessy and D.A. Patterson, *Computer architecture: A quantitative Approach (5th edition)*, Morgan Kaufmann, 2011.
- [5] X.C. Cai, W.D. Gropp, D.E. Keyes, and M.D. Tidriri, *Newton-Krylov-Schwarz methods in CFD*, in F. Hebeker and R. Rannacher, editors, Proceedings of the International Workshop on Numerical Methods for the Navier-Stokes Equations, Notes in Numerical Fluid Dynamics, volume 47, pages 17-30, Vieweg Verlag, 1994.
- [6] R. Walters, E. Nielsen, W.K. Anderson and D. Keyes, *Application of Newton-Krylov methodology to a three dimensional unstructured Euler code*, in AIAA 95-1733-CP, June, 1995.
- [7] B. Koobus, S. Camarri, M.V. Salvetti, S. Wornom and A. Dervieux, *Parallel Simulation of Three-Dimensional Complex Flows: Application to turbulent wakes and two-phase compressible flows*, Advances in Engineering Software, Vol. 38, pp 328-337, 2007.
- [8] S.C. Brenner, *Two-level additive Schwarz preconditioners for plate elements*, Wuhan university of Natural Sciences, 1:658-667, 1996.
- [9] R. Aitbayev, X.-C. Cai, and M. Paraschivoiu, *Parallel two-level methods for three-dimensional transonic compressible flow simulations on unstructured meshes*, in J. Périaux, D.E. Keyes, A. Ecer and N. Satofuka, editors, Proceedings of Parallel CFD'99, Williamsburg, Virginia, May 23-26, 1999. Elsevier, 2000.
- [10] R.A. Nicolaides, *Deflation of conjugate gradients with applications to boundary value problem*, SIAM J. Numer. Anal., Vol. 24, pp. 355-365, 1987.
- [11] J. Mandel, *Balancing domain decomposition*, Communications in Numerical Methods in Engineering, 9:233-241, 1993.
- [12] R. Nabben and C. Vuik, *A comparison of deflation and the balancing preconditioner*, SIAM J. Sci. Comput., Vol. 27, No5, pp. 1742-1759, 2006.
- [13] R. Nabben and C. Vuik, *A comparison of abstract versions of deflation, balancing and additive coarse grid correction preconditioners*, Numer. Linear Algebra Appl., 15:355-372, 2008.
- [14] J.M. Tang, R. Nabben, C. Vuik, Y.A. Erlangga, *Comparison of Two-Level Preconditioners Derived from Deflation, Domain Decomposition and Multigrid Methods*, J Sci Comput (2009), 39:340-370.

- [15] J. Frank and C. Vuik, *On the construction of deflation-based preconditioners*, SIAM J. Sci. Comput., Vol. 23, No2, pp. 442-462, 2001.
- [16] J.M.Tang, S.P. Maclachlan, R. Nabben and C. Vuik, *A comparison of two-level preconditioners based on multigrid and deflation*, SIAM J. Matrix Anal. Appl., Vol.31, No4, pp. 1715-1739, 2010.
- [17] R. Nabben and C. Vuik, *A comparison of deflation and coarse grid correction applied to porous media flow*, SIAM J. Numer. Anal., Vol. 42, No4, pp. 1631-1647, 2004..
- [18] H.L. Stone, *Iterative solution of implicit approximations of multidimensional partial differential equations*, SIAM Journal of Numerical Analysis 5 (3):530-538 (1968).apport synonyme
- [19] X.-C. Cai, C. Farhat, and M. Sarkis. A minimum overlap restricted additive schwarz preconditioner and applications in 3D flow simulations. In C. Farhat J. Mandel and AMS X.-C. Cai, eds, editors, *The Tenth International Conference on Domain Decomposition Methods for Partial Differential Equations, August 10-14, 1997 Boulder, Colorado*, volume 218. Contemporary Mathematics, 1998.
- [20] Y. Saad et M.H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Stat. Comput., 7:856-869, 1986.
- [21] X.-C. Cai and M. Sarkis. A restricted additive schwarz preconditioner for general sparse linear systems. *SIAM J. Sci. Comput.*, 21:792–797, 1999.
- [22] M. Sarkis and B. Koobus. A scaled and minimum overlap restricted additive schwarz method with application on aerodynamics. *Computer Methods in Applied Mechanics and Engineering*, 184:391–400, 2000.
- [23] A. Frommer and D.B. Szyld. An algebraic convergence theory for restricted additive schwarz methods using weighted max norms. *SIAM J. Numer. Anal.*, 39:2:463–479, 2001.
- [24] B. Koobus and C. Farhat, *A Variational Multiscale Method for the Large Eddy Simulation of Compressible Turbulent Flows on Unstructured Meshes - Application to Vortex Shedding*, Computer Methods in Applied Mechanics and Engineering, Vol. 193, pp. 1367-1383, 2004
- [25] D.E. Keyes, *How scalable is domain decomposition in practice?*, In C.-H. Lai et al., editor, Proceedings of the 11th International Conference on Domain Decomposition Methods, London, 1998. DDM.org, Augsburg, 1999.
- [26] I.S. Duff P.R. Amestoy and J.-Y. L'Excellent. Multifrontal parallel distributed symmetric and unsymmetric solvers. *Comput. Methods in Appl. Mech. Eng.*, 184:501–520, 2000.

[27] <http://graal.ens-lyon.fr/MUMPS>.