Calcul haute performance pour la simulation d'interactions fluide-structure

Vincent Partimbene

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :

Institut National Polytechnique de Toulouse (Toulouse INP)

Discipline ou spécialité :

Mathématiques Appliquées

Présentée et soutenue par :

M. VINCENT PARTIMBENE

le mercredi 25 avril 2018

Titre :

Calcul haute performance pour la simulation d'interactions fluide-structure

---

Ecole doctorale :

Mathématiques, Informatique, Télécommunications de Toulouse (MITT)

Unité de recherche :

Institut de Recherche en Informatique de Toulouse (I.R.I.T.)

Directeur(s) de Thèse :

M. PIERRE SPITERI

M. PHILIPPE MARTHON

Rapporteurs :

M. DAMIEN TROMEUR-DERVOUT, UNIVERSITE LYON 1

M. FRÉDÉRIC MAGOULES, CENTRALESUPELEC GIF SUR YVETTE

Membre(s) du jury :

M. SERGE GRATTON, INP TOULOUSE, Président

M. LEON RATSIFANDRIHANA, SEGULA TECHNOLOGIES, Membre

Mme EMMANUELLE JACQUET, UNIVERSITE DE FRANCHE COMTE, Membre

M. PIERRE SPITERI, INP TOULOUSE, Membre

M. RAPHAËL COUTURIER, UNIVERSITE DE FRANCHE COMTE, Membre

M. THIERRY GARCIA, UNIVERSITE PARIS-SACLAY, Membre

TABLE DES MATIÈRES

TABLE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ALGORITHMES

LISTE DES SYMBOLES

Fluide

$\mathbf{q}$	Vecteur flux de chaleur	W
$\mathbf{u}^*$	Vecteur vitesse de grille	$\mathrm{m\,s^{-1}}$
$\mathbf{u}$	Vecteur vitesse	$\mathrm{m\,s^{-1}}$
$\mathcal{D}$	Fonction de dissipation	
μ^f	Viscosité dynamique	$\mathrm{Pa\,s}$
ν^f	Viscosité cinématique	$\mathrm{m^2\,s^{-1}}$
ρ^f	Masse volumique du fluide	$\mathrm{kg\,m^{-3}}$
τ_{ij}	Contraintes visqueuses	N
c	Chaleur spécifique	$\mathrm{J\,kg^{-1}\,K^{-1}}$
E^f	Énergie du fluide	J
E_{int}	Énergie interne	J
k	Conductivité thermique	$\mathrm{W\,m^{-1}\,K^{-1}}$
S_i	Sources	
T	Température	K

Structure

λ	Premier coefficient de Lamé	Pa
$\mathbb{E}$	Tenseur des déformations de Green-Lagrange	
$\mathbf{a}$	Vecteur accélération	$\mathrm{m\,s^{-2}}$
$\mathbf{D}$	Vecteur déplacement	m
$\mathbf{F}$	Vecteur somme des forces	N
$\mathbf{f}^s$	Vecteur des forces externes	N
μ	Second coefficient de Lamé ou module de cisaillement	Pa
ν	Coefficient de Poisson	
ρ^s	Masse volumique du solide	$\mathrm{kg\,m^{-3}}$
σ	Tenseur des contraintes	
ε	Tenseur des déformations linéarisé	
C_0	Configuration initiale	
C_t	Configuration courante	
E	Module de Young	Pa
V	Parallélépipède élémentaire	$\mathrm{m^3}$

Remerciements

Au terme de ces trois années passées au sein de l'Institut de Recherche en Informatique de Toulouse (IRIT) et de SEGULA Technologies, je tiens tout d'abord à remercier les personnes qui ont rendu cette aventure possible. Léon RATSIFANDRIHANA, qui a été mon responsable scientifique industriel, mais surtout sans qui ce projet n'aurait pas vu le jour. Je le remercie vivement pour avoir cru en moi et pour nous avoir permis de monter ce projet.

Je tiens évidemment à remercier sincèrement mes directeurs de thèse, Pierre SPITERI et Philippe MARTHON, qui ont accepté d'assurer la direction scientifique de ces travaux. Leurs compétences, leurs expertises et leurs rigueurs scientifiques m'ont beaucoup apporté. Je les remercie de leur encadrement et des conseils avisés qu'ils ont su me prodiguer tout au long de ces trois années.

J'adresse également mes vifs remerciements à Thierry GARCIA, Maître de Conférences à l'Université de Paris-Saclay-UVSQ, pour son aide et sa contribution essentielles à ces travaux de recherche. Ses compétences en informatiques et ses qualités pédagogiques m'ont été très précieuses et m'ont permis de découvrir de nouvelles nouvelles notions en programmation parallèle.

Je suis très reconnaissant à Frédéric MAGOULÈS, Professeur à CentraleSupélec, et Damien TROMEUR-DERVOUT, Professeur à l'Université Lyon 1, pour m'avoir fait l'honneur d'accepter d'être rapporteurs de cette thèse. Je remercie également Raphaël COUTURIER, Professeur à l'Université de Franche-Comté, et Emmanuelle JACQUET, Maître de Conférences à l'Université de Franche-Comté, d'avoir participé au jury ainsi que Serge GRATTON, Professeur à l'INP Toulouse, pour l'avoir présidé.

Je souhaite exprimer ma gratitude à tout le personnel de l'IRIT et plus spécialement au secrétariat, Sylvie, Annabelle, Isabelle et Muriel, qui ont toujours été disponibles, sympathiques et très efficaces. Je remercie également le service informatique du laboratoire pour leur disponibilité et leur aide.

Merci à tous mes collègues de SEGULA Technologies et particulièrement aux autres doctorants Bruno, Nicolas et Antoine. Je remercie également les autres doctorants du laboratoire que j'ai eu le plaisir de côtoyer durant ces trois années : Abdelghafour, Halim, Mohanad, Nora, etc.

Je tiens finalement à remercier chaleureusement ma famille, dont mes parents et ma sœur, pour leurs soutien et encouragements et sans qui rien n'aurait été possible. Je remercie également ma tante pour avoir accepté de relire mon manuscrit.

Résumé

Cette thèse aborde la résolution des problèmes d'interaction fluide-structure par un algorithme consistant en un couplage entre deux solveurs : un pour le fluide et un pour la structure. Pour assurer la cohérence entre les maillages fluide et structure, on considère également une discrétisation de chaque domaine par volumes finis. En raison des difficultés de décomposition du domaine en sous-domaines, nous considérons pour chaque environnement un algorithme parallèle de multi-splitting (ou multi-décomposition) qui correspond à une présentation unifiée des méthodes de sous-domaines avec ou sans recouvrement. Cette méthode combine plusieurs applications de points fixes contractantes et nous montrons que, sous des hypothèses appropriées, chaque application de points fixes est contractante dans des espaces de dimensions finies normés par des normes hilbertiennes et non-hilbertiennes. De plus, nous montrons qu'une telle étude est valable pour les résolutions parallèles synchrones et plus généralement asynchrones de grands systèmes linéaires apparaissant lors de la discrétisation des problèmes d'interaction fluide-structure et peut être étendue au cas où le déplacement de la structure est soumis à des contraintes. Par ailleurs, nous pouvons également considérer l'analyse de la convergence de ces méthodes de multi-splitting parallèles asynchrones par des techniques d'ordre partiel, lié au principe du maximum discret, aussi bien dans le cadre linéaire que dans celui obtenu lorsque les déplacements de la structure sont soumis à des contraintes. Nous réalisons des simulations parallèles pour divers cas test fluide-structure sur différents clusters, en considérant des communications bloquantes et non bloquantes. Dans ce dernier cas nous avons eu à résoudre une difficulté d'implémentation dans la mesure où une erreur irrécupérable survenait lors de l'exécution ; cette difficulté a été levée par introduction d'une méthode assurant la terminaison de toutes les communications non bloquantes avant la mise à jour du maillage. Les performances des simulations parallèles sont présentées et analysées. Enfin, nous appliquons la méthodologie présentée précédemment à divers contextes d'interaction fluide-structure de type industriel sur des maillages non structurés, ce qui constitue une difficulté supplémentaire.

Mots clés :

- interaction fluide-structure ;
- multi-splitting ;
- volumes finis ;
- algorithmes parallèles ;
- message passing interface.

Abstract

This thesis deals with the solution of fluid-structure interaction problems by an algorithm consisting in the coupling between two solvers: one for the fluid and one for the structure. In order to ensure the consistency between fluid and structure meshes, we also consider a discretization of each domain by finite volumes. Due to the difficulties of decomposing the domain into sub-domains, we consider a parallel multi-splitting algorithm for each environment which represents a unified presentation of sub-domain methods with or without overlapping. This method combines several contracting fixed point mappings and we show that, under appropriate assumptions, each fixed point mapping is contracting in finite dimensional spaces normalized by Hilbertian and non-Hilbertian norms. In addition, we show that such a study is valid for synchronous parallel solutions and more generally asynchronous of large linear systems arising from the discretization of fluid-structure interaction problems and can be extended to cases where the displacement of the structure is subject to constraints. Moreover, we can also consider the analysis of the convergence of these asynchronous parallel multi-splitting methods by partial ordering techniques, linked to the discrete maximum principle, both in the linear frame and in the one obtained when the structure's displacements are subjected to constraints. We carry out parallel simulations for various fluid-structure test cases on different clusters considering blocking and non-blocking communications. In the latter case, we had to solve an implementation problem due to the fact that an unrecoverable error occurred during execution; this issue has been overcome by introducing a method to ensure the termination of all non-blocking communications prior to the mesh update. Performances of parallel simulations are presented ans analyzed. Finally, we apply the methodology presented above to various fluid-structure interaction cases on unstructured meshes, which represents an additional difficulty.

Key words:

- fluid-structure interaction;
- multi-splitting;
- finite volumes;
- parallel algorithms;
- message passing interface.

CHAPITRE

1

INTRODUCTION GÉNÉRALE

Les interactions fluide-structure (FSI) interviennent dans de nombreux domaines tels que l'aéronautique, l'automobile, la biomécanique, la construction navale, l'industrie ferroviaire, le génie civil, etc. Il s'agit de prédire avec précision les effets aérodynamiques ou hydrodynamiques sur une structure soumise à un écoulement fluide, ainsi que les couplages éventuels qui peuvent s'avérer destructifs pour la structure.

Le procédé d'interaction fluide-structure a été modélisé de façon mathématique par de nombreux auteurs [10, 15, 16, 26, 39, 44, 68], ce qui conduit à l'élaboration d'un modèle couplé comprenant d'une part les équations de Navier-Stokes qui permettent de décrire le comportement du fluide et d'autre part l'équation de Navier modélisant le comportement de la structure mécanique. Ces modèles sont complétés par des conditions initiales et des conditions aux limites décrivant les phénomènes physiques intervenant sur les bords des domaines Ω^f et Ω^s respectivement pour le fluide et la structure. L'interaction entre le fluide et la structure est modélisé par la prise en compte des valeurs de pression et de contrainte de cisaillement à l'interface fluide-structure pour la structure, constituant ainsi pour l'équation de Navier, une partie des conditions aux limites. Pour le fluide, la déformation de la structure représente une évolution du domaine où sont définies les équations de Navier-Stokes.

Cette interaction fluide-structure a pour effet de déformer la structure. Lors d'une résolution numérique du problème, cette déformation de la structure aura pour effet de déformer le maillage lié à la discrétisation du domaine Ω^s où repose la structure, ces déplacements permettent de déterminer le nouveau maillage. Pour prendre en compte cette modification de Ω^s, il sera donc nécessaire d'adapter le maillage de Ω^f. Cette nécessité d'adaptation de maillage constitue une difficulté supplémentaire ; habituellement les uti-

lisateurs discrétisent le domaine Ω^s par éléments finis et le domaine Ω^f par volumes finis avec l'inconvénient que le raccordement des deux maillages introduit des erreurs d'approximation dues aux extrapolations nécessaires des grandeurs calculées. Pour éviter cet inconvénient nous proposons dans ce travail une discrétisation globale des deux domaines Ω^f et Ω^s par la méthode des volumes finis, avec pour conséquence une cohérence dans la numérotation des points de grille.

Par ailleurs, les équations de Navier-Stokes et de Navier sont complexes à résoudre, en particulier du fait que le modèle mathématique décrivant le comportement du fluide est fortement non-linéaire. De nombreuses méthodes numériques ont été élaborées pour résoudre cette dernière ; en particulier la méthode SIMPLE (i.e. *Semi-Implicit Method for Pressure Linked Equations*) proposée par PATANKAR et SPALDING [67]. Elle a ensuite été améliorée par ce dernier ce qui a conduit à la méthode SIMPLER (i.e. *SIMPLE Revised*) [66]. Cependant, pour les problèmes instationnaires, ces dernières méthodes ont été notablement améliorées par ISSA [41] qui a proposé la méthode PISO (i.e. *Pressure-Implicit with Splitting of Operator*). Cette dernière correspond à un schéma de prédiction-correction en temps comportant un pas prédicteur permettant de donner une première approximation des composantes de vitesse à partir de la valeur obtenue au pas précédent de la pression, puis à l'aide de deux pas correcteurs, de déterminer la pression et d'améliorer les valeurs des vitesses obtenues au pas prédicteur. Ces étapes de prédiction-correction amènent à résoudre de systèmes linéaires de grandes dimensions. Il est à noter que, lorsque Ω^f est un domaine tri-dimensionnel, le pas prédicteur conduit à la résolution de trois systèmes linéaires découplés, propriété très intéressante dans l'optique d'une parallélisation de cette partie de la méthode, alors que la résolution des deux pas correcteurs conduit à résoudre deux systèmes linéaires mal conditionnés pour déterminer les corrections de pression ainsi que des équations de récurrence pour améliorer les valeurs des composantes de vitesse.

L'équation de Navier est apparemment plus simple à résoudre ; cependant une résolution implicite conduit à des instabilités numériques qui ont pour effet de faire diverger le traitement. Pour remédier à cet inconvénient, JASAK et al. [42] ont proposé un algorithme semi-implicite qui permet d'équilibrer le découpage car les termes explicites contiennent plus d'informations que leurs équivalents implicites. Ceci a pour effet d'améliorer grandement la résolution de l'équation régissant la structure.

Aussi bien les systèmes linéaires provenant de la mise en œuvre de la méthode PISO que ceux découlant de la résolution numérique de l'équation de Navier ont un caractère creux. De plus, si on souhaite obtenir une bonne précision des valeurs approchées, il est nécessaire de considérer des maillages très fins. Ainsi, la résolution du problème d'interaction fluide-structure conduit à chaque pas de temps à résoudre six systèmes linéaires creux de très grandes dimensions. Compte tenu de ces deux dernières caractéristiques, on préfère en calcul scientifique utiliser des **méthodes itératives** qui ne nécessitent pas de modification des matrices et qui comportent peu d'opérations arithmétiques à cause de la prise en compte du caractère creux des matrices de discrétisation. Ceci est d'autant plus souhaitable que certaines de ces matrices sont mal conditionnées, en particulier celles intervenant dans les pas correcteurs de pression de la méthode PISO ; dans

ce cas, la propagation des erreurs d'arrondi aurait pour effet de dénaturer la solution calculée, ce qui est atténué ici par le caractère creux des matrices qui limitera le nombre d'opérations arithmétiques.

Précisons également que la dimension de tels systèmes peut être de plusieurs millions, notamment pour le traitement d'applications industrielles telles que l'aéronautique, l'automobile, le naval, le ferroviaire, etc. Actuellement, de nombreuses études sont menées dans la communauté scientifique pour élaborer des solveurs performants permettant la résolution numérique de ce type de système linéaire. Par exemple, on assiste depuis un certain nombre d'années au développement d'algorithmes de gradient conjugué préconditionnés qui permettent d'accélérer la vitesse de convergence de l'algorithme de gradient conjugué classique [48, 72]. Par ailleurs, un autre type de méthode est apparu, la méthode multi-grille qui permet de combiner une méthode de lissage, obtenue par exemple en utilisant une méthode de relaxation, permettant de réduire les hautes fréquences de l'erreur lorsque l'on décompose celle-ci dans la base des vecteurs propres de la matrice, avec une correction sur grille grossière qui traite efficacement les basses fréquences de l'erreur [14, 38]. Cette méthode multi-grille est également très efficace. D'autres solveurs plus ou moins performants ont également été étudiés et mis en œuvre. Cependant, il n'existe pas de méthode universelle pour résoudre de grands systèmes linéaires creux ; en fonction des caractéristiques des matrices, par exemple la symétrie ou la non-symétrie, les propriétés de diagonale dominance et/ou d'irréductibilité, etc., certaines sont mieux adaptées que d'autres.

Malgré les progrès notables obtenus grâce à ces études algorithmiques, l'accroissement des vitesses de convergence de ces méthodes ainsi que la diminution des temps de calcul obtenus ne sont cependant pas suffisants. Depuis de nombreuses années, l'effort s'est porté d'une part sur l'utilisation de composants permettant d'obtenir des cycles de base des machines moins importants et d'autre part sur l'architecture des machines [17, 23, 30, 34, 46]. L'amélioration des composants ne concerne pas à proprement parler le cadre de la présente étude. Par contre, l'évolution des architectures des machines, vers des machines multi-processeurs, permet d'envisager de nouveaux types d'algorithmes numériques issus des algorithmes classiques listés précédemment. En utilisant ces machines de calcul à haute performance, on est conduit à paralléliser le traitement numérique. Cette parallélisation sera aisée lorsqu'on pourra mettre en évidence des tâches indépendantes, comme par exemple lors du traitement du pas prédicteur dans la méthode PISO. En revanche, lorsque les tâches sont couplées, la parallélisation devient beaucoup plus ardue à mettre en œuvre. Cette dépendance des tâches nécessite l'introduction de synchronisations qui ont pour effet de rendre inactifs des processeurs en attente de résultats calculés par les autres processeurs, ce qui induit des pertes de temps.

En 1969, Chazan et Miranker [20] ont développé une classe de **méthodes parallèles chaotiques** pour la résolution des systèmes linéaires où les valeurs des composantes prenant en compte les interactions entre les sous-systèmes résolus en parallèle sont retardées. En 1974 et 1975, Miellou a étendu cette étude au cas de la résolution de systèmes fortement non-linéaires [51, 52]. Cette étude a été prolongée par Baudet au cas où les communications entre les processeurs sont asynchrones [9]. D'autres nombreux

travaux sur ces itérations asynchrones ont été menés de par le monde dans de nombreux instituts universitaires. On a pu, grâce à ces travaux, mesurer dans quels cas l'utilisation d'algorithmes parallèles asynchrones était intéressante. Pour résumer, lorsque la vitesse convergence des méthodes itératives est élevée, les synchronisations ne dégradent pas les performances parallèles. Par contre, lorsque la convergence est lente, à cause du nombre élevé de synchronisations, les méthodes parallèles asynchrones présentent un grand intérêt, surtout lorsque le réseau d'intercommunication est lent.

Ces méthodes asynchrones ont pu être modélisées sur le plan mathématique de différentes manières, compte tenu notamment de la souplesse des communications interprocesseur [54]. Il est intéressant de remarquer que, dans ce modèle d'algorithme, les méthodes synchrones constituent un cas particulier des méthodes asynchrones ; de plus, dans le cas des méthodes synchrones on retrouve comme cas particulier la formulation classique des méthodes de relaxation séquentielles. Cette modélisation des algorithmes permet d'étudier le comportement des méthodes itératives parallèles, en particulier la convergence de ces méthodes et également la vitesse de convergence. Les techniques d'analyse de la convergence de ces méthodes sont, d'une part, des techniques de contraction [7, 9, 19, 20, 28, 35, 51, 55] qui permettent en plus d'avoir une estimation de la vitesse de convergence, grâce à l'estimation de la constante de contraction, et d'autre part, des techniques d'ordre partiel [31, 52, 53, 54] qui permettent de mettre en évidence des propriétés de convergence monotone des itérés. Si les tests d'arrêt des itérations parallèles synchrones ne posent pas de grandes difficultés, ceux des méthodes asynchrones présentent une grande difficulté. On peut cependant concevoir des tests d'arrêt liés à des propriétés purement numériques telles que l'encadrement de solution par des techniques d'ordre partiel ou encore en prenant en compte que, à cause des erreurs d'arrondi, les itérés successifs appartiennent à des boules emboîtées centrées en la solution et que, par estimation du diamètre de ces boules, on peut exhiber des tests d'arrêt numériques fiables [11, 56, 57, 78]. Par ailleurs, les techniques de *snapshot*, permettent de construire un vecteur résidu global et de mettre en évidence des tests d'arrêt basés sur des considérations informatiques [4, 18, 45].

Cependant, vis-à-vis de l'implémentation des algorithmes subsiste une autre difficulté liée à la décomposition en sous-domaines géométriques du domaine où est définie l'équation aux dérivées partielles. En effet, lorsque les domaines de définition sont quelconques, cette décomposition géométrique n'est pas toujours aisée à effectuer. On lui substitue alors une décomposition de type algébrique basée sur la méthode de **multi-splitting** (ou multi-décomposition) [60, 85] qui permet, assez aisément et de manière unifiée, de séparer en sous-problèmes le problème global décomposé en grands blocs avec ou sans recouvrement. Par ailleurs, il est possible de concevoir une méthode de multi-splitting asynchrone [3, 5, 6, 32, 33, 79, 80, 84] qui permet en plus de prendre en compte des contraintes inégalité sur la solution du problème à résoudre, ce qui présente un intérêt dans le cas d'interactions fluide-structure où des contraintes sur le déplacement peuvent apparaître physiquement. Cette dernière difficulté conduit à la résolution de problèmes multivoques fortement non-linéaires. Par ailleurs, dans ce cadre d'étude on peut considérer des normes variées pour définir les distances dans les espaces de dimension finie

où sont considérés les systèmes discrétisés.

Compte tenu de la nature itérative de ces méthodes de multi-splitting, il est nécessaire d'étudier leur convergence dans le cas général d'une implémentation parallèle asynchrone. Pour mener à bien cet objectif, on considère d'abord une analyse de convergence par des techniques de contraction ; on utilise alors un résultat établi en 1997 par Bahi et al. [5] qui garantit cette propriété de convergence, à condition toutefois que les applications de point fixe intervenant dans la méthode de multi-splitting soient toutes contractantes dans l'espace de travail considéré, normé par une norme avec poids, les poids étant constitués par les composantes des vecteurs propres de la matrice de contraction, ces dernières composantes étant strictement positives, par application du théorème de Perron-Frobenius [82]. Les matrices de discrétisation intervenant d'une part dans la méthode PISO et d'autre part dans la méthode proposée par Jasak pour résoudre l'équation modélisant le comportement de la structure, étant des M-matrices, on montre en utilisant la notion de splitting régulier [61] que les applications de point fixe associées à ces systèmes sont contractantes, quelle que soit la décomposition en grands blocs du problème ; ce qui permet alors d'appliquer le résultat de Bahi et al. [5]. On obtient ainsi un résultat de contraction de l'application de point fixe associée à la méthode de multi-splitting, ce qui permet de conclure sur la convergence de ces méthodes lorsqu'elles sont parallélisées avec échanges asynchrones entre les processeurs. En plus, on peut estimer la constante de contraction des méthodes de multi-spitting parallèles asynchrones comme le maximum des constantes de contraction de chaque application de point fixe intervenant dans la méthode de multi-splitting considérée, permettant ainsi de donner une estimation de la vitesse de convergence. Dans le cas où les déplacements sont soumis à des contraintes, pour prendre en compte ces dernières, on doit formuler l'équation de Navier perturbée par le sous-différentiel de la fonction indicatrice de l'ensemble convexe modélisant ces contraintes. Compte tenu de la propriété de monotonie du sous-différentiel, les résultats de convergence de la méthode de multi-splitting dans le cas de problèmes linéaires définis avec des M-matrices, peuvent être étendus à cette situation de problèmes multivoques fortement non linéaires.

Par ailleurs, on peut également considérer une analyse de la convergence de la méthode de multi-splitting parallèle asynchrone par des techniques d'ordre partiel, liées au principe du maximum, aussi bien dans le cadre de problèmes linéaires définis avec des M-matrices que dans le cadre de problèmes pseudo-linéaires multivoques prenant en compte des contraintes sur le déplacement de la structure. En effet, lorsque l'équation à résoudre est une application affine définie avec une M-matrice, éventuellement perturbée par une application diagonale monotone, comme c'est le cas lorsque le déplacement de la structure est soumis à des contraintes, on montre alors que la méthode de multi-splitting est associée à un problème étendu, ayant les mêmes caractéristiques, c'est-à-dire qu'on a à résoudre un système défini soit par une application affine avec une M-matrice, soit par le même type de problème perturbé par une application diagonale monotone ; dans ces deux cas le problème global déduit de la méthode de multi-splitting est une M-application au sens de Rheinboldt [61, 70], généralisation au cas non linéaire de la notion de M-matrice. En appliquant un résultat de Miellou et al. [54, 79], on aboutit

à une convergence monotone des itérés issus de la méthode de multi-splitting parallèle asynchrone.

L'implémentation des méthodes de résolution numériques a été facilitée par l'utilisation de la bibliothèque de calcul OpenFOAM (i.e. *Open Field Operation And Manipulation*) qui, de par son aspect *Open Source*, permet de créer ou d'implémenter des fonctionnalités complémentaires. L'implémentation des méthodes comportant des communications synchrones n'a pas posé de problèmes insurmontables. Par contre l'implémentation des méthodes comportant des communications asynchrones où les réceptions sont non bloquantes a posé de grandes difficultés. En effet, dans ce mode de communications, survenait une erreur lors de l'exécution. Ce problème a été résolu par l'introduction d'une méthode assurant la terminaison de toutes les communications non bloquantes avant la mise à jour du maillage.

Ces différentes méthodes de résolution parallèle sont utilisées dans le cadre de plusieurs applications. Une première application de résolution de l'équation de la chaleur constituant un cas test, nous permet de tester le bon fonctionnement des deux méthodes de parallélisation utilisées. Deux applications fluide-structure ont pour objectif de valider le solveur couplé et enfin la dernière application constitue un exemple d'utilisation des méthodes développées dans un contexte de type industriel.

Pour résumer, les objectifs de ces travaux de recherche sont, dans un premier temps, la résolution des modèles mathématiques régissant le couplage fluide-structure en utilisant le même formalisme de discrétisation pour les deux sous-domaines. Ceci permet d'assurer la cohérence des échanges entre les solveurs et de limiter les erreurs d'approximation. Dans un deuxième temps, la parallélisation des solveurs devient nécessaire afin de réduire les temps de restitution des calculs importants. Pour ce faire, des méthodes de sous-domaines sont employées ; il est donc nécessaire d'étudier la convergence. Enfin, les implémentations de différentes méthodes de parallélisation doivent être validées par leurs utilisations sur diverses applications d'interaction fluide-structure.

Ce manuscrit est organisé en quatre chapitres. Dans le chapitre 2, la modélisation mathématique des interactions fluide-structure est présentée. Dans un premier temps, nous détaillons les équations de Navier-Stokes qui régissent le comportement du fluide, puis, dans un second temps, l'équation de Navier qui modélise le déplacement d'une structure dans le contexte de l'élasticité linéaire. Enfin, les phénomènes de couplage ainsi que les méthodes de résolution des deux sous-systèmes couplés sont exposés.

Dans le chapitre 3, nous nous intéressons à la résolution numérique du problème d'interaction fluide-structure avec, dans un premier temps, l'exposé de méthodes numériques telles que la méthode de discrétisation par volumes finis ainsi que les différents algorithmes permettant de résoudre les équations régissant les deux sous-systèmes. Dans un second temps, nous développons la résolution numérique parallèle, avec notamment une présentation unifiée de la méthode de multi-splitting.

Le chapitre 4, qui se concentre sur l'implémentation des algorithmes parallèles, est essentiellement de nature bibliographique des différents types d'architecture informatiques et des différentes méthodes de parallélisation ; nous exposons aussi une présentation de différents indicateurs de performance de ces algorithmes. Nous présentons également

les différents solveurs utilisés pour la résolution du problème couplé ainsi que leurs implémentations parallèles et en particulier comment nous avons surmonté les difficultés d'implémentation en mode asynchrone.

Dans le chapitre 5, différentes applications des méthodes développées dans les chapitres précédents sont présentées. Tout d'abord, un cas test de résolution de l'équation de la chaleur afin de démontrer le bon fonctionnement des méthodes de parallélisation, puis plusieurs applications d'interaction fluide-structure : des applications simples pour établir le bon fonctionnement du solveur FSI et des applications à visé industrielle.

Enfin, nous concluons et donnons les perspectives de ces travaux de recherche.

Références

[3] J. Arnal, V. Migallón et J. Penadés. "Non-Stationary Parallel Multisplitting Algorithms for Almost Linear Systems". In : *Numerical Linear Algebra with Applications* 6 (1999), p. 79–92.

[4] J. Bahi, S. Contassot-Vivier et R. Couturier. *Parallel Iterative Algorithms : From Sequential to Grid Computing*. Numerical Analysis and Scientific Computing. Chapman & Hall/CRC, 2007.

[5] J. Bahi, J.-C. Miellou et K. Rhofir. "Asynchronous Multisplitting Methods for Nonlinear Fixed Point Problems". In : *Numerical Algorithms* 15 (1997), p. 315–345.

[6] Z.-Z. Bai. "The Monotone Convergence Rate of the Parallel Nonlinear AOR Method". In : *Computers & Mathematics with Applications* 31 (1996), p. 1–8.

[7] Z.-Z. Bai, V. Migallón, J. Penadés et D. B. Szyld. "Block and Asynchronous Two-stage Methods for Mildly Nonlinear Systems". In : *Numerische Mathematik* 82 (1999), p. 1–20.

[9] G. Baudet. "Asynchronous Iterative Methods for Multiprocessors". In : *Journal of the Association for Computing Machinery* 25 (1978), p. 226–244.

[10] Y. Bazilevs, K. Takizawa et T. E. Tezduyar. *Computational Fluid-Structure Interaction : Methods and Applications*. Wiley, 2013.

[11] D. P. Bertsekas et J. N. Tsitsiklis. "Some Aspects of Parallel and Distributed Iterative Algorithms – A Survey". In : *Automatica* 27 (1991), p. 3–21.

[14] W. L. Briggs, V. E. Henson et S. F. McCormick, éds. *A Multigrid Tutorial, Second Edition*. Other Titles in Applied Mathematics. SIAM, 2000.

[15] H.-J. Bungartz, M. Mehl et M. Schafer, éds. *Fluid-Structure Interaction II : Modelling, Simulation, Optimization*. Berlin, Heidelberg : Springer, 2010.

[16] H.-J. Bungartz et M. Schafer, éds. *Fluid-Structure Interaction : Modelling, Simulation, Optimization*. Berlin, Heidelberg : Springer, 2006.

[17] R. Buyya, J. Broberg et A. M. Goscinski. *Cloud Computing : Principles and Paradigms*. Wiley, 2011.

[18] M. Chau. “Algorithmes Parallèles Asynchrones pour la Simulation Numérique”. Thèse de doct. Institut National Polytechnique de Toulouse, 2005.

[19] M. Chau, T. Garcia et P. Spiteri. “Asynchronous Schwarz Methods Applied to Constrained Mechanical Structures in Grid Environment”. In : *Advances in Engineering Software* 74 (2014), p. 1–15.

[20] D. Chazan et W. Miranker. “Chaotic Relaxation”. In : *Linear Algebra and its Applications* 2 (1969), p. 199–222.

[23] R. Couturier. *Designing Scientific Applications on GPUs*. Numerical Analysis and Scientific Computing. Chapman & Hall/CRC, 2013.

[26] E. de Langre. *Fluide et Solide*. Ellipses, 2001.

[28] M. N. El Tarazi. “Some Convergence Results for Asynchronous Algorithms”. In : *Numerische Mathematik* 39 (1982), p. 325–240.

[30] M. J. Flynn. “Very High-Speed Computing Systems”. In : *Proceedings of the IEEE 54*. 1966, p. 1901–1909.

[31] A. Frommer. “On Asynchronous Iterations in Partially Ordered Spaces”. In : *Numerical Functional Analysis and Optimization* 12 (1991), p. 315–325.

[32] A. Frommer et G. Mayer. “Convergence of Relaxed Parallel Multisplitting Methods”. In : *Linear Algebra and its Applications* 119 (1989), p. 141–152.

[33] A. Frommer et G. Mayer. “On the Theory and Practice of Multisplitting Methods in Parallel Computation”. In : *Computing* 49 (1992), p. 63–74.

[34] M. Gengler, S. Ubéda et F. Desprez. *Initiation au Parallélisme : Concept, Architectures et Algorithmes*. Masson, 1996.

[35] L. Giraud et P. Spiteri. “Résolution Parallèle de Problèmes aux Limites Non Linéaires”. In : *ESAIM : Mathematical Modelling and Numerical Analysis* 25 (1991), p. 579–606.

[38] W. Hackbusch et U. Trottenberg, éds. *Multigrid Methods - Proceedings of the Conference Held at Köln-Porz*. Lecture Notes in Mathematics. Berlin, Heidelberg : Springer, 1981.

[39] P. Hémon. *Vibrations des Structures Couplées avec le Vent*. Ellipses, 2006.

[41] R. I. Issa. “Solution of the Implicitly Discretised Fluid Flow Equation by Operator-Splitting”. In : *Journal of Computational Physics* 62 (1986), p. 40–65.

[42] H. Jasak et H. G. Weller. “Application of the Finite Volume Method and Unstructured Meshes to Linear Elasticity”. In : *International Journal for Numerical Methods in Engineering* 48 (2000), p. 267–287.

[44] L. S. Lai, C. H. Lai, A.-K. Cheik Ahamed et F. Magoulès. “Coupling and Simulation of Fluid-Structure Interaction Problems for Automotive Sun-Roof on Graphics Processing Unit”. In : *2014 IEEE International Conference on High Performance Computing and Communications*. 2014, p. 137–144.

[45] F. Magoulès et G. Gbikpi-Benissan. "Distributed Computation of Convergence Residual Under Asynchronous Iterations". In : *IEEE Transactions on Parallel and Distributed Systems* (to be published).

[46] F. Magoulès et F.-X. Roux. *Calcul Scientifique Parallèle - 2e Édition*. Sciences Sup. Dunod, 2017.

[48] G. Meurant. *Computer Solution of Large Linear Systems*. Studies in Mathematics and its Applications. North Holland, 1999.

[51] J.-C. Miellou. "Algorithmes de Relaxation Chaotique à Retards". In : *ESAIM : Mathematical Modelling and Numerical Analysis* 9 (1975), p. 55–82.

[52] J.-C. Miellou. "Itérations Chaotiques à Retards, Étude de la Convergence dans le Cas d'Espaces Partiellement Ordonnés". In : *CRAS* 280 (1975), p. 233–236.

[53] J.-C. Miellou. "Asynchronous Iterations and Order Intervals". In : *Parallel Algorithms and Architectures*. Sous la dir. de M. Cosnard et D. Trystram. North Holland, 1986, p. 85–96.

[54] J.-C. Miellou, D. El Baz et P. Spiteri. "A New Class of Asynchronous Iterative Algorithms with Order Intervals". In : *Mathematics of Computation* 67 (1998), p. 237–255.

[55] J.-C. Miellou et P. Spiteri. "Un Critère de Convergence pour des Méthodes Générales de Point Fixe". In : *ESAIM : Mathematical Modelling and Numerical Analysis* 19 (1985), p. 645–669.

[56] J.-C. Miellou et P. Spiteri. "Stopping Criteria for Parallel Asynchronous Iterations for Fixed Point Methods". In : *Developments in Parallel, Distributed, Grid and Cloud Computing for Engineering*. Sous la dir. de B. H. V. Topping et P. Iványi. Stirlingshire, UK : Saxe-Coburg Publications, 2013. Chap. 12, p. 277–308.

[57] J.-C. Miellou, P. Spiteri et D. El Baz. "A New Stopping Criterion for Linear Perturbed Asynchronous Iterations". In : *Journal of Computational and Applied Mathematics* 219 (2008), p. 471–483.

[60] D. P. O'Leary et R. E. White. "Multi-Splittings of Matrices and Parallel Solution of Linear Systems". In : *SIAM : Journal on Algebraic Discrete Methods* 6 (1985), p. 630–640.

[61] J. M. Ortega et W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Computer Science and Applied Mathematics. New-York : Academic Press, 1970.

[66] S. V. Patankar. *Numerical Heat Transfer and Fluid Flow*. Series in Computational Methods in Mechanics and Thermal Science. New-York : Hemisphere Publishing Corporation, 1980.

[67] S. V. Patankar et D. B. Spalding. "A Calculation Procedure for Heat, Mass and Momentum Transfer in Three-Dimensional Parabolic Flows". In : *International Journal of Heat and Mass Transfer* 15 (1972), p. 1787–1806.

[68] S. Piperno. *Interactions Fluide-Structure*. Mastère de Mécanique Numérique - École Nationale Supérieure des Mines de Paris. 2005–2006.

[70] W. Rheinboldt. "On M-functions and Their Application to Nonlinear Gauss-Seidel Iterations and to Network Flows". In : *Journal of Mathematical Analysis and Applications* 32 (1970), p. 274–307.

[72] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Other Titles in Applied Mathematics. SIAM, 2003.

[78] P. Spiteri. "Finite Precision Computation for Linear Fixed Point Methods of Parallel Asynchronous Iterations". In : *Techniques for Parallel, Distributed and Cloud Computing in Engineering*. Sous la dir. de P. Iványi et B. H. V. Topping. Stirlingshire, UK : Saxe-Coburg Publications, 2015. Chap. 8, p. 163–196.

[79] P. Spiteri, J.-C. Miellou et D. El Baz. "Parallel Asynchronous Schwarz and Multisplitting Methods for a Nonlinear Diffusion Problem". In : *Numerical Algorithms* 33 (2003), p. 461–474.

[80] D. B. Szyld. "Different Models Of Parallel Asynchronous Iterations With Overlapping Blocks". In : *Computational and Applied Mathematics* 17 (1998), p. 101–115.

[82] R. S. Varga. *Matrix Iterative Analysis*. Computational Mathematics. Berlin, Heidelberg : Springer, 2000.

[84] D.-R. Wang, Z.-Z. Bai et D. J. Evans. "On the Monotone Convergence of Multisplitting Method for a Class of Systems of Weakly Nonlinear Equations". In : *International Journal of Computer Mathematics* 60 (1996), p. 229–242.

[85] R. E. White. "Parallel Algorithms for Nonlinear Problems". In : *SIAM : Journal on Algebraic Discrete Methods* 7 (1986), p. 137–149.

Le travail présenté dans ce manuscrit a donné lieu à :

– deux *proceedings* dans des conférences internationales :

  - V. Partimbene, P. Spiteri, P. Marthon et L. Ratsifandrihana. "A Two-Dimensional Numerical Case Study of Fluid-Structure Interaction". In : *Proceedings of the Third International Conference on Railway Technology : Research, Development and Maintenance*. Sous la dir. de J. Pombo. Stirlingshire, UK : Civil-Comp Press, 2016. doi : 10.4203/ccp.110.43 ;
  - V. Partimbene, T. Garcia, P. Spiteri, P. Marthon et L. Ratsifandrihana. "A Parallel Method for the Solution of Fluid-Structure Interaction Problems". In : *Proceedings of the Fifth International Conference on Parallel, Distributed, Grid and Cloud Computing for Engineering*. Sous la dir. de P. Iványi, B. H. V. Topping et G. Várady. Stirlingshire, UK : Civil-Comp Press, 2017. doi : 10.4203/ccp.111.20 ;

– un rapport de recherche :

  - V. Partimbene, P. Spiteri, P. Marthon et L. Ratsifandrihana. *A Fluid-Structure Interaction Study on 3-D Structures Under Crosswind*. Rapport de Recherche - Institut de Recherche en Informatique de Toulouse. 2016 ;

– un article soumis :

  - V. Partimbene, T. Garcia, P. Spiteri, P. Marthon et L. Ratsifandrihana. "Asynchronous Multi-Splitting Method for the Solution of Fluid-Structure Interaction Problems". In : *Advances in Engineering Software* (soumis).

CHAPITRE

2

MODÉLISATION DES INTERACTIONS FLUIDE-STRUCTURE

Sommaire

2.1 Introduction

Les phénomènes d'interactions fluide-structure interviennent dans de nombreux domaines tels que l'aéronautique, l'automobile, la biomécanique, l'industrie ferroviaire, la construction navale, le bâtiment, etc. Ces phénomènes mettent en jeu le couplage entre les équations régissant le comportement du fluide et celles régissant le comportement de la structure. Le couplage se fait par l'intermédiaire d'une interface (une surface pour les problèmes en trois dimensions) sur laquelle le fluide exerce des forces ce qui se traduit par une déformation de la structure et par conséquent une modification du domaine fluide.

Dans ce chapitre, les modèles mathématiques utilisés pour représenter les phénomènes physiques présents dans l'interaction fluide-structure ainsi que le phénomène de couplage entre les sous-systèmes sont décrits.

Dans un premier temps, la modélisation mathématique du comportement d'un fluide newtonien, incompressible et isotrope est décrite en détail pour arriver au système d'équations dites de Navier-Stokes qui regroupe l'équation de continuité (ou de conservation de la masse), l'équation de conservation de la quantité de mouvement et l'équation d'énergie. Une présentation des différents formalismes utilisés en mécanique des milieux continus est faite avec l'accent mis sur le formalisme ALE (Arbitraire Lagrangien-Eulérien) utilisé pour résoudre les équations régissant le fluide dans un domaine mobile.

Dans une deuxième partie, la démarche ainsi que les hypothèses qui permettent d'arriver à la modélisation mathématique du comportement d'une structure homogène sont décrites. En partant des équations d'équilibre des contraintes sur un volume élémentaire solide et en faisant l'hypothèse de petits déplacements et de petites déformations classique en élasticité linéaire, les équations de Navier apparaissent et permettent notamment de calculer le déplacement de la structure.

Enfin, dans une troisième partie, le couplage régissant les interactions fluide-structure ainsi que les stratégies de simulation numérique de tels phénomènes sont décrits.

2.2 Fluide

Un **fluide** est un milieu matériel continu, déformable et s'écoulant [83]. On regroupe sous cette appellation les liquides, les gaz et les plasmas. Gaz et plasmas sont très compressibles, alors que les liquides le sont très peu, à peine plus que les solides.

Les équations régissant le comportement d'un écoulement fluide sont la traduction mathématique des lois de conservation de la physique :

– la **masse** d'un fluide se conserve ;

– la variation de la **quantité de mouvement** est égale à la somme des forces sur une « particule » de fluide, ce qui correspond à la seconde loi de Newton ;

– la variation de l'**énergie** d'une « particule » de fluide est égale à la quantité d'énergie échangée par transfert thermique (chaleur) et transfert mécanique dû au travail des forces conformément au premier principe de la thermodynamique.

2.2.1 Particule fluide

On appelle **particule fluide** (ou particule de fluide) un paquet de molécules du fluide regroupées autour d'un point du fluide et dont le volume est petit à l'échelle macroscopique mais grand à l'échelle microscopique.[1] On considère que le fluide se divise en un grand nombre de particules fluides et que son mouvement (sa déformation et son écoulement) peut être décrit par le mouvement de l'ensemble des particules fluides qui le composent.[2] En vertu de la loi de conservation de la masse, la masse d'une particule fluide restera donc constante lors du mouvement du fluide ; mais pas son volume si le fluide est compressible.

2.2.2 Élément fluide

On appelle **élément fluide** un volume cubique de très petites dimensions à l'intérieur du fluide. Ce volume reste toujours fixe, à la différence du volume d'une particule fluide qui varie si le fluide est compressible. Les dimensions des côtés de l'élément fluide seront notées dx, dy et dz (voir la figure 2.1).

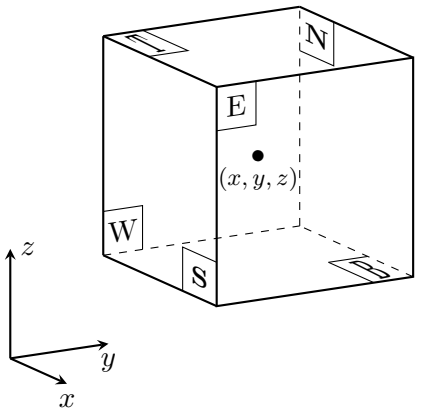


FIGURE 2.1 – Élément de fluide

[1] Le volume d'une particule fluide est à l'échelle mésoscopique (typiquement de l'ordre du micromètre). L'échelle mésoscopique est une échelle intermédiaire entre l'échelle microscopique qui caractérise les atomes et les molécules et l'échelle macroscopique qui caractérise les corps dans leur ensemble.

[2] Il est important de noter que chaque particule fluide suit la déformation et l'écoulement du fluide à la différence des « éléments » fluides qui eux restent fixes.

Les six faces sont dénotées N, S, E, W, T et B, représentant respectivement les quatre points cardinaux et les faces supérieure (Top) et inférieure (Bottom). Le centre de l'élément est situé à la position (x, y, z).

Toutes les propriétés du fluide sont fonctions de l'espace et du temps et devraient donc s'écrire $\rho^f(x, y, z, t)$, $p(x, y, z, t)$, $T(x, y, z, t)$ et $\mathbf{u}(x, y, z, t)$ pour la masse volumique, la pression, la température et le vecteur vitesse respectivement. Pour éviter des notations trop lourdes, les dépendances aux coordonnées d'espace et au temps ne seront pas explicitées dans la suite.

L'élément de fluide considéré est petit de telle sorte que les propriétés du fluide aux faces peuvent être exprimées avec suffisamment de précision par les deux premiers termes d'un développement en série de Taylor. Par exemple, la pression aux faces W et E, qui sont toutes deux à une distance $\frac{1}{2}dx$ du centre de l'élément, peut s'exprimer

$$p - \frac{\partial p}{\partial x} \cdot \frac{1}{2}dx \quad \text{et} \quad p + \frac{\partial p}{\partial x} \cdot \frac{1}{2}dx. \tag{2.1}$$

2.2.3 Conservation de la masse (approche eulérienne)

L'approche eulérienne consiste à raisonner sur un élément fluide. Pour traduire la conservation de la masse totale du fluide, on écrit que pour chaque élément fluide, le taux de croissance de sa masse[3] est égal au débit massique de l'élément fluide à travers ses faces. L'équilibre de masse pour un élément de fluide peut s'écrire

Croissance de la masse dans un élément fluide	=	Débit net de masse dans un élément fluide

Le taux de croissance de la masse dans un élément de fluide est donné par

$$\frac{\partial}{\partial t}(dm) = \frac{\partial}{\partial t}(\rho^f \, dx \, dy \, dz) = \frac{\partial \rho^f}{\partial t} dx \, dy \, dz. \tag{2.2}$$

Notons $\Phi(X)$ le flux de masse entrant(ou sortant) dans (de) l'élément fuide par la face X ; alors le débit massique de l'élément fluide à travers ses six faces vaut

$$\begin{array}{rcl} \text{Débit massique de l'élément de fluide} & = & [\Phi(W) - \Phi(E)] + [\Phi(S) - \Phi(N)] + [\Phi(B) - \Phi(T)]. \end{array} \tag{2.3}$$

Le flux de masse à travers une face X de l'élément est donné par le produit de la masse volumique, de l'aire et de la composante de vitesse normale à la face. D'après la figure 2.2 et l'équation (2.3) le débit massique du fluide dans un élément à travers ses frontières

[3] La masse d'un élément fluide est la masse du fluide contenue dans cet élément.

est donné par

$$\begin{aligned}\Big(\rho^f u \quad & -\frac{\partial(\rho^f u)}{\partial x}\cdot\frac{1}{2}dx\Big)\,dy\,dz - \Big(\rho^f u + \frac{\partial(\rho^f u)}{\partial x}\cdot\frac{1}{2}dx\Big)\,dy\,dz \\ & + \Big(\rho^f v - \frac{\partial(\rho^f v)}{\partial y}\cdot\frac{1}{2}dy\Big)\,dx\,dz - \Big(\rho^f v + \frac{\partial(\rho^f v)}{\partial y}\cdot\frac{1}{2}dy\Big)\,dx\,dz \\ & + \Big(\rho^f w - \frac{\partial(\rho^f w)}{\partial z}\cdot\frac{1}{2}dz\Big)\,dx\,dy - \Big(\rho^f w + \frac{\partial(\rho^f w)}{\partial z}\cdot\frac{1}{2}dz\Big)\,dx\,dy.\end{aligned} \tag{2.4}$$

Les flux qui sont dirigés vers l'élément produisent une augmentation de la masse dans l'élément et ont un signe positif alors que les flux qui quittent l'élément ont un signe négatif.

En égalant les deux grandeurs (2.2) et (2.4), en plaçant tous les termes à gauche du signe égal et en divisant par le volume de l'élément $dx\,dy\,dz$, on trouve

$$\frac{\partial\rho^f}{\partial t} + \frac{\partial(\rho^f u)}{\partial x} + \frac{\partial(\rho^f v)}{\partial y} + \frac{\partial(\rho^f w)}{\partial z} = 0, \tag{2.5}$$

ou en notation vectorielle

$$\frac{\partial\rho^f}{\partial t} + \nabla\cdot(\rho^f\mathbf{u}) = 0. \tag{2.6}$$

L'équation (2.6) est l'**équation de conservation de la masse instationnaire tridimensionnelle** ou **équation de continuité** en un point d'un **fluide compressible**. Le premier terme de gauche représente l'évolution de la masse volumique au cours du temps. Le second terme de gauche représente les flux de masse à travers les frontières de l'élément et est appelé le terme convectif.

Pour un **fluide incompressible**, la masse volumique ρ^f est constante et l'équation (2.6) devient

$$\nabla\cdot\mathbf{u} = 0. \tag{2.7}$$

2.2.4 Conservation de la masse (approche lagrangienne)

L'approche lagrangienne consiste à raisonner sur une particule fluide. Pour traduire la conservation de la masse totale du fluide, on écrit la conservation de la masse de chaque particule fluide, soit

$$\frac{D}{Dt}(dm) = 0, \tag{2.8}$$

où dm est la masse de la particule fluide et D/Dt est la dérivée totale par rapport au temps t. Or

$$dm = \rho^f dV = \rho^f dx\,dy\,dz, \tag{2.9}$$

et

$$\frac{D}{Dt}(dm) = \frac{D}{Dt}\left(\rho^f dV\right) = dV\frac{D}{Dt}(\rho^f) + \rho^f\frac{D}{Dt}(dV). \tag{2.10}$$

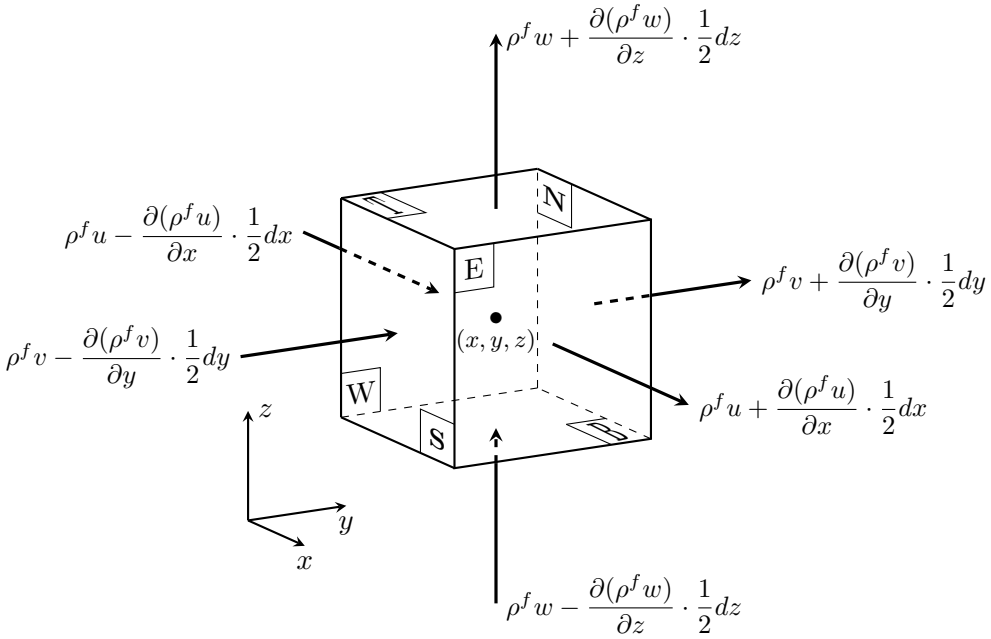


FIGURE 2.2 – Flux de masse à travers un élément de fluide

On vérifie que :[4]

$$\frac{D}{Dt}(dV) = dV \nabla \cdot \mathbf{u} = dV \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \right), \tag{2.11}$$

d'où

$$\frac{D}{Dt}(dm) = dV \left(\frac{D}{Dt}(\rho^f) + \rho^f \nabla \cdot \mathbf{u} \right) = 0, \tag{2.12}$$

et

$$\frac{D}{Dt}(\rho^f) + \rho^f \nabla \cdot \mathbf{u} = 0. \tag{2.13}$$

Or

$$\frac{D}{Dt}(\rho^f) = \frac{\partial \rho^f}{\partial t} + u \frac{\partial \rho^f}{\partial x} + v \frac{\partial \rho^f}{\partial y} + w \frac{\partial \rho^f}{\partial z}. \tag{2.14}$$

Reportant dans (2.13) , il vient

$$\frac{\partial \rho^f}{\partial t} + \nabla \cdot (\rho^f \mathbf{u}) = 0. \tag{2.15}$$

qui est l'équation (2.6).

Les deux approches eulérienne et lagrangienne conduisent à la même équation de conservation de la masse totale du fluide.

[4]Remarquons ici que si la divergence du vecteur vitesse $\mathbf{u}$ est positive, le volume de la particule fluide croît alors que si elle est négative, son volume décroît.

2.2.5 Équation de quantité de mouvement

La deuxième loi du mouvement de Newton stipule que dans un repère galiléen, la dérivée de la quantité de mouvement d'une particule fluide est égale à la somme des forces[5] externes qui s'exercent sur cette particule[6] :

Dérivée de la quantité de mouvement d'une particule de fluide	=	Somme des forces sur la particule de fluide

La dérivée de la quantité de mouvement d'une particule fluide vaut :

$$\frac{D}{Dt}(dm\ \mathbf{u}) = \frac{D}{Dt}(dm)\mathbf{u} + dm\frac{D}{Dt}(\mathbf{u}) = dm\frac{D}{Dt}(\mathbf{u}) = \rho^f dV\left(\frac{Du}{Dt}, \frac{Dv}{Dt}, \frac{Dw}{Dt}\right), \quad (2.16)$$

car d'après l'équation (2.8), $\frac{D}{Dt}(dm) = 0$.

Parmi les forces externes, on distinguera les forces de contact qui s'exercent à la surface de la particule fluide des autres forces (dites forces à distance) :

- les forces de contact :
  - forces de pression ;
  - forces visqueuses ;
- les forces à distance :
  - force gravitationnelle ;
  - force centrifuge ;
  - force de Coriolis ;
  - force électrostatique ;
  - force electromagnétique.

L'état des forces de contact sur une particule fluide est défini par un terme de pression et par les neuf composantes de la contrainte visqueuse représentées sur la figure 2.3. La pression notée p est une contrainte normale. Les contraintes visqueuses sont notées τ_{ij}, l'indice i indique que la contrainte agit sur une surface normale à la direction i alors que l'indice j indique la direction dans laquelle elle agit.

[5] En mécanique classique, une **force** est la modélisation d'une interaction, quelle que soit la nature de celle-ci. La force résulte de l'action d'un objet sur un autre. C'est le cas en particulier des interactions de contact (pression, frottement, interaction dans une liaison) ou à distance (force gravitationnelle, force électrostatique, force électromagnétique). La force est représentée par un vecteur ayant un point d'application, une direction, un sens et une intensité (en newtons).

[6] Son énoncé original était : « les changements qui arrivent dans le mouvement sont proportionnels à la force motrice et se font dans la ligne droite dans laquelle cette force a été imprimée ». Dans sa version moderne, on l'appelle encore principe fondamental de la dynamique de translation.

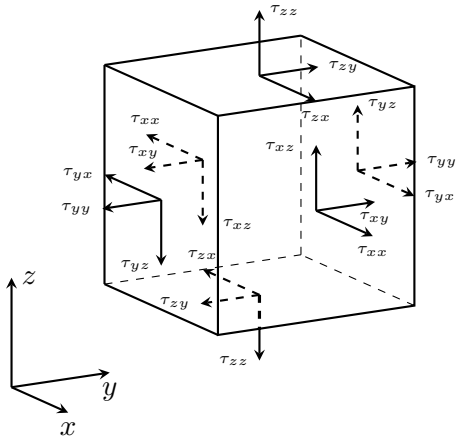


FIGURE 2.3 – Composantes des contraintes sur les faces d'un élément de fluide

Dans un premier temps, déterminons la composante en x, F_x^c, de la force de contact s'exerçant sur une particule fluide. L'intensité d'une force résultant d'une contrainte de surface est le produit de la contrainte et de la surface. Les forces ayant la même direction qu'un axe du repère ont un signe positif et celles qui agissent dans la direction opposée ont un signe négatif. La force de contact dans la direction x est la somme des composantes des forces de contact agissant sur l'élément fluide dans cette direction.

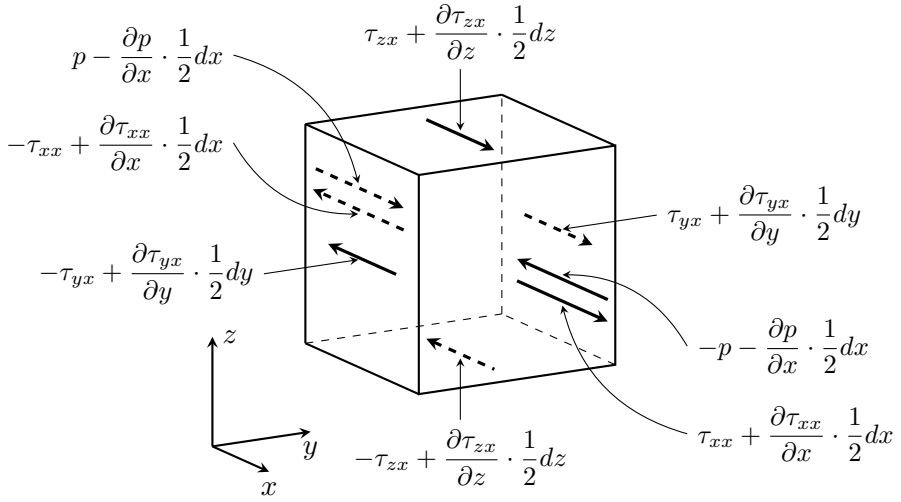


FIGURE 2.4 – Composantes des contraintes dans la direction x

Sur la paire de faces (E, W), la force de contact vaut

$$\begin{aligned}\left[\left(p-\frac{\partial p}{\partial x}\cdot\frac{1}{2}dx\right)\right. &-\left(\tau_{xx}-\frac{\partial \tau_{xx}}{\partial x}\cdot\frac{1}{2}dx\right)\Bigg]dy\,dz+\left[-\left(p+\frac{\partial p}{\partial x}\cdot\frac{1}{2}dx\right)\right.\\ &\left.+\left(\tau_{xx}+\frac{\partial \tau_{xx}}{\partial x}\cdot\frac{1}{2}dx\right)\right]dy\,dz=\left(-\frac{\partial p}{\partial x}+\frac{\partial \tau_{xx}}{\partial x}\right)dx\,dy\,dz.\end{aligned} \tag{2.17}$$

La force nette dans la direction x sur la paire de faces (N, S) est

$$-\left(\tau_{yx}-\frac{\partial \tau_{yx}}{\partial y}\cdot\frac{1}{2}dy\right)dx\,dz+\left(\tau_{yx}+\frac{\partial \tau_{yx}}{\partial y}\cdot\frac{1}{2}dy\right)dx\,dz=\frac{\partial \tau_{yx}}{\partial y}dx\,dy\,dz. \tag{2.18}$$

Enfin, la force nette dans la direction x sur la paire de faces (T, B) est

$$-\left(\tau_{zx}-\frac{\partial \tau_{zx}}{\partial z}\cdot\frac{1}{2}dz\right)dx\,dy+\left(\tau_{zx}+\frac{\partial \tau_{zx}}{\partial z}\cdot\frac{1}{2}dz\right)dx\,dy=\frac{\partial \tau_{zx}}{\partial z}dx\,dy\,dz. \tag{2.19}$$

La force totale de contact dans la direction x est donc la somme de (2.17), (2.18) et (2.19) :

$$\left(\frac{\partial(-p+\tau_{xx})}{\partial x}+\frac{\partial \tau_{yx}}{\partial y}+\frac{\partial \tau_{zx}}{\partial z}\right)dV. \tag{2.20}$$

Dans la direction x, la seconde loi de Newton s'écrit donc :

$$\left(\frac{D}{Dt}(dm\ \mathbf{u})\right)_x=F_x^c+F_x^d=\left(\frac{\partial(-p+\tau_{xx})}{\partial x}+\frac{\partial \tau_{yx}}{\partial y}+\frac{\partial \tau_{zx}}{\partial z}\right)dV+F_x^d, \tag{2.21}$$

où F_x^d représente les forces à distance s'exerçant sur la particule fluide dans la direction x.

Définissons maintenant la quantité S_M, dite terme « source », comme étant la force à distance s'exerçant sur la particule fluide par unité de volume, soit $S_{M_x}dV=F_x^d$; alors en remplaçant le premier membre de l'équation (2.21) par le second membre de l'équation (2.16), puis en divisant par dV, on obtient :

$$\rho^f\frac{Du}{Dt}=\frac{\partial(-p+\tau_{xx})}{\partial x}+\frac{\partial \tau_{yx}}{\partial y}+\frac{\partial \tau_{zx}}{\partial z}+S_{M_x}. \tag{2.22}$$

De la même façon, la composante en y de l'équation de quantité de mouvement est donnée par

$$\rho^f\frac{Dv}{Dt}=\frac{\partial \tau_{xy}}{\partial x}+\frac{\partial(-p+\tau_{yy})}{\partial y}+\frac{\partial \tau_{zy}}{\partial z}+S_{M_y}, \tag{2.23}$$

et la composante en z est donnée par

$$\rho^f\frac{Dw}{Dt}=\frac{\partial \tau_{xz}}{\partial x}+\frac{\partial \tau_{yz}}{\partial y}+\frac{\partial(-p+\tau_{zz})}{\partial z}+S_{M_z}. \tag{2.24}$$

Le signe associé à la pression est opposé à celui associé à la contrainte visqueuse normale car la convention habituelle prévoit qu'un effort de traction soit une contrainte normale positive et ainsi la pression, qui est par définition un effort de compression normal, hérite d'un signe négatif.

Les termes source S_{M_x}, S_{M_y} et S_{M_z} modélisent les contributions des forces à distance. Une force à distance due à la gravité serait par exemple exprimée par $S_{M_x}=0$, $S_{M_y}=0$ et $S_{M_z}=-\rho^f g$.

2.2.6 Équation d'énergie en trois dimensions

L'équation d'énergie provient du premier principe de la thermodynamique qui stipule que la variation de l'énergie d'une particule de fluide est égale à l'apport de chaleur à la particule plus le travail effectué sur la particule :

Taux de croissance de l'énergie de la particule de fluide	=	Taux net de chaleur ajoutée à la particule	+	Taux net de travail effectué sur la particule

Comme précédemment, une équation pour le taux de croissance de l'énergie d'une particule de fluide par unité de volume sera dérivée et donnée par

$$\rho^f \frac{DE^f}{Dt}. \tag{2.25}$$

2.2.6.1 Travail effectué par les forces de surface

La travail effectué sur la particule de fluide dans l'élément par une force de surface est égal au produit de la force et de la composante de vitesse dans la direction de la force. Par exemple, les forces données par (2.17), (2.18) et (2.19) agissent toutes dans la direction x. Le travail effectué par ces forces est donné par

$$\begin{aligned}\Bigg[\Bigg(pu &- \frac{\partial(pu)}{\partial x}\frac{1}{2}dx\Bigg) - \Bigg(\tau_{xx}u - \frac{\partial(\tau_{xx}u)}{\partial x}\frac{1}{2}dx\Bigg)\\ &- \Bigg(pu + \frac{\partial(pu)}{\partial x}\frac{1}{2}dx\Bigg) + \Bigg(\tau_{xx}u + \frac{\partial(\tau_{xx}u)}{\partial x}\frac{1}{2}dx\Bigg)\Bigg]dy\,dz\\ &+ \Bigg[-\Bigg(\tau_{yx}u - \frac{\partial(\tau_{yx}u)}{\partial y}\frac{1}{2}dy\Bigg) + \Bigg(\tau_{yx}u + \frac{\partial(\tau_{yx}u)}{\partial y}\frac{1}{2}dy\Bigg)\Bigg]dx\,dz\\ &+ \Bigg[-\Bigg(\tau_{zx}u - \frac{\partial(\tau_{zx}u)}{\partial z}\frac{1}{2}dz\Bigg) + \Bigg(\tau_{zx}u + \frac{\partial(\tau_{zx}u)}{\partial z}\frac{1}{2}dz\Bigg)\Bigg]dx\,dy.\end{aligned} \tag{2.26}$$

Le taux net de travail effectué par ces forces de surface agissant dans la direction x est donné par

$$\left[\frac{\partial(u(-p+\tau_{xx}))}{\partial x} + \frac{\partial(u\tau_{yx})}{\partial y} + \frac{\partial(u\tau_{zx})}{\partial z}\right]dx\,dy\,dz. \tag{2.27}$$

Les composantes de la contrainte de surface dans les directions y et z travaillent elles aussi sur la particule. De la même manière, les travaux effectués sur la particule fluide et dus à ces forces sont

$$\left[\frac{\partial(v\tau_{xy})}{\partial x} + \frac{\partial(v(-p+\tau_{yy}))}{\partial y} + \frac{\partial(v\tau_{zy})}{\partial z}\right]dx\,dy\,dz, \tag{2.28}$$

et

$$\left[\frac{\partial(w\tau_{xz})}{\partial x} + \frac{\partial(w\tau_{yz})}{\partial y} + \frac{\partial(w(-p+\tau_{zz}))}{\partial z}\right]dx\,dy\,dz. \tag{2.29}$$

Le travail total par unité de volume effectué sur la particule de fluide par toutes les forces de surface est donné par la somme de (2.27), (2.28) et (2.29) divisée par le volume

$dx\,dy\,dz$. Les termes faisant intervenir la pression peuvent être rassemblés et écrits sous une forme vectorielle plus compacte

$$-\frac{\partial(up)}{\partial x}-\frac{\partial(vp)}{\partial y}-\frac{\partial(wp)}{\partial z}=-\nabla\cdot(p\mathbf{u}). \tag{2.30}$$

Ce qui donne le travail total effectué sur la particule fluide par les contraintes de surface :

$$\begin{aligned}[-\nabla\cdot(p\mathbf{u})]+\quad&\Big[\frac{\partial(u\tau_{xx})}{\partial x}+\frac{\partial(u\tau_{yx})}{\partial y}+\frac{\partial(u\tau_{zx})}{\partial z}\\&+\frac{\partial(v\tau_{xy})}{\partial x}+\frac{\partial(v\tau_{yy})}{\partial y}+\frac{\partial(v\tau_{zy})}{\partial z}\\&+\frac{\partial(w\tau_{xz})}{\partial x}+\frac{\partial(w\tau_{yz})}{\partial y}+\frac{\partial(w\tau_{zz})}{\partial z}\Big].\end{aligned} \tag{2.31}$$

2.2.6.2 Flux d'énergie dû à la conduction thermique

Le vecteur flux de chaleur $\mathbf{q}$ possède trois composantes q_x, q_y et q_z (figure 2.5).

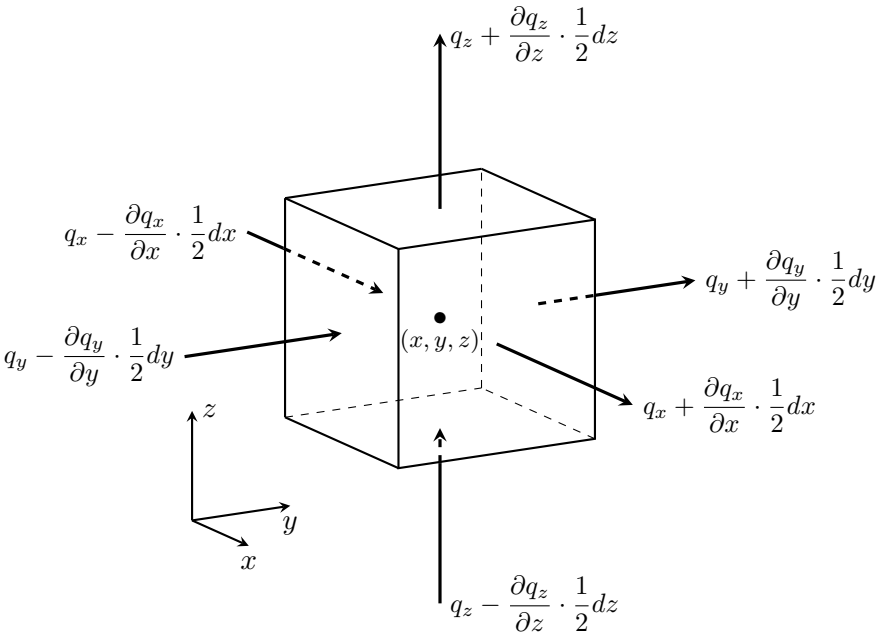


FIGURE 2.5 – Composantes du vecteur flux de chaleur

Le taux de transfert de chaleur vers la particule de fluide dû aux flux thermiques dans la direction x est donné par la différence entre l'apport de chaleur à travers la face W et la perte de chaleur à travers la face E :

$$\left[\left(q_x-\frac{\partial q_x}{\partial x}\frac{1}{2}dx\right)-\left(q_x+\frac{\partial q_x}{\partial x}\frac{1}{2}dx\right)\right]dy\,dz=-\frac{\partial q_x}{\partial x}dx\,dy\,dz. \tag{2.32}$$

De la même manière, dans les directions y et z il vient

$$-\frac{\partial q_y}{\partial y} dx\, dy\, dz, \tag{2.33}$$

et

$$-\frac{\partial q_z}{\partial z} dx\, dy\, dz. \tag{2.34}$$

L'apport total de chaleur à la particule fluide par unité de volume dû aux flux thermiques à travers ses frontières est la somme de (2.32), (2.33) et (2.34) divisée par le volume $dx\, dy\, dz$:

$$-\frac{\partial q_x}{\partial x} - \frac{\partial q_y}{\partial y} - \frac{\partial q_z}{\partial z} = -\nabla \cdot \mathbf{q}. \tag{2.35}$$

La loi de Fourier pour la conduction thermique associe le flux de chaleur au gradient local de température, donc

$$q_x = -k\frac{\partial T}{\partial x}, \quad q_y = -k\frac{\partial T}{\partial y} \quad \text{et} \quad q_z = -k\frac{\partial T}{\partial z}, \tag{2.36}$$

où k est la conductivité thermique et qui peut s'écrire sous la forme

$$\mathbf{q} = -k\nabla T. \tag{2.37}$$

La combinaison de (2.35) et (2.37) donne la forme finale du taux d'apport de chaleur à la particule fluide dû à la conduction thermique à travers les frontières de l'élément :

$$-\nabla \cdot \mathbf{q} = \nabla \cdot (k\nabla T). \tag{2.38}$$

2.2.6.3 Équation d'énergie

L'énergie d'un fluide est souvent définie comme la somme de l'énergie interne c'est-à-dire thermique, E_{int}, de l'énergie cinétique $\frac{1}{2}(u^2 + v^2 + w^2)$ et de l'énergie potentielle due à la gravité. Cette définition considère que l'élément fluide stocke de l'énergie potentielle gravitationnelle. Il est aussi possible de considérer la force gravitationnelle comme une force interne qui agit sur l'élément de fluide se déplaçant dans le champ gravitationnel.

Comme précédemment, une source d'énergie S_E par unité de volume et de temps est définie. La conservation de l'énergie de la particule de fluide est assurée en égalant la variation de l'énergie de la particule (2.25) à la somme du travail effectué sur la particule (2.31) ajouté au taux d'apport de chaleur au fluide (2.38) et au taux de croissance de l'énergie due aux sources. L'équation d'énergie est

$$\begin{aligned} \rho^f \frac{DE^f}{Dt} = \; & -\nabla \cdot (\rho^f \mathbf{u}) + \left[\frac{\partial(u\tau_{xx})}{\partial x} + \frac{\partial(u\tau_{yx})}{\partial y} + \frac{\partial(u\tau_{zx})}{\partial z} + \frac{\partial(v\tau_{xy})}{\partial x}\right. \\ & \left. + \frac{\partial(v\tau_{yy})}{\partial y} + \frac{\partial(v\tau_{zy})}{\partial z} + \frac{\partial(w\tau_{xz})}{\partial x} + \frac{\partial(w\tau_{yz})}{\partial y} + \frac{\partial(w\tau_{zz})}{\partial z}\right] \\ & + \nabla \cdot (k\nabla T) + S_E. \end{aligned} \tag{2.39}$$

Dans l'équation (2.39), $E^f = E_{int} + \frac{1}{2}(u^2 + v^2 + w^2)$.

Même si (2.39) est une équation d'énergie adéquate, il est en pratique courant d'extraire les variations de l'énergie cinétique pour obtenir une équation en fonction de l'énergie interne E_{int} ou en fonction de la température T. La partie de l'équation d'énergie attribuable à l'énergie cinétique peut être trouvée en multipliant l'équation de la quantité de mouvement selon x (2.22) par la composante de vitesse u, selon y (2.23) par v et selon z (2.24) par w et en sommant les résultats. Il vient alors l'équation de conservation pour l'énergie cinétique

$$\begin{aligned}\rho^f \frac{D[\frac{1}{2}(u^2+v^2+w^2)]}{Dt} = \quad & -\mathbf{u}\cdot\nabla p + u\left(\frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{yx}}{\partial y} + \frac{\partial \tau_{zx}}{\partial z}\right) \\ & + v\left(\frac{\partial \tau_{xy}}{\partial x} + \frac{\partial \tau_{yy}}{\partial y} + \frac{\partial \tau_{zy}}{\partial z}\right) \\ & + w\left(\frac{\partial \tau_{xz}}{\partial x} + \frac{\partial \tau_{yz}}{\partial y} + \frac{\partial \tau_{zz}}{\partial z}\right) + \mathbf{u}\cdot\mathbf{S_M}.\end{aligned} \tag{2.40}$$

Par soustraction de (2.40) et de (2.39) et en définissant un nouveau terme source comme $S_i = S_E - \mathbf{u}\cdot\mathbf{S_M}$, il vient l'équation d'énergie interne

$$\begin{aligned}\rho^f \frac{Di}{Dt} = \quad & -p\nabla\cdot\mathbf{u} + \nabla\cdot(k\nabla T) + \tau_{xx}\frac{\partial u}{\partial x} + \tau_{yx}\frac{\partial u}{\partial y} + \tau_{zx}\frac{\partial u}{\partial z} \\ & + \tau_{xy}\frac{\partial v}{\partial x} + \tau_{yy}\frac{\partial v}{\partial y} + \tau_{zy}\frac{\partial v}{\partial z} \\ & + \tau_{xz}\frac{\partial w}{\partial x} + \tau_{yz}\frac{\partial w}{\partial y} + \tau_{zz}\frac{\partial w}{\partial z} + S_i.\end{aligned} \tag{2.41}$$

Dans le cas d'un fluide **incompressible**, $E_{int} = cT$, où c est la chaleur spécifique et $\nabla\cdot\mathbf{u} = 0$. Cela permet de reformuler (2.41) en équation de température

$$\begin{aligned}\rho^f c\frac{DT}{Dt} = \quad & \nabla\cdot(k\nabla T) + \tau_{xx}\frac{\partial u}{\partial x} + \tau_{yx}\frac{\partial u}{\partial y} + \tau_{zx}\frac{\partial u}{\partial z} + \tau_{xy}\frac{\partial v}{\partial x} \\ & + \tau_{yy}\frac{\partial v}{\partial y} + \tau_{zy}\frac{\partial v}{\partial z} + \tau_{xz}\frac{\partial w}{\partial x} + \tau_{yz}\frac{\partial w}{\partial y} + \tau_{zz}\frac{\partial w}{\partial z} + S_i.\end{aligned} \tag{2.42}$$

2.2.7 Équations de Navier-Stokes pour un fluide newtonien

Les équations qui régissent les écoulements fluides contiennent d'autres inconnues : les composantes de la contrainte visqueuse τ_{ij}. C'est la modélisation des ces contraintes visqueuses qui engendre les formes les plus utiles des équations de conservations pour un écoulement fluide. Dans beaucoup d'écoulements, les contraintes visqueuses peuvent être exprimées en fonction du taux de déformation ou de la vitesse de déformation locale. En trois dimensions, la vitesse de déformation locale est composée de la vitesse de déformation linéaire et de la vitesse de déformation volumétrique.

Tous les gaz ainsi que de nombreux liquides sont isotropes. La vitesse de déformation linéaire d'un élément de fluide en trois dimensions possède neuf composantes dont six sont indépendantes pour les fluides isotropes (Schlichting, 1979). Elles sont notées s_{ij},

les indices ayant la même signification que pour la contrainte visqueuse. Il y a trois composantes de l'allongement linéaire :

$$s_{xx} = \frac{\partial u}{\partial x}, \quad s_{yy} = \frac{\partial v}{\partial y} \quad \text{et} \quad s_{zz} = \frac{\partial w}{\partial z}. \tag{2.43}$$

Il y a six composantes de déformation de cisaillement linéaire :

$$\begin{aligned} s_{xy} = s_{yx} &= \frac{1}{2}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right), \quad s_{xz} = s_{zx} = \frac{1}{2}\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right), \\ s_{xy} = s_{yz} &= \frac{1}{2}\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right). \end{aligned} \tag{2.44}$$

La déformation volumétrique est donnée par

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = \nabla \cdot \mathbf{u}. \tag{2.45}$$

Pour un fluide newtonien, les contraintes visqueuses sont proportionnelles aux taux de déformation. La forme tridimensionnelle de la loi de viscosité de Newton pour un fluide compressible suppose deux constantes de proportionnalité : la première viscosité (ou viscosité dynamique), μ^f, associée aux déformations linéaires, et la seconde viscosité, λ^f, associée à la déformation volumétrique. Les neuf composantes de la contrainte visqueuse sont

$$\begin{aligned} \tau_{xx} &= 2\mu^f \frac{\partial u}{\partial x} + \lambda^f \nabla \cdot \mathbf{u}, \quad \tau_{xy} = \tau_{yx} = \mu^f \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right), \\ \tau_{yy} &= 2\mu^f \frac{\partial v}{\partial y} + \lambda^f \nabla \cdot \mathbf{u}, \quad \tau_{xz} = \tau_{zx} = \mu^f \left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right), \\ \tau_{zz} &= 2\mu^f \frac{\partial w}{\partial z} + \lambda^f \nabla \cdot \mathbf{u}, \quad \tau_{yz} = \tau_{zy} = \mu^f \left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right). \end{aligned} \tag{2.46}$$

Ici, le fluide étant considéré **incompressible**, l'équation de conservation de la masse est $\nabla \cdot \mathbf{u} = 0$ et les contraintes visqueuses sont égales à deux fois le taux de déformation linéaire local multiplié par la viscosité dynamique.

Par substitution des équations de cisaillement ci-dessus (2.46) dans les équations de quantité de mouvement (2.22), (2.23) et (2.24), il vient les équations de Navier-Stokes, dues à deux scientifiques du XIX$^{\text{e}}$ siècle qui les ont dérivées indépendamment :

$$\begin{aligned} \rho^f \frac{Du}{Dt} = \; & -\frac{\partial p}{\partial x} + \frac{\partial}{\partial x}\left[2\mu^f \frac{\partial u}{\partial x} + \lambda^f \nabla \cdot \mathbf{u}\right] + \frac{\partial}{\partial y}\left[\mu^f \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\right] \\ & + \frac{\partial}{\partial z}\left[\mu^f \left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right)\right] + S_{M_x}, \end{aligned} \tag{2.47}$$

$$\begin{aligned} \rho^f \frac{Dv}{Dt} = \; & -\frac{\partial p}{\partial y} + \frac{\partial}{\partial x}\left[\mu^f \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\right] + \frac{\partial}{\partial y}\left[2\mu^f \frac{\partial u}{\partial y} + \lambda^f \nabla \cdot \mathbf{u}\right] \\ & + \frac{\partial}{\partial z}\left[\mu^f \left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right)\right] + S_{M_y}, \end{aligned} \tag{2.48}$$

$$\begin{aligned}\rho^f \frac{Dw}{Dt} = & -\frac{\partial p}{\partial z} + \frac{\partial}{\partial x}\left[\mu^f\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right)\right] + \frac{\partial}{\partial y}\left[\mu^f\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right)\right] \\ & + \frac{\partial}{\partial z}\left[2\mu^f \frac{\partial w}{\partial z} + \lambda^f \nabla \cdot \mathbf{u}\right] + S_{M_z},\end{aligned} \tag{2.49}$$

Il peut très souvent être utile de réarranger la contrainte visqueuse comme suit :

$$\begin{aligned}\frac{\partial}{\partial x} & \left[2\mu^f \frac{\partial u}{\partial x} + \lambda^f \nabla \cdot \mathbf{u}\right] + \frac{\partial}{\partial y}\left[\mu^f\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\right] + \frac{\partial}{\partial z}\left[\mu^f\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right)\right] \\ & = \frac{\partial}{\partial x}\left(\mu^f \frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(\mu^f \frac{\partial u}{\partial y}\right) + \frac{\partial}{\partial z}\left(\mu^f \frac{\partial u}{\partial z}\right) \\ & + \left[\frac{\partial}{\partial x}\left(\mu^f \frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(\mu^f \frac{\partial v}{\partial x}\right) + \frac{\partial}{\partial z}\left(\mu^f \frac{\partial w}{\partial x}\right) + \frac{\partial}{\partial x}(\lambda^f \nabla \cdot \mathbf{u})\right].\end{aligned} \tag{2.50}$$

La même chose peut être faite pour les composantes en y et en z des contraintes visqueuses. Les équations de quantité de mouvement peuvent être simplifiées en négligeant les petites contributions entre crochets par rapport aux termes de contrainte visqueuse présents dans la source en définissant une nouvelle source comme

$$S_M = S_M + [s_M], \tag{2.51}$$

les équations de Navier-Stokes peuvent alors être réécrites sous une forme plus compacte :

$$\rho^f \frac{Du}{Dt} = -\frac{\partial p}{\partial x} + \nabla \cdot (\mu^f \nabla u) + S_{M_x}, \tag{2.52}$$

$$\rho^f \frac{Dv}{Dt} = -\frac{\partial p}{\partial y} + \nabla \cdot (\mu^f \nabla v) + S_{M_y}, \tag{2.53}$$

$$\rho^f \frac{Dw}{Dt} = -\frac{\partial p}{\partial z} + \nabla \cdot (\mu^f \nabla w) + S_{M_z}. \tag{2.54}$$

La notation vectorielle des équations de Navier-Stokes pour un fluide **newtonien**, **incompressible** et **isotrope** est donnée par

$$\rho^f \left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u}\right) + \nabla p - \mu^f \nabla^2 \mathbf{u} - \mathbf{S} = 0, \tag{2.55}$$

où ρ^f est la masse volumique du fluide, μ^f est sa viscosité dynamique, $\mathbf{u}$ le vecteur vitesse, p est la pression et $\mathbf{S}$ est le vecteur des sources.

En utilisant le modèle newtonien pour les contraintes visqueuses dans l'équation d'énergie en température (2.42), il vient

$$\rho^f c \frac{DT}{Dt} = \nabla \cdot (k \nabla T) + \mathcal{D} + S_i. \tag{2.56}$$

Tous les effets dus aux contraintes visqueuses dans l'équation d'énergie sont décrits par la fonction de dissipation $\mathcal{D}$ qui, après des calculs considérables, est classiquement égale à

$$\begin{aligned}\mathcal{D} = \quad & \mu^f \Bigg\{ 2 \left[\left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial v}{\partial y} \right)^2 + \left(\frac{\partial w}{\partial z} \right)^2 \right] \\ & + \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} \right)^2 + \left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \right)^2 \Bigg\} \\ & + \lambda^f \left(\nabla \cdot \mathbf{u} \right)^2 .\end{aligned} \tag{2.57}$$

2.2.8 Équations régissant l'écoulement fluide

Les équations régissant l'écoulement d'un **fluide newtonien**, **incompressible** et **isotrope** se résument donc à l'équation de continuité (ou conservation de la masse) :

$$\nabla \cdot \mathbf{u} = 0, \tag{2.58}$$

et à l'équation de conservation de la quantité de mouvement :

$$\rho^f \left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} \right) + \nabla p - \mu^f \nabla^2 \mathbf{u} - \mathbf{S} = 0, \tag{2.59}$$

ainsi que l'équation d'énergie :

$$\rho^f c \left(\frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T \right) = \nabla \cdot (k \nabla T) + \mathcal{D} + S_i. \tag{2.60}$$

2.2.9 Formulation ALE

2.2.9.1 Formalisme lagrangien

Dans ce formalisme, le système de référence est lié à la matière ; chaque nœud est toujours lié à la même particule de matière. À tout instant, la configuration de la matière peut être déduite de la configuration de référence grâce à la loi du mouvement. Le maillage se déforme donc avec la matière, sa vitesse est égale à celle de la matière. C'est une formulation adoptée en mécanique des solides car elle permet de bien représenter l'évolution des surfaces libres et les conditions aux limites s'appliquent alors directement aux nœuds.

En revanche, comme le maillage se déforme avec la matière, pour de grandes déformations, les éléments du maillage se dégradent progressivement ce qui réduit alors la précision de la discrétisation spatiale ; il est donc nécessaire de remailler en faisant à chaque fois un transport d'information d'un maillage à l'autre.

2.2.9.2 Formalisme eulérien

Dans ce formalisme, le système de référence est fixe dans l'espace. Le maillage ne se déforme pas : on appelle d'ailleurs les éléments des volumes de contrôle. On cherche à

étudier les propriétés du fluide dans chaque volume de contrôle. Le fluide passe à travers des volumes et il faut donc résoudre des équations de transport. La vitesse du maillage est nulle ; on peut alors modéliser des grandes déformations puisque la qualité du maillage ne sera pas altérée au cours du temps.

La gestion des surfaces libres pose malheureusement de gros problèmes. En effet, ces frontières ne coïncident généralement pas avec le maillage et leurs positions doivent être calculées à chaque instant pour pouvoir y appliquer des conditions aux limites. De plus, comme il faut résoudre des équations de transport à chaque instant, il faudra différentes méthodes de résolution toujours plus complexes lorsque la vitesse du fluide augmentera.

2.2.9.3 Formalisme ALE

Dans le cadre des interactions fluide-structure, il devient nécessaire d'utiliser un formalisme plus général capable de conserver les avantages des formalismes lagrangien et eulérien. Le formalisme arbitraire Lagrangien-Eulérien (ALE) a donc été développé. Un nouveau système de référence qui peut se mouvoir indépendamment de la matière avec une vitesse arbitraire est alors défini [13]. Il y a donc un découplage entre le mouvement de la matière et celui du maillage. Il devient alors possible de garder un maillage de bonne qualité tout au long du calcul malgré de grandes déformations ; ce formalisme permet en outre de traiter des problèmes de surfaces libres bien plus simplement que pour un formalisme purement eulérien.

Cette formulation a d'abord été introduite pour traiter des problèmes de mécanique des fluides par les méthode des différences finies puis a ensuite été utilisée pour des problèmes d'interactions fluide-structure.

Sur la figure 2.6, les trois formalismes sont représentés de façon schématique pour une configuration de référence de la matière et du maillage à gauche et pour un instant ultérieur à droite. Dans le formalisme eulérien, la matière quitte l'espace de calcul et dans le formalisme lagrangien, le maillage subit des distorsions très importantes qui peuvent dégrader la qualité de la discrétisation spatiale. Dans le cas du formalisme ALE, le maillage s'est adapté pour garder une bonne qualité tout en respectant les frontières de la matière.

C'est cette formulation, bien adaptée à la simulation d'interactions fluide-structure, qui est utilisée dans cette étude pour résoudre les équations de Navier-Stokes.

2.2.9.4 Formulation mathématique

Chaque particule d'un milieu continu peut être repérée dans un des trois sytèmes de référence suivant :

- le système de référence matériel (SRM), dans lequel les coordonnées sont notées X ;
- le système de référence spatial (SRS), dans lequel les coordonnées sont notées x ;
- le système de référence relatif à la grille de calcul (SRG), dans lequel les coordonnées sont notées χ.

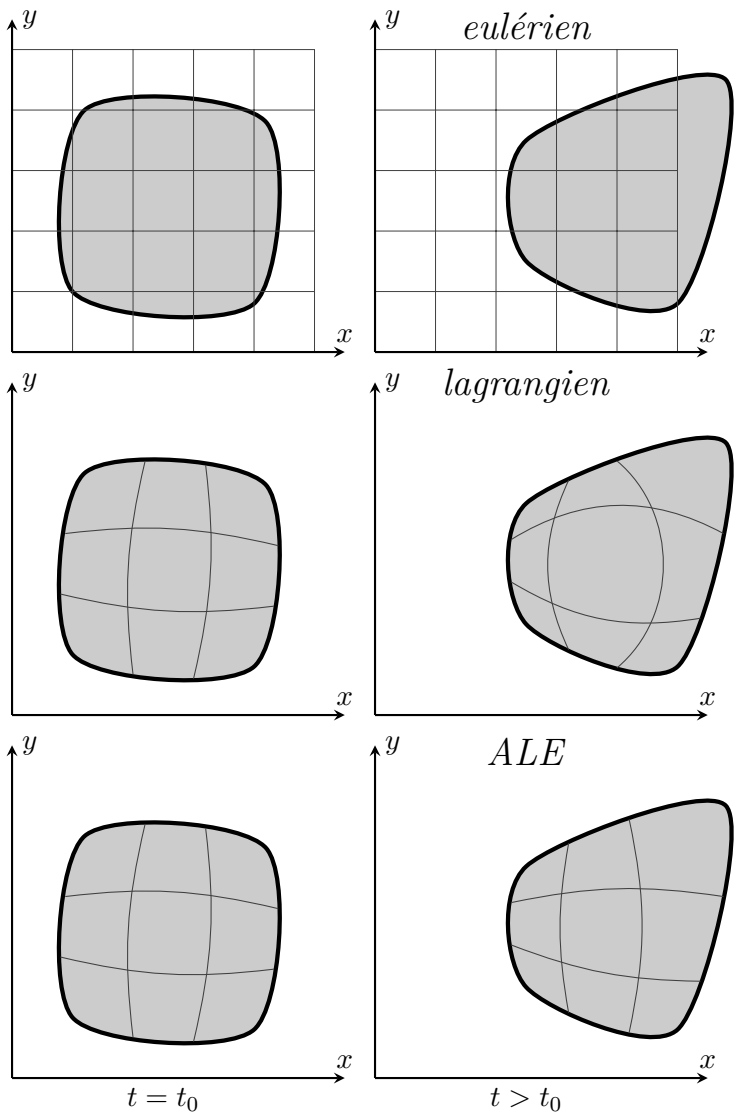


FIGURE 2.6 – Les trois formalismes utilisés en mécanique des milieux continus [13]

La figure 2.7 représente la cinématique de la matière et de la grille de calcul ; trois applications ϕ, ϕ^* et ψ^* sont définies pour passer d'un système de référence à un autre. Il est par exemple possible de relier le SRG et le SRM par la relation suivante en combinant les deux transformations principales ϕ et ϕ^* :

$$\chi = \phi^{*-1}(x,t) = \phi^{*-1}(\phi(X,t),t) = \psi^*(X,t). \tag{2.61}$$

Il est alors possible de retrouver facilement la formulation lagrangienne en identifiant les vecteurs X et χ (ce qui revient à dire que ψ^* est la fonction identité) ; le SRG est alors confondu avec le SRM. En identifiant les vecteurs χ et x, la formulation eulérienne

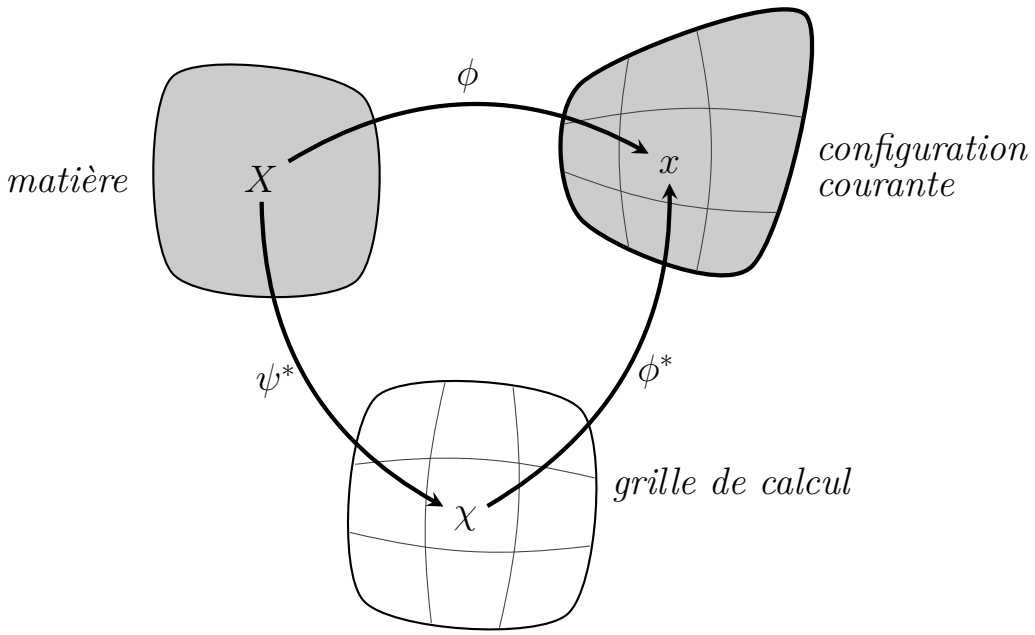


FIGURE 2.7 – Cinématique de la matière (en gris) et de la grille de calcul [13]

apparaît et le SRG et le SRS sont confondus.

Une grandeur physique arbitraire f peut être exprimée de plusieurs manières selon le système de référence, par exemple sur le SRG, f est donnée par $f = f(\chi, t)$. Donc, si f représente la position courante x, sa dérivée par rapport au temps dans le SRG représente la **vitesse de grille** $\mathbf{u}^*$ ou vitesse SRG :

$$\mathbf{u}^*(\chi, t) = \left.\frac{\partial x(\chi, t)}{\partial t}\right|_{\chi} = \left.\frac{\partial \phi^*(\chi, t)}{\partial t}\right|_{\chi}. \tag{2.62}$$

De manière plus générale, les dérivées temporelles de la grandeur physique f peuvent être exprimées dans les systèmes SRM et SRG puis les règles de dérivation d'une fonction composée peuvent être appliquées :

$$\left.\frac{\partial f(x,t)}{\partial t}\right|_X = \left.\frac{\partial f(\phi(X,t),t)}{\partial t}\right|_X = \left.\frac{\partial f}{\partial t}\right|_\phi + \left.\frac{\partial \phi}{\partial t}\right|_X \cdot \frac{\partial f}{\partial \phi} = \left.\frac{\partial f}{\partial t}\right|_x + \mathbf{u} \cdot \nabla_x f, \tag{2.63}$$

où $\mathbf{u}$ est la **vitesse matérielle**. De même dans le SRG :

$$\left.\frac{\partial f(x,t)}{\partial t}\right|_X = \left.\frac{\partial f}{\partial t}\right|_x + \mathbf{u}^* \cdot \nabla_x f. \tag{2.64}$$

Finalement, la relation fondamentale du formalisme ALE s'écrit

$$\left.\frac{\partial f(x,t)}{\partial t}\right|_X = \left.\frac{\partial f(x,t)}{\partial t}\right|_\chi + (\mathbf{u} - \mathbf{u}^*) \cdot \nabla_x f. \tag{2.65}$$

La **vitesse convective** $(\mathbf{u} - \mathbf{u}^*)$ est alors introduite ; elle représente la vitesse relative entre SRM et SRG. La relation fondamentale (2.65) permet de retrouver les deux cas particuliers contenus dans la formulation ALE, à savoir :

- la formulation lagrangienne si $\mathbf{u}^* = \mathbf{u}$, le maillage est alors lié à la matière ;
- la formulation eulérienne si $\mathbf{u}^* = 0$, le maillage est fixe dans l'espace.

Dans le cadre des équations de Navier-Stokes, l'équation de conservation de la quantité de mouvement (2.55) dans un formalisme ALE devient

$$\rho^f \left(\left. \frac{\partial \mathbf{u}}{\partial t} \right|_\chi + (\mathbf{u} - \mathbf{u}^*) \cdot \nabla \mathbf{u} \right) + \nabla p - \mu^f \nabla^2 \mathbf{u} - \mathbf{S} = 0. \tag{2.66}$$

où $\mathbf{u}^*$ vérifie une équation permettant de déterminer la déformation du maillage. Dans notre cas nous choisissons une équation de diffusion homogène avec des conditions aux limites adaptées au problème considéré :

$$\nabla^2 \left(\Gamma_f \mathbf{u}^* \right) = 0, \tag{2.67}$$

où Γ_f est un coefficient de diffusion.

2.3 Structure

Dans cette section, on suppose le matériau homogène.

2.3.1 Équations d'équilibre des contraintes

Soit un parallélépipède cartésien en trois dimensions noté V ayant des côtés de longueurs dx, dy et dz. Il possède six facettes notées dS, on note $\mathbf{n}$ le vecteur unitaire normal à la facette dirigé vers l'extérieur de V. Chacune de ses facettes est soumise à un vecteur contrainte défini par :

$$\mathbf{T} = \frac{d\mathbf{F}}{dS} \quad , \tag{2.68}$$

ce vecteur se décompose en sa composante normale suivant $\mathbf{n}$ appelée contrainte normale et sa projection sur la facette appelée contrainte de cisaillement. La figure 2.8 représente les contraintes agissant sur un élément cartésien en deux dimensions. Le premier indice indique une des deux facettes sur laquelle la contrainte agit alors que le second indice indique la direction dans laquelle elle agit.

Par exemple, les contraintes agissant sur le plan x de gauche sont σ_{xx}, τ_{xy} et τ_{xz}, les contraintes agissant sur le plan x de droite sont $(\sigma_{xx} + \Delta\sigma_{xx})$, $(\tau_{xy} + \Delta\tau_{xy})$ et $(\tau_{xz} + \Delta\tau_{xz})$. Avec l'hypothèse des petites déformations, elles peuvent être approximées avec une décomposition en série de Taylor de la forme

$$\sigma_{xx} + \Delta\sigma_{xx} = \sigma_{xx} + \frac{\partial \sigma_{xx}}{\partial x} dx, \tag{2.69}$$

$$\tau_{xy} + \Delta\tau_{xy} = \tau_{xy} + \frac{\partial \tau_{xy}}{\partial y} dy, \tag{2.70}$$

$$\tau_{xz} + \Delta\tau_{xz} = \tau_{xz} + \frac{\partial \tau_{xz}}{\partial z} dz. \tag{2.71}$$

Des expressions similaires peuvent être écrites pour les plans y et z. Le volume de contrôle peut aussi être sujet à des forces internes telles que la gravité, les forces inertielles, les forces magnétiques, etc. Les trois composantes du vecteur de forces externes sont f_x^s, f_y^s et f_z^s.

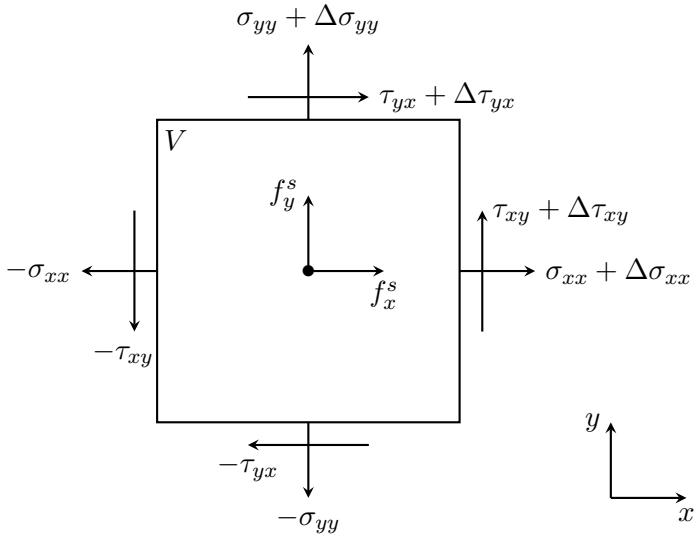


FIGURE 2.8 – Forces internes (contraintes) et forces externes agissant sur l'élément V

D'après la seconde loi de Newton, le vecteur somme des forces $\mathbf{F}$ (internes et externes) sur V est égal à sa masse m multipliée par son vecteur accélération $\mathbf{a}$:

$$\mathbf{F} = m\mathbf{a}. \tag{2.72}$$

Donc la condition d'équilibre des forces dans la direction x peut s'écrire

$$\sum F_x = \frac{\partial^2(\rho^s D_x)}{\partial t^2} dx\, dy\, dz, \tag{2.73}$$

où F_x est la composante selon x du vecteur somme des forces $\mathbf{F}$, ρ^s est la masse volumique du matériau et D_x est la composante selon x du vecteur déplacement ; par conséquent sa dérivée seconde en fonction du temps représente l'accélération de l'élément dans la direction x.

Par sommation des forces agissant sur V dans la direction x, la condition d'équilibre peut s'écrire

$$\begin{aligned}\frac{\partial^2(\rho^s D_x)}{\partial t^2}dx\,dy\,dz = \quad & [-\sigma_{xx} + (\sigma_{xx} + \Delta\sigma_{xx})]\,dy\,dz \\ & + [-\tau_{xy} + (\tau_{xy} + \Delta\tau_{xy})]\,dx\,dz \\ & + [-\tau_{xz} + (\tau_{xz} + \Delta\tau_{xz})]\,dx\,dy + f_x^s dx\,dy\,dz.\end{aligned} \tag{2.74}$$

Après substitution des équations (2.69), (2.70) et (2.71), les équations ci-dessus deviennent

$$\begin{aligned}\frac{\partial^2(\rho^s D_x)}{\partial t^2}dx\,dy\,dz = \quad & \left[-\sigma_{xx} + \left(\sigma_{xx} + \frac{\partial\sigma_{xx}}{\partial x}dx\right)\right]dy\,dz \\ & + \left[-\tau_{xy} + \left(\tau_{xy} + \frac{\partial\tau_{xy}}{\partial y}dy\right)\right]dx\,dz \\ & + \left[-\tau_{xz} + \left(\tau_{xz} + \frac{\partial\tau_{xz}}{\partial z}dz\right)\right]dx\,dy + f_x^s dx\,dy\,dz,\end{aligned} \tag{2.75}$$

qui, après annulation, se simplifie :

$$\frac{\partial^2(\rho^s D_x)}{\partial t^2} - \left(\frac{\partial\sigma_{xx}}{\partial x} + \frac{\partial\tau_{xy}}{\partial y} + \frac{\partial\tau_{xz}}{\partial z}\right) = f_x^s. \tag{2.76}$$

De la même façon, les équations d'équilibre des forces peuvent s'écrire dans les directions y et z :

$$\frac{\partial^2(\rho^s D_y)}{\partial t^2} - \left(\frac{\partial\tau_{xy}}{\partial x} + \frac{\partial\sigma_{yy}}{\partial y} + \frac{\partial\tau_{yz}}{\partial z}\right) = f_y^s, \tag{2.77}$$

$$\frac{\partial^2(\rho^s D_z)}{\partial t^2} - \left(\frac{\partial\tau_{xz}}{\partial x} + \frac{\partial\tau_{yz}}{\partial y} + \frac{\partial\sigma_{zz}}{\partial z}\right) = f_z^s. \tag{2.78}$$

Ce qui, en notation vectorielle, donne

$$\frac{\partial^2(\rho^s \mathbf{D})}{\partial t^2} - \nabla\cdot\sigma = \mathbf{f}^s, \tag{2.79}$$

où σ est le tenseur des contraintes de Cauchy, $\mathbf{D}$ est le vecteur déplacement et $\mathbf{f}^s$ est le vecteur des forces externes :

$$\sigma = \begin{bmatrix}\sigma_{xx} & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \sigma_{yy} & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \sigma_{zz}\end{bmatrix}, \quad \mathbf{D} = \begin{pmatrix}D_x \\ D_y \\ D_z\end{pmatrix}, \quad \mathbf{f}^s = \begin{pmatrix}f_x^s \\ f_y^s \\ f_z^s\end{pmatrix}. \tag{2.80}$$

Le tenseur des contraintes est symétrique ce qui traduit l'équilibre statique, et signifie que $\tau_{xy} = \tau_{yx}$, $\tau_{xz} = \tau_{zx}$ et $\tau_{yz} = \tau_{zy}$; d'où la notation de Voigt du tenseur des contraintes :

$$\{\sigma\} = (\sigma_{xx} \quad \sigma_{yy} \quad \sigma_{zz} \quad \tau_{xy} \quad \tau_{xz} \quad \tau_{yz})^T. \tag{2.81}$$

2.3.2 Déformation

Sous l'action des forces appliquées, les points d'un solide se déplacent. Il en résulte, pour des fibres infinitésimales de matière, des variations de longueur et des variations d'angle appelées déformations.

On appelle configuration courante C_t, le volume occupé par le solide à l'instant t et C_0 la configuration initiale.

Le vecteur infiniment petit $d\mathbf{x_0}$ en un point M_0 de C_0 devient $d\mathbf{x}$ en M dans la configuration courante C_t.

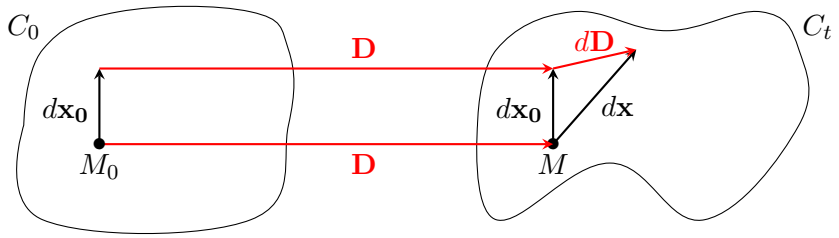


FIGURE 2.9 – Transformation du vecteur $d\mathbf{x_0}$

Notant ds_0 la longueur du vecteur $d\mathbf{x_0}$ et ds celle du vecteur $d\mathbf{x}$, on montre que :

$$ds^2 + ds_0^2 = 2(d\mathbf{x_0})^T \mathbb{E} d\mathbf{x_0}, \tag{2.82}$$

où

$$\begin{aligned} \mathbb{E} &= \tfrac{1}{2}\left(L^T + L\right) + \tfrac{1}{2}L^T L \\ &= \tfrac{1}{2}\left(\nabla\mathbf{D} + (\nabla\mathbf{D})^T\right) + \tfrac{1}{2}\nabla\mathbf{D}^T\nabla\mathbf{D}, \end{aligned} \tag{2.83}$$

avec

$$L = \begin{bmatrix} \dfrac{\partial D_x}{\partial x_0} & \dfrac{\partial D_x}{\partial y_0} & \dfrac{\partial D_x}{\partial z_0} \\ \dfrac{\partial D_y}{\partial x_0} & \dfrac{\partial D_y}{\partial y_0} & \dfrac{\partial D_y}{\partial z_0} \\ \dfrac{\partial D_z}{\partial x_0} & \dfrac{\partial D_z}{\partial y_0} & \dfrac{\partial D_z}{\partial z_0} \end{bmatrix} = \nabla\mathbf{D}, \tag{2.84}$$

$\mathbb{E}$ est appelé tenseur des déformations de Green-Lagrange.

2.3.2.1 Hypothèse des petits déplacements et des petites déformations - Élasticité linéaire

Les deux hypothèses suivantes seront faites :

1. les déplacements sont petits par rapport aux dimensions du solide ;
2. les dérivées des déplacements par rapport à x_0, y_0 et z_0 sont petites devant l'unité :

$$\left|\frac{\partial D_x}{\partial x_0}\right| \ll 1, \quad \left|\frac{\partial D_x}{\partial y_0}\right| \ll 1, \ \ldots \tag{2.85}$$

Dans ces conditions, si f est une fonction de x_0, y_0 et z_0, on a

$$\frac{\partial f}{\partial x_0} = \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y_0} = \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z_0} = \frac{\partial f}{\partial z}. \tag{2.86}$$

Le tenseur des déformations de Green-Lagrange se réduit alors à

$$\mathbb{E} = \frac{1}{2}\left(\nabla \mathbf{D} + (\nabla \mathbf{D})^T\right) = \varepsilon, \tag{2.87}$$

où ε est le tenseur des déformations linéarisé :

$$\varepsilon = \begin{bmatrix} \varepsilon_{xx} & \varepsilon_{xy} & \varepsilon_{xz} \\ \varepsilon_{yx} & \varepsilon_{yy} & \varepsilon_{yz} \\ \varepsilon_{zx} & \varepsilon_{zy} & \varepsilon_{zz} \end{bmatrix} = \begin{bmatrix} \frac{\partial D_x}{\partial x} & \frac{1}{2}\left(\frac{\partial D_x}{\partial y} + \frac{\partial D_y}{\partial x}\right) & \frac{1}{2}\left(\frac{\partial D_x}{\partial z} + \frac{\partial D_z}{\partial x}\right) \\ \frac{1}{2}\left(\frac{\partial D_x}{\partial y} + \frac{\partial D_y}{\partial x}\right) & \frac{\partial D_y}{\partial y} & \frac{1}{2}\left(\frac{\partial D_y}{\partial z} + \frac{\partial D_z}{\partial y}\right) \\ \frac{1}{2}\left(\frac{\partial D_x}{\partial z} + \frac{\partial D_z}{\partial x}\right) & \frac{1}{2}\left(\frac{\partial D_y}{\partial z} + \frac{\partial D_z}{\partial y}\right) & \frac{\partial D_z}{\partial z} \end{bmatrix}. \tag{2.88}$$

2.3.3 Loi de Hooke

Pour un matériau isotrope, en un point donné du solide, le matériau a les mêmes propriétés dans toutes les directions, la déformation est proportionnelle à la contrainte appliquée,[7] ce qui peut s'écrire :

$$\begin{aligned} \varepsilon_{xx} &= \tfrac{1}{E}\left(\sigma_{xx} - \nu(\sigma_{yy} + \sigma_{zz})\right), \\ \varepsilon_{yy} &= \tfrac{1}{E}\left(\sigma_{yy} - \nu(\sigma_{xx} + \sigma_{zz})\right), \\ \varepsilon_{zz} &= \tfrac{1}{E}\left(\sigma_{zz} - \nu(\sigma_{xx} + \sigma_{yy})\right), \\ 2\varepsilon_{xy} &= \frac{\tau_{xy}}{\mu}, \\ 2\varepsilon_{xz} &= \frac{\tau_{xz}}{\mu}, \\ 2\varepsilon_{yz} &= \frac{\tau_{yz}}{\mu}, \end{aligned} \tag{2.89}$$

où E est le module de Young ou module d'élasticité longitudinale, ν est le coefficient de Poisson ($0 \leq \nu \leq 1/2$), μ est le module de cisaillement donné par $\mu = E/(2(1+\nu))$.

Ces relations se réécrivent en utilisant le tenseur des contraintes σ et le tenseur des déformations ε :

$$\varepsilon = \frac{1+\nu}{E}\sigma - \frac{\nu}{E}(\mathrm{tr}(\sigma))\mathbf{I}. \tag{2.90}$$

Exprimant σ en fonction de ε, il vient

$$\begin{Bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \tau_{xy} \\ \tau_{xz} \\ \tau_{yz} \end{Bmatrix} = \begin{bmatrix} \lambda+2\mu & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda+2\mu & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & \lambda+2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & 2\mu & 0 & 0 \\ 0 & 0 & 0 & 0 & 2\mu & 0 \\ 0 & 0 & 0 & 0 & 0 & 2\mu \end{bmatrix} \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \varepsilon_{xy} \\ \varepsilon_{xz} \\ \varepsilon_{yz} \end{Bmatrix}, \tag{2.91}$$

[7] La loi de Hooke est un principe physique, Robert Hooke a dit : « ut tensio, sic vis », ce qui signifie « l'allongement est proportionnel à la force »

où λ et μ sont les constantes de Lamé :

$$\lambda = \frac{\nu E}{(1+\nu)(1-2\nu)}, \tag{2.92}$$

$$\mu = \frac{E}{2(1+\nu)}. \tag{2.93}$$

Le tenseur des déformations est définie en fonction de $\mathbf{D}$:

$$\varepsilon = \frac{1}{2}\left(\nabla\mathbf{D} + (\nabla\mathbf{D})^T\right). \tag{2.94}$$

Ainsi, la loi de Hooke peut être écrite en notation vectorielle :

$$\sigma = 2\mu\varepsilon + \lambda\left(\nabla\cdot\mathbf{D}\right)\mathbf{I}, \tag{2.95}$$

ou

$$\sigma = 2\mu\varepsilon + \lambda\,\mathrm{tr}(\varepsilon)\mathbf{I}. \tag{2.96}$$

2.3.4 Équation du déplacement de Navier

En utilisant l'équation (2.95), l'équation (2.79) d'équilibre peut être réécrite avec le vecteur déplacement $\mathbf{D}$ comme variable principale. Par exemple, en coordonnées cartésiennes dans la direction x il vient

$$\begin{aligned}
\sigma_{xx} &= 2\mu\frac{\partial D_x}{\partial x} + \lambda\left(\frac{\partial D_x}{\partial x} + \frac{\partial D_y}{\partial y} + \frac{\partial D_z}{\partial z}\right),\\
\tau_{xy} &= \mu\left(\frac{\partial D_x}{\partial y} + \frac{\partial D_y}{\partial x}\right),\\
\tau_{xz} &= \mu\left(\frac{\partial D_x}{\partial z} + \frac{\partial D_z}{\partial x}\right).
\end{aligned} \tag{2.97}$$

Après substitution des équations ci-dessus dans l'équation (2.76), il vient

$$\begin{aligned}
\frac{\partial^2(\rho^s D_x)}{\partial t^2} \quad & -\frac{\partial}{\partial x}\left[2\mu\frac{\partial D_x}{\partial x} + \lambda\left(\frac{\partial D_x}{\partial x} + \frac{\partial D_y}{\partial y} + \frac{\partial D_z}{\partial z}\right)\right]\\
& -\frac{\partial}{\partial y}\left[\mu\left(\frac{\partial D_x}{\partial y} + \frac{\partial D_y}{\partial x}\right)\right] - \frac{\partial}{\partial z}\left[\mu\left(\frac{\partial D_x}{\partial z} + \frac{\partial D_z}{\partial x}\right)\right] = f_x^s,
\end{aligned} \tag{2.98}$$

et

$$\begin{aligned}
\frac{\partial^2(\rho^s D_x)}{\partial t^2} \quad & -\left[2\mu\frac{\partial^2 D_x}{\partial x^2} + \lambda\frac{\partial^2 D_x}{\partial x^2} + \lambda\frac{\partial^2 D_y}{\partial x\partial y} + \lambda\frac{\partial^2 D_z}{\partial x\partial z}\right.\\
& \left.+\mu\frac{\partial^2 D_x}{\partial y^2} + \mu\frac{\partial^2 D_y}{\partial x\partial y} + \mu\frac{\partial^2 D_x}{\partial z^2} + \mu\frac{\partial^2 D_z}{\partial x\partial z}\right] = f_x^s,
\end{aligned} \tag{2.99}$$

qui peut être réécrite sous la forme

$$\begin{aligned}
\frac{\partial^2(\rho^s D_x)}{\partial t^2} \quad & -(\lambda+\mu)\frac{\partial}{\partial x}\left[\frac{\partial D_x}{\partial x} + \frac{\partial D_y}{\partial y} + \frac{\partial D_z}{\partial z}\right]\\
& -\mu\left[\frac{\partial^2 D_x}{\partial x^2} + \frac{\partial^2 D_x}{\partial y^2} + \frac{\partial^2 D_x}{\partial z^2}\right] = f_x^s.
\end{aligned} \tag{2.100}$$

Des équations similaires peuvent être écrites pour les directions y et z :

$$\begin{aligned}\frac{\partial^2(\rho^s D_y)}{\partial t^2} \quad & -(\lambda+\mu)\frac{\partial}{\partial y}\left[\frac{\partial D_x}{\partial x}+\frac{\partial D_y}{\partial y}+\frac{\partial D_z}{\partial z}\right] \\ & -\mu\left[\frac{\partial^2 D_y}{\partial x^2}+\frac{\partial^2 D_y}{\partial y^2}+\frac{\partial^2 D_y}{\partial z^2}\right]=f_y^s,\end{aligned} \tag{2.101}$$

$$\begin{aligned}\frac{\partial^2(\rho^s D_z)}{\partial t^2} \quad & -(\lambda+\mu)\frac{\partial}{\partial z}\left[\frac{\partial D_x}{\partial x}+\frac{\partial D_y}{\partial y}+\frac{\partial D_z}{\partial z}\right] \\ & -\mu\left[\frac{\partial^2 D_z}{\partial x^2}+\frac{\partial^2 D_z}{\partial y^2}+\frac{\partial^2 D_z}{\partial z^2}\right]=f_z^s.\end{aligned} \tag{2.102}$$

En notation vectorielle, l'équation de Navier pour l'élasticité linéaire est donnée par :

$$\frac{\partial^2(\rho^s \mathbf{D})}{\partial t^2}-(\lambda+\mu)\nabla(\nabla\cdot\mathbf{D})-\mu\nabla^2\mathbf{D}=\mathbf{f}^s, \tag{2.103}$$

ou, après substitution de l'équation (2.95) dans (2.79) :

$$\frac{\partial^2(\rho^s \mathbf{D})}{\partial t^2}-\nabla\cdot\left[\mu\left(\nabla\mathbf{D}+(\nabla\mathbf{D})^T\right)+\lambda\mathbf{I}\left(\nabla\cdot\mathbf{D}\right)\right]=\mathbf{f}^s, \tag{2.104}$$

ou de l'équation (2.96) dans (2.79) :

$$\frac{\partial^2(\rho^s \mathbf{D})}{\partial t^2}-\nabla\cdot\left[\mu\left(\nabla\mathbf{D}+(\nabla\mathbf{D})^T\right)+\lambda\mathbf{I}\,\mathrm{tr}\left(\nabla\mathbf{D}\right)\right]=\mathbf{f}^s. \tag{2.105}$$

On peut réécrire ces équations de différentes manières :

$$\frac{\partial^2(\rho^s \mathbf{D})}{\partial t^2}-\nabla\cdot\left[(2\mu+\lambda)\nabla\mathbf{D}-(\mu+\lambda)\nabla\mathbf{D}+\mu\left(\nabla\mathbf{D}\right)^T+\lambda\mathbf{I}\,\mathrm{tr}\left(\nabla\mathbf{D}\right)\right]=\mathbf{f}^s, \tag{2.106}$$

$$\frac{\partial^2(\rho^s \mathbf{D})}{\partial t^2}-\nabla\cdot\left[(2\mu+\lambda)\nabla\mathbf{D}\right]-\nabla\cdot\left[\mu\left(\nabla\mathbf{D}\right)^T-(\mu+\lambda)\nabla\mathbf{D}+\lambda\mathbf{I}\,\mathrm{tr}\left(\nabla\mathbf{D}\right)\right]=\mathbf{f}^s, \tag{2.107}$$

$$\frac{\partial^2(\rho^s \mathbf{D})}{\partial t^2}-(2\mu+\lambda)\nabla^2\mathbf{D}-\nabla\cdot\left[\mu\left(\nabla\mathbf{D}\right)^T-(\mu+\lambda)\nabla\mathbf{D}+\lambda\mathbf{I}\,\mathrm{tr}\left(\nabla\mathbf{D}\right)\right]=\mathbf{f}^s. \tag{2.108}$$

En trois dimensions et en coordonnées cartésiennes selon la direction x :

$$\begin{aligned}\frac{\partial^2(\rho^s D_x)}{\partial t^2}-(2\mu+\lambda) \quad & \left[\frac{\partial^2 D_x}{\partial x^2}+\frac{\partial^2 D_x}{\partial y^2}+\frac{\partial^2 D_x}{\partial z^2}\right] \\ -\mu \quad & \left[\frac{\partial^2 D_x}{\partial x^2}+\frac{\partial^2 D_y}{\partial x^2}+\frac{\partial^2 D_z}{\partial x^2}\right] \\ -\lambda \quad & \left[\frac{\partial^2 D_x}{\partial x^2}+\frac{\partial^2 D_y}{\partial x\partial y}+\frac{\partial^2 D_z}{\partial x\partial z}\right] \\ +(\mu+\lambda) \quad & \left[\frac{\partial^2 D_x}{\partial x^2}+\frac{\partial^2 D_x}{\partial y^2}+\frac{\partial^2 D_x}{\partial z^2}\right]=f_x^s,\end{aligned} \tag{2.109}$$

selon la direction y :

$$\begin{aligned}
\frac{\partial^2(\rho^s D_y)}{\partial t^2} - (2\mu+\lambda) & \left[\frac{\partial^2 D_y}{\partial x^2} + \frac{\partial^2 D_y}{\partial y^2} + \frac{\partial^2 D_y}{\partial z^2}\right] \\
-\mu & \left[\frac{\partial^2 D_x}{\partial y^2} + \frac{\partial^2 D_y}{\partial y^2} + \frac{\partial^2 D_z}{\partial y^2}\right] \\
-\lambda & \left[\frac{\partial^2 D_x}{\partial x \partial y} + \frac{\partial^2 D_y}{\partial y^2} + \frac{\partial^2 D_z}{\partial y \partial z}\right] \\
+(\mu+\lambda) & \left[\frac{\partial^2 D_y}{\partial x^2} + \frac{\partial^2 D_y}{\partial y^2} + \frac{\partial^2 D_y}{\partial z^2}\right] = f_y^s,
\end{aligned} \tag{2.110}$$

et selon la direction z :

$$\begin{aligned}
\frac{\partial^2(\rho^s D_z)}{\partial t^2} - (2\mu+\lambda) & \left[\frac{\partial^2 D_z}{\partial x^2} + \frac{\partial^2 D_z}{\partial y^2} + \frac{\partial^2 D_z}{\partial z^2}\right] \\
-\mu & \left[\frac{\partial^2 D_x}{\partial z^2} + \frac{\partial^2 D_y}{\partial z^2} + \frac{\partial^2 D_z}{\partial z^2}\right] \\
-\lambda & \left[\frac{\partial^2 D_x}{\partial x \partial z} + \frac{\partial^2 D_y}{\partial y \partial z} + \frac{\partial^2 D_z}{\partial z^2}\right] \\
+(\mu+\lambda) & \left[\frac{\partial^2 D_z}{\partial x^2} + \frac{\partial^2 D_z}{\partial y^2} + \frac{\partial^2 D_z}{\partial z^2}\right] = f_z^s.
\end{aligned} \tag{2.111}$$

2.4 Couplage fluide-structure

Des phénomènes sont dits couplés quand l'évolution de chacun des deux éléments dépend de celle de l'autre [26, 39]. Les interactions fluide-structure font partie des phénomènes couplés qui mettent en jeu une structure toujours mobile, rigide ou déformable, et un fluide liquide ou gazeux, en écoulement autour ou contre une partie de la structure. Il existe un très grand nombre d'exemples d'interactions fluide-structure, deux catégories se distinguent cependant :

- les phénomènes hydroélastiques (fluide en phase liquide) : écoulements autour d'un navire, d'un sous-marin, écoulements liquides à l'intérieur de conduites, mouvement de liquides dans un réservoir, etc.
- les phénomènes aéroélastiques (fluide en phase gazeuse) : écoulements autour des véhicules aériens (avions, missiles, etc.) et terrestres (trains à grande vitesse, automobiles, etc.), influence du vent sur les constructions souples (ponts suspendus, etc.).

L'approche « monolithique » qui consiste à écrire une formulation complète couplant toutes les inconnues est très lourde et très difficile à développer à l'exception de certains cas simples. C'est une approche qui demande un développement spécifique ce qui rend très difficile sa réutilisation pour d'autres cas.

L'approche la plus pratique et la plus largement utilisée consiste à introduire un couplage plus ou moins fort entre des sous-systèmes. Le système global couplé est résolu sous-système par sous-système de manière à pouvoir réutiliser des codes déjà existants et ayant été optimisés pour chaque sous-système. Cette approche plus flexible permet également de s'adapter plus facilement à de nouveaux problèmes. Des variables sont échangées à l'interface fluide-structure, ces communications correspondent aux forces d'action et de réaction échangées au niveau physique. Contrairement à la plupart des problèmes couplés, le couplage physique ne se fait que par l'intermédiaire d'une surface et non pas de manière volumique, les échanges d'informations entre les solveurs fluide et structure peuvent donc se limiter à l'interface fluide-structure.

La résolution numérique de problème d'interaction fluide-structure nécessite donc les éléments suivants :

- un **code de dynamique des structures** qui permet de calculer l'évolution dans le temps de la structure, c'est-à-dire résoudre les équations aux dérivées partielles choisies pour modéliser la structure, ici les équations de Navier (2.103). Il doit être capable de recevoir les informations représentant les forces physiques exercées par le fluide sur la structure à l'interface fluide-structure, d'avancer d'un pas de temps et d'envoyer les déplacements voire les vitesses au code de dynamique des fluides ;

- un **code de dynamique des fluides** qui permet de calculer l'évolution dans le temps du fluide, c'est-à-dire résoudre les équations au dérivées partielles choisies pour modéliser le fluide, ici les équations de Navier-Stokes (2.58), (2.59) et (2.60). Il doit être capable de recevoir les informations représentant le déplacement d'une partie du bord de son domaine de calcul ; c'est-à-dire l'interface entre le fluide et la structure, d'avancer d'un pas de temps dans un domaine déformable en utilisant la formulation ALE et d'envoyer les forces exercées sur l'interface fluide-structure au code de dynamique des structures ;

- une **interface de couplage en espace** qui permet l'adaptation des forces et des déplacements lorsque les maillages ou les formulations adoptées dans chacun des sous-systèmes sont différents ;

- une **interface de couplage en temps** qui gère les échanges d'informations entre les solveurs dans le temps. C'est cette interface qui va déterminer, de part la fréquence des échanges, la force du couplage fluide-structure.

2.4.1 Couplage en espace

La complexité du couplage en espace varie en fonction des types de discrétisation en espace choisis pour chacun des sous-système. La discrétisation comprend non seulement le maillage mais aussi le type d'approximation utilisé : volumes finis, éléments finis, etc.

Dans le cas idéal, les maillages et les formulations sont identiques pour les deux solveurs comme illustré sur la figure 2.10. Le transfert d'informations est alors intuitif

puisqu'il suffit de communiquer les degrés de liberté correspondants. L'interface de couplage en espace se limite donc à l'adaptation des forces et des déplacements entre les deux codes :

- le code de dynamique des fluides transfère au code de dynamique des structures les pressions et les contraintes de cisaillement dues à la viscosité du fluide ;
- le code de dynamique des structures transfère les déplacements voire les vitesses afin que le code de dynamique des fluides prenne en compte la déformation de son domaine de calcul.

Le couplage en espace devient plus complexe si l'on considère des formulations identiques mais des maillages différents ou des maillages identiques mais des formulations différentes. Dans le pire des cas, on peut imaginer des maillages et des formulations différents, le rôle de l'interface de couplage en espace sera alors de traduire et d'approximer les informations afin de pouvoir effectuer les échanges entre chaque code. Ces approximations dans l'échange des forces peuvent modifier significativement les résultats numériques et altérer leur interprétation. C'est pourquoi nous nous limiterons ici à des maillages et des approximations par les volumes finis identiques pour les deux modèles physiques.

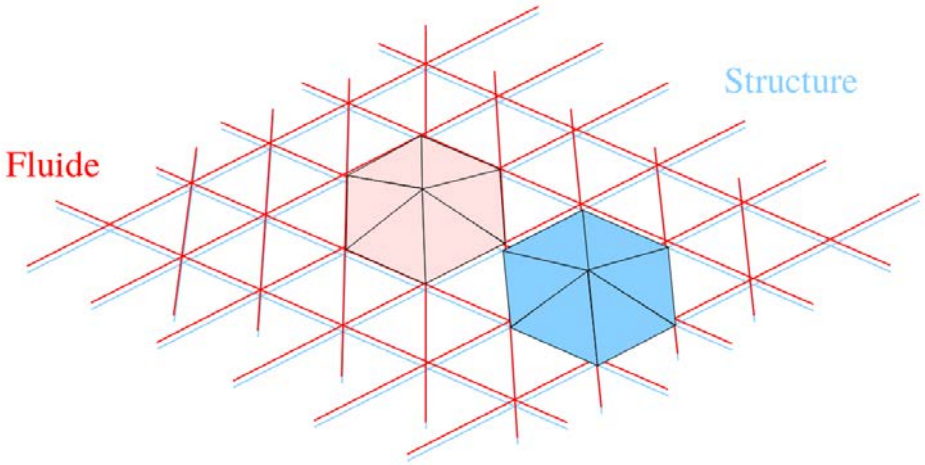


FIGURE 2.10 – Maillages et formulations identiques [68]

2.4.2 Couplage en temps

L'interface de couplage en temps a pour rôle d'organiser les échanges d'informations entre les solveurs dans le temps. Dans la réalité, ces échanges sont effectifs à tout moment, vérifiant le principe d'action et de réaction. Du fait de la discrétisation en temps et de l'utilisation de solveurs séparés pour résoudre les équations gouvernant le fluide et la structure, il est presque impossible de vérifier le principe d'action et de réaction à chaque

instant. L'objectif de l'algorithme de couplage en temps est donc de respecter au mieux ce principe physique qui régit l'interaction fluide-structure.

Plusieurs catégories d'algorithmes de couplage en temps peuvent être distinguées :

- les **algorithmes décalés** où chaque sous-système est avancé en temps successivement ;
- les **algorithmes parallèles** où les deux sous-systèmes sont avancés en temps parallèlement ;
- les **algorithmes itérés** où chaque pas de temps peut donner lieu à plusieurs calculs afin d'assurer la conservation de l'énergie du système.

L'algorithme itéré peut s'avérer beaucoup plus coûteux que l'algorithme décalé (n fois plus coûteux s'il converge en n itérations) et que l'algorithme parallèle, mais il est nécessaire lorsqu'on traite des problèmes d'interaction fluide-structure avec un couplage fort, c'est-à-dire avec des échanges à chaque pas de temps ou encore lorsque les deux sous-systèmes ont des inerties comparables. L'algorithme itéré sera décrit plus en détail dans la suite.

2.4.3 Procédure de couplage FSI

L'algorithme 1 résume la procédure générale de l'implémentation numérique du couplage fluide-structure.

Algorithme 1 Procédure générale de couplage FSI

```
pour t = 1 à t_f faire
    tant que l'itération FSI n'a pas convergé faire
        mettre à jour le maillage
        résoudre Navier-Stokes sur Ω_f
        résoudre Navier sur Ω_s
        calculer la déformation du maillage
    fin
fin
```

Le code de dynamique des fluides doit donc être capable de résoudre les équations de Navier-Stokes dans un formalisme ALE :

$$\nabla \cdot \mathbf{u} = 0, \tag{2.112}$$

$$\rho^f \left(\left. \frac{\partial \mathbf{u}}{\partial t} \right|_\chi + (\mathbf{u} - \mathbf{u}^*) \cdot \nabla \mathbf{u} \right) + \nabla p - \mu^f \nabla^2 \mathbf{u} - \mathbf{S} = 0. \tag{2.113}$$

Le code de dynamique des structures doit être capable de résoudre l'équation de Navier :

$$\frac{\partial^2 (\rho^s \mathbf{D})}{\partial t^2} - (\lambda + \mu) \nabla (\nabla \cdot \mathbf{D}) - \mu \nabla^2 \mathbf{D} = \mathbf{f}^s. \tag{2.114}$$

Les conditions aux limites étant déterminées en fonction des phénomènes physiques propres à chaque application.

La prise en compte de la déformation du maillage modélisée par l'équation (2.67) est détaillée dans le chapitre 4 au paragraphe 4.5.3.2.

Références

[13] R. Boman. "Développement d'un formalisme Arbitraire Lagrangien Eulérien tridimensionel en dynamique implicite. Application aux opérations de mise à forme". Thèse de doct. Université de Liège, 2010.

[26] E. de Langre. *Fluide et Solide*. Ellipses, 2001.

[39] P. Hémon. *Vibrations des Structures Couplées avec le Vent*. Ellipses, 2006.

[68] S. Piperno. *Interactions Fluide-Structure*. Mastère de Mécanique Numérique - École Nationale Supérieure des Mines de Paris. 2005–2006.

[83] H. K. Versteeg et W. Malalasekera. *An Introduction to Computational Fluid Dynamics : The Finite Volume Method (2nd Edition)*. Pearson, 2007.

CHAPITRE

3

ALGORITHMES PARALLÈLES ASYNCHRONES

Sommaire

3.1 Introduction

Le chapitre précédent a permis de présenter la modélisation de l'interaction fluide-structure. Ce modèle comprend les équations de Navier-Stokes décrivant le comportement du fluide, équations aux dérivées partielles fortement non-linéaires, couplées à l'équation de Navier modélisant l'évolution de la structure. Ce problème d'interaction

fluide-structure n'est pas simple à résoudre analytiquement ; c'est pourquoi la communauté scientifique [10, 15, 16, 44] a adopté une résolution par voie numérique.

Dans ce travail, nous utilisons la méthode PISO (i.e. *Pressure-Implicit with Splitting of Operator*) [41] pour résoudre les équations de Navier-Stokes et, pour la structure, nous adoptons la méthode proposée par JASAK et al. dans [42] pour résoudre l'équation de Navier. Il est à noter que ces deux méthodes sont bien adaptées à la résolution de chacun des modèles. La mise en œuvre de ces deux méthodes conduit à résoudre 6 systèmes linéaires. La résolution numérique de ce problème est effectuée en considérant un schéma en temps implicite pour l'équation de Navier et un schéma prédicteur-correcteur pour les équations de Navier-Stokes. Par ailleurs, la discrétisation en espace est effectuée par la méthode des volumes finis pour les deux modèles fluide et structure.

La discrétisation de ces équations par la méthode des volumes finis présente plusieurs avantages ; le même formalisme est utilisé pour les deux sous-domaines, ce qui permet d'éliminer les erreurs d'approximation qui apparaissent lors du traitement des variables entre deux méthodes de discrétisation différentes. C'est une méthode conservative dans chacun des volumes de contrôle et bien adaptée pour résoudre des problèmes compliqués, fortement non-linéaires et couplés. La mise en œuvre de ces méthodes conduit à résoudre, à chaque pas de temps, six systèmes linéaires creux de très grande taille, compte tenu du fait qu'on souhaite obtenir une bonne précision de calcul. Dans ces conditions, pour résoudre ces systèmes linéaires, nous avons opté d'une part pour des méthodes itératives qui permettent de limiter l'effet de la propagation d'erreur d'arrondis, et d'autre part, pour des méthodes parallèles où on considère des communications bloquantes et non bloquantes, dans ce dernier cas, afin de minimiser les temps d'inactivité de certains processeurs. On oriente donc la résolution du problème vers des méthodes de sous-domaines. Cependant, pour une structure mécanique de forme quelconque, la décomposition en sous-domaines peut être problématique. C'est pourquoi nous considérons la méthode de multi-splitting qui permet une présentation unifiée des méthodes de sous-domaines avec ou sans recouvrement. On renvoie à la bibliographie en fin de chapitre pour des références concernant l'étude des méthodes de multi-splitting [3, 5, 6, 7, 32, 31, 33, 52, 60, 61, 80, 84, 85].

De plus, dans ce contexte algorithmique, nous présentons une analyse de ces méthodes par des techniques de contraction qui permet d'une part de prendre en compte n'importe quelle norme hilbertienne ou non-hilbertienne sur l'espace de travail de dimension finie et d'autre part, sous des hypothèses adéquates, à savoir que les matrices de discrétisation sont des M-matrices, de montrer que les applications de point fixe associées aux systèmes à résoudre, sont contractantes quelle que soit la décomposition en grands blocs du problème, ce qui permet d'appliquer un résultat de [5]. On obtient ainsi un résultat de contraction de l'application de point fixe étendue associée à la méthode de multi-splitting et on déduit la convergence des méthodes parallèles considérées quel que soit le mode de communication bloquant ou non bloquant. Par ailleurs, on peut également considérer une analyse de la convergence de la méthode de multi-splitting par des techniques d'ordre partiel, ce qui permet d'obtenir une convergence monotone des itérés.

Enfin, il convient de noter que ces résultats d'analyse de convergence, aussi bien par des techniques de contraction que par des techniques d'ordre partiel, peuvent être étendus au cas de problèmes semi-linéaires constitués par une application affine perturbée par un opérateur diagonal croissant. Dans ce même contexte, on peut aussi concevoir des problèmes où le déplacement de la structure est soumis à des contraintes de type inégalités. Classiquement, on obtient alors un problème multivoque où l'application affine associée au système linéaire à résoudre est perturbée par un opérateur monotone diagonal multivoque [5, 8, 19, 27, 35, 36, 51, 55] qui rend compte des contraintes sur le déplacement.

Ce chapitre se décompose en deux paragraphes. Le premier paragraphe détaille les méthodes numériques de résolution utilisées pour le problème d'interaction fluide-structure : la méthode de discrétisation ainsi que les algorithmes de résolution des équations de Navier-Stokes et de l'équation de Navier. Le deuxième paragraphe détaille les méthodes de résolution parallèles du problème ; une présentation des algorithmes parallèles est effectuée puis une analyse par des techniques de contraction et d'ordre partiel de ces algorithmes est présentée ainsi que des résultats de convergence. Enfin la méthode de multi-splitting est détaillée.

3.2 Méthodes numériques de résolution du problème FSI

3.2.1 Discrétisation par volumes finis

L'utilisation de la méthode de discrétisation par volumes finis nécessite la décomposition du domaine physique en volumes de contrôle définis par une grille numérique [66]. Les équations régissant le comportement physique de la matière sont intégrées sur tous ces volumes de contrôle ; les termes qui apparaissent sont alors approximés par des relations de type différences finies. De cette discrétisation, résulte un système de N_{CV} équations algébriques, N_{CV} étant le nombre de volumes de contrôle, qui sont ensuite résolues avec des méthodes adaptées.

Le principal avantage de la méthode de discrétisation par les volumes finis est qu'elle est une méthode conservative dans chacun des volumes de contrôle plutôt que sur le domaine en entier comme c'est le cas pour la méthode des éléments finis ; c'est-à-dire que le flux entrant dans un volume donné est égal au flux sortant du volume adjacent. Cela signifie que des solutions conservatives peuvent être obtenues à bas coût de calcul avec des maillages grossiers. L'utilisation de maillages grossiers introduit cependant des erreurs numériques qui réduisent la précision des simulations.

La méthode des volumes finis est intrinsèquement bonne pour résoudre des problèmes couplés et non-linéaires et devient une alternative plus intéressante que la méthode des éléments finis quand le modèle mathématique devient plus complexe. La méthode des volumes finis est très répandue en dynamique des fluides alors que c'est plutôt la méthode des éléments finis qui prévaut en dynamique des structures. Cependant, la méthode des volumes finis donne des résultats similaires en dynamique des structures [42] et son utilisation pour la simulation des interactions fluide-structure est particulièrement

intéressante car des couplages très forts peuvent être simulés sans introduire d'erreurs d'approximation dues à l'utilisation de deux formulations différentes.

La figure 3.1 représente un volume de contrôle en trois dimensions. Le nœud central est noté P, les nœuds voisins sont notés par les initiales en majuscule de leurs positions cardinales par rapport à P : Top, Bottom, North, South, East, West. Les centres des facettes correspondantes sont notés avec les initiales en minuscule. Ses dimensions sont notées δx_1, δx_2 et δx_3.

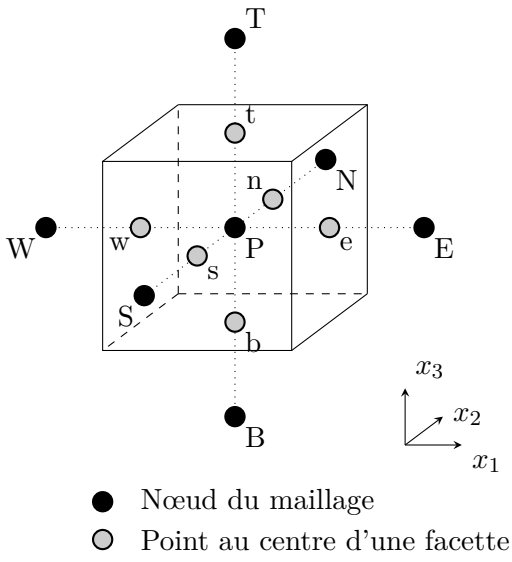


FIGURE 3.1 – Nœuds autour d'un volume de contrôle

L'équation de convection-diffusion stationnaire d'une propriété ϕ dans un champ de propagation $\mathbf{u}$ est donnée par

$$\nabla \cdot (\ \rho^f \mathbf{u}\phi - \Gamma \nabla \phi) = \mathbf{f}^f, \tag{3.1}$$

où Γ est le coefficient de diffusion. En intégrant l'équation (3.1) sur un volume de contrôle arbitraire V_P et en appliquant le théorème de Gauss, il vient

$$\oint_{\partial V_P} (\rho^f \mathbf{u}\phi - \Gamma \nabla \phi)\mathbf{n} dS = \int_{V_P} \mathbf{f}^f dV, \tag{3.2}$$

où ∂V_P est la surface englobant le volume de contrôle et $\mathbf{n}$ est le vecteur unitaire normal à la surface. L'intégrale de surface peut être découpée en la somme des six intégrales de surface sur chaque facette du volume de contrôle ∂V_c ($c =$ t, b, n, s, e, w)

$$\sum_c \oint_{\partial V_c} (\rho^f \mathbf{u}\phi - \Gamma \nabla \phi)\mathbf{n} dS = \int_{V_P} \mathbf{f}^f dV. \tag{3.3}$$

Le lecteur est renvoyé à [74] pour la procédure de discrétisation complète. Par exemple, si un schéma de discrétisation décentré amont est utilisé pour le terme convectif et un

schéma centré pour le terme diffusif sur une grille cartésienne, il vient alors l'approximation de l'équation d'équilibre (3.3) en trois dimensions donnée par

$$a_P\phi_P = a_E\phi_E + a_W\phi_W + a_N\phi_N + a_S\phi_S + a_T\phi_T + a_B\phi_B + b_P, \tag{3.4}$$

où les coefficients sont donnés par

$$\begin{aligned}
a_E &= \frac{\Gamma}{(x_E - x_P)(x_e - x_w)},\\
a_W &= \frac{\rho^f u_1}{x_e - x_w} + \frac{\Gamma}{(x_P - x_W)(x_e - x_w)},\\
a_N &= \frac{\Gamma}{(y_N - y_P)(y_n - y_s)},\\
a_S &= \frac{\rho^f u_2}{y_n - y_s} + \frac{\Gamma}{(y_P - y_S)(y_n - y_s)},\\
a_T &= \frac{\Gamma}{(z_T - z_P)(z_t - z_b)},\\
a_B &= \frac{\rho^f u_3}{z_t - z_b} + \frac{\Gamma}{(z_P - z_B)(z_t - z_b)},\\
a_P &= \frac{\rho^f u_1}{x_e - x_w} + \frac{\Gamma(x_E - x_W)}{(x_P - x_W)(x_E - x_P)(x_e - x_w)}\\
&\quad + \frac{\rho^f u_2}{y_n - y_s} + \frac{\Gamma(y_N - y_S)}{(y_P - y_S)(y_N - y_P)(y_n - y_s)}\\
&\quad + \frac{\rho^f u_3}{z_t - z_b} + \frac{\Gamma(z_T - z_B)}{(z_P - z_B)(z_T - z_P)(z_t - z_b)},\\
b_P &= f_P^f.
\end{aligned} \tag{3.5}$$

où les x et y sont les coordonnées spatiales de chaque point. Notons que les coefficients diagonaux a_P sont positifs alors que les coefficients hors-diagonaux a_C sont négatifs.

La discrétisation par les volumes finis produit donc un système d'équations algébriques de la forme

$$a_P\phi_P + \sum_{C\neq P} a_C\phi_C = b_P, \tag{3.6}$$

où l'indice C désigne chaque point impliqué dans l'approximation résultante du schéma de discrétisation utilisé et

$$a_P = -\sum_{C\neq P} a_C \tag{3.7}$$

découle de la propriété de conservation de la méthode des volumes finis. On a donc des matrices à diagonale dominante irréductibles, qui compte tenu des signes des coefficients sont des M-matrices (cf. annexe A.3).

3.2.2 Algorithme PISO pour résoudre Navier-Stokes

L'algorithme PISO (i.e. *Pressure-Implicit with Splitting of Operator*) est un algorithme itératif de type prédicteur-correcteur [41] basé sur la séparation de la résolution des

équations de vitesse et des équations de pression. Il décompose les opérateurs en un pas prédicteur implicite et deux pas correcteurs explicites et est conçu pour satisfaire la condition de conservation de la masse grâce à des étapes de prédiction-correction. C'est une extension de l'algorithme SIMPLE (i.e. *Semi-Implicit Method for Pressure Linked Equations*) [67] avec l'ajout d'une étape de correction afin d'améliorer son efficacité.

Les équations de transport (continuité et quantité de mouvement), sous leur forme incompressible, sont données par les équations (2.58) et (2.59). Dans le but d'expliquer la méthode de l'algorithme PISO de façon claire et aisée, les équations de transport peuvent être discrétisées par différences finies avec le schéma semi-implicite d'Euler.

Si n et $n+1$ représentent des numéros de temps successifs, les équations qui régissent l'écoulement d'un fluide incompressible peuvent s'exprimer en différences finies de la façon suivante :

$$\frac{\rho^f}{\delta t}(\mathbf{u}^{n+1} - \mathbf{u}^n) = H(\mathbf{u}^{n+1}) - \Delta p^{n+1}, \tag{3.8}$$

et

$$\Delta \mathbf{u}^{n+1} = 0. \tag{3.9}$$

où l'opérateur Δ représente l'approximation de $\partial/\partial\mathbf{x}$ en différences finies, et dans l'équation (3.8), H est la matrice de discrétisation de l'opérateur de convection-diffusion résultant du schéma itératif semi-implicite, il prend habituellement la forme

$$H(\mathbf{u}) = \mathcal{C}_m \mathbf{u}_m, \tag{3.10}$$

où l'indice m indique la position sur la grille de calcul et la sommation est faite sur tous les nœuds concernés par la représentation des flux dans l'espace en différences finies. Le coefficient $\mathcal{C}$ est fonction des vitesses, des masses volumiques, etc. et par conséquent, H est un opérateur non-linéaire ; mais en supposant que $\mathcal{C}$ reste constant sur l'intervalle δt, l'opérateur H peut être linéarisé de cette façon.

L'équation de pression est obtenue en prenant la divergence de l'équation (3.8) et en la substituant dans l'équation (3.9) d'où

$$\Delta^2 p^{n+1} = \Delta H(\mathbf{u}^{n+1}) + \frac{\rho^f}{\delta t}\Delta \mathbf{u}^n. \tag{3.11}$$

Notons par les exposants †, ††, et ††† les valeurs intermédiaires des champs des variables obtenus lors de l'opération de séparation des équations.

3.2.2.1 Étape de prédiction

Le champ de pression à l'instant t^n est fixé dans l'équation discrétisée (3.8) afin d'obtenir le champ de vitesse $\mathbf{u}^\dagger$ donné par :

$$\frac{\rho^f}{\delta t}(\mathbf{u}^\dagger - \mathbf{u}^n) = H(\mathbf{u}^\dagger) - \Delta p^n. \tag{3.12}$$

Cette équation peut être résolue par une technique standard telle que les méthodes de relaxation ou de gradient conjugué, afin d'obtenir le champ $\mathbf{u}^\dagger$ qui ne satisfait vraisemblablement pas l'équation de continuité (3.9) et a donc besoin d'être corrigé.

3.2.2.2 Première étape de correction

Un nouveau champ de vitesse $\mathbf{u}^{\dagger\dagger}$ et de pression $p^{\dagger}$ sont déterminés afin de satisfaire l'équation de continuité

$$\Delta \mathbf{u}^{\dagger\dagger} = 0, \tag{3.13}$$

et l'équation de quantité de mouvement correspondante

$$\frac{\rho^f}{\delta t}(\mathbf{u}^{\dagger\dagger} - \mathbf{u}^n) = H(\mathbf{u}^{\dagger}) - \Delta p^{\dagger}, \tag{3.14}$$

qui est explicite puisque H opère sur $\mathbf{u}^{\dagger}$. Cette forme explicite est utile pour calculer les composantes de vitesse. Auparavant on doit résoudre le système linéaire suivant associé à l'équation de correction de pression :

$$\Delta^2 p^{\dagger} = \Delta H(\mathbf{u}^{\dagger}) + \frac{\rho^f}{\delta t}\Delta \mathbf{u}^n, \tag{3.15}$$

puisque la partie de droite contient les termes en $\mathbf{u}^{\dagger}$et en $\mathbf{u}^n$ qui sont connus. Le champ $p^{\dagger}$ est ensuite introduit dans l'équation (3.14) pour obtenir $\mathbf{u}^{\dagger\dagger}$ qui satisfait désormais l'équation (3.13).

3.2.2.3 Seconde étape de correction

De la même manière que pour la première étape de correction, de nouveaux champs de vitesse $\mathbf{u}^{\dagger\dagger\dagger}$ et de pression $p^{\dagger\dagger}$ sont déterminés pour satisfaire l'équation de continuité

$$\Delta \mathbf{u}^{\dagger\dagger\dagger} = 0. \tag{3.16}$$

De façon analogue, l'équation de quantité de mouvement de type explicite correspondante s'écrit alors

$$\frac{\rho^f}{\delta t}(\mathbf{u}^{\dagger\dagger\dagger} - \mathbf{u}^n) = H(\mathbf{u}^{\dagger\dagger}) - \Delta p^{\dagger\dagger}, \tag{3.17}$$

et est calculée en résolvant préalablement le système linéaire suivant associé à l'équation de correction de pression correspondante

$$\Delta^2 p^{\dagger\dagger} = \Delta H(\mathbf{u}^{\dagger\dagger}) + \frac{\rho^f}{\delta t}\Delta \mathbf{u}^n. \tag{3.18}$$

Comme précédemment, le champ $p^{\dagger\dagger}$ peut facilement être déterminé puisque la partie de droite de l'équation (3.18) est connue, et, une fois sa valeur introduite dans l'équation (3.17), le champ $\mathbf{u}^{\dagger\dagger\dagger}$ peut être calculé.

D'autres étapes de correction peuvent être utilisées mais, comme montré par Issa dans [41], la précision avec laquelle $\mathbf{u}^{\dagger\dagger\dagger}$ et $p^{\dagger\dagger}$ approchent les solutions exactes $\mathbf{u}^{n+1}$ et p^{n+1} est suffisante pour la majorité des applications.

Finalement, on a 5 systèmes linéaires à résoudre et des formules explicites à calculer.

Algorithme 2 L'algorithme PISO

1: **tant que** $t^n < t_f$ **faire**
2: résoudre les équations de quantité de mouvement discrétisées (3.12)
3: résoudre la première équation de correction de pression (3.15)
4: corriger la pression et les vitesses (3.14)
5: résoudre la seconde équation de correction de pression (3.18)
6: corriger la pression et les vitesses (3.17)
7: résoudre les équations de transport discrétisées restantes
8: **fin**

3.2.3 Équation de Navier, méthode de Jasak et al.

Pour rappel, l'équation du déplacement de Navier peut s'écrire sous la forme :

$$\frac{\partial^2(\rho^s \mathbf{D})}{\partial t^2} - \nabla \cdot \left[\mu \left(\nabla \mathbf{D} + (\nabla \mathbf{D})^T\right) + \lambda \mathbf{I} \operatorname{tr}(\nabla \mathbf{D})\right] = \mathbf{f}^s. \tag{3.19}$$

La discrétisation par la méthode des volumes finis utilise la forme intégrale de l'équation (3.19) sur un volume de contrôle de volume V_P autour d'un point P. En utilisant le théorème de Gauss, il vient :

$$\int_{V_P} \frac{\partial^2(\rho^s \mathbf{D})}{\partial t^2} dV - \oint_{\partial V_P} d\mathbf{s} \cdot \left[\mu \left(\nabla \mathbf{D} + (\nabla \mathbf{D})^T\right) + \lambda \mathbf{I} \operatorname{tr}(\nabla \mathbf{D})\right] = \int_{V_P} \mathbf{f}^s dV. \tag{3.20}$$

Cette équation est discrétisée de manière ségréguée, c'est-à-dire que chaque composante du déplacement est résolue séparément et le couplage inter-composante est traité explicitement. Il en résulte des matrices de discrétisation creuses et à diagonale dominante, de formes idéales pour une résolution par des solveurs itératifs.

En utilisant la méthode développée par Jasak et al. dans [42], la dérivée temporelle est calculée en utilisant les valeurs de $\mathbf{D}$ aux deux pas de temps précédents :

$$\frac{\partial^2 \mathbf{D}}{\partial t^2} \approx \frac{\mathbf{D}(t+\delta t) - 2\mathbf{D}(t) + \mathbf{D}(t-\delta t)}{\delta t^2} + o(\delta t^2). \tag{3.21}$$

Les intégrales de volume sont évaluées en utilisant la formule du point-milieu :

$$\int_{V_P} \phi dV = \phi_P V_P. \tag{3.22}$$

Le terme en Div-Grad ($\nabla \cdot \nabla$) est divisé en une somme d'intégrales sur toutes les facettes du volume de contrôle en utilisant la formule du point-milieu :

$$\int_{V_P} \nabla \cdot (\mu \nabla \mathbf{D}) dV = \oint_{\partial V_P} (\mu \nabla \mathbf{D}) \mathbf{n} dS = \sum_c \mu (\nabla \mathbf{D})_c \mathbf{n} S_c, \tag{3.23}$$

où S_c représente l'aire de la facette. Deux types de discrétisation sont identifiés, une discrétisation implicite qui donne

$$\oint_{\partial V P} (\mu \nabla \mathbf{D}) \mathbf{n} dS = \bar{a}_P + \sum_C a_C D_C, \tag{3.24}$$

où

$$a_C = \mu \frac{|\mathbf{n}S_c|}{|\mathbf{d}_C|},\ \bar{a}_P = \sum_C -a_C \text{ et } \mathbf{d}_C = \overline{PC}, \tag{3.25}$$

et une discrétisation explicite :

$$(\nabla \mathbf{D})_c = f_x(\nabla \mathbf{D})_P + (1 - f_x)(\nabla \mathbf{D})_C, \tag{3.26}$$

où f_x est un coefficient d'extrapolation.

Le gradient au centre du volume de contrôle est calculé par la méthode des moindres carrés. En supposant une variation linéaire d'une variable quelconque ϕ et en minimisant l'erreur en C il vient l'expression suivante :

$$(\nabla \phi)_P = \sum_C w_C^2 \mathbf{G}^{-1} \cdot \mathbf{d}_C(\phi_C - \phi_P), \tag{3.27}$$

où $w_c = 1/|\mathbf{d}_C|$ et $\mathbf{G}$ est la matrice 3×3

$$\mathbf{G} = \sum_C w_C^2 \mathbf{d}_C \mathbf{d}_C^T. \tag{3.28}$$

En assemblant l'équation (3.20) avec les équations (3.21) à (3.24), (3.26) et (3.27), il vient :

$$\bar{a}_P \mathbf{D}_P - \sum_C a_C \mathbf{D}_C = \mathbf{r}_P, \tag{3.29}$$

où $\bar{a}_P$ et $\mathbf{r}_P$ incluent les contributions du terme temporel et des conditions aux limites. $\mathbf{D}_P$ dépend des valeurs dans les volumes de contrôle voisins, créant ainsi le système d'équations algébriques suivant :

$$[A][\mathbf{D}] = [\mathbf{r}], \tag{3.30}$$

où $[A]$ est une matrice creuse contenant les coefficients $\bar{a}_P$ sur la diagonale et les coefficients a_C en dehors de la diagonale. $[\mathbf{D}]$ est le vecteur des valeurs des déplacements pour tous les volumes de contrôle. Notons que la matrice $[A]$ est symétrique et à diagonale dominante stricte du fait de l'utilisation d'un schéma implicite en temps.

L'équation (3.19) est donc discrétisée de la façon suivante :

$$\frac{\partial^2(\rho^s \mathbf{D})}{\partial t^2} - \underbrace{\nabla \cdot (\mu \nabla \mathbf{D})}_{\text{implicite}} - \underbrace{\nabla \cdot \left[\mu(\nabla \mathbf{D})^T + \lambda \mathbf{I}\,\mathrm{tr}(\nabla \mathbf{D})\right]}_{\text{explicite}} = \mathbf{f}^s. \tag{3.31}$$

Malheureusement, ce découpage ne converge que marginalement car les termes explicites contiennent plus d'informations que leurs équivalents implicites. La solution proposée par Jasak et al. dans [42] est de réécrire l'équation (3.31) de la façon suivante :

$$\frac{\partial^2(\rho^s \mathbf{D})}{\partial t^2} - \underbrace{\nabla \cdot [(2\mu + \lambda)\nabla \mathbf{D}]}_{\text{implicite}} - \underbrace{\nabla \cdot \left[\mu(\nabla \mathbf{D})^T + \lambda \mathbf{I}\,\mathrm{tr}(\nabla \mathbf{D}) - (\mu + \lambda)\nabla \mathbf{D}\right]}_{\text{explicite}} = \mathbf{f}^s. \tag{3.32}$$

Après discrétisation, la matrice a été « sur-relaxée » et la convergence de la méthode est bien meilleure. La matrice de discrétisation correspondante à la partie implicite a des coefficients diagonaux positifs, des coefficients hors-diagonaux négatifs et est symétrique définie positive ; c'est une matrice de Stieltjes i.e. [61]. Le fait que la matrice soit symétrique va orienter notre choix vers un algorithme de résolution de type gradient conjugué préconditionné.

3.3 Résolution numérique parallèle du problème FSI

Afin de résoudre numériquement le problème d'interaction fluide-structure, il faut résoudre 7 systèmes linéaires à chaque pas de temps : 5 pour Navier-Stokes, 1 pour Navier et 1 pour la vitesse de grille $\mathbf{u}^*$ intervenant dans l'équation (2.66). Dans le paragraphe 3.2, on a remarqué que les matrices associées aux systèmes linéaires, symétriques ou non, sont des M-matrices. Cette propriété est intéressante et joue un rôle algorithmique important dans ce qui va suivre. C'est pourquoi, dans la suite, nous présentons un résultat original pour la résolution de systèmes linéaires par la méthode de multi-splitting permettant de s'affranchir d'une décomposition formelle en sous-domaines. Notons cependant que cette analyse est aussi valable pour des problèmes pseudo-linéaires univoques ou multivoques, ce dernier cas correspondant au cas où des contraintes sont appliquées sur la solution.

Les principales contributions développées dans cette section sont les suivantes :

- dans les paragraphes 3.3.1 et 3.3.2, pour des problèmes linéaires, nous analysons la convergence des méthodes parallèles asynchrones pour l'application fluide-structure quelle que soit la décomposition du problème en grands blocs ;
- en particulier dans le paragraphe 3.3.2.4, sous des hypothèses appropriées, nous montrons un résultat de contraction pour les applications de point fixe associées à une décomposition réaliste en grands blocs en calcul parallèle, ce qui permet d'appliquer un résultat de Bahi et al. dans [5] pour la méthode de multi-splitting ;
- dans le paragraphe 3.3.4, nous utilisons la méthode de multi-splitting, qui permet de s'affranchir d'une décomposition effective en sous-domaines, dans un contexte unifié en utilisant les normes classiques hilbertiennes ou non-hilbertiennes qui sont plus pratiques à utiliser (cf. remarque 4), la méthode asynchrone de Jacobi par blocs apparaissant alors comme un cas particulier ;
- les résultats sont étendus aux cas des problèmes pseudo-linéaires univoques ou multivoques ;
- en ce qui concerne les applications fluide-structure, nous considérons dans ce travail une discrétisation globale des domaines fluide et structure par la méthode des volumes finis, qui permet d'obtenir la propriété de M-matrice dans les systèmes discrétisés. Une autre conséquence est la cohérence dans la numérotation des points de grille ce qui limite les erreurs d'approximation quand on passe des points dans le domaine fluide à ceux dans le domaine structure.

3.3.1 Itérations asynchrones

Considérons la solution $\tilde{X}^*$ du système linéaire suivant :

$$A\tilde{X}^* = \tilde{B}, \tag{3.33}$$

où A est le résultat de la discrétisation des équations de Navier-Stokes ou de l'équation de Navier provenant du couplage fluide-structure et est une M-matrice (cf. annexe A.3).

Remarque 1. *Dans l'équation* (3.33)*, on note $\tilde{X}$ l'inconnue du problème ; cette notation sera conservée dans les paragraphes 3.3.1 et 3.3.2. En revanche, dans le paragraphe 3.3.4, nous aurons à introduire m vecteurs $\tilde{X}$ pour définir un multi-vecteur X intervenant dans la méthode de multi-splitting.*

Notons $M \in \mathbb{N}$ la dimension de la matrice A. Soit $\mathcal{E} = \mathbb{R}^M$ un espace vectoriel normé par une norme hilbertienne ou non-hilbertienne et soit α un autre entier naturel positif, correspondant au nombre de blocs associé à la décomposition naturelle, tel que l'espace $\mathcal{E} = \prod_{i=1}^{\alpha} \mathcal{E}_i$ est décomposé en un produit fini d'α sous-espaces notés $\mathcal{E}_i = \mathbb{R}^{m_i}$, avec $\sum_{i=1}^{\alpha} m_i = M$, où m_i représente la dimension du i-ème sous-espace $\mathcal{E}_i$; à noter que $\mathcal{E}_i$ est aussi un espace vectoriel normé.

Dans la suite, on considère le problème général de point fixe suivant, associé à l'équation (3.33) :

$$\text{Trouver } \tilde{X}^* \in \mathcal{E} \text{ tel que } \tilde{X}^* = F(\tilde{X}^*), \tag{3.34}$$

où $F : \mathcal{D}(F) \subset \mathcal{E} \mapsto \mathcal{D}(F)$, $\tilde{X} \mapsto F(\tilde{X})$ est une application de point fixe associée à une décomposition particulière de la matrice A. Pour être intéressante d'un point de vue algorithmique, une telle application de point fixe doit être contractante ; par conséquent, d'après des travaux similaires dans la littérature [5, 60], nous considérerons dans la suite une décomposition régulière de la matrice A définie par $A = \tilde{\Delta} - (\tilde{L} + \tilde{R})$, où ici $\tilde{\Delta}^{-1}$, $\tilde{L}$ et $\tilde{R}$ sont des matrices non négatives puisque A est une M-matrice. Cette application de point fixe sera choisie de manière appropriée, c'est-à-dire de telle sorte qu'elle soit contractante. Dans la section suivante, il sera montré comment l'application F est liée au problème à résoudre (3.33). Compte tenu de la décomposition $\mathcal{E}$, considérons la décomposition en α blocs de F et $\tilde{Y}$ correspondante

$$\begin{array}{rcl} \tilde{Y} & = & (\tilde{Y}_1, \ldots, \tilde{Y}_\alpha), \\ F(\tilde{Y}) & = & (F_1(\tilde{Y}), \ldots, F_\alpha(\tilde{Y})), \end{array} \tag{3.35}$$

où pour tout $\tilde{Y} \in \mathcal{E}$, on note $\tilde{Y}_l$ le bloc-composante de $\tilde{Y}$ pour $l \in \{1, \ldots, \alpha\}$.

Dans le but de résoudre le problème de point fixe (3.34), considérons maintenant les itérations parallèles asynchrones définies comme suit : soit $\tilde{X}^0 \in \mathcal{E}$ donné ; alors, pour tout $r \in \mathbb{N}$, $\tilde{X}^{(r+1)}$ est défini récursivement par

$$\tilde{X}_l^{(r+1)} = \begin{cases} F_l\left(\tilde{X}_1^{(\kappa_1(r))}, \ldots, \tilde{X}_k^{(\kappa_k(r))}, \ldots, \tilde{X}_\alpha^{(\kappa_\alpha(r))}\right) \text{ si } l \in s(r), \\ \tilde{X}_l^r \text{ si } l \notin s(r), \end{cases} \tag{3.36}$$

où

$$\begin{cases} \forall r \in \mathbb{N}, s(r) \subset \{1, \ldots, \alpha\} \text{ et } s(r) \neq \emptyset, \\ \forall l \in \{1, \ldots, \alpha\}, \{r \mid l \in s(r)\} \text{ est dénombrable,} \end{cases} \tag{3.37}$$

et $\forall k \in \{1, \ldots, \alpha\}$,

$$\begin{cases} \forall r \in \mathbb{N}, \kappa_k(r) \in \mathbb{N}, 0 \leq \kappa_k(r) \leq r, \\ \kappa_k(r) = r \text{ si } k \in s(r) \text{ et } \lim\limits_{r \to \infty} \kappa_k(r) = +\infty. \end{cases} \tag{3.38}$$

Les modèles de schéma de calcul itératif asynchrone précédent sont réalisés en parallèle sans ordre ni synchronisation ; de tels algorithmes décrivent une méthode de sous-domaines avec ou sans recouvrement, correspondant en fait à un découpage du domaine où est défini le problème aux limites en α parties.. En particulier, ils permettent de considérer des calculs distribués au cours desquels les processeurs calculent à leur rythme en fonction de leurs caractéristiques intrinsèques et de leur charge propre de calcul. Le parallélisme entre les processeurs est bien décrit par l'ensemble $s(r)$ qui contient le numéro des composantes relaxées par chaque processeur de manière parallèle à chaque relaxation tandis que l'utilisation de composantes retardées dans (3.36) permet de modéliser un comportement non déterministe et n'implique pas d'inefficacité du schéma de calcul distribué considéré. Notons que, théoriquement, chaque composante du vecteur doit être relaxée un nombre infini de fois. Le choix des composantes relaxées peut être guidé par n'importe quel critère et, en particulier, un critère naturel est de prendre les valeurs les plus récentes disponibles des composantes calculées par les processeurs.

Remarque 2. *De telles itérations asynchrones décrivent différentes classes d'algorithmes parallèles, tel que des itérations parallèles synchrones si* $\forall k \in \{1, \ldots, \alpha\}$, $\forall r \in \mathbb{N}$, $\kappa_k(r) = r$. *Dans le contexte synchrone, pour un choix particulier de* $s(r)$, *alors* (3.36)-(3.38) *décrivent des algorithmes séquentiels classiques ; tels que la méthode de Jacobi si* $\forall r \in \mathbb{N}$, $s(r) = \{1, \ldots, \alpha\}$ *et la méthode Gauss-Seidel si* $\forall r \in \mathbb{N}$, $s(r) = \{1 + r \bmod \alpha\}$ *(cf. [51]).*

3.3.2 Notion de M-minorantes, analyse en contraction

3.3.2.1 Définitions fondamentales

Considérons la résolution du problème (3.33) ainsi que la décomposition de la matrice A en α blocs ; en utilisant les mêmes notations, cette décomposition par blocs est définie comme suit :

$$A_{l,k} \in \mathcal{L}(\mathbb{R}^{m_k}; \mathbb{R}^{m_l}) = \mathcal{L}(\mathcal{E}_k; \mathcal{E}_l), \text{ avec } \sum_{l=1}^{\alpha} m_l = M. \tag{3.39}$$

De plus, pour $l = 1, \ldots, \alpha$, notons $< ., . >_l$ le produit de dualité[1] entre $\mathcal{E}_l$ et $\mathcal{E}_l'$ son espace dual topologique ; soit $| \ . \ |_l$ la norme associée dans $\mathcal{E}_l$, tel que $G_l(\tilde{X}_l)$ est l'application de

[1]Forme bilinéaire de $\mathcal{E} \times \mathcal{E}'$ dans $\mathbb{R}$. Si $\mathcal{E}_l$ est un espace de Hilbert, alors le produit de dualité est le produit scalaire de $\mathcal{E}_l$.

dualité définie par

$$G_l(\tilde{X}_l) = \{g_l(\tilde{X}_l) \in \mathcal{E}_i' \mid < \tilde{X}_l, g_l >_l = |\tilde{X}_l|_l^2 \text{ et } |g_l|_l^* = |\tilde{X}_l|_l\}, \tag{3.40}$$

où $|g_l|_l^*$ représente la norme de $g_l \in \mathcal{E}_l'$. Si on considère l'application $\tilde{X}_l \mapsto \phi_l(\tilde{X}_l) = \frac{1}{2}|\tilde{X}_l|_l^2$ alors, on sait classiquement que $G_l(\tilde{X}_l)$ coïncide avec le sous-différentiel de l'application $\phi_l(\tilde{X}_l)$ [8] définie par

$$\phi_l(\tilde{Y}_l) \geq \phi_l(\tilde{X}_l) + < \tilde{Y}_l - \tilde{X}_l, g_l >_l , \tag{3.41}$$

où $g_l \in G_l(\tilde{X}_l)$ est le sous-gradient (cf. annexe A.1).

Remarque 3. *Si ϕ_l est Gâteaux-différentiable ou Fréchet-différentiable (cf. annexes A.1, définitions 12 et 13), alors g_l est la dérivée de $\phi_l(\cdot)$. Si on considère la norme euclidienne $\phi_l(\tilde{X}_l) = \frac{1}{2}\|\tilde{X}_l\|_2^{l,2}$ alors, par un calcul simple la Gâteaux-dérivée est égale à $\tilde{X}_l$.*

Remarque 4. *Même si, dans les espaces de dimension finie, toutes les normes de Holder sont équivalentes mathématiquement, quand la dimension M de l'espace devient grande, la mesure de distance ξ entre deux éléments à l'aide de ces normes, pose numériquement, quelques problèmes de cohérence. Par exemple, si, en utilisant la norme uniforme, on a la condition $|\xi|_\infty \leq \varepsilon$, comme classiquement pour la norme euclidienne on a $|\xi|_2 \leq M^{1/2}|\xi|_\infty$, on obtient alors que $|\xi|_2 \leq M^{1/2}\varepsilon$. De même, pour la norme l_1, on obtiendrait $|\xi|_1 \leq M|\xi|_\infty \leq M\varepsilon$. Par conséquent, même si la norme uniforme donne un résultat de proximité acceptable, les distances mesurées par la norme l_1 ou par la norme euclidienne peuvent être importantes. Pour cette raison et par soucis de généralité, dans la suite, nous utiliserons aussi bien des normes hilbertiennes que non-hilbertiennes.*

Définition 1. La matrice $A \in \mathcal{L}(\mathbb{R}^M)$ admet une L-minorante (ou Z-minorante) notée $N(A)$ pour la décomposition en α blocs et les normes définies dans $\mathcal{E}_l, l = 1, \ldots, \alpha$ si, et seulement si, il existe $\eta_{l,l} > 0$ et il existe $g_l \in G_l(\tilde{X}_l)$ tel que

$$< A_{l,l}\tilde{X}_l, g_l >_l \geq \eta_{l,l}|\tilde{X}_l|_l^2, \forall \tilde{X}_l \in \mathcal{E}_l, \, l = 1, \ldots, \alpha, \tag{3.42}$$

et pour tout $l, k \in \{1, \ldots, \alpha\}$, $l \neq k$, $\forall \tilde{X}_l \in \mathcal{E}_l$, $\forall \tilde{X}_k \in \mathcal{E}_k$, $\exists\ \eta_{l,k} \geq 0$ tel que

$$| < A_{l,k}\tilde{X}_k, g_l >_l | \leq \eta_{l,k}|\tilde{X}_k|_k|\tilde{X}_l|_l, \tag{3.43}$$

où $N(A)$ est définie par

$$N(A) = \begin{pmatrix} \eta_{1,1} & \cdots & \cdots & \cdots & \cdots & \cdots & -\eta_{1,\alpha} \\ \vdots & \ddots & & & & & \vdots \\ \vdots & & \ddots & & & & \vdots \\ \vdots & -\eta_{l,k} & \cdots & \eta_{l,l} & \cdots & \cdots & \vdots \\ \vdots & & & & \ddots & & \vdots \\ \vdots & & & & & \ddots & \vdots \\ -\eta_{\alpha,1} & \cdots & \cdots & \cdots & \cdots & \cdots & \eta_{\alpha,\alpha} \end{pmatrix}. \tag{3.44}$$

La L-minorante $N(A)$ est une M-minorante de A si, et seulement si $N(A)$ est une M-matrice inversible, i.e. $n_{l,k} \leq 0$ pour $l \neq k$ et $(N(A))^{-1} \geq 0$.

Définition 2. Une application de point fixe $F : \mathcal{D}(F) \subset \mathcal{E} \mapsto \mathcal{D}(F)$ associée au système linéaire algébrique $A\tilde{X} = \tilde{B}$, où A admet une L-minorante, admet une matrice majorante $J_\alpha \in \mathcal{L}(\mathbb{R}^\alpha)$, où J_α est une matrice non négative ($J_\alpha \geq 0$) pour la décomposition en α blocs de A et pour la norme définie dans $\mathcal{E}_l$ si, et seulement si pour $l = 1, \ldots, \alpha$ et pour tout $\tilde{X}, \tilde{X}' \in \mathcal{E}$ on a

$$|F_l(\tilde{X}_1, \ldots, \tilde{X}_k, \ldots, \tilde{X}_\alpha) - F_l(\tilde{X}'_1, \ldots, \tilde{X}'_k, \ldots, \tilde{X}'_\alpha)|_l \leq \sum_{\substack{k=1 \\ k \neq l}}^{\alpha} \frac{\eta_{l,k}}{\eta_{l,l}} |\tilde{X}_k - \tilde{X}'_k|_k, \tag{3.45}$$

avec

$$\tilde{X} = \{\tilde{X}_1, \ldots, \tilde{X}_k \ldots, \tilde{X}_\alpha\} \in \mathcal{E}, \tag{3.46}$$

et

$$\tilde{X}' = \{\tilde{X}'_1, \ldots, \tilde{X}'_k, \ldots, \tilde{X}'_\alpha\} \in \mathcal{E}, \tag{3.47}$$

où $\tilde{X}_k$ et $\tilde{X}'_k$, $k = 1, \ldots, \alpha$, sont les bloc-composantes de X et X' et la matrice J_α associée à la L-minorante $N(A)$ étant définie par

$$J_\alpha = J_{N(A)} = \begin{pmatrix} 0 & \frac{\eta_{1,2}}{\eta_{1,1}} & \cdots & \cdots & \cdots & \cdots & \frac{\eta_{1,\alpha}}{\eta_{1,1}} \\ \frac{\eta_{2,1}}{\eta_{2,2}} & \ddots & & & & & \vdots \\ \vdots & & \ddots & & & & \vdots \\ \vdots & \frac{\eta_{l,k}}{\eta_{l,l}} & \cdots & 0 & \cdots & \cdots & \vdots \\ \vdots & & & & \ddots & & \vdots \\ \vdots & & & & & \ddots & \vdots \\ \frac{\eta_{\alpha,1}}{\eta_{\alpha,\alpha}} & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \end{pmatrix}. \tag{3.48}$$

Définition 3. Si en plus $N(A)$ est une M-minorante, alors le rayon spectral de J_α est strictement inférieur à 1, et J_α est une matrice de contraction pour la décomposition en α blocs dans $\mathcal{E}$ normé par n'importe quelle norme.

Par conséquent, dans le cadre d'une décomposition en α blocs, soit $q(\cdot)$ la norme vectorielle définie dans $\mathcal{E} = \mathbb{R}^M = \prod_{l=1}^{\alpha} \mathcal{E}_l$ par

$$q(\tilde{X}) = \left\{ |\tilde{X}_1|_1, \ldots, |\tilde{X}_\alpha|_\alpha \right\}. \tag{3.49}$$

Alors, en utilisant cette notation, l'équation (3.45) s'écrit

$$q(F(\tilde{X}) - F(\tilde{X}')) \leq J_\alpha \cdot q(\tilde{X} - \tilde{X}'). \tag{3.50}$$

Remarque 5. *La notion de norme vectorielle correspond à l'introduction d'un outil mathématique bien adapté à la décomposition du problème global en α sous-problèmes interconnectés. Compte tenu de la grande taille des problèmes à résoudre et du caractère*

creux des matrices, les méthodes itératives parallèles sont bien adaptées à la résolution des systèmes algébriques incriminés. Les normes vectorielles permettent donc d'analyser le comportement de ces méthodes itératives parallèles. Cet outil mathématique, indissociable de la notion de minorante, a été utilisé par de nombreux auteurs comme [22, 29, 49, 50, 51, 71]. La condition (3.45) *correspond à une condition de Lipschitz vectorielle qui devient une propriété de contraction en norme vectorielle* (3.50) *si* J_α *est une matrice non-négative de rayon spectral strictement inférieur à* 1.

Remarque 6. *Pour la décomposition en* α *blocs* $J_\alpha = J_{N(A)}$ *est la matrice de Jacobi de* $N(A)$.

Remarque 7. *De plus, au lieu d'une décomposition en* α *blocs, on peut considérer une décomposition par points ; si A est une M-matrice, elle est toujours sa propre M-minorante associée à la décomposition par points, ce qui correspond au cas limite où*

$$\left\{ \begin{array}{l} A_{i,j} = (a_{i,j}) \in \mathcal{L}(\mathbb{R}) = \mathcal{L}(\bar{\mathcal{E}}_i;\bar{\mathcal{E}}_j) \\ \text{avec, dans ce cas } i \equiv l \text{ ; } j \equiv k \text{ ; } i,j \in \{1,\ldots,M\} \\ \text{et } \bar{\mathcal{E}}_i \equiv \bar{\mathcal{E}}_j \equiv \mathbb{R}. \end{array} \right. \tag{3.51}$$

3.3.2.2 Calcul des minorantes

La détermination de $\eta_{l,k}$, quand $l \neq k$, ne pose aucun problème puisque, par utilisation de la formule de Schwartz, $\eta_{l,k}$ est égale à la norme matricielle (cf. annexe A.2) de la sous-matrice $A_{l,k}$. Par contre, pour calculer les éléments diagonaux $\eta_{l,l}$ de la minorante, on a besoin d'un lemme technique qui prend en compte la norme définie dans $\mathcal{E}_l$. On a donc le résultat suivant.

Lemme 1. *Considérons l'inégalité suivante :*

$$< A_{l,l}\tilde{X}_l, g_l(\tilde{X}_l) >_l \geq \eta_{l,l}|\tilde{X}_l|_l^2. \tag{3.52}$$

- *Si l'espace* $\mathcal{E}_l$ *est normé par la norme euclidienne, une condition nécessaire et suffisante pour que* (3.52) *soit vérifiée est que la matrice* $A_{l,l}$ *soit fortement définie positive, i.e. qu'il existe un réel positif* $\eta_{l,l}$ *tel que*

$$< A_{l,l}\tilde{X}_l, \tilde{X}_l >_l \geq \eta_{l,l}|\tilde{X}_l|_{l,2}^2, \ \forall \tilde{X}_l \in \mathcal{E}_l, \ \tilde{X}_l \neq 0. \tag{3.53}$$

- *Si l'espace* $\mathcal{E}_l$ *est normé par la norme* l_1[2], *une condition nécessaire et suffisante pour que* (3.52) *soit vérifiée est que la matrice* $A_{l,l}$ *vérifie les propriétés suivantes :*

$$\begin{array}{l} a_{l,l} \geq \eta_{l,l}, \\ a_{l,l} - \displaystyle\sum_{\substack{k=1 \\ k\neq l}}^{M} |a_{k,l}| \geq \eta_{l,l} \text{ (diagonale strictement dominante par colonne).} \end{array} \tag{3.54}$$

[2]Norme l_1 : $\|X\|_1 = \displaystyle\sum_{l=1}^{M} |x_l|$.

– *Si l'espace $\mathcal{E}_l$ est normé par la norme l_∞[3], une condition nécessaire et suffisante pour que* (3.52) *soit vérifiée est que la matrice $A_{l,l}$ vérifie les propriétés suivantes :*

$$\begin{aligned} &a_{l,l} \geq \eta_{l,l}, \\ &a_{l,l} - \sum_{\substack{k=1 \\ k\neq l}}^{M} |a_{l,k}| \geq \eta_{l,l} \text{ (diagonale strictement dominante par ligne).} \end{aligned} \tag{3.55}$$

Démonstration. Pour les démonstrations de ces résultats, le lecteur est renvoyé à [35] et [76]. □

Les espaces $\mathcal{E}_l$, $l = 1, \ldots, \alpha$, peuvent être normés par toutes les normes classiques telles que la norme euclidienne mais aussi par la norme uniforme l_∞ ou par la norme l_1. Dans ces deux derniers contextes topologiques, c'est-à-dire quand $\mathcal{E}_l$ est normé par une norme non-hilbertienne, il faut utiliser des outils mathématiques plus sophistiqués. En effet, dans un contexte hilbertien, comme on l'a vu, $\tilde{X}$ est la dérivée au sens de Gâteaux ou de Fréchet de la norme euclidienne $\frac{1}{2}|\tilde{X}|^2_{l,2}$ et le produit de dualité entre $\mathcal{E}$ et son espace dual topologique est le produit scalaire. Quand l'espace est normé par la norme uniforme l_∞ ou par la norme l_1, puisque l'application $z \mapsto |z|$, $\forall z \in \mathbb{R}$, la valeur absolue de z, n'est pas dérivable, il faut considérer pour g_l un élément du sous-différentiel de $|\tilde{X}|_\infty$ ou de $|\tilde{X}|_1$ respectivement ; à noter que pour les normes non-hilbertiennes l_1 et l_∞, le sous-différentiel de la norme au carré utilise la fonction sign$(\cdot)$ présentée à l'exemple 5 de l'annexe A.1. Une telle situation est plus complexe du point de vue mathématique mais mène à des critères pratiques plus faciles pour calculer les coefficients diagonaux $\eta_{l,l}$, $l = 1, \ldots, \alpha$ de la L-minorante. D'après le résultat établi dans [35] et rappelé au lemme 1 en (3.54) et (3.55), dans chaque cas, les sous-matrices blocs diagonales sont des matrices strictement diagonale dominante et ont des coefficients diagonaux strictement plus grands que $\eta_{l,l}$, avec $\eta_{l,l} = a_{l,l} - \sum_{k\neq l}|a_{l,k}|$, $\eta_{l,l} > 0$, où $a_{l,k}$ représentent les coefficients diagonaux des sous-matrices $A_{l,l}$. Le calcul des $\eta_{l,k}$, $k \neq l$ est effectué en calculant les normes matricielles des blocs hors-diagonaux $A_{l,k}$, $k \neq l$ en utilisant la norme subordonnée utilisée dans $\mathcal{E}_k$. De cette façon, on peut calculer tous les coefficients de la L-minorante. Dans le cas contraire, si les espaces $\mathcal{E}_l$, $l = 1, \ldots, \alpha$, sont normés par la norme euclidienne, alors les coefficients diagonaux de la L-minorante sont les plus petites valeurs propres des sous-matrices blocs diagonales $A_{l,l}$, $l = 1, \ldots, \alpha$, calculées par la méthode de la puissance inverse, alors que le calcul des normes matricielles des sous-matrices blocs hors-diagonales nécessite l'utilisation de la méthode de la puissance itérée ; de tels calculs sont très coûteux en comparaison des calculs utilisant les normes non-hilbertiennes. Dans le cas général, il est bon de noter que de tels critères liés aux propriétés de diagonale dominance stricte des sous-matrices blocs diagonales sont plus faciles à mettre en œuvre que ceux qui requièrent le calcul des valeurs propres des sous-matrices blocs diagonales, calcul toujours délicat à cause de la propagation d'erreurs d'arrondis.

[3]Norme l_∞ : $\|X\|_\infty = \max_{l\in 1,\ldots,M} |x_l|$.

Remarque 8. *Lorsque l'espace est normé par les normes l_p, une propriété analogue à* (3.52) *dans ce contexte topologique, permet d'avoir une caractérisation des M-matrices et des H-matrices. Cependant, ce résultat conduit à des critères moins pratiques que ceux énoncés dans le lemme 1 (cf. [77]).*

Notons maintenant $\bar{\eta}_{l,l} = \eta_{l,l}$ et $\bar{\eta}_{l,k} = -\eta_{l,k}$, $l \neq k$; pour la décomposition en α blocs, on peut alors caractériser une L-minorante de la matrice A comme suit.

Proposition 1. *Les deux hypothèses* (3.42) *et* (3.43) *sont équivalentes à*

$$< \sum_{k=1}^{\alpha} A_{l,k}\tilde{X}_k, g_l >_l \geq \sum_{k=1}^{\alpha} \bar{\eta}_{l,k}|\tilde{X}_k|_k|\tilde{X}_l|_l, \ \forall l \in \{1,\ldots,\alpha\}. \tag{3.56}$$

Démonstration. On peut directement vérifier que (3.42) et (3.43) impliquent (3.56) ; en effet

$$< A_{l,l}\tilde{X}_l, g_l >_l \geq \eta_{l,l}|\tilde{X}_l|_l, \ \forall \tilde{X}_l \in \mathcal{E}_l, \ l = 1,\ldots,\alpha, \tag{3.57}$$

et, pour $l, k = 1,\ldots,\alpha$, $l \neq k$,

$$-\eta_{l,k}|\tilde{X}_k|_k \leq | < A_{l,k}\tilde{X}_k, g_l >_l | \leq \eta_{l,k}|\tilde{X}_k|_k \, ; \tag{3.58}$$

puis, par sommation, on obtient (3.56). Pour la réciproque, le lecteur est renvoyé à [55] et [76]. □

Remarque 9. *Considérons la solution au problème* (3.33) *et considérons une décomposition en α blocs de la matrice A. Écrivons les équations linéaires pour $\tilde{X}$ et $\tilde{X}'$; on soustrait membre à membre et on multiplie par g_l ; alors, avec une telle décomposition, on obtient*

$$< \sum_{k=1}^{\alpha} A_{l,k}(\tilde{X}_k - \tilde{X}'_k), g_l(\tilde{X}_l - \tilde{X}'_l) >_l = 0, \ \forall l \in \{1,\ldots,\alpha\}. \tag{3.59}$$

La norme vectorielle $q(\cdot)$ associée à la décomposition en α blocs étant définie par (3.49)*, si la matrice A admet une M-minorante, grace à* (3.56) *et* (3.59) *on obtient de façon évidente et directement une propriété de contraction en norme vectorielle $q(\cdot)$ telle que* (3.50)*.*

Remarque 10. *Lorsqu'une matrice A est symétrique définie positive, classiquement, l'inégalité suivante est vérifiée :*

$$< A\tilde{X}, \tilde{X} > \geq \bar{\eta}\|\tilde{X}\|_2^2, \tag{3.60}$$

où $\bar{\eta}$ est la plus petite valeur propre de A ; la relation (3.60) *s'obtient aisément compte tenu du fait que les vecteurs propres forment une base orthonormée. En fait, l'inégalité* (3.56) *est l'analogue de la relation précédente qui tient compte ici, de la décomposition du problème global en α sous-problèmes. Notons que si la matrice A n'est plus*

symétrique mais est définie positive, alors on obtient l'analogue de l'inégalité (3.60) *par un raisonnement de compacité. En effet, soit*

$$S = \{\tilde{X} \in \mathbb{R}^M,\ \|\tilde{X}\|_2 = 1\}, \tag{3.61}$$

la boule unité de $\mathbb{R}^M$ *et* Q *l'ensemble des sous-espaces vectoriels de* $\mathbb{R}^M$*. Soit* $S_Q = S \cap Q$*, l'intersection des ensembles* S *et* Q*. Le produit scalaire étant une fonction continue, l'application* $\tilde{X} \mapsto J_\alpha(\tilde{X})$ *de* $\mathbb{R}^M$ *dans* $\mathbb{R}$ *est continue ; l'ensemble* S_Q *étant fermé et borné, la restriction de* J_α *à* S_Q *est également continue ; donc il existe un vecteur* $\tilde{X}_0 \in S_Q$ *tel que* $J_\alpha(\tilde{X}_0) = \bar{\eta}$ *réalise le minimum de* $J_\alpha(\tilde{X})$ *sur l'ensemble* S_Q *; la matrice* A *étant définie positive, on a* $J_\alpha(\tilde{X}_0) = \bar{\eta} > 0$*. Soit un vecteur* $\tilde{Y} \in Q$*,* $\tilde{Y} \neq 0$ *; posons*

$$\tilde{X} = \frac{\tilde{Y}}{\|\tilde{Y}\|_2} \in S_Q, \tag{3.62}$$

donc

$$< A\tilde{X}, \tilde{X} > = < A\frac{\tilde{Y}}{\|\tilde{Y}\|_2}, \frac{\tilde{Y}}{\|\tilde{Y}\|_2} > \geq J_\alpha(\tilde{X}_0) = \bar{\eta}, \tag{3.63}$$

ce qui entraîne $< A\tilde{Y}, \tilde{Y} > \geq \bar{\eta}\|\tilde{Y}\|_2^2$*.*

3.3.2.3 Exemples

Exemple 1. Considérons le problème classique de Dirichlet bi-dimensionnel avec conditions aux limites de Dirichlet homogènes dans un domaine carré $\Omega =]0,1[\times]0,1[$ de frontière $\partial\Omega$:

$$\begin{cases} -\nabla^2 X = f \text{ dans } \Omega, \\ X|_{\partial\Omega} = 0. \end{cases} \tag{3.64}$$

La discrétisation du problème (3.64) par différences finies classiques ou éléments finis P_1 ou volumes finis conduit à déterminer un vecteur $X^* \in \mathbb{R}^M$ solution du système

$$AX^* = B, \tag{3.65}$$

où X^*, B sont des vecteurs de $\mathbb{R}^M$ et A est la matrice triple-diagonale suivante :

$$A = \begin{pmatrix} D & -\mathbb{I} & & \\ -\mathbb{I} & D & -\mathbb{I} & \\ & -\mathbb{I} & D & -\mathbb{I} \\ & & -\mathbb{I} & D \end{pmatrix}, \tag{3.66}$$

où $\mathbb{I}$ représente la matrice identité et

$$D = \begin{pmatrix} 4 & -1 & & & & & \\ -1 & \ddots & \ddots & & 0 & & \\ & \ddots & \ddots & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ & & 0 & & \ddots & \ddots & -1 \\ & & & & & -1 & 4 \end{pmatrix}. \tag{3.67}$$

Contexte hilbertien, norme euclidienne. Calculons la minorante associée à A lorsque l'espace est normé par la norme euclidienne. On peut écrire que

$$D = \tilde{D} + 2\mathbb{I}, \tag{3.68}$$

où $\tilde{D}$ est une matrice tri-diagonale représentée par :

$$\tilde{D} = \begin{pmatrix} 2 & -1 & & & & & \\ -1 & \ddots & \ddots & & 0 & & \\ & \ddots & \ddots & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ & & 0 & & \ddots & \ddots & -1 \\ & & & & & -1 & 2 \end{pmatrix}. \tag{3.69}$$

Comme $\tilde{D}$ est une matrice symétrique définie positive, on peut écrire

$$< DX, X >\geq 2\|X\|_{l,2}^2. \tag{3.70}$$

Par ailleurs, lorsque les sous-espaces $\mathcal{E}_k$ sont normés par la norme euclidienne, la norme matricielle induite est évidemment égale à

$$]|\mathbb{I}|[= 1. \tag{3.71}$$

Donc en appliquant le lemme 1, compte tenu du fait que l'espace est normé par la norme euclidienne, la minorante associée à la matrice A est

$$N(A) = \begin{pmatrix} 2 & -1 & & & & & \\ -1 & \ddots & \ddots & & 0 & & \\ & \ddots & \ddots & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ & & 0 & & \ddots & \ddots & -1 \\ & & & & & -1 & 2 \end{pmatrix}. \tag{3.72}$$

De plus, c'est une M-matrice car c'est une matrice à diagonale dominante irréductible est donc $N(A)$ est une M-minorante.

On vérifie aisément que les valeurs propres de la matrice de Jacobi associée à la minorante sont données par :

$$\hat{\delta}(J_\alpha) = \cos(\pi h), \tag{3.73}$$

où h est le pas de discrétisation en espace, supposé uniforme pour des questions de simplicité de présentation. Évidemment $|\hat{\delta}| < 1$ et J_α est une matrice de contraction.

Contexte non-hilbertien, normes l_1 et l_∞. Si à présent on considère que l'espace est normé par la norme l_1, en partant de (3.68) et en appliquant le résultat du lemme 1, on trouve une M-minorante identique à (3.72). De plus, si on norme l'espace par la norme uniforme l_∞, la même démarche conduit à la même détermination d'une M-minorante (3.72).

Par ailleurs, si nous considérons le problème

$$\begin{cases} -\nabla^2 X + \theta h^2 X = f \text{ dans } \Omega,\ \theta > 0, \\ X|_{\partial\Omega} = 0, \end{cases} \tag{3.74}$$

où θ pourrait provenir de la discrétisation implicite ou semi-implicite d'un problème d'évolution du premier ordre, i.e. un problème parabolique, $\theta = 1/\delta t$ (δt étant le pas de discrétisation en temps), alors, lorsque l'espace $\mathcal{E}$ est normé par l'une des normes quelconque précédentes, en effectuant le même calcul, on détermine aisément une minorante

$$\tilde{N}(A) = N(A) + \theta h^2 \mathbb{I}, \tag{3.75}$$

où $N(A)$ est définie par (3.72).

Remarque 11. *Dans le cas de problèmes tri-dimensionnels, on peut de manière identique déterminer des M-minorantes de l'opérateur laplacien et on vérifie qu'elles sont constituées des matrices de discrétisation bi-dimensionnelles (cf. exemple 3).*

Remarque 12. *Dans le cas de la résolution numérique de l'équation des ondes ou de l'équation de Navier, le développement précédent concernant l'équation de la chaleur est encore valable mais avec ici $\theta = 1/(\delta t)^2$; cette estimation est encore plus intéressante que celle développée à l'exemple 1 puisque la M-minorante a des propriétés de diagonale dominance plus intéressantes. Cela aura pour conséquence une très grande vitesse de convergence à cause du fait que δt étant petit pour assurer la consistance des schémas, la variable θ sera par conséquent plus grande et donc le rayon spectral de la matrice de Jacobi associée à $N(A)$ sera petit ce qui implique une vitesse de convergence élevée.*

Exemple 2. On considère à présent l'exemple d'une matrice qui n'est pas une M-matrice mais qui admet une M-minorante pour une décomposition quelconque. Soit $c \in \mathbb{R}$ un nombre réel différent de 0 aussi grand que l'on veut, et $a, b \in \mathbb{R}$ deux nombres réels vérifiant

$$1 - a \times b > 0. \tag{3.76}$$

On considère la matrice

$$A = \begin{pmatrix} 1 & c & 0 \\ -c & 1 & -a \\ -b & 0 & 1 \end{pmatrix}. \tag{3.77}$$

De manière évidente, A n'est pas une M-matrice. On considère la décomposition de A en deux blocs

$$\begin{aligned} A_{1,1} &= \begin{pmatrix} 1 & c \\ -c & 1 \end{pmatrix}, \quad A_{1,2} = \begin{pmatrix} 0 \\ -a \end{pmatrix}, \\ A_{1,2} &= \begin{pmatrix} -b & 0 \end{pmatrix}, \quad A_{2,2} = \begin{pmatrix} 1 \end{pmatrix}. \end{aligned} \tag{3.78}$$

Pour cette décomposition, on définit les sous-vecteurs X_i, $i = 1, 2$, de manière compatible avec la décomposition en blocs précédente

$$X_1 = (x_1, x_2) \text{ et } X_2 = (x_3). \tag{3.79}$$

Puisque $A_{1,1}$ est anti-symétrique, on a

$$X_1^T A_{1,1} X_1 = \|X_1\|_2^2. \tag{3.80}$$

De plus, on calcule la L-minorante associée à la matrice A et qui est donnée par

$$N(A) = \begin{pmatrix} 1 & -a \\ -b & 1 \end{pmatrix}. \tag{3.81}$$

qui est une M-matrice puisque $1 - a \times b > 0$.

Exemple 3. On considère le problème de convection-diffusion tri-dimensionnel suivant

$$\begin{cases} -\nabla^2 X + \delta \nabla X = f \text{ dans } \Omega = [0,1]^3, \\ X|_{\partial\Omega} = 0, \end{cases} \tag{3.82}$$

où $\delta = \{\delta_x, \delta_y, \delta_z\}$ représente le coefficient de convection.

La discrétisation du problème (3.82) est effectuée d'abord par des différences centrées. On généralise ce qui a été vu dans l'exemple 2 à ce problème. La discrétisation par différences finies centrées conduit à une matrice de discrétisation de la forme suivante :

$$H = A + C, \tag{3.83}$$

où A est la matrice de discrétisation du terme de diffusion et C est la matrice de discrétisation du terme de convection. Remarquons tout de suite que la matrice C est anti-symétrique. Donc, si on considère que l'espace $\mathcal{E}$ est normé par la norme euclidienne, on peut calculer aisément la minorante et compte tenu de la propriété d'anti-symétrie de C, on retrouve la minorante $N(A)$ donnée par (3.72) lors de l'étude de l'exemple 1.

Par ailleurs, si l'espace $\mathcal{E}$ est à présent normé par la norme l_∞ ou l_1, des propriétés de diagonale dominance stricte des matrices blocs diagonales sont nécessaires compte tenu du résultat du lemme 1. Pour obtenir de telles propriétés, on considère non plus un

schéma centré comme précédemment, mais un schéma décentré défini comme suit, pour le terme de convection $\delta_x \partial X/\partial x$:

$$\delta_x \frac{\partial X}{\partial x} = \begin{cases} \delta_x \dfrac{X(x,y,z) - X(x-h,y,z)}{h} \text{ si } \delta_x > 0, \\ \delta_x \dfrac{X(x+h,y,z) - X(x,y,z)}{h} \text{ si } \delta_x < 0, \end{cases} \tag{3.84}$$

et de manière identique pour la dérivée par rapport à y et z. Dans ce cas, on obtient une matrice bloc diagonale à diagonale dominante irréductible, et on peut calculer la minorante. Donc par exemple, pour $\delta_x > 0$, $\delta_y > 0$ et $\delta_z > 0$, on a la minorante $N(H)$ suivante :

$$N(H) = \left(\begin{array}{cccc|cccc|cccc} a & -c & & & -e & & & & & & & \\ -b & \ddots & \ddots & & & \ddots & & & & & & \\ & \ddots & \ddots & -c & & & \ddots & & & & & \\ & & -b & a & & & & -e & & & & \\ \hline -d & & & & \ddots & \ddots & & & -e & & & \\ & \ddots & & & \ddots & \ddots & \ddots & & & \ddots & & \\ & & \ddots & & & \ddots & \ddots & \ddots & & & \ddots & \\ & & & -d & & & \ddots & \ddots & & & & -e \\ \hline & & & & -d & & & & a & -c & & \\ & & & & & \ddots & & & -b & \ddots & \ddots & \\ & & & & & & \ddots & & & \ddots & \ddots & -c \\ & & & & & & & -d & & & -b & a \end{array}\right), \tag{3.85}$$

où

$$a = \frac{4}{h^2} + \frac{\delta_y + \delta_z}{h},\ b = d = \frac{1}{h^2},\ c = \left(\frac{1}{h^2} + \frac{\delta_z}{h}\right) \text{ et } e = \left(\frac{1}{h^2} + \frac{\delta_y}{h}\right), \tag{3.86}$$

qui est aussi diagonale dominante irréductible et c'est une M-minorante.

Les valeurs propres de J_α sont données par

$$\mathcal{V}(J_\alpha) = -2\frac{\sqrt{bc}}{a}\cos(k\pi h) - 2\frac{\sqrt{de}}{a}\cos(l\pi h),\ k,l = 1,\ldots,n, \tag{3.87}$$

où $h = 1/(n+1)$ est le pas de discrétisation en espace, supposé uniforme pour des questions de simplicité de présentation, avec $M = n^2$. Puisque $N(H)$ est une M-matrice alors le rayon spectral de J_α donné par

$$\hat{\delta}(J_\alpha) = \frac{2}{a}\left(\sqrt{bc} + \sqrt{de}\right)\cos(\pi h), \tag{3.88}$$

est bien inférieur à 1 et J_α est une matrice de contraction.

Remarque 13. *On retrouve la matrice de discrétisation du problème de convection-diffusion bi-dimensionnel.*

3.3.2.4 Résultat général de convergence pour une décomposition en grands blocs

En calcul parallèle, l'utilisateur n'a pas accès à autant de processeurs que de blocs composantes. Dans la pratique, le système algébrique à résoudre est décomposé en β blocs adjacents, $\beta \leq \alpha$ (en pratique, $\beta \ll \alpha$), correspondant à une décomposition en sous-domaines plus grossière constituée par un regroupement de blocs adjacents de la décomposition la plus fine (cf. figure 3.2) ; les communications se font alors par des échanges des valeurs des composantes entre les processeurs voisins. De telles conditions, revenant à considérer une décomposition plus grossière d'une décomposition donnée, ont été étudiées dans [55] où, il a été prouvé que si l'application de point fixe est contractante pour une décomposition donnée, alors l'application de point fixe est aussi contractante pour une décomposition plus grossière ; donc, dans les deux cas, les itérations parallèles asynchrones convergent.

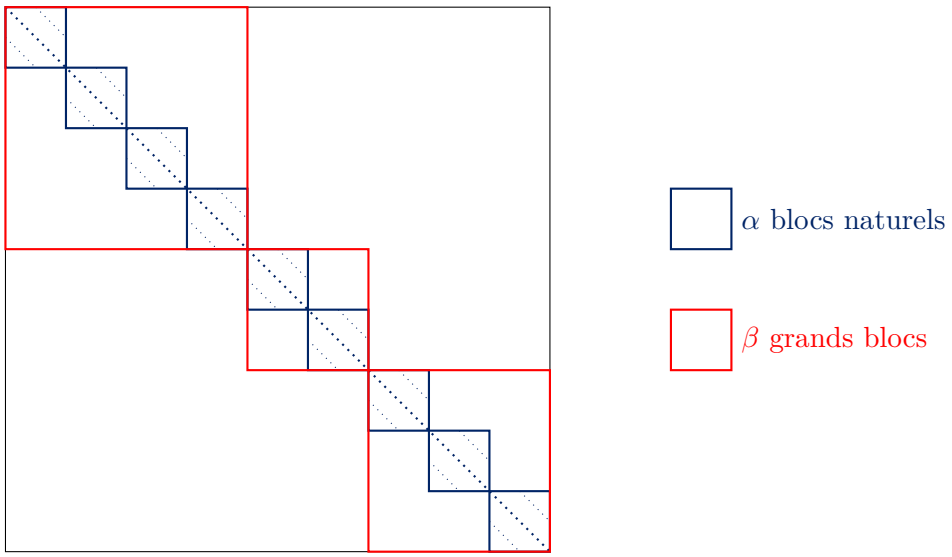


FIGURE 3.2 – Construction de la $\beta-$décomposition

Dans la suite, nous considérerons que $N(A) \in \mathcal{L}(\mathbb{R}^\alpha)$ est une M-minorante, tel que l'application de point fixe F associée à la décomposition en α blocs est contractante d'après (3.45) ou (3.50) et nous montrerons que, dans ce contexte, l'application de point fixe $\hat{F}$ associée à la décomposition plus grossière en β blocs, $\beta \leq \alpha$ est aussi contractante.

$$\left\{ \begin{array}{l} \text{Soit } \beta \in \mathbb{N} \text{ et } \{\alpha_i\} \text{ pour } i \in \{1, \ldots, \beta\} \\ \text{un ensemble d'entiers tels que } \alpha_i \neq 0 \text{ et } \displaystyle\sum_{i=1}^{\beta} \alpha_i = \alpha, \end{array} \right. \tag{3.89}$$

où, pour $i = 1, \ldots, \beta$, α_i représente le nombre de blocs de la décomposition plus fine, i.e. la α-décomposition, ces blocs étant regroupés pour définir le i-ème grand bloc de la décomposition plus grossière, i.e. la β-décomposition.

$$\left\{\begin{array}{l} \text{Soit } \forall i \in \{2,\ldots,\beta+1\},\ \beta_i = \displaystyle\sum_{j=1}^{i-1} \alpha_j \text{ tel que } \beta_1 = 0 \\ \\ \text{et } \beta_{\beta+1} = \alpha, \text{ et } \forall i \in \{1,\ldots,\beta\},\ \hat{\mathcal{E}}_i = \displaystyle\prod_{l=\beta_i+1}^{\beta_{i+1}} \mathcal{E}_l, \end{array}\right. \tag{3.90}$$

où le i-ème grand bloc de la β-décomposition est constitué en regroupant le (β_i+1)-ème bloc jusqu'au β_{i+1}-ème bloc. Il vient alors

$$\mathcal{E} = \prod_{i=1}^{\beta} \hat{\mathcal{E}}_i = \prod_{i=1}^{\alpha} \mathcal{E}_i, \tag{3.91}$$

et

$$\tilde{X} = \left\{\hat{\tilde{X}}_1,\ldots,\hat{\tilde{X}}_\beta\right\} \in \prod_{i=1}^{\beta} \hat{\mathcal{E}}_i, \tag{3.92}$$

où $\hat{\tilde{X}}_i$, $i = 1,\ldots,\beta$, représente le bloc-composante de $\tilde{X}$ exprimé dans la β-décomposition.

Remarque 14. *La hiérarchie de blocs considérée permet de démontrer que sous des hypothèses convenables, vérifiées si la matrice de discrétisation est une M-matrice, ce qui est le cas pour les systèmes linéaires* (3.8), (3.12), (3.15), (3.18), (3.30) *ainsi que pour l'équation régissant la vitesse de grille* $\mathbf{u}^*$*, les applications de point fixe associées à la décomposition en grands blocs sont contractantes quelle que soit la décomposition en grands blocs considérée. Ce résultat est énoncé dans la suite à la proposition 2 et au corollaire 3. Ceci permet ensuite d'appliquer et d'étendre utilement un résultat de* Bahi *et al. [5] pour l'analyse des méthodes de multi-splitting.*

Notons $\Lambda :\ D(\Lambda) \subset \mathcal{E} \mapsto \mathcal{E}$ l'application suivante définie par

$$\Lambda_l(\tilde{X}_1,\ldots,\tilde{X}_{l-1},\tilde{X}_l,\tilde{X}_{l+1},\ldots,\tilde{X}_\alpha) = \sum_{k=1}^{\alpha} A_{l,k}\tilde{X}_k, \tag{3.93}$$

tel que nous pouvons écrire le problème initial (3.33) comme suit :

$$\Lambda_l(\tilde{X}_1,\ldots,\tilde{X}_{l-1},\tilde{X}_l,\tilde{X}_{l+1},\ldots,\tilde{X}_\alpha) - \tilde{B}_l = 0,\ \forall l \in \{1,\ldots,\alpha\}. \tag{3.94}$$

Une décomposition plus grossière du problème (3.33) est définie par

$$\hat{\Lambda}_i(\hat{\tilde{X}}_1,\ldots,\hat{\tilde{X}}_{i-1},\hat{\tilde{X}}_i,\hat{\tilde{X}}_{i+1},\ldots,\hat{\tilde{X}}_\beta) - \hat{\tilde{B}}_i = 0,\ \forall i \in \{1,\ldots,\beta\}, \tag{3.95}$$

où

$$\hat{\Lambda}_i(\hat{\tilde{X}}_1,\ldots,\hat{\tilde{X}}_{i-1},\hat{\tilde{X}}_i,\hat{\tilde{X}}_{i+1},\ldots,\hat{\tilde{X}}_\beta) = \sum_{j=1}^{\beta} \hat{A}_{i,j}\hat{\tilde{X}}_j,\ \forall i \in \{1,\ldots,\beta\}, \tag{3.96}$$

les matrices $\hat{A}_{i,j}$, $i,j = 1,\ldots,\beta$, étant obtenues en regroupant de manière adéquate les blocs $A_{l,k}$ de la décomposition en α blocs. On considère alors la décomposition du problème (3.33) en β sous-problèmes :

$$\hat{\Lambda}_i(\hat{\tilde{Y}}_1,\ldots,\hat{\tilde{Y}}_{i-1},\hat{\tilde{X}}_i,\hat{\tilde{Y}}_{i+1},\ldots,\hat{\tilde{Y}}_\beta) - \hat{\tilde{B}}_i = 0,\ \forall i \in \{1,\ldots,\beta\}, \tag{3.97}$$

où, pour simplifier les notations, $\hat{\tilde{Y}}_j$ pour $j \neq i$, $j = 1,\ldots,\beta$, représentent les valeurs retardées d'interaction définies par $\hat{\tilde{Y}}_j = \hat{\tilde{X}}_j^{(\kappa_j(r))}$ et où, pour une telle décomposition, $\hat{\tilde{X}}_i$ est défini par

$$\hat{\tilde{X}}_i = \{\ldots,\tilde{X}_l,\ldots\},\ \text{pour } l \in \{\beta_i+1,\ldots,\beta_{i+1}\}. \tag{3.98}$$

Considérons aussi le même problème pour un vecteur $\tilde{X}'$ distinct, tel que

$$\hat{\Lambda}_i(\hat{\tilde{Y}}'_1,\ldots,\hat{\tilde{Y}}'_{i-1},\hat{\tilde{X}}'_i,\hat{\tilde{Y}}'_{i+1},\ldots,\hat{\tilde{Y}}'_\beta) - \hat{\tilde{B}}_i = 0,\ \forall i \in \{1,\ldots,\beta\}, \tag{3.99}$$

où de la même façon

$$\hat{\tilde{X}}'_i = \{\ldots,\tilde{X}'_l,\ldots\},\ \text{pour } l \in \{\beta_i+1,\ldots,\beta_{i+1}\}. \tag{3.100}$$

En partant de la β-décomposition, la décomposition la plus grossière, nous pouvons reformuler le même problème pour la α-décomposition, la décomposition la plus fine ; en soustrayant et en multipliant par $g_l(\tilde{X}_l - \tilde{X}'_l)$, on obtient, pour $l \in \{\beta_i+1,\ldots,\beta_{i+1}\}$,

$$<\Lambda_l(\tilde{Y}_1,\ldots,\tilde{Y}_{l-1},\tilde{X}_l,\tilde{Y}_{l+1},\ldots,\tilde{Y}_\alpha) - \Lambda_l(\tilde{Y}'_1,\ldots,\tilde{Y}'_{l-1},\tilde{X}'_l,\tilde{Y}'_{l+1},\ldots,\tilde{Y}'_\alpha), g_l(\tilde{X}_l - \tilde{X}'_l)>_l = 0, \tag{3.101}$$

et, puisque (3.56) est vérifiée, alors, après simplification par $|\tilde{X}_l - \tilde{X}'_l|_l$, il vient

$$\left(\sum_{k=1}^{\beta_i} \bar{\eta}_{l,k}|\tilde{Y}_k - \tilde{Y}'_k|_k + \sum_{k=\beta_i+1}^{\beta_{i+1}} \bar{\eta}_{l,k}|\tilde{X}_k - \tilde{X}'_k|_k + \sum_{k=\beta_{i+1}+1}^{\alpha} \bar{\eta}_{l,k}|\tilde{Y}_k - \tilde{Y}'_k|_k\right) \leq 0\,; \tag{3.102}$$

puisque $\bar{\eta}_{l,k} \leq 0$, pour $k \neq l$ on obtient pour $l \in \{\beta_i+1,\ldots,\beta_{i+1}\}$,

$$\sum_{k=\beta_i+1}^{\beta_{i+1}} \bar{\eta}_{l,k}|\tilde{X}_k - \tilde{X}'_k|_k \leq \left(-\sum_{k=1}^{\beta_i} \bar{\eta}_{l,k}|\tilde{Y}_k - \tilde{Y}'_k|_k - \sum_{k=\beta_{i+1}+1}^{\alpha} \bar{\eta}_{l,k}|\tilde{Y}_k - \tilde{Y}'_k|_k\right). \tag{3.103}$$

Considérons la décomposition par blocs de la M-minorante $N(A) = \Delta - (L+R)$ en β blocs $\hat{N}(A)_{i,j}$, $i,j = 1,\ldots,\beta$ tel que pour $i,j \in \{1,\ldots,\beta\}$, les coefficients du bloc $\hat{N}(A)_{i,j}$ sont $\bar{\eta}_{l,k}$ pour $l \in \{\beta_i+1,\ldots,\beta_{i+1}\}$ et $k \in \{\beta_j+1,\ldots,\beta_{j+1}\}$. Par conséquent, on définit la matrice bloc diagonal $\hat{\Delta}$ avec les blocs diagonaux $\hat{N}(A)_{i,i} = (\hat{\Delta}_{i,i})$, par $\hat{\Delta}_{i,i} = (\bar{\eta}_{l,k})$, $l,k \in \{\beta_i+1,\ldots,\beta_{i+1}\}$, $i \in \{1,\ldots,\beta\}$. On définit aussi la partie triangulaire inférieure stricte par blocs $\hat{L} = (\hat{L}_{i,j})$ par $\hat{L}_{i,j} = (-\bar{\eta}_{l,k})$, $l \in \{\beta_i+1,\ldots,\beta_{i+1}\}$, $k \in \{\beta_j+1,\ldots,\beta_{j+1}\}$ pour $i \in \{2,\ldots,\beta\}$ et $j \in \{1,\ldots,i-1\}$, i.e. $j < i$ et, de la même manière, la partie triangulaire supérieure stricte par blocs $\hat{R} = (\hat{R}_{i,j})$ par $\hat{R}_{i,j} = (-\bar{\eta}_{l,k})$,

$l \in \{\beta_i+1,\ldots,\beta_{i+1}\}$, $k \in \{\beta_j+1,\ldots,\beta_{j+1}\}$, pour $i \in \{1,\ldots,\beta-1\}$ et $j \in \{i+1,\ldots,\beta\}$, i.e. $j > i$; en d'autres mots, on a

$$\hat{L}_{i,j} = \begin{cases} -\hat{N}(A)_{i,j} & \text{si } i > j, \\ 0 & \text{si } i \leq j, \end{cases} \quad \text{et } \hat{R}_{i,j} = \begin{cases} -\hat{N}(A)_{i,j} & \text{si } i < j, \\ 0 & \text{si } i \geq j. \end{cases} \tag{3.104}$$

Donc, la matrice $\hat{\Delta}$ est de dimension α ; on peut écrire l'inégalité (3.103) de la façon suivante :

$$\hat{\Delta} \cdot q(\tilde{X} - \tilde{X}') \leq (\hat{L} + \hat{R}) \cdot q(\tilde{Y} - \tilde{Y}'). \tag{3.105}$$

Rappelons la définition suivante (cf. [61] aux pages 56 à 58).

Définition 4. Soit $N(A)$, $\hat{\Delta}$, $\hat{L}$, $\hat{R} \in \mathcal{L}(\mathcal{R}^\alpha)$. Alors $\hat{\Delta} - (\hat{L} + \hat{R})$ est un partitionnement régulier de $N(A)$ si $\hat{\Delta}$ est inversible avec $\hat{\Delta}^{-1} \geq 0$, et $(\hat{L} + \hat{R}) \geq 0$.

De plus, d'après [61], nous avons le lemme suivant.

Lemme 2. *Soit $N(A) \in \mathcal{L}(\mathcal{R}^\alpha)$; supposons que $\hat{\Delta} - (\hat{L} + \hat{R})$ est un partitionnement régulier de $N(A)$. Alors, le rayon spectral $\hat{\delta}$ de $(\hat{\Delta}^{-1} \cdot (\hat{L} + \hat{R}))$ satisfait la condition $\hat{\delta} < 1$ si, et seulement si $\hat{\Delta}^{-1}$ existe et est non négative.*

Puisque $N(A)$ est une M-matrice, on peut appliquer le lemme précédent ; en effet $\hat{\Delta}$ est une M-matrice et $\hat{\Delta} - (\hat{L} + \hat{R})$ est un partitionnement régulier de $N(A)$; alors, le rayon spectral $\hat{\delta}$ de la matrice $\hat{J} = \hat{\Delta}^{-1}(\hat{L} + \hat{R})$ satisfait la condition

$$\hat{\delta} < 1. \tag{3.106}$$

Par conséquent, comme $F(\tilde{Y}) = \hat{F}(\hat{\tilde{Y}})$ puisqu'on ne fait que réécrire les mêmes relations correspondant à des décompositions différentes, où F est l'application de point fixe associée à la α-décomposition (la plus fine) et $\hat{F}$ est l'application de point fixe associée à la β-décomposition (la plus grossière), pour le partitionnement correspondant à la décomposition la plus grossière, la matrice bloc $\hat{J}$ est une matrice de contraction pour la norme vectorielle $q(\cdot)$, et on obtient

$$q(\hat{F}(\hat{\tilde{X}}) - \hat{F}(\hat{\tilde{X}'})) \leq \hat{J} \cdot q(\hat{\tilde{Y}} - \hat{\tilde{Y}'}). \tag{3.107}$$

On résume ce résultat comme suit.

Proposition 2. *Supposons que $N(A)$ est une M-minorante pour la décomposition en α blocs. Considérons une décomposition plus grossière en β blocs telle que $\beta \leq \alpha$. Alors, l'inégalité* (3.50) *est valable et $\hat{J} = \hat{\Delta}^{-1}(\hat{L} + \hat{R})$ est une matrice de contraction pour l'application de point fixe associée à la β-décomposition.*

Rappelons que la matrice $\hat{J}$, constituée de coefficients diagonaux nuls et de coefficients hors-diagonaux égaux à $-\eta_{l,k}/\eta_{l,l}$, est la matrice de Jacobi de la matrice $N(A)$ (cf. remarque 6). Puisque $N(A)$ est une M-matrice, alors $\hat{J}$ est un matrice non négative et toutes les valeurs propres de $\hat{J}$ ont un module inférieur à un. Notons Θ le vecteur propre

associé au rayon spectral $\hat{\delta}$ de $\hat{J}$. Grâce au théorème de Perron-Frobenius et au lemme 5 de l'annexe A.3, toutes les composantes de Θ sont strictement positives et l'inégalité suivante $\hat{J}\Theta \leq \hat{\delta}\Theta$, $0 \leq \hat{\delta} < 1$ est valable. On a donc

$$q(\hat{F}(\hat{\tilde{X}}) - \hat{F}(\hat{\tilde{X}}')) \leq \hat{J} \cdot q(\hat{\tilde{Y}} - \hat{\tilde{Y}}'). \tag{3.108}$$

En passant aux composantes, on a donc, pour $i \in \{1, \ldots, \beta\}$,

$$|\hat{F}_i(\hat{\tilde{X}}) - \hat{F}_i(\hat{\tilde{X}}')|_i \leq \sum_{j \neq i} \hat{J}_{i,j} |\hat{\tilde{Y}}_j - \hat{\tilde{Y}}'_j|_j, \tag{3.109}$$

ce qu'on peut encore écrire

$$|\hat{F}_i(\hat{\tilde{X}}) - \hat{F}_i(\hat{\tilde{X}}')|_i \leq \sum_{j \neq i} \hat{J}_{i,j} \Theta_j \frac{|\hat{\tilde{Y}}_j - \hat{\tilde{Y}}'_j|_j}{\Theta_j}, \tag{3.110}$$

soit encore

$$|\hat{F}_i(\hat{\tilde{X}}) - \hat{F}_i(\hat{\tilde{X}}')|_i \leq \max_j \left(\frac{|\hat{\tilde{Y}}_j - \hat{\tilde{Y}}'_j|_j}{\Theta_j} \right) \cdot \sum_{j \neq i} \hat{J}_{i,j} \Theta_j. \tag{3.111}$$

Or, $\sum_{j \neq i} \hat{J}_{i,j} \Theta_j = \hat{\delta} \Theta_i$, soit

$$\max_i \left(\frac{|\hat{F}_i(\hat{\tilde{X}}) - \hat{F}_i(\hat{\tilde{X}}')|_i}{\Theta_i} \right) \leq \hat{\delta}_i \max_j \left(\frac{|\hat{\tilde{Y}}_j - \hat{\tilde{Y}}'_j|_j}{\Theta_j} \right), \tag{3.112}$$

soit encore

$$\|\hat{F}(\hat{\tilde{X}}) - \hat{F}(\hat{\tilde{X}}')\|_\Theta \leq \hat{\delta} \|\hat{\tilde{Y}} - \hat{\tilde{Y}}'\|_{\nu,\Theta}, \, \forall \hat{\tilde{Y}}, \hat{\tilde{Y}}' \in \mathcal{E}, \, 0 \leq \hat{\delta} < 1, \tag{3.113}$$

où $\|\cdot\|_\Theta$ est une norme pondérée uniforme définie par

$$\|\hat{\tilde{X}}\|_\Theta = \max_{l=1,\ldots,\alpha} \left(\frac{|\hat{\tilde{X}}_l|_l}{\Theta_i} \right). \tag{3.114}$$

En utilisant l'inégalité (3.113), l'application de point fixe F associée à la α-décomposition et à n'importe quelle application de point fixe $\hat{F}$ associée à la β-décomposition sont des contractions et, dans chaque cas, on obtient un résultat d'existence et d'unicité du point fixe de F et de $\hat{F}$ ainsi que de la solution du problème (3.33). Alors, quelle que soit l'estimation initiale $\tilde{X}^0$, la convergence des itérations parallèles asynchrones et synchrones et des itérations séquentielles décrites par (3.36)-(3.38), découle soit de [9] ou [51] associé à la propriété de contraction pour la norme vectorielle (3.50) des applications de point fixe ou directement de la propriété de contraction selon la norme définie par (3.114) en appliquant le résultat de [28].

Le résultat de la proposition 2 peut être appliqué au cas où A est une M-matrice et la α-décomposition considérée est la décomposition par points. Par conséquent, dans la suite, nous considérerons l'hypothèse suivante :

$$A \text{ est une M-matrice.} \tag{3.115}$$

Dans ce cas, d'après la définition 1 et la remarque 7, pour la décomposition par points, la matrice A est sa propre M-minorante. Donc, d'une manière directe et similaire à celle utilisée pour prouver la proposition 2, nous pouvons établir le résultat suivant.

Corollaire 1. *Considérons le problème* (3.33)*. Supposons que l'hypothèse* (3.115) *soit vérifiée, i.e. A est une M-matrice. Considérons une quelconque décomposition plus grossière en β blocs tel que $\beta < M$; soit $\hat{F}$ l'application de point fixe associée à cette β-décomposition. Alors, quelle que soit la décomposition par blocs, $\hat{F}$ satisfait l'inégalité en norme vectorielle*

$$|\hat{F}(\hat{\tilde{X}}) - \hat{F}(\hat{\tilde{X}}')| \leq (\hat{\Delta})^{-1} \cdot (\hat{L} + \hat{R}) \cdot |\hat{\tilde{Y}} - \hat{\tilde{Y}}'|,\ \forall \hat{\tilde{Y}}, \hat{\tilde{Y}}' \in \mathcal{E}, \tag{3.116}$$

pour laquelle $\hat{J}$ est une matrice de contraction pour l'application de point fixe $\hat{F}$, où $\hat{J}$ est donnée par

$$\hat{J} = (\hat{\Delta})^{-1}(\hat{L} + \hat{R}), \tag{3.117}$$

et on a

$$|\hat{F}(\hat{\tilde{X}}) - \hat{F}(\hat{\tilde{X}}'))| \leq \hat{J}|\hat{\tilde{Y}} - \hat{\tilde{Y}}'|,\ \forall \hat{\tilde{X}}, \hat{\tilde{X}}' \in \mathbb{R}^M. \tag{3.118}$$

De plus, $\hat{F}$ est aussi contractante dans $\mathcal{E}$ normé par la norme uniforme pondérée

$$\|\hat{\tilde{X}}\|_{\bar{\Theta}} = \max_{l=1,\ldots,M} \left(\frac{|\tilde{x}_l|}{\bar{\theta}_i} \right),\ \bar{\theta} > 0,\ \bar{\theta}_l \in \mathbb{R},\ \forall l = 1, \ldots, M. \tag{3.119}$$

Donc, quelle que soit l'approximation initiale $\tilde{X}^0$, la méthode parallèle asynchrone associée à n'importe quelle décomposition par blocs converge vers $\tilde{X}^\star$, point fixe de $\hat{F}$.

Remarque 15. *Il est à noter que plus la valeur de $\hat{\delta}$ est petite comparée à 1, et plus la méthode itérative converge vite. Par conséquent, on peut considérer que $\hat{\delta}$ nous donne une indication sur la vitesse de convergence du processus itératif.*

Pour illustrer les résultats de la proposition 2 et du corollaire 1, on considère l'exemple suivant.

Exemple 4. Considérons la matrice 3×3 suivante

$$A = \begin{bmatrix} 1 & -\dfrac{1}{1+\varepsilon} & -a \\ -\dfrac{1}{1+\varepsilon} & 1 & 0 \\ 0 & -d & 1 \end{bmatrix},\ \forall a, d, \varepsilon > 0, \tag{3.120}$$

qui est une L-matrice (ou Z-matrice). En utilisant une propriété classique des M-matrices, il est facile de prouver par un simple calcul des déterminants mineurs principaux que A est une M-matrice si, et seulement si

$$\det(A) = 1 - \frac{1}{(1+\varepsilon)^2} - \frac{1}{1+\varepsilon} ad > 0. \tag{3.121}$$

L'inégalité précédente se traduit alors par la condition suivante :

$$\frac{ad}{\varepsilon} \times \frac{(\varepsilon+1)}{(\varepsilon+2)} < 1. \tag{3.122}$$

D'après la remarque 7, si la condition (3.122) est satisfaite, alors la matrice A est sa propre M-minorante pour la décomposition par points et dans l'espace $\mathcal{E}$ normé par la norme euclidienne. On envisage à présent deux décompositions distinctes par blocs.

Première décomposition par blocs. Considérons une décomposition par blocs de la matrice A en 4 blocs telle que

$$A = \begin{pmatrix} A_{1,1}^{(1)} & A_{1,2}^{(1)} \\ A_{2,1}^{(1)} & A_{2,2}^{(1)} \end{pmatrix}, \tag{3.123}$$

avec

$$A_{1,1}^{(1)} = \begin{pmatrix} 1 & -\frac{1}{1+\varepsilon} \\ -\frac{1}{1+\varepsilon} & 1 \end{pmatrix}, \; A_{1,2}^{(1)} = \begin{pmatrix} -a \\ 0 \end{pmatrix}, \; A_{2,1}^{(1)} = \begin{pmatrix} 0 & -d \end{pmatrix} \text{ et } A_{2,2}^{(1)} = 1. \tag{3.124}$$

À cette décomposition par blocs de A correspond une décomposition en sous-vecteurs des vecteurs dans $\mathbb{R}^3$ normé par la norme euclidienne ; soit

$$\forall X \in \mathbb{R}^3,\, X = \{x_1, x_2, x_3\}^t = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}, \tag{3.125}$$

la première décomposition est donnée par

$$\forall X \in \mathbb{R}^3,\, X_1^{(1)} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \; X_2^{(1)} = \begin{pmatrix} x_3 \end{pmatrix}. \tag{3.126}$$

Notons que la décomposition par blocs précédente est une décomposition plus grossière. D'après le principe de démonstration de la proposition 2, en partant de la M-minorante associée à la décomposition par points de A, nous considérons la décomposition suivante de cette M-matrice :

$$A = \hat{D}^{(1)} - (\hat{L}^{(1)} + \hat{R}^{(1)}), \tag{3.127}$$

où $\hat{D}^{(1)}$ est donnée par

$$\hat{D}^{(1)} = \left(\begin{array}{cc|c} 1 & -\frac{1}{1+\varepsilon} & 0 \\ -\frac{1}{1+\varepsilon} & 1 & 0 \\ \hline 0 & 0 & 1 \end{array}\right) \text{ et } \hat{L}^{(1)} + \hat{R}^{(1)} = \left(\begin{array}{cc|c} 0 & 0 & a \\ 0 & 0 & 0 \\ \hline 0 & d & 0 \end{array}\right). \tag{3.128}$$

$\hat{D}^{(1)}$ est une matrice à diagonale dominance stricte avec des coefficients diagonaux strictement positifs et des coefficients hors-diagonaux strictement négatifs ou nuls ; donc $\hat{D}^{(1)}$ est une M-matrice et $(\hat{D}^{(1)})^{-1}$ possède des coefficients non négatifs ; par un calcul direct on trouve

$$(\hat{D}^{(1)})^{-1} = \left(\begin{array}{cc|c} \dfrac{1}{1-\frac{1}{(1+\varepsilon)^2}} & \dfrac{1}{1-\frac{1}{(1+\varepsilon)^2}}\cdot\dfrac{1}{1+\varepsilon} & 0 \\ \dfrac{1}{1-\frac{1}{(1+\varepsilon)^2}}\cdot\dfrac{1}{1+\varepsilon} & \dfrac{1}{1-\frac{1}{(1+\varepsilon)^2}} & 0 \\ \hline 0 & 0 & 1 \end{array}\right), \tag{3.129}$$

et

$$\hat{J}^{(1)} = (\hat{D}^{(1)})^{-1}(\hat{L}^{(1)} + \hat{R}^{(1)}) = \left(\begin{array}{cc|c} 0 & 0 & \dfrac{a}{1-\frac{1}{(1+\varepsilon)^2}} \\ 0 & 0 & \dfrac{a}{1-\frac{1}{(1+\varepsilon)^2}}\cdot\dfrac{1}{1+\varepsilon} \\ \hline 0 & d & 0 \end{array}\right). \tag{3.130}$$

$\hat{J}^{(1)}$ est donc une matrice non négative car $(\hat{D}^{(1)})^{-1}$ et $(\hat{L}^{(1)} + \hat{R}^{(1)})$ sont des matrices non négatives. De plus, un calcul direct des valeurs propres $\mathcal{V}^{(1)}$ de $\hat{J}^{(1)}$ donne

$$\mathcal{V}_1^{(1)} = -\sqrt{\frac{ad}{\varepsilon}\cdot\frac{\varepsilon+1}{\varepsilon+2}},\ \mathcal{V}_2^{(1)} = 0 \text{ et } \mathcal{V}_3^{(1)} = \sqrt{\frac{ad}{\varepsilon}\cdot\frac{\varepsilon+1}{\varepsilon+2}}. \tag{3.131}$$

En prenant en compte la condition (3.122), il vient

$$|\mathcal{V}_i^{(1)}| < 1 \text{ pour } i = 1, 2, 3, \tag{3.132}$$

et par conséquent, la matrice $\hat{J}^{(1)}$ associée à cette première décomposition plus grossière est une matrice de contraction.

Deuxième décomposition par blocs. On peut aussi considérer une autre décomposition par blocs de la matrice A :

$$A = \begin{pmatrix} A_{1,1}^{(2)} & A_{1,2}^{(2)} \\ A_{2,1}^{(2)} & A_{2,2}^{(2)} \end{pmatrix}, \tag{3.133}$$

avec

$$A_{1,1}^{(2)} = 1,\ A_{1,2}^{(2)} = \begin{pmatrix} -\dfrac{1}{1+\varepsilon} & -a \end{pmatrix},\ A_{2,1}^{(2)} = \begin{pmatrix} -\dfrac{1}{1+\varepsilon} \\ 0 \end{pmatrix} \text{ et } A_{2,2}^{(2)} = \begin{pmatrix} 1 & 0 \\ -d & 1 \end{pmatrix}. \tag{3.134}$$

À cette deuxième décomposition par blocs de A correspond une décomposition en sous-vecteurs des vecteurs $X \in \mathbb{R}^3$ donnée par

$$\forall X \in \mathbb{R}^3,\ X_1^{(2)} = \begin{pmatrix} x_1 \end{pmatrix},\ X_2^{(2)} = \begin{pmatrix} x_2 \\ x_3 \end{pmatrix}. \tag{3.135}$$

De la même manière que pour la première décomposition par blocs, on considère la décomposition suivante de la M-minorante :

$$A = \hat{D}^{(2)} - (\hat{L}^{(2)} + \hat{R}^{(2)}), \tag{3.136}$$

où $\hat{D}^{(2)}$ est donnée par

$$\hat{D}^{(2)} = \left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & -d & 1 \end{array}\right) \text{ et } \hat{L}^{(2)} + \hat{R}^{(2)} = \left(\begin{array}{c|cc} 0 & \frac{1}{1+\varepsilon} & a \\ \hline \frac{1}{1+\varepsilon} & 0 & 0 \\ 0 & 0 & 0 \end{array}\right). \tag{3.137}$$

Alors

$$(\hat{D}^{(2)})^{-1} = \left(\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & d & 1 \end{array}\right) \text{ et } \hat{J}^{(2)} = \left(\begin{array}{c|cc} 0 & \frac{1}{1+\varepsilon} & a \\ \hline \frac{1}{1+\varepsilon} & 0 & 0 \\ \frac{1}{1+\varepsilon} & 0 & 0 \end{array}\right), \tag{3.138}$$

où $\hat{J}^{(2)}$ est définie par $\hat{J}^{(2)} = (\hat{D}^{(2)})^{-1}(\hat{L}^{(2)} + \hat{R}^{(2)})$. Un calcul direct des valeurs propres $\mathcal{V}^{(2)}$ de $\hat{J}^{(2)}$ donne

$$\mathcal{V}_1^{(2)} = -\sqrt{\frac{ad}{\varepsilon+1} + \frac{1}{(\varepsilon+1)^2}},\ \mathcal{V}_2^{(2)} = 0 \text{ et } \mathcal{V}_3^{(2)} = \sqrt{\frac{ad}{\varepsilon+1} + \frac{1}{(\varepsilon+1)^2}}. \tag{3.139}$$

Ainsi, grace à la condition (3.121), on a aussi

$$|\mathcal{V}_i^{(2)}| < 1 \text{ pour } i = 1, 2, 3, \tag{3.140}$$

et par conséquent, la matrice $\hat{J}^{(2)}$ associée à cette deuxième décomposition est une matrice de contraction.

En conclusion, nous avons vérifié directement sur cet exemple que si une M-minorante existe pour la décomposition par points, pour toute décomposition plus grossière, les matrices $\hat{J}^{(k)}$, $k = 1, 2$ sont des matrices de contraction.

3.3.3 Analyse selon un ordre partiel

Pour analyser la convergence, on peut aussi considérer des techniques d'ordre partiel associées au principe du maximum discret correspondant au fait que l'inverse de l'opérateur considéré $\mathcal{A}$ est inverse monotone ; c'est-à-dire que $\mathcal{A}(\tilde{X}) \leq \mathcal{A}(\tilde{Y})$ entraîne que $\tilde{X} \leq \tilde{Y}$. Notons que pour une M-matrice inversible, cette propriété est bien vérifiée puisque son inverse est non négative.

3.3.3.1 Notations et définitions

Pour $i \in \{1, \ldots, \alpha\}$, chaque sous-espace $\mathcal{E}_i$ est à présent muni de l'ordre partiel naturel (i.e. composante par composante) noté $\tilde{X}_i \underset{i}{\leq} \tilde{Y}_i$ pour $\tilde{X}_i, \tilde{Y}_i \in \mathcal{E}_i$. Soit $\leq$ l'ordre partiel considéré pour $\mathcal{E} = \prod_{i=1}^{\alpha} \mathcal{E}_i$ défini par

$$\begin{cases} \text{Soit } \tilde{X} \text{ et } \tilde{Y} \text{ deux vecteurs de } \mathcal{E} \\ \text{alors } \tilde{X} \leq \tilde{Y} \text{ si } \tilde{X}_i \underset{i}{\leq} \tilde{Y}_i, \, \forall i \in \{1, \ldots, \alpha\}. \end{cases} \tag{3.141}$$

De plus, la notion d'intervalle d'ordre, défini comme suit, sera utilisée.

Définition 5. Soit $\tilde{X}_i, \tilde{Y}_i \in \mathcal{E}_i$ tel que $\tilde{Y}_i \leq \tilde{X}_i$; alors $\langle \tilde{Y}_i, \tilde{X}_i \rangle = \{\tilde{Z}_i \in \mathcal{E}_i \mid \tilde{Y}_i \leq \tilde{Z}_i \leq \tilde{X}_i\}$.

Cette défintion est similaire pour l'ensemble $\mathcal{E}$; $\langle \tilde{Y}, \tilde{X} \rangle$, $\tilde{X}, \tilde{Y} \in \mathcal{E}$.

Soit $\mathcal{A}$ une application continue de $\mathbb{R}^M$ dans lui-même. Rappelons les définitions suivantes (cf. [61] et [70]).

Définition 6. L'application $\mathcal{A}$ est une M-fonction si $\mathcal{A}$ est inverse monotone, i.e.

$$\mathcal{A}(\tilde{X}) \leq \mathcal{A}(\tilde{Y}) \text{ implique } \tilde{X} \leq \tilde{Y}, \, \forall \tilde{X}, \tilde{Y} \in \mathbb{R}^M, \tag{3.142}$$

et, si de plus, $\mathcal{A}$ est hors-diagonale décroissante (antitone), i.e. pour tout $\tilde{X} \in \mathbb{R}^M$, les fonctions définies par

$$\begin{cases} \mathcal{A}_{lk} : \{\tau \in \mathbb{R} \mid \tilde{X} + \tau e_k \in \mathbb{R}^M\} \mapsto \mathbb{R}, \\ \mathcal{A}_{lk}(\tau) = \mathcal{A}_l(\tilde{X} + \tau e_k), \, l \neq k, \, l, k = 1, \ldots, M, \end{cases} \tag{3.143}$$

sont décroissantes, où $e_k \in \mathbb{R}^M$, $k = 1, \ldots, M$, sont les vecteurs unitaires de la base canonique. Pour des hypothèses et des propriétés supplémentaires sur les M-fonctions, le lecteur est renvoyé à [61] et [54].

Lemme 3. *Si $\mathcal{A}$ est une M-matrice inversible $A = (a_{l,k})$ alors $\mathcal{A}$ est une M-fonction linéaire.*

Démonstration. En effet, dans ce cas, pour $l \neq k$, en considérant une décomposition par points, on a $\mathcal{A}_l(\tilde{X} + \tau e_k) = \sum_{j=1}^{\alpha} a_{l,j} \tilde{X}_j + a_{l,k}\tau$; supposons que $\tau_2 \leq \tau_1$, alors

$$\mathcal{A}_l(\tilde{X} + \tau_2 e_k) = \sum_{j=1}^{\alpha} a_{l,j} \tilde{X}_j + a_{l,k}\tau_2 \geq \sum_{j=1}^{\alpha} a_{l,j} \tilde{X}_j + a_{l,k}\tau_1 = \mathcal{A}_l(\tilde{X} + \tau_1 e_k), \tag{3.144}$$

et $\mathcal{A}$ est hors-diagonale décroissante ; de plus, A étant une M-matrice inversible, alors A^{-1} est une matrice non négative donc $A^{-1}\tilde{X} \leq A^{-1}\tilde{Y}$, implique $\tilde{X} \leq \tilde{Y}$, donc que $\mathcal{A}$ est inverse monotone, ce qui établit la démonstration. □

Remarque 16. *Notons qu'un tel résultat découle aussi du fait que l'application $t \mapsto \mathcal{A}_l(\tilde{X} + te_k)$ admet une dérivée négative par rapport à τ est égale à $a_{l,k}$.*

Remarque 17. *Plus généralement, une fonction affine $A\tilde{X} + B$ est une M-fonction linéaire si, et seulement si A est une M-matrice ; voir le lemme* 2.9 *à la page* 278 *dans [70] ou dans [61] à la page* 468.

Supposons aussi que

$$\mathcal{A} \text{ est une M-fonction surjective.} \tag{3.145}$$

Soit $\mathcal{A}$ une M-fonction surjective et considérons la solution du système d'équations suivant :

$$\mathcal{A}(\tilde{X}) = 0. \tag{3.146}$$

D'après les hypothèses précédentes, le problème (3.146) admet une solution unique $\tilde{X}^*$ (cf. [61] et [70]). Compte tenu de la décomposition par blocs de $\tilde{X}$, on considère la décomposition par blocs de $\mathcal{A}$ correspondante

$$\mathcal{A}(\tilde{X}) = \{\mathcal{A}_1(\tilde{X}), \ldots, \mathcal{A}_i(\tilde{X}), \ldots, \mathcal{A}_\alpha(\tilde{X})\} \in \prod_{i=1}^{\alpha} \mathcal{E}_i. \tag{3.147}$$

Pour tout $i \in \{1, \ldots, \alpha\}$, on introduit l'application suivante de $\mathcal{E}_i$ dans lui-même

$$\tilde{X}_i \mapsto \mathcal{A}_i(\tilde{X}_i\,;\,\tilde{Y}) = \mathcal{A}_i(\tilde{Y}_1, \ldots, \tilde{Y}_{i-1}, \tilde{X}_i, \tilde{Y}_{i+1}, \ldots, \tilde{Y}_\alpha). \tag{3.148}$$

Puisque $\mathcal{A}$ est une M-fonction continue et surjective, il résulte du théorème 3.5 de [70] que pour tout $i \in \{1, \ldots, \alpha\}$, et tout $\tilde{Y} \in \mathcal{E}$, l'application $\tilde{X}_i \mapsto \mathcal{A}_i(\tilde{X}_i\,;\,\tilde{Y})$ est une M-fonction continue et surjective. De plus, pour tout $i \in \{1, \ldots, \alpha\}$ et tout $\tilde{Y} \in \mathcal{E}$, le système,

$$\mathcal{A}_i(\tilde{X}_i\,;\,\tilde{Y}) = 0, \tag{3.149}$$

admet une solution unique. Par conséquent, on peut définir une application de point fixe $F : \mathcal{E} \mapsto \mathcal{E}$, associée au problème (3.146) tel que

$$F(\tilde{Y}) = \tilde{X} = \{\tilde{X}_1, \ldots, \tilde{X}_i, \ldots, \tilde{X}_\alpha\}\,; \tag{3.150}$$

l'application F est bien définie et croissante monotone sur $\mathcal{E}$, i.e. pour tout $\tilde{X}, \tilde{Y} \in \mathcal{E}$ tel que $\tilde{X} \leq \tilde{Y}$, $F(\tilde{X}) \leq F(\tilde{Y})$ (cf. [53]).

Pour résoudre le problème (3.146), nous considérons une méthode itérative générale de point fixe dans laquelle l'approximation initiale des itérations de point fixe satisfait le concept suivant.

Définition 7. Un vecteur $\tilde{Y} \in \mathbb{R}$ est une sur-solution $\mathcal{A}$ (sous-solution $\mathcal{A}$ respectivement) si $\mathcal{A}(\tilde{Y}) \geq 0$ ($\mathcal{A}(\tilde{Y}) \leq 0$ respectivement).

3.3.3.2 Algorithmes parallèles asynchrones et convergence monotone

Pour l'analyse en ordre partiel des itérations parallèles asynchrones, on modifie légèrement l'algorithme (3.36) dans la mesure où la donnée initiale est soit une sur-solution soit une sous-solution ; soit $\tilde{X}^0 \in \mathcal{E}$ une sur-solution de $\mathcal{A}$, i.e. $\mathcal{A}(\tilde{X}^0) \geq 0$, ou une sous-solution de $\mathcal{A}$, i.e. $\mathcal{A}(\tilde{X}^0) \leq 0$. On a donc l'algorithme suivant :

$$\tilde{X}_l^{(r+1)} = \begin{cases} F_l\left(\tilde{X}_1^{(\kappa_1(r))}, \ldots, \tilde{X}_k^{(\kappa_k(r))}, \ldots, \tilde{X}_\alpha^{(\kappa_\alpha(r))}\right) \text{ si } l \in s(r), \\ \tilde{X}_l^r \text{ si } l \notin s(r), \end{cases} \tag{3.151}$$

où $s(r)$ et $\kappa_k(r)$ sont définis comme en (3.37) et (3.38) respectivement.

On considère maintenant les itérations de point fixe parallèles asynchrones $\{\tilde{X}^r\}_{r\in\mathbb{N}}$ définies par (3.36)-(3.38). On rappelle un résultat important (cf. [54]).

Proposition 3. *Soit $\mathcal{A}$ une M-fonction continue surjective, F l'application de point fixe associée à $\mathcal{A}$ définie par* (3.149) *et* (3.150)*, $\tilde{X}^0 \in \mathcal{E}$ une sur-solution de $\mathcal{A}$ ($\tilde{Y}^0 \in \mathcal{E}$ une sous-solution de $\mathcal{A}$ respectivement). Alors, l'itération asynchrone $\{\tilde{X}^r\}_{r\in\mathbb{N}}$ ($\{\tilde{Y}^r\}_{r\in\mathbb{N}}$ respectivement) donnée par* (3.151) *(avec* (3.37)-(3.38) *évidemment vérifiées) est bien définie et satisfait*

$$\tilde{X}^r \downarrow \tilde{X}^\star,\ r \to \infty, \tag{3.152}$$

et

$$\tilde{Y}^r \uparrow \tilde{X}^\star,\ r \to \infty, \tag{3.153}$$

respectivement, où (3.152) *signifie que* $\lim_{r\to\infty} \tilde{X}^r = \tilde{X}^\star$ *et* $\tilde{X}^\star \leq \ldots \leq \tilde{X}^{(r+1)} \leq \tilde{X}^r \leq \ldots \leq \tilde{X}^0$*, et* (3.153) *signifie que* $\lim_{r\to\infty} \tilde{Y}^r = \tilde{X}^\star$ *et* $\tilde{X}^\star \geq \ldots \geq \tilde{Y}^{(r+1)} \geq \tilde{Y}^r \geq \ldots \geq \tilde{Y}^0$*.*

L'arrêt des itérations parallèles asynchrones est un problème difficile à implémenter. Cependant, le résultat de la proposition 3 permet d'obtenir un critère d'arrêt adapté, quand on considère deux suites de vecteurs itérés associées à une sur-solution de $\mathcal{A}$ et à une sous-solution de $\mathcal{A}$ respectivement.

Corollaire 2. *Si on considère les itérations de point fixe parallèles asynchrones* (3.151) *avec, d'une part, l'estimation $\tilde{X}^0$, une sur-solution de $\mathcal{A}$ et, d'autre part, $\tilde{Y}^0$, une sous-solution de $\mathcal{A}$, alors on a*

$$\tilde{Y}^0 \leq \tilde{Y}^1 \leq \ldots \leq \tilde{Y}^r \leq \ldots \leq \tilde{X}^\star \leq \ldots \leq \tilde{X}^q \leq \ldots \leq \tilde{X}^1 \leq \tilde{X}^0, \tag{3.154}$$

et, si ε est la tolérance considérée pour l'arrêt du processus itératif, alors on peut prendre le critère d'arrêt suivant :

$$\tilde{X}^q - \tilde{Y}^r \leq \varepsilon,\ \forall q, r \in \mathbb{N}. \tag{3.155}$$

Remarque 18. *Le critère d'arrêt* (3.155) *est certes coûteux en quantité de calcul mais cependant fiable. Par ailleurs, notons que la terminaison des algorithmes parallèles asynchrones est un problème difficile.*

3.3.3.3 Détermination de la sur-solution et de la sous-solution de $\mathcal{A}$

On considère le problème discrétisé (3.33) où la matrice de décomposition par blocs A vérifie (3.115) ; on considère la décomposition par blocs du problème (3.33)

$$\sum_{k=1}^{\alpha} A_{l,k}\tilde{X}_k = B_l,\ \forall l \in \{1,\ldots,\alpha\}, \tag{3.156}$$

on considère aussi l'application $\tilde{X} \mapsto \mathcal{A}(\tilde{X})$ définie par

$$\mathcal{A}(\tilde{X}) = A\tilde{X} - B, \tag{3.157}$$

où $\mathcal{A}$ est une M-fonction. Soit $\mathcal{A}_i(\tilde{X}_i\,;\,\tilde{Y})$ définie par

$$\mathcal{A}_i(\tilde{X}_i\,;\,\tilde{Y}) = A_{i,i}\tilde{X}_i + \sum_{j\neq i} A_{i,j}\tilde{Y}_j - B_i. \tag{3.158}$$

On peut définir implicitement une application de point fixe F de $\mathbb{R}^M$ dans $\mathbb{R}^M$ associée au sous-problème (3.156). Soit, $\tilde{Z}$, $\tilde{X}^0$ et $\tilde{Y}^0$ trois vecteurs de $\mathbb{R}^M$ tels que

$$\mathcal{A}_i(\tilde{Z}_i\,;\,\tilde{Z}) \geq 0,\ \tilde{X}_i^0 = \tilde{Z}_i \text{ et } \tilde{Y}_i^0 = F_i(\tilde{Z}),\ \forall i \in \{1,\ldots,\alpha\}. \tag{3.159}$$

Pour $i \in \{1,\ldots,\alpha\}$, soit $\theta_i = \min(\mathcal{A}_i(\tilde{Z}_i\,;\,\tilde{Z}), 0)$ et $\gamma_i = \max(\mathcal{A}_i(\tilde{Z}_i\,;\,\tilde{Z}), 0)$. Notons que si $\theta_i = 0$, alors $\gamma_i = \mathcal{A}_i(\tilde{Z}_i\,;\,\tilde{Z})$; alors, soit $\tilde{Y}^0 = \mathcal{A}^{-1}(\theta)$ tel que $\tilde{X}^0 = \mathcal{A}^{-1}(\gamma)$ avec $\mathcal{A}(\tilde{Y}^0) = 0$ et $\mathcal{A}(\tilde{X}^0) = \mathcal{A}(\tilde{Z})$. Inversement, si $\theta_i = \mathcal{A}_i(\tilde{Z}_i\,;\,\tilde{Z})$, alors $\gamma_i = 0$ et, de la même façon $\tilde{Y}^0 = \mathcal{A}(\tilde{Z})$ et $\tilde{X}^0 = 0$. Alors, on a

$$\mathcal{A}(\tilde{Y}^0) \leq \mathcal{A}(\tilde{X}^\star) = 0 \leq \mathcal{A}(\tilde{X}^0), \tag{3.160}$$

et, d'après les hypothèses précédentes, l'application $\mathcal{A}_i$ est une M-fonction sur l'intervalle d'ordre $\langle \tilde{Y}_i^0, \tilde{X}_i^0 \rangle_i$. Alors, en partant de n'importe quel vecteur $\tilde{Z}$, on obtient une sur-solution et une sous-solution de $\mathcal{A}$.

3.3.4 Multi-splitting

3.3.4.1 Multi-splitting et contraction

Afin d'obtenir une bonne précision lors des simulations d'interactions fluide-structure, il est nécessaire d'utiliser un maillage très fin ; la discrétisation des problèmes modélisant les équations fluide-structure conduit alors à résoudre de très grands systèmes algébriques. De plus, dans les applications industrielles, les matrices de discrétisation ne sont pas structurées et le découpage en grands blocs est difficile (cf. figure 3.3). Dans cette sous-section, nous présentons le principe des méthodes de multi-splitting qui permet de donner une présentation unifiée de la plupart des méthodes de sous-domaines et par conséquent, permet d'analyser le comportement des toutes ces méthodes d'une façon unique. Rappelons que l'introduction de ces méthodes permet également de s'affranchir, du moins théoriquement, d'une décomposition en sous-domaines.

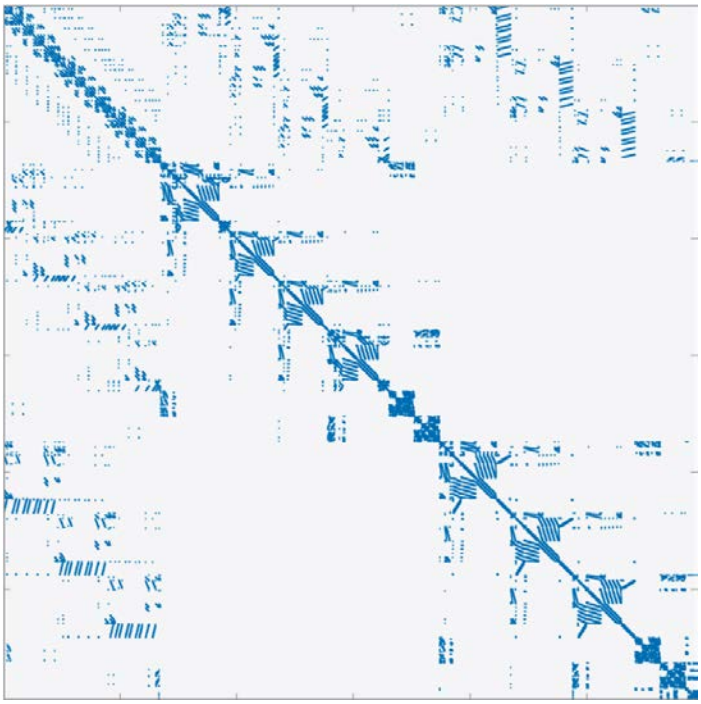


FIGURE 3.3 – Exemple de matrice creuse issue d'une discrétisation

On considère maintenant la solution du problème (3.33) lorsque A satisfait l'hypothèse (3.115) et, dans la suite, par souci de généralité, nous considérerons uniquement la décomposition par points. Pour $m \in \mathbb{N}$, soit la famille de décomposition de la matrice A suivante :

$$A = M^l - N^l,\ l = 1, \ldots, m, \tag{3.161}$$

où $(M^l)^{-1} \geq 0$ et $N^l \geq 0$; soit $\hat{F}^l : \mathcal{D}(\hat{F}^l) \mapsto \mathcal{D}(\hat{F}^l)$, $\mathcal{D}(\hat{F}^l) \subset \mathcal{E}$, $l = 1, \ldots, m$, m applications de point fixe associées au problème (3.33) et définies par

$$\begin{cases} \hat{F}^l(\tilde{Y}) = \tilde{X},\ l = 1, \ldots, m,\ \tilde{Y} \in \mathcal{E},\ \tilde{X} \in \mathcal{E} \\ \text{tel que } M^l \tilde{X} = N^l \tilde{Y} + \tilde{B}. \end{cases} \tag{3.162}$$

Supposons que l'espace $\mathcal{E}$ soit normé par une norme pondérée uniforme, définie d'une façon similaire à (3.119) par

$$\|\tilde{Y}\|_{\bar{\Theta}} = \max_{i=1,\ldots,M} \left(\frac{|\tilde{y}_i|}{\bar{\theta}_i} \right),\ \bar{\theta}_i > 0,\ \bar{\theta}_i \in \mathbb{R}, \tag{3.163}$$

où $|\tilde{y}_i|$ représente la valeur absolue de $\tilde{y}_i$, $\tilde{y}_i \in \mathbb{R}$; supposons que pour tout $l = 1, \ldots, m$, $\hat{F}^l$ est contractante pour $\tilde{X}^*$, son point fixe, tel que l'inégalité suivante soit vérifiée :

$$\|\hat{F}^l(\tilde{Y}) - \tilde{X}^*\|_{\bar{\Theta}_l} \leq \hat{\delta}_l \cdot \|\tilde{Y} - \tilde{X}^*\|_{\bar{\Theta}_l},\ l = 1, \ldots, m, \tag{3.164}$$

où $0 \leq \hat{\delta}_l < 1$.

Un multi-splitting formel associé au problème (3.33) est défini par la famille de problèmes de point fixe (cf. [5])

$$\tilde{X}^* = \hat{F}^l(\tilde{X}^*),\ l = 1, \ldots, m,\ \tilde{X}^* \in \mathcal{E}. \tag{3.165}$$

Soit $E = \mathcal{E}^m$ et considérons maintenant la décomposition par blocs suivante de E :

$$E = \prod_{l=1}^{m} E_l, \tag{3.166}$$

où $E_l = \mathcal{E}$. Soit $X \in E$, défini par

$$X = (\tilde{X}^1, \ldots, \tilde{X}^l, \ldots, \tilde{X}^m) \in \prod_{l=1}^{m} E_l, \tag{3.167}$$

où $\tilde{X}^1, \ldots, \tilde{X}^l, \ldots, \tilde{X}^m$ représentent m vecteurs de $\mathcal{E}$.

Définition 8. L'application de point fixe étendue $T : E \mapsto E$ associée au multi-splitting formel est donnée par

$$\begin{cases} T(Y) = X, \text{ tel que } \tilde{X}^l = \hat{F}^l(\tilde{Z}^l) \\ \text{avec } \tilde{Z}^l = \displaystyle\sum_{k=1}^{m} W_{lk}\tilde{Y}^k,\ l = 1, \ldots, m, \end{cases} \tag{3.168}$$

où les W_{lk} sont des matrices de pondération diagonales non négatives satisfaisant

$$\sum_{k=1}^{m} W_{lk} = \mathbb{I}_l,\ \forall l \in \{1, \ldots, m\}, \tag{3.169}$$

$\mathbb{I}_l$ étant la matrice identité dans $\mathcal{L}(E_l)$.

Puisque $\hat{F}^l(\mathcal{D}(\hat{F}^l)) \subset \mathcal{D}(\hat{F}^l)$, alors $T(\mathcal{U}(T)) \subset \mathcal{U}(T)$ où $\mathcal{U}(T) = \prod_{l=1}^{m} \mathcal{D}(\hat{F}^l)$.

Notons que des économies de calcul considérables peuvent être possibles car une composante de $\tilde{Y}^k$ n'a pas besoin d'être calculée si le coefficient diagonal correspondant des matrices de pondération est nul ; le rôle de ces matrices de pondération peut être vu comme la détermination de la distribution de la charge de calcul sur chaque processeur. La figure 3.4 représente une décomposition quelconque de la matrice A.

Soit la décomposition par blocs de l'application T suivante

$$T(X) = \{T^1(X), \ldots, T^l(X), \ldots, T^m(X)\} \in \prod_{l=1}^{m} E_l. \tag{3.170}$$

Notons que pour un choix particulier des matrices de pondération W_{lk}, on peut obtenir des méthodes itératives différentes et en particulier une méthode de sous-domaine sans recouvrement ou la méthode alternée de Schwarz classique (cf. [5]). En effet, d'après [5], la méthode de Jacobi par blocs correspond au choix de M^l suivant :

$$M^l = diag(\mathbb{I}_1, \ldots, \mathbb{I}_{l-1}, A_{l,l}, \mathbb{I}_{l+1}, \ldots, \mathbb{I}_m), \tag{3.171}$$

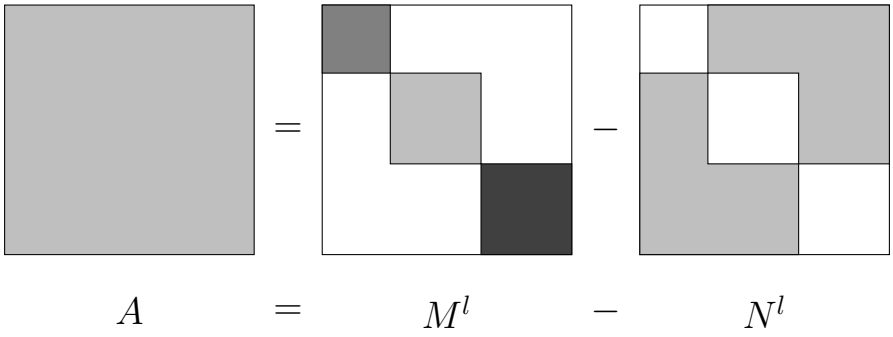


FIGURE 3.4 – Décomposition de la matrice A

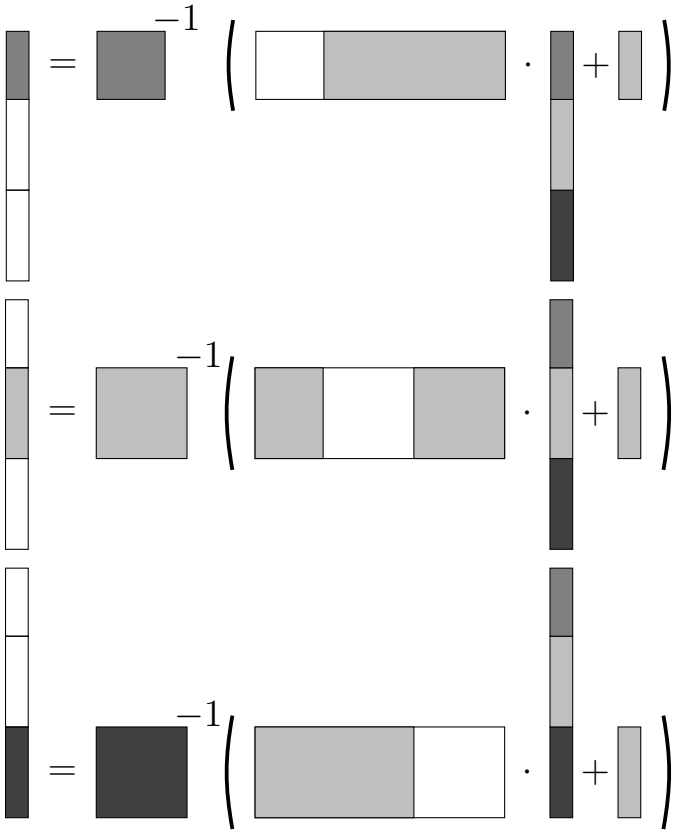


FIGURE 3.5 – Décomposition pour la méthode de Jacobi par blocs

et au choix de $W_{lk} \equiv \bar{W}_l$ donné par

$$\bar{W}_l = diag(0, \ldots, 0, \mathbb{I}_l, 0, \ldots, 0), \tag{3.172}$$

d'après la représentation de la figure 3.5.

Notons que pour la méthode de Jacobi par blocs en parallèle, il n'y a pas de recou-

vrement et les matrices diagonales de pondération ne peuvent avoir que des coefficients égaux à zéro ou à un.

Pour la méthode alternée de Schwarz, il y a recouvrement et, par conséquent, certains coefficients des matrices diagonales de pondération ont des valeurs strictement comprises entre zéro et un comme illustré sur la figure 3.6.

Pour la méthode de multi-splitting d'O'Leary et White [60], $W_{lk} = \bar{W}_k$ satisfait

$$\sum_{k=1}^{m} \bar{W}_k = \mathbb{I} \text{ and } (\bar{W}_k)_{j,j} = 0, \forall j \notin \mathcal{J}_k, \tag{3.173}$$

où, pour $k \in \{1, \ldots, m\}$, $\mathcal{J}_k$ est un sous-ensemble de $\{1, \ldots, M\}$ et $\alpha = m$, où α est le nombre de blocs. Cependant, ici, chaque processeur gère l'une des m décompositions ; dans ce cas, les matrices diagonales de pondération $\bar{W}_k$ n'ont que des coefficients égaux à zéro ou un, ce qui correspond au fait que les sous-vecteurs à calculer sont des sous-vecteurs indépendants et la méthode de multi-splitting n'utilise pas de recouvrement (cf. [5]).

Rappelons maintenant un résultat de [5].

Proposition 4. *Notons $X^* = (\tilde{X}^*, \ldots, \tilde{X}^*)$ où $\tilde{X}^*$ est la solution du problème* (3.33) *; alors, si l'hypothèse* (3.164) *est vérifiée, l'application de point fixe étendue T est $\| \ . \ \|_{\bar{\Theta}}$ contractante en X^*, où $\| \ . \ \|_{\bar{\Theta}_l}$ est définie par*

$$\|X\|_{\bar{\Theta}} = \max_{1 \le l \le m} \left(\max_{i=1,\ldots,M} \left(\frac{|\tilde{x}_i|}{(\bar{\Theta}_l)_i} \right) \right) \ ; \tag{3.174}$$

la constante de contraction associée étant

$$\hat{\delta} = \max_{1 \le l \le m} (\hat{\delta}_l) < 1. \tag{3.175}$$

En combinant les résultats du corollaire 1 et de la proposition 4, on obtient le résultat suivant.

Corollaire 3. *Considérons la solution au problème* (3.33)*, si l'hypothèse* (3.115) *est vérifiée, i.e. la matrice A est une M-matrice, alors, quelle que soit la décomposition du problème, le multi-splitting formel est contractant pour X^* et, n'importe quelle méthode de multi-splitting séquentielle, parallèle synchrone ou asynchrone partant de $\tilde{X}^0 \in \mathcal{U}$ converge vers la solution de* (3.33)*.*

Démonstration. En effet, si l'hypothèse (3.115) est vérifiée, d'après le corollaire 1, chaque application de point fixe $\hat{F}^l$ est contractante pour $l = 1, \ldots, m$; donc les hypothèses de la proposition 4 sont vérifiées. □

Remarque 19. *En fait, d'un point de vue de l'implémentation des algorithmes, dans les méthodes de multi-splitting, on peut obtenir deux niveaux de parallélisme ; en effet,*

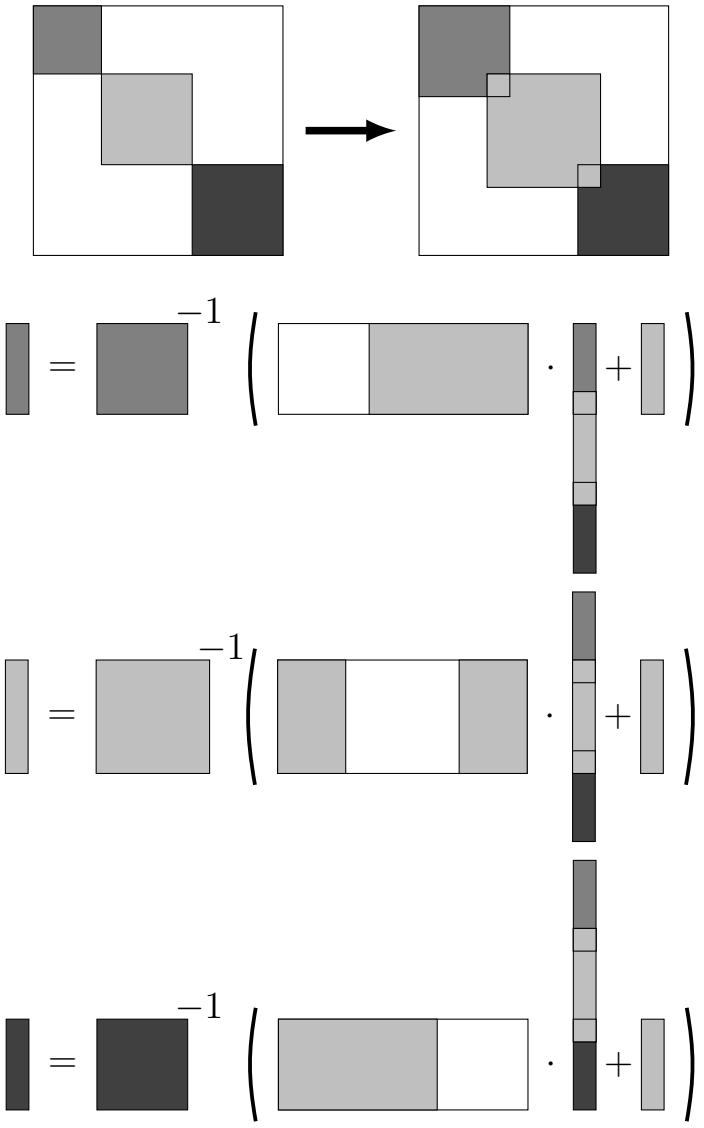


FIGURE 3.6 – Décomposition de la matrice A avec recouvrement

– *le premier niveau de parallélisme peut être obtenu en attribuant l'une des m décomposition* (3.161) *à un cluster tel que chacune des m décompositions soit traitée indépendamment des autres ;*

– *le second niveau de parallélisme correspond au fait que chaque ensemble d'équations associé à chaque décomposition en β blocs est résolu par un algorithme parallèle asynchrone identique à celui présenté précédemment dans* (3.36)-(3.38).

3.3.4.2 Multi-splitting et ordre partiel

L'application de point fixe étendue T est associée au problème étendu suivant :

$$a(X^*) = 0, \tag{3.176}$$

où l'application $a : \mathcal{E} \mapsto \mathcal{E}$ est donnée par

$$a(X) = A^e X - B^e, \tag{3.177}$$

et

$$A^e_l \tilde{X} = M^l \tilde{X}_l - N^l \sum_{k=1}^{m} W_{l,k} \tilde{Y}_k, \; \forall l \in \{1, \ldots, m\}. \tag{3.178}$$

Dans la suite, $a_l(\tilde{Y}_1, \ldots, \tilde{Y}_{l-1}, \tilde{X}_l, \tilde{Y}_{l+1}, \ldots, \tilde{Y}_m)$ sera aussi noté $a_l(\tilde{X}_l\,;\,\tilde{Y})$.

Proposition 5. *Considérons vérifiée l'hypothèse* (3.115). *L'application $X \mapsto a(X)$ est une M-fonction continue surjective.*

Démonstration. En effet, conformément à la remarque 17, on peut vérifier que $a(X)$ est hors-diagonale décroissante ; de plus, en appliquant le théorème 3.4 de [70] page 286, puisque $a(X)$ est évidemment continue et hors-diagonale décroissante, alors $a(X)$ est une M-fonction surjective. □

Alors il résulte de la proposition 5 que l'on est dans cadre théorique de l'étude développée précédemment au paragraphe 3.3.3. Par conséquent, on peut déduire la convergence monotone des itérations asynchrones. Il résulte de la proposition 5 que

$$\left\{ \begin{array}{l} \text{Pour tout } l \in \{1, \ldots, m\} \text{ et tout } X \in E, \text{ l'application } \tilde{X}_l \mapsto a_l(\tilde{X}_l\,;\,\tilde{Y}) \\ \text{est une M-fonction continue surjective de } E \text{ dans } E_l\,; \end{array} \right. \tag{3.179}$$

pour plus de détails, le lecteur est renvoyé au théorème 3.5 de [70]. De plus il résulte aussi de la proposition 5 que

$$\left\{ \begin{array}{l} \text{Pour tout } l \in \{1, \ldots, m\} \text{ et tout } X \in E, \text{ le problème } a_l(\tilde{X}_l\,;\,\tilde{Y}) = 0 \\ \text{possède une solution unique } \tilde{X}_l. \end{array} \right. \tag{3.180}$$

Des hypothèses précédentes, on déduit aussi que T est croissante sur E (cf. [53]). Alors, la méthode de multi-splitting parallèle asynchrone initialisée par une sur-solution ou une sous-solution de $\mathcal{A}$ converge de façon monotone vers la solution du problème discrétisé (3.33).

Remarque 20. *Pour les méthodes parallèles asynchrones alternée de Schwarz et de multi-splitting avec communications flexibles où on ne considère plus des itérés retardés comme dans* (3.36)-(3.38)*, mais où on considère les valeurs disponibles des interactions calculées par les autres processeurs, le lecteur est renvoyé à [79], qui correspond à un modèle plus général d'itérations parallèles asynchrones. En fait, la méthode alternée de Schwarz est une classe particulière de méthode de sous-domaine. Plus généralement, on peut considérer la méthode de multi-splitting parallèle synchrone ou asynchrone correspondant à une méthode itérative plus générale. La convergence de cette méthode de multi-splitting parallèle asynchrone avec communications flexibles peut être aussi étudiée par des techniques d'ordre partiel [79]. Néanmoins, l'implémentation de telles méthodes itératives parallèles peut être délicate du fait qu'il peut être très compliqué de minimiser pratiquement le poids des communications.*

3.3.5 Extension aux problèmes pseudo-linéaires

L'analyse et les résultats précédents sont toujours valables quand le système algébrique linéaire est perturbé par un opérateur non-linéaire diagonal continu et croissant $\tilde{X} \mapsto \Phi(\tilde{X})$:

$$A\tilde{X} + \Phi(\tilde{X}) = \tilde{B}, \tag{3.181}$$

ou encore par un opérateur multivoque monotone diagonal prenant en compte des contraintes sur la solution

$$A\tilde{X} + \partial\Gamma(\tilde{X}) - \tilde{B} \ni 0, \tag{3.182}$$

où $\partial\Gamma(\tilde{X})$ est le sous-différentiel de la fonction indicatrice de l'ensemble convexe définissant les contraintes sur la solution, dont on rappelle les définitions ci après. La fonction indicatrice d'un ensemble convexe [8] K est donnée par

$$\Gamma(\mathbf{x}) = \begin{cases} 0 & \text{si } \mathbf{x} \in K, \\ +\infty & \text{sinon.} \end{cases} \tag{3.183}$$

Classiquement, compte tenu de la convexité de la fonction indicatrice, son sous-différentiel est donné par

$$\partial\Gamma(\mathbf{x}) = \{\mathbf{x}^* \in E^* \mid < \mathbf{x} - \mathbf{y}, \mathbf{x}^* > \geq 0, \forall \mathbf{y} \in K\}, \tag{3.184}$$

où $\mathbf{D}(\partial\Gamma) = \mathbf{D}(\Gamma) = K$ et $\partial\Gamma_i(\mathbf{x}) = \{0\}$ si $\mathbf{x} \in \mathring{K}$. De plus, si $\mathbf{x}$ est situé à la frontière de K, alors $\partial\Gamma(\mathbf{x})$ coïncide avec le cône des normales à K au point $\mathbf{x}$ [8].

En effet, puisque le système linéaire est perturbé par un opérateur diagonal monotone maximal dans chacun des cas précédents, alors, dans cette situation non-linéaire, les inégalités comme (3.42) sont toujours valables pour la partie diagonale du fait de la monotonie de la perturbation [8], la partie hors-diagonale restant inchangée. Par conséquent, les résultats des propositions 1 et 2 et du corollaire 1 sont toujours valables pour l'analyse des itérations parallèles asynchrones.

Dans le cadre des interactions fluide-structure, ce type de contraintes sur la solution peut, par exemple, apparaître sur le déplacement dans l'équation de Navier.

Cependant, la formulation (3.182) est formelle et permet d'exprimer que la solution $\tilde{X}$ est soumise à des contraintes. Son intérêt réside principalement dans le fait qu'elle permet d'analyser de manière commode, grâce à la propriété de monotonie du sous-différentiel, le comportement convergent des itérations. Sur le plan algorithmique, il suffit de tester si les contraintes sont saturées ou non ; si elle ne sont pas saturées, on ne change pas la mise à jour obtenue par l'algorithme (3.36)-(3.38). Sinon, on doit projeter cette dernière sur l'ensemble convexe définissant la contrainte.

Par exemple, considérons à présent des problèmes ou le déplacement $\mathbf{D}$ est soumis à des contraintes inégalités. On considérera les situations suivantes :

$$\begin{cases} \mathbf{D}_{min} \leq \mathbf{D} \leq \mathbf{D}_{max}, \\ \mathbf{D} \leq \mathbf{D}_{max}, \\ \mathbf{D}_{min} \leq \mathbf{D}. \end{cases} \tag{3.185}$$

Dans chacun des cas, la solution $\mathbf{D}$ est un élément de l'ensemble convexe, qui dans chaque cas est respectivement défini comme suit :

$$\begin{cases} K_1 = \{\mathbf{D} \mid \mathbf{D}_{min} \leq \mathbf{D} \leq \mathbf{D}_{max}\}, \\ K_2 = \{\mathbf{D} \mid \mathbf{D} \leq \mathbf{D}_{max}\}, \\ K_3 = \{\mathbf{D} \mid \mathbf{D}_{min} \leq \mathbf{D}\}. \end{cases} \tag{3.186}$$

Il en découle que dans le premier cas où $\mathbf{D}_{min} \leq \mathbf{D} \leq \mathbf{D}_{max}$, on a

$$\partial\Gamma_1 = \begin{cases}]-\infty, 0] & \text{si } \mathbf{D} = \mathbf{D}_{min}, \\ 0 & \text{si } \mathbf{D}_{min} < \mathbf{D} < \mathbf{D}_{max}, \\ [0, +\infty[& \text{si } \mathbf{D} = \mathbf{D}_{max}, \\ \emptyset & \text{sinon}, \end{cases} \tag{3.187}$$

et le graphe correspondant de $\partial\Gamma_1$ est représenté sur la figure 3.7.

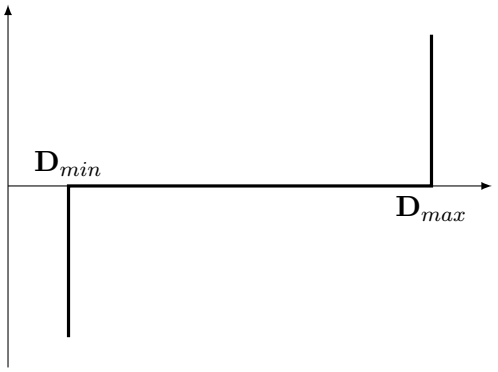


FIGURE 3.7 – Graphe de $\partial\Gamma_1$

Dans le cas où $\mathbf{D} \leq \mathbf{D}_{max}$, on a

$$\partial\Gamma_2 = \begin{cases} 0 & \text{si } \mathbf{D} < \mathbf{D}_{max}, \\ [0, +\infty[& \text{si } \mathbf{D} = \mathbf{D}_{max}, \\ \emptyset & \text{si } \mathbf{D} > \mathbf{D}_{max}, \end{cases} \tag{3.188}$$

et le graphe correspondant de $\partial\Gamma_2$ est représenté sur la figure 3.8.

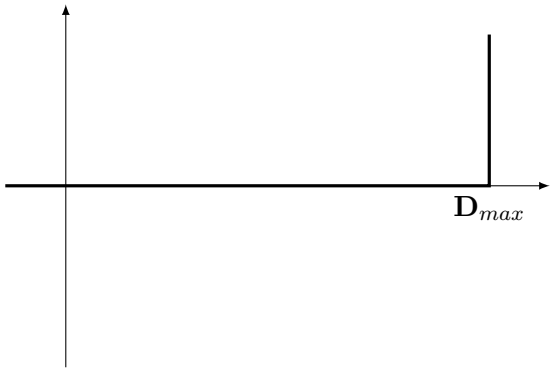


FIGURE 3.8 – Graphe de $\partial\Gamma_2$

Enfin, dans le cas où $\mathbf{D}_{min} \leq \mathbf{D}$, on a

$$\partial\Gamma_3 = \begin{cases}]-\infty, 0] & \text{si } \mathbf{D} = \mathbf{D}_{min}, \\ 0 & \text{si } \mathbf{D}_{min} < \mathbf{D} < \mathbf{D}_{max}, \\ \emptyset & \text{si } \mathbf{D} < \mathbf{D}_{min}, \end{cases} \tag{3.189}$$

et le graphe correspondant de $\partial\Gamma_1$ est représenté sur la figure 3.9.

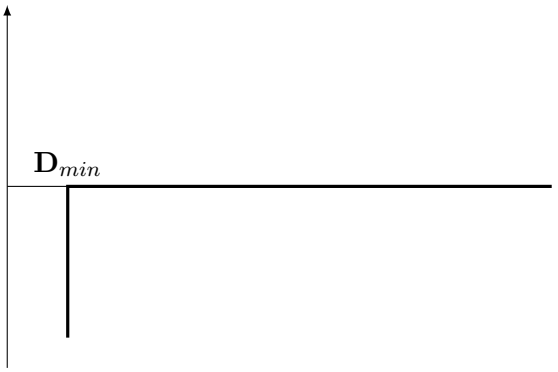


FIGURE 3.9 – Graphe de $\partial\Gamma_3$

La prise en compte des contraintes sur les déplacements conduit donc classiquement [8, 27, 36] à la formulation suivante :

$$A\tilde{X} + \partial\Gamma_i(\tilde{X}) - \tilde{B} \ni 0,\ i = 1, 2, 3. \tag{3.190}$$

Sur le plan physique, de telles contraintes inégalité peuvent être représentées par des limites géométriques, par exemple une butée, sur la structure.

Remarque 21. *Pour les méthodes de multi-splitting, on peut également analyser la convergence des itérés vers la solution des problèmes pseudo-linéaires éventuellement multivoques par des techniques d'ordre partiel [52, 53].*

Références

[3] J. Arnal, V. Migallón et J. Penadés. "Non-Stationary Parallel Multisplitting Algorithms for Almost Linear Systems". In : *Numerical Linear Algebra with Applications* 6 (1999), p. 79–92.

[5] J. Bahi, J.-C. Miellou et K. Rhofir. "Asynchronous Multisplitting Methods for Nonlinear Fixed Point Problems". In : *Numerical Algorithms* 15 (1997), p. 315–345.

[6] Z.-Z. Bai. "The Monotone Convergence Rate of the Parallel Nonlinear AOR Method". In : *Computers & Mathematics with Applications* 31 (1996), p. 1–8.

[7] Z.-Z. Bai, V. Migallón, J. Penadés et D. B. Szyld. "Block and Asynchronous Two-stage Methods for Mildly Nonlinear Systems". In : *Numerische Mathematik* 82 (1999), p. 1–20.

[8] V. Barbu. *Nonlinear Semigroups and Differential Equations in Banach Spaces*. Noordhoff International Publishing, 1976.

[9] G. Baudet. "Asynchronous Iterative Methods for Multiprocessors". In : *Journal of the Association for Computing Machinery* 25 (1978), p. 226–244.

[10] Y. Bazilevs, K. Takizawa et T. E. Tezduyar. *Computational Fluid-Structure Interaction : Methods and Applications*. Wiley, 2013.

[15] H.-J. Bungartz, M. Mehl et M. Schafer, éds. *Fluid-Structure Interaction II : Modelling, Simulation, Optimization*. Berlin, Heidelberg : Springer, 2010.

[16] H.-J. Bungartz et M. Schafer, éds. *Fluid-Structure Interaction : Modelling, Simulation, Optimization*. Berlin, Heidelberg : Springer, 2006.

[19] M. Chau, T. Garcia et P. Spiteri. "Asynchronous Schwarz Methods Applied to Constrained Mechanical Structures in Grid Environment". In : *Advances in Engineering Software* 74 (2014), p. 1–15.

[22] P. Comte, J.-C. Miellou et P. Spiteri. "La Notion de H-accrétivité, Applications". In : *CRAS* 283 (1976), p. 655–658.

[27] G. Duvaut et J.-L. Lions. *Les Inéquations en Mécanique et Physique*. Dunod, 1972.

[28] M. N. El Tarazi. "Some Convergence Results for Asynchronous Algorithms". In : *Numerische Mathematik* 39 (1982), p. 325–240.

[29] D. G. Feingold et R. S. Varga. "Block Diagonally Dominant Matrices and Generalizations of the Gerschgorin Circle Theorem". In : *Pacific Journal of Mathematics* 12 (1962), p. 1241–1250.

[31] A. Frommer. "On Asynchronous Iterations in Partially Ordered Spaces". In : *Numerical Functional Analysis and Optimization* 12 (1991), p. 315–325.

[32] A. Frommer et G. Mayer. "Convergence of Relaxed Parallel Multisplitting Methods". In : *Linear Algebra and its Applications* 119 (1989), p. 141–152.

[33] A. Frommer et G. Mayer. “On the Theory and Practice of Multisplitting Methods in Parallel Computation”. In : *Computing* 49 (1992), p. 63–74.

[35] L. Giraud et P. Spiteri. “Résolution Parallèle de Problèmes aux Limites Non Linéaires”. In : *ESAIM : Mathematical Modelling and Numerical Analysis* 25 (1991), p. 579–606.

[36] R. Glowinski, J.-L. Lions et R. Tremolieres. *Analyse Numérique des Inéquations Variationnelles*. T. 1 – 2. Dunod, 1976.

[41] R. I. Issa. “Solution of the Implicitly Discretised Fluid Flow Equation by Operator-Splitting”. In : *Journal of Computational Physics* 62 (1986), p. 40–65.

[42] H. Jasak et H. G. Weller. “Application of the Finite Volume Method and Unstructured Meshes to Linear Elasticity”. In : *International Journal for Numerical Methods in Engineering* 48 (2000), p. 267–287.

[44] L. S. Lai, C. H. Lai, A.-K. Cheik Ahamed et F. Magoulès. “Coupling and Simulation of Fluid-Structure Interaction Problems for Automotive Sun-Roof on Graphics Processing Unit”. In : *2014 IEEE International Conference on High Performance Computing and Communications*. 2014, p. 137–144.

[49] J.-C. Miellou. “Méthodes de Jacobi, Gauss-Seidel, Sur-Relaxation par Blocs, appliquées à une Classe de Problèmes Non Linéaires”. In : *CRAS* 273 (1971), p. 1257–1260.

[50] J.-C. Miellou. “Sur une Variante de la Méthode de Relaxation, Appliquée à des Problèmes Comportant un Opérateur Somme d’un Opérateur Différentiable et d’un Opérateur Maximal Monotone Diagonal”. In : *CRAS* 275 (1972), p. 1107–1110.

[51] J.-C. Miellou. “Algorithmes de Relaxation Chaotique à Retards”. In : *ESAIM : Mathematical Modelling and Numerical Analysis* 9 (1975), p. 55–82.

[52] J.-C. Miellou. “Itérations Chaotiques à Retards, Étude de la Convergence dans le Cas d’Espaces Partiellement Ordonnés”. In : *CRAS* 280 (1975), p. 233–236.

[53] J.-C. Miellou. “Asynchronous Iterations and Order Intervals”. In : *Parallel Algorithms and Architectures*. Sous la dir. de M. Cosnard et D. Trystram. North Holland, 1986, p. 85–96.

[54] J.-C. Miellou, D. El Baz et P. Spiteri. “A New Class of Asynchronous Iterative Algorithms with Order Intervals”. In : *Mathematics of Computation* 67 (1998), p. 237–255.

[55] J.-C. Miellou et P. Spiteri. “Un Critère de Convergence pour des Méthodes Générales de Point Fixe”. In : *ESAIM : Mathematical Modelling and Numerical Analysis* 19 (1985), p. 645–669.

[60] D. P. O’Leary et R. E. White. “Multi-Splittings of Matrices and Parallel Solution of Linear Systems”. In : *SIAM : Journal on Algebraic Discrete Methods* 6 (1985), p. 630–640.

[61] J. M. Ortega et W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Computer Science and Applied Mathematics. New-York : Academic Press, 1970.

[66] S. V. Patankar. *Numerical Heat Transfer and Fluid Flow*. Series in Computational Methods in Mechanics and Thermal Science. New-York : Hemisphere Publishing Corporation, 1980.

[67] S. V. Patankar et D. B. Spalding. "A Calculation Procedure for Heat, Mass and Momentum Transfer in Three-Dimensional Parabolic Flows". In : *International Journal of Heat and Mass Transfer* 15 (1972), p. 1787–1806.

[70] W. Rheinboldt. "On M-functions and Their Application to Nonlinear Gauss-Seidel Iterations and to Network Flows". In : *Journal of Mathematical Analysis and Applications* 32 (1970), p. 274–307.

[71] F. Robert. "Recherche d'une Matrice Parmi les Minorants d'un Opérateur Linéaire". In : *Numerische Mathematik* 9 (1966), p. 189–199.

[74] M. Schafer. *Computational Engineering - Introduction to Numerical Methods*. Berlin, Heidelberg : Springer, 2006.

[76] P. Spiteri. "Contribution à l'étude de grands systèmes non-linéaires, comportement d'algorithmes itératifs, stabilité de systèmes continus". Thèse de doct. Université de Besançon, 1984.

[77] P. Spiteri. "A New Characterization of M-matrices and H-matrices". In : *BIT Numerical Mathematics* 43 (2003), p. 1019–1032.

[79] P. Spiteri, J.-C. Miellou et D. El Baz. "Parallel Asynchronous Schwarz and Multisplitting Methods for a Nonlinear Diffusion Problem". In : *Numerical Algorithms* 33 (2003), p. 461–474.

[80] D. B. Szyld. "Different Models Of Parallel Asynchronous Iterations With Overlapping Blocks". In : *Computational and Applied Mathematics* 17 (1998), p. 101–115.

[84] D.-R. Wang, Z.-Z. Bai et D. J. Evans. "On the Monotone Convergence of Multisplitting Method for a Class of Systems of Weakly Nonlinear Equations". In : *International Journal of Computer Mathematics* 60 (1996), p. 229–242.

[85] R. E. White. "Parallel Algorithms for Nonlinear Problems". In : *SIAM : Journal on Algebraic Discrete Methods* 7 (1986), p. 137–149.

CHAPITRE

4

IMPLÉMENTATION PARALLÈLE

Sommaire

4.1 Introduction

Dans le chapitre 3, nous avons présenté les algorithmes de multi-splitting parallèles ; il est clair que sur les machines ne comportant qu'une seule unité de calcul ces méthodes présentent moins d'intérêt.

Depuis les années 1960, les architectures des ordinateurs ont considérablement évolué de manière à ce que des algorithmes parallèles puissent être exécutés de façon à diminuer le temps de restitution des travaux. On peut citer de nombreuses contributions à ce sujet et notamment [4, 9, 11, 18, 47, 81].

Dans ce chapitre, nous présentons d'abord un état de l'art concernant l'évolution des architectures des calculateurs parallèles. Dans un premier paragraphe, nous présentons donc la taxonomie de Flynn [30] qui correspond à une classification classiquement utilisée. Cette classification permet de présenter l'organisation de la mémoire habituellement adoptée dans la construction des machines parallèles. Pour fixer les idées, on distingue traditionnellement deux classes d'organisation de machines parallèles :

- architecture à mémoire partagée ;
- architecture à mémoire distribuée ;

à partir de cette organisation de base, les constructeurs peuvent concevoir une organisation mixte. Actuellement, les utilisateurs de calculateurs parallèles ont accès à différents types de plateformes comme des grappes de calcul, des grilles de calcul, des clouds ou encore des architectures GPU (*Graphics Processing Unit*). Cependant, les performances obtenues lors de l'utilisation de calculateurs parallèles sont variables, et il peut arriver, si l'utilisateur ne développe pas l'application cible avec soin, que la parallélisation ne permette pas une diminution du temps de restitution, but de l'utilisation de ce type de machine. De plus, l'architecture des machines parallèles, en particulier la conception du réseau de communication, a une influence sur les performances des algorithmes.

Pour mesurer les performances en termes de diminution du temps de restitution, un certain nombre d'indicateurs permet de mesurer le gain (ou non) obtenu lors de l'utilisation d'un calculateur parallèle. Le cas idéal correspond au fait que le rapport du temps séquentiel sur le temps parallèle (i.e. l'accélération) soit égal au nombre de processeurs. Cependant ceci ne peut pas, en pratique, se produire car la parallélisation du code de calcul nécessite le lancement des processeurs ce qui pénalise les performances obtenues compte tenu du nombre de cycles supplémentaires nécessaires à ce lancement. Nous détaillons au paragraphe 4.3 les notions classiques permettant de mesurer les performances.

Par ailleurs, nous avons vu au chapitre 3 une classe générale d'algorithmes parallèles asynchrones qui tend à minimiser les temps d'attente des processeurs lors des synchronisations. Nous présentons donc schématiquement au paragraphe 4.4.1 une classification des méthodes synchrones et asynchrones que nous appelons SISC (i.e. *Synchronous Iterations, Synchronous Communications*) et SIAC (i.e. *Synchronous Iterations, Asynchronous Communications*). Ces rappels étant effectués, nous pouvons à présent revenir sur notre application cible constituée par l'interaction fluide-structure. Pour mener à bien les

simulations, nous avons utilisé la bibliothèque OpenFOAM. L'implémentation des méthodes avec communications synchrones (SISC) n'a pas posé de problème. En revanche, celle des méthodes avec itérations synchrones mais communications asynchrones (SIAC) où les réceptions sont non bloquantes a posé une réelle difficulté qui a finalement pu être surmontée.

Enfin, la parallélisation du code fluide-structure nécessite, comme on l'a vu au chapitre 3, une décomposition par grands blocs du problème global. On est donc amené à résoudre des sous-problèmes où les interactions avec les autres sous-problèmes sont gérées de manière synchrone ou asynchrone. Plus précisément, la méthode de multi-splitting conduit à des codes de calcul comportant :

- une itération externe, où les communications sont synchrones ou asynchrones ;
- une itération interne, constituée entre autre par un solveur de type gradient conjugué, qui est de nature essentiellement synchrone.

Pour augmenter l'efficacité de la méthode de calcul, on considère comme solveur interne des méthodes de gradient conjugué préconditionné qui sont performantes. Pour le solveur fluide, compte tenu du fait que les matrices de discrétisation des équations de Navier-Stokes sont non-symétriques, on considère pour le calcul des vitesses un algorithme bien adapté à ce cas qui est la méthode du gradient bi-conjugué préconditionné par la factorisation incomplète LU, alors que pour le calcul des pressions, on utilise une méthode multi-grille. La résolution de l'équation de Navier est beaucoup moins complexe car, comme on l'a vu au chapitre 3, la matrice de discrétisation par volumes finis est symétrique et le solveur interne est constitué par une méthode de gradient conjugué préconditionné par une factorisation incomplète de Cholesky.

Le présent chapitre, essentiellement bibliographique, s'articule comme suit. Le paragraphe 4.2 fait état des différents types d'architecture de calculateurs parallèles. Le paragraphe 4.3 présente différents indicateurs permettant de mesurer les performances des algorithmes parallèles. Le paragraphe 4.4 détaille les différentes méthodes de parallélisation ainsi que leur implémentation pour l'application fluide-structure. Enfin, dans le paragraphe 4.5, nous présentons les différents solveurs utilisés pour la résolution du problème d'interaction fluide-structure ainsi que le couplage entre les différents solveurs.

4.2 Architectures

4.2.1 Classification des architectures

La diversité des architectures parallèles étant très importante, il est nécessaire d'utiliser des classifications de ces dernières afin d'en dégager les concepts sous-jacents. Il existe un grand nombre de classifications (ou taxonomies) mais nous nous baserons principalement sur celle introduite par Flynn en 1966 [30]. La taxonomie de Flynn des architectures parallèles ne se base pas sur la structure des machines mais plutôt sur la façon dont la machine associe les instructions aux données traitées [34, 40]. Un flot (*stream*) est une séquence d'objets (instructions ou données) exécutés ou traités par un processeur. On

distingue classiquement quatre grandes classes d'architectures parallèles, en fonction du fait que les flots d'instructions et de données sont uniques ou multiples.

4.2.1.1 SISD - *Single Instruction stream, Single Data stream*

C'est la machine séquentielle classique de von Neumann pour laquelle il n'y a qu'un seul flot d'instructions (et par conséquent, en pratique, une seule unité de traitement d'instructions ou *séquenceur*) et chaque instruction initie une opération, ce qui conduit à un flot de données et de résultats associés unique.

4.2.1.2 SIMD - *Single Instruction stream, Multiple Data streams*

C'est une machine qui ne possède qu'un seul séquenceur mais opère sur des données différentes et donne lieu à un grand nombre d'opérations. Elle correspond aux processeurs vectoriels et aux calculateurs possédant un grand nombre d'unités de calcul. Les processeurs de calcul SIMD exécutent donc la même instruction mais opérant sur des données différentes. Les données traitées peuvent se trouver dans un espace mémoire qui est global à toute la machine ou dans des espaces mémoire propres aux différents éléments de calcul.

4.2.1.3 MISD - *Multiple Instruction streams, Single Data stream*

La classe des machines MISD applique plusieurs flots d'instructions à un flot unique de données. Peu de calculateurs de ce type ont existé dans la pratique car le nombre d'applications qui peuvent être mises en œuvre sur ce genre d'architecture est très réduit.

4.2.1.4 MIMD - *Multiple Instruction streams, Multiple Data streams*

Un flot multiple d'instructions provenant de plusieurs séquenceurs agit sur un flot multiple de données. Cette classe d'architecture inclut par conséquent toutes les configurations multi-processeurs. Chaque processeur traite une donnée distincte, mais, en plus, il suit son propre flot d'instructions. Dans cette classe d'architecture, la localisation des données permet de distinguer deux sous-classes d'architectures : les architectures MIMD à mémoire partagée et les architectures MIMD à mémoire distribuée.

4.2.2 Organisation de la mémoire

Comme mentionnée précédemment, la localisation des données peut faire l'objet d'un critère de classification des architectures. En effet, l'organisation interne de la mémoire fait apparaître deux classes de machines MIMD bien distinctes : à mémoire partagée (*shared memory*) et à mémoire distribuée (*distributed memory*).

4.2.2.1 Mémoire partagée

Dans cette première classe d'architectures, les processeurs peuvent accéder directement à l'intégralité de la mémoire disponible. L'accès à la mémoire se fait par l'intermédiaire

d'un réseau d'interconnexion performant, avec un temps d'accès rapide et équitable entre les processeurs. Chaque processeur peut accéder à toute la mémoire, on parle d'espace d'adressage global (*global address space*). Les processeurs peuvent opérer indépendamment mais les données du calcul sont placées dans une mémoire commune ; la communication entre les processeurs se fait donc en accédant aux informations stockées en mémoire et n'est pas explicite. Toute écriture en mémoire peut être vue comme une communication implicite vers tout autre processeur qui accède ultérieurement à cette mémoire. Ce type d'organisation de la mémoire nécessite une programmation rigoureuse afin d'éviter les conflits d'accès à un même emplacement mémoire et nécessite une programmation par verrous des sections critiques. La figure 4.1 illustre un exemple d'architecture à mémoire partagée.

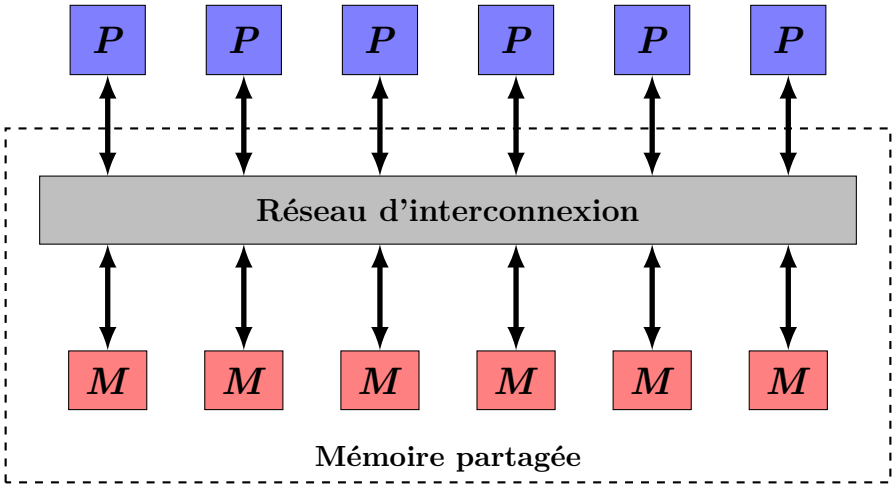


FIGURE 4.1 – Architecture à mémoire partagée

4.2.2.2 Mémoire distribuée

Pour les architectures à mémoire distribuée, chaque processeur dispose d'une mémoire locale qui lui est réservée. Un processeur ne peut donc accéder qu'à sa mémoire locale, on parle d'espace d'adressage local (*local address space*) ou, plus couramment, d'absence d'adressage global. La communication entre les processeurs ne peut alors plus se faire par l'intermédiaire de la mémoire, elle se fait par l'envoi explicite de messages entre les processeurs par le biais d'un réseau d'interconnexion aussi appelé réseau de communication (*communication network*), et l'architecture est dite à passage de messages (*message passing*). La figure 4.2 illustre un exemple d'architecture à mémoire distribuée.

Si une architecture à mémoire partagée permet un partage des données entre les tâches plus rapide qu'une architecture à mémoire distribuée du fait de la proximité entre espace mémoire et processeur, des inconvénients tels que la gestion des synchronisations dans la mémoire globale ou encore la difficulté à augmenter le nombre de processeurs (accroissement du trafic sur le chemin d'accès à la mémoire partagée) rendent le choix

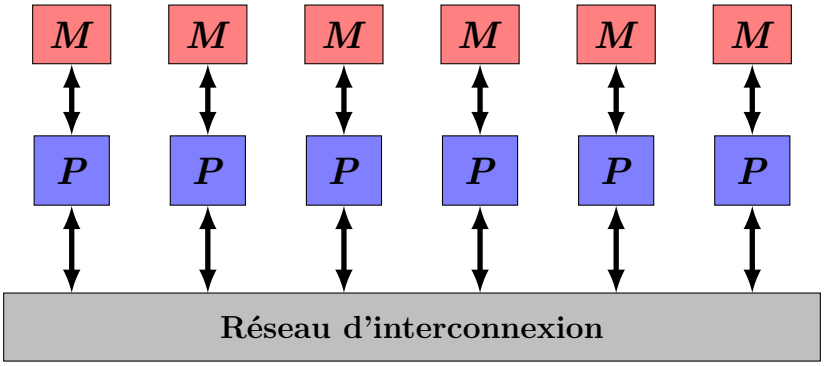


FIGURE 4.2 – Architecture à mémoire distribuée

d'architectures à mémoire distribuée plus viable pour la mise en œuvre de plateformes de calcul parallèle. En effet, le nombre de processeurs et la quantité de mémoire peuvent être augmentés proportionnellement, un accès rapide à la mémoire locale sur chaque processeur est possible sans surcoût de gestion des conflits d'accès et le coût reste raisonnable (i.e. PCs en réseau).

4.2.3 Plateformes de calcul parallèle

4.2.3.1 Grappe de calcul

Une grappe de calcul (*cluster*) est un ensemble de calculateurs faiblement ou fortement interconnectés par un réseau local à haut débit et qui fonctionnent ensemble de telle façon qu'ils peuvent être vus comme un seul calculateur parallèle. Chaque calculateur est appelé un nœud de calcul, possède un ou plusieurs processeurs ainsi que sa propre mémoire locale et exécute sa propre instance du système d'exploitation. Dans la plupart des cas, tous les nœuds possèdent la même architecture et sont gérés par le même système d'exploitation mais il n'est pas impossible d'avoir une grappe de calcul hétérogène constituée de nœuds aux caractéristiques différentes.

La figure 4.3 illustre une grappe de calcul constituée de 5 nœuds possédant chacun 4 unités de calcul ainsi qu'une mémoire locale. En général, une grappe de calcul possède un nœud, appelé nœud frontal, dédié à la gestion des ressources et à la distribution des tâches aux autres nœuds.

4.2.3.2 Grille de calcul

Contrairement à une grappe de calcul, une grille de calcul (*grid*) est une architecture distribuée constituée de calculateurs hétérogènes, géographiquement distants et qui peuvent effectuer des tâches différentes et indépendantes. Une grille de calcul est un ensemble de ressources hétérogènes, c'est-à-dire qu'elle peut être composée d'ordinateurs personnels, de serveurs, de clusters, etc. Contrairement à une grappe de calcul, l'administration des différentes ressources d'une grille n'est pas commune.

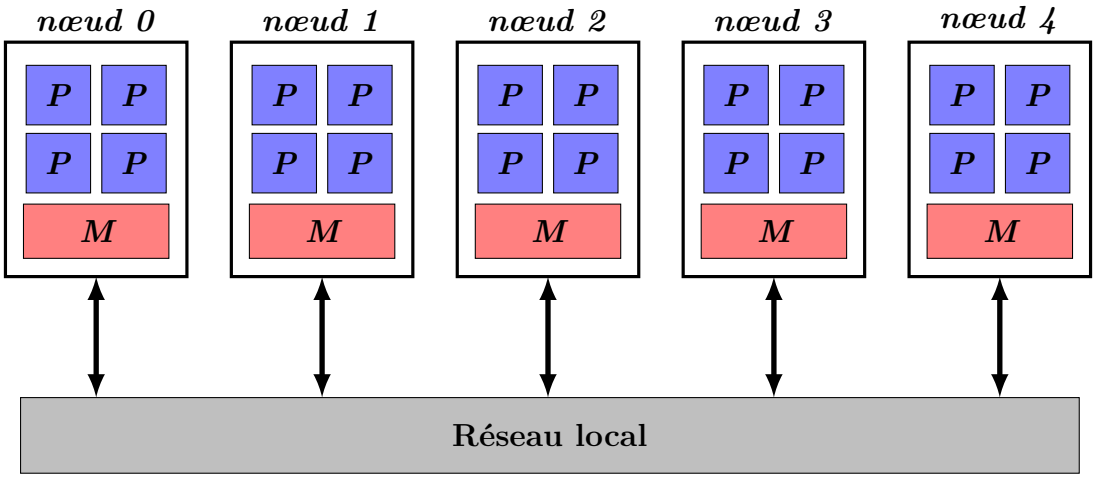


FIGURE 4.3 – Exemple d'architecture d'une grappe de calcul

L'objectif d'une grille est d'utiliser les ressources d'un très grand nombre de plateformes de calcul quelles que soient leurs localisations géographiques afin de résoudre des problèmes de très grande taille donc nécessitant beaucoup de mémoire et/ou nécessitant des temps de calcul très importants sur des calculateurs classiques. Le développement de ces grilles de calcul est rendu possible grâce aux réseaux haut-débit longue distance (e.g. réseau Ethernet). La figure 4.4a illustre un exemple de grille de calcul composée d'ordinateurs personnels (*desktop* ou portables), de serveurs et de clusters tous reliés entre eux par le réseau Internet.

Un exemple de ce type d'architecture est la grille de calcul Grid'5000 [12] constituée de plusieurs sites en France (cf. figure 4.4b) et regroupant environ 1000 nœuds, 8000 cœurs et utilisant des technologies variées. Pour des exemples de travaux réalisés grâce à cette plateforme de calcul, le lecteur est renvoyé à [24, 25, 43].

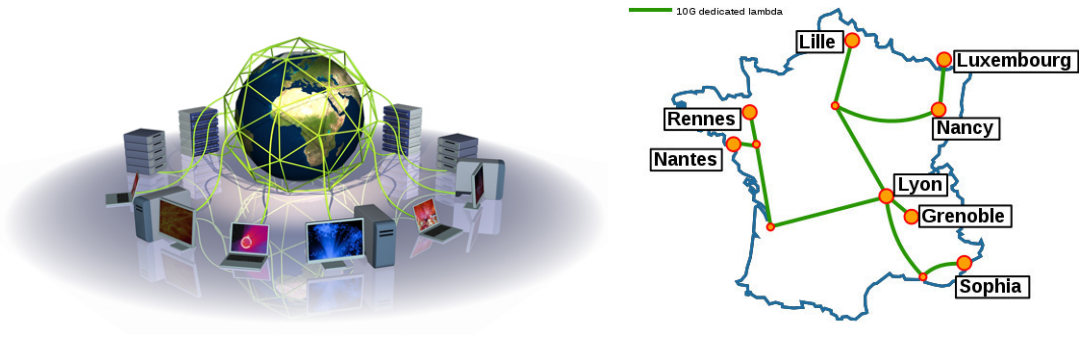
À noter cependant que les algorithmes asynchrones sont particulièrement adaptés à ce type d'architecture où les nœuds sont reliés par des réseaux lents et éloignés géographiquement [19].

4.2.3.3 Cloud computing

Le cloud computing est un modèle qui permet un accès omniprésent, pratique et à la demande, à un réseau partagé et à un ensemble de ressources informatiques configurables, comme par exemple : des réseaux, des serveurs, du stockage, des applications et des services (cf. figure 4.5) qui peuvent être provisionnées et libérées avec un minimum d'administration [2, 17, 59].

Ce modèle est composé de 5 caractéristiques essentielles et de 3 modèles de services. Les 5 caractéristiques essentielles sont :

– ressources en libre-service partout dans le monde ;



(a) Illustration d'une grille (b) Grid'5000 [37]

FIGURE 4.4 – Exemples de grille de calcul

- élasticité : la puissance et la capacité de stockage peuvent être adaptées automatiquement en fonction des besoins des utilisateurs ;
- ouverture : les services sont accessibles depuis un ordinateur, un téléphone ou une tablette ;
- mutualisation : il combine des ressources hétérogènes (matériel, logiciel, trafic réseau) pour servir plusieurs utilisateurs à qui les ressources sont automatiquement distribuées ;
- paiement à l'usage : la quantité de service est mesurée à des fins de contrôle, d'adaptation des moyens techniques et de facturation.

Les trois principaux modèles de services sont :

- IaaS (*Infrastructure as a Service*) : c'est le service de plus bas niveau, il consiste à offrir un accès à un parc informatique virtualisé à l'utilisateur qui peut y installer un système d'exploitation et des applications ;
- PaaS (*Platform as a Service*) : c'est le service situé juste au dessus du précédent, le système d'exploitation et les outils d'infrastructure sont sous la responsabilité du fournisseur mais l'utilisateur garde le contrôle des applications et peut ajouter ses propres outils ;
- SaaS (*Software as a Service*) : dans ce type de service, des applications sont mises à la disposition de l'utilisateur qui n'a pas à se soucier du maintien de ces dernières, les services de courrier électronique sont un exemple de ce modèle.

Par rapport à une grille de calcul, le cloud computing dispose de surcouches logicielles qui permettent de faire abstraction des problématiques matérielles au niveau utilisateur. Bien que dans le présent travail nous n'ayons pas utilisé des services de cloud computing, les algorithmes à itérations asynchrones peuvent présenter un intérêt dans ce type d'environnement par rapport à leurs équivalents synchrones.

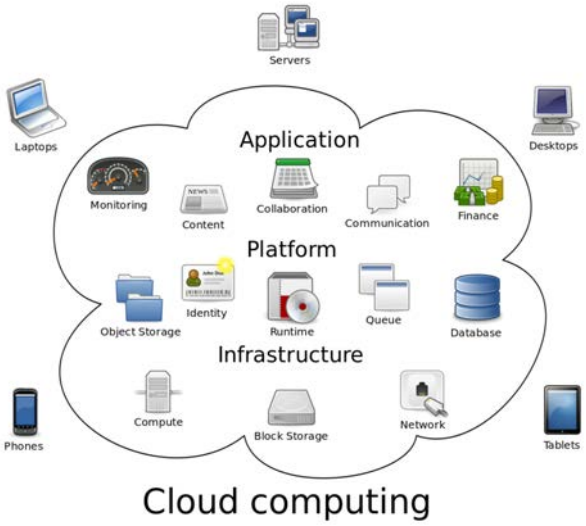


FIGURE 4.5 – Illustration du cloud computing [21]

4.2.3.4 Architecture GPU

Les cartes graphiques (*GPU : Graphics Processing Unit*) ont été introduites dans les ordinateurs personnels afin de produire des graphismes de haute qualité plus rapidement qu'un processeur classique (CPU) et afin de soulager ce dernier de ce travail [23, 86]. Ces fonctions d'affichage sont très répétitives et très spécifiques ; c'est pourquoi les fabricants produisent des cartes graphiques de plus en plus sophistiquées. Ces cartes graphiques possèdent leur propre mémoire dédiée afin d'effectuer ces opérations. Depuis 2000, les utilisateurs de GPU peuvent effectuer des opérations arithmétiques sur une série de pixels. Des chercheurs ont alors essayé d'appliquer ces opérations sur d'autres types de données afin d'utiliser la puissance des GPUs à des fins plus génériques. Malheureusement, la programmation était très complexe et très dépendante des contraintes matérielles, c'est pourquoi, en 2007, la société NVIDIA a proposé la technologie CUDA qui a permis d'unifier la programmation sur leurs modèles de cartes graphiques. Cette avancée a rendu le calcul générique sur un processeur graphique (*GPGPU : General Purpose Graphics Processing Unit*) beaucoup plus accessible.

Les unités de traitement composant un GPU sont beaucoup plus simples que celles d'un CPU traditionnel mais il est aussi beaucoup plus facile d'intégrer un grand nombre de ces unités dans une carte graphique que dans un CPU. La figure 4.6 illustre la comparaison entre le nombre d'unités de traitement dans un CPU et dans un GPU. La vitesse d'horloge d'un CPU en général d'environ 3 GHz alors que celle d'un GPU est d'environ 1.5 GHz ; même si la vitesse d'horloge d'un GPU est moins importante, le nombre de cœurs permet de fournir plus de puissance.

Même si de nos jours les GPUs sont capables de fournir une grande puissance de calcul, ils ne sont efficaces que pour certains types d'opérations telles que des tâches répétitives dans lesquelles seules les données changent [87]. Ce type d'architecture ne

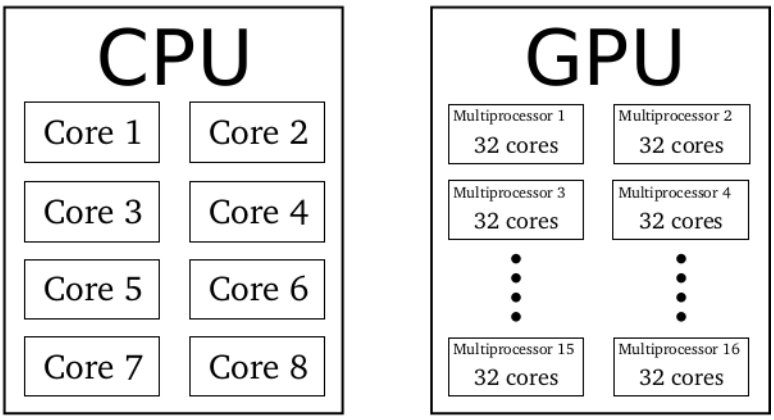


FIGURE 4.6 – Comparaison du nombre de cœurs dans un CPU et dans un GPU [23]

semble pas convenir aux types d'algorithmes implicites utilisés dans les applications fluide-structure. Par contre, elle est bien adaptée à l'utilisation d'algorithmes explicites.

4.3 Mesures de performances

4.3.1 Notions d'accélération et d'efficacité

Définition 9. (Accélération) Soit $T_1(n)$ le temps nécessaire à l'algorithme pour résoudre un problème de taille n avec un seul processeur et soit $T_P(n)$ celui que la résolution parallèle prend avec P processeurs. On appelle accélération (*speed-up*) le rapport $S(n, P) = T_1(n)/T_P(n)$.

On peut définir un second nombre qui permet d'apprécier le taux de parallélisme mis en œuvre pour un problème donné. Ce nombre est appelé efficacité (*efficiency*).

Définition 10. (Efficacité) Soit $T_1(n)$ le temps nécessaire à l'algorithme pour résoudre un problème de taille n avec un seul processeur, soit $T_P(n)$ celui que la résolution prend avec P processeurs et soit $S(n, P) = T_1(n)/T_P(n)$ le facteur d'accélération. On appelle efficacité de l'algorithme le nombre

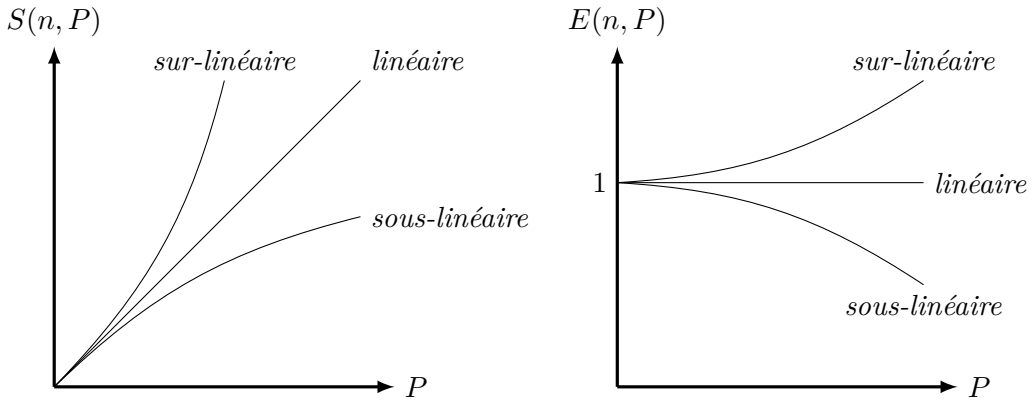
$$E(n, P) = \frac{S(n, P)}{P} = \frac{T_1(n)}{T_P(n) \cdot P}. \tag{4.1}$$

L'efficacité correspond à une normalisation du facteur d'accélération par le nombre de processeurs. Les figures 4.7a et 4.7b illustrent les notions d'accélération et d'efficacité. L'accélération est dite *linéaire* lorsque $S(n, P) = P$, c'est-à-dire lorsque $E(n, P) = 1$. Elle est dite *sous-linéaire* ou *sub-linéaire* lorsque $S(n, P) < P$, c'est-à-dire lorsque $E(n, P) < 1$. Enfin, elle est dite *sur-linéaire* ou *supra-linéaire* lorsque $S(n, P) > 1$, c'est-à-dire lorsque $E(n, P) > 1$.

Pour une accélération linéaire, le gain en temps est égal au nombre de processeurs ; cela signifie que chacun d'eux a un taux d'activité de 100 %. Une accélération sous-linéaire

correspond à un gain de temps inférieur au nombre de processeurs, les processeurs ne travaillent donc pas tous à 100 % ; l'algorithme ne tire pas entièrement parti de la puissance totale de la machine. Une accélération sur-linéaire qui implique un taux d'activité des processeurs supérieur à 100 % n'est atteint que très rarement. Par exemple, dans les méthodes de sous-domaine, ce comportement se produit lorsque le découpage a pour effet d'accélérer la convergence des méthodes itératives. Cependant, le gain au dessus de 100 % est faible.

Cela peut cependant se produire et s'expliquer, d'une part par l'organisation de la mémoire : en effet, on peut imaginer qu'une machine mono-processeur fasse de nombreux défauts de cache, là où une machine parallèle n'en fera que rarement. L'algorithme s'exécute alors plus rapidement du seul fait des caractéristiques matérielles. D'autre part, on peut aussi imaginer que certains algorithmes parallèles fassent l'économie d'instructions.



(a) Accélération $S(n,P)$ (b) Efficacité $E(n,P)$

FIGURE 4.7 – Illustration des notions d'accélération et d'efficacité

4.3.2 Loi d'Amdahl

Considérons une machine fictive, dans laquelle toute instruction prend un temps unitaire et un algorithme qui, pour une donnée de taille n, est exécuté en temps séquentiel $T(n)$ sur cette machine, nécessitant donc $T(n)$ instructions. Supposons en plus que le temps de calcul séquentiel puisse être découpé de la manière suivante : $T(n) = T_s(n) + T_\parallel(n)$, où $T_s(n)$ est le temps pris pour l'exécution de la partie de l'algorithme qui ne peut être parallélisée et où $T_\parallel(n)$ est le temps pris pour l'exécution de la partie de l'algorithme qui peut être parallélisée. En posant $T_s(n) = f(n) \cdot T(n)$ et $T_\parallel(n) = (1 - f(n)) \cdot T(n)$, on peut définir le temps $T_1(n)$ d'exécution séquentielle de la manière suivante :

$$T_1(n) = T_s(n) + T_\parallel(n) = T(n). \tag{4.2}$$

Le meilleur taux d'exécution sur P processeurs, en vertu du principe énoncé par

Amdahl en 1967 [1], est le temps

$$T_P(n) = T_s(n) + \frac{T_\parallel(n)}{P} = \left(f(n) + \frac{1 - f(n)}{P}\right) \cdot T(n), \tag{4.3}$$

qui est obtenu lorsque la partie parallélisable de l'algorithme est répartie de manière optimale sur les P processeurs, c'est-à-dire lorsque les $T_\parallel(n)$ instructions de la partie parallélisable sont exécutées sur la machine parallèle. Notons que les algorithmes séquentiel et parallèle sont identiques au sens où ils comportent exactement les mêmes instructions.

Sous ces hypothèses, la loi d'Amdahl donne comme borne pour le facteur d'accélération $S(n, P)$ l'inverse de la partie séquentielle de l'algorithme parallèle :

$$\lim_{P\to\infty} S(n, P) = \lim_{P\to\infty} \frac{1}{f(n) + (1 - f(n))/P} = \frac{1}{f(n)}. \tag{4.4}$$

Cette formulation implique que pour un algorithme ayant une partie séquentielle de $f(n) = 10\,\%$, l'accélération est limitée à 10, quel que soit le nombre de processeurs utilisé. En conséquence, on ne peut espérer des accélérations importantes que pour des algorithmes ayant une faible proportion d'instructions séquentielles. De surcroît, cette proportion doit diminuer avec n. Fort heureusement, ceci est très souvent le cas. Pour certains algorithmes, la partie séquentielle $T_s(n)$ est constante. Pour d'autres elle augmente, mais beaucoup plus lentement que $T_\parallel(n)$.

Toujours sous les mêmes hypothèses, l'efficacité s'exprime par la formule suivante :

$$E(n, P) = \frac{1}{1 + (P - 1) \cdot f(n)}. \tag{4.5}$$

Sur un seul processeur, elle est bien sûr unitaire. Pour n fixé, l'efficacité tend vers 0 avec P grandissant. Ceci signifie qu'en fixant la taille du problème à traiter on limite implicitement le nombre de processeurs que l'on pourra utiliser de manière efficace. Par ailleurs, l'efficacité est la plupart du temps inférieure à l'unité dans les méthodes itératives, interdisant toute accélération sur-linéaire.

4.4 Parallélisation

4.4.1 Méthodes de parallélisation

4.4.1.1 SISC - *Synchronous Iterations, Synchronous Communications*

Les solveurs itératifs les plus courants sont ceux utilisant des itérations synchrones et des communications synchrones (SISC). Le synchronisme des itérations est dû au fait que chaque processeur ne peut commencer à calculer la nouvelle itération que s'il a reçu les données calculées à l'itération précédente par tous ses voisins. Par conséquent, tous les processeurs commencent les calculs de la même itération en même temps et échangent les données à la fin de chaque itération par le biais de communications globales synchrones.

La figure 4.8 illustre l'exécution d'un algorithme parallèle SISC. Le synchronisme des itérations d'un algorithme SISC implique un nombre d'itérations identique à celui

d'un l'algorithme séquentiel correspondant. Donc, les conditions de convergence des algorithmes parallèles SISC sont les mêmes que celles d'un algorithme séquentiel. Cependant, le synchronisme des communications peut souvent être la cause de périodes d'inactivité des processeurs car ils doivent attendre avant de communiquer leurs données ou encore car la vitesse du réseaux engendre des communications trop lentes.

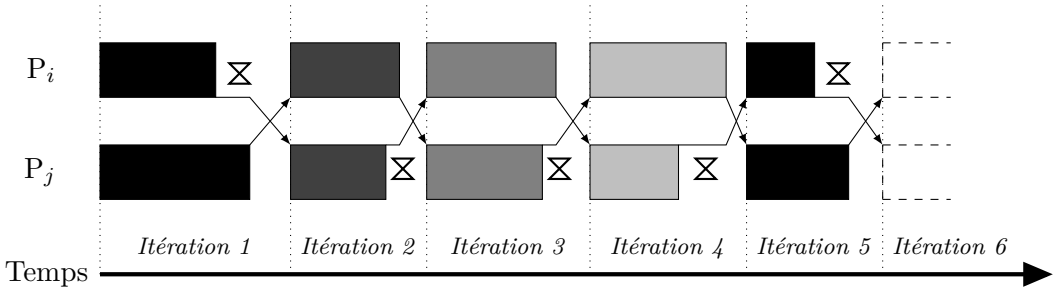


FIGURE 4.8 – Exemple d'un schéma SISC sur deux processeurs P_i et P_j

4.4.1.2 SIAC - *Synchronous Iterations, Asynchronous Communications*

Les solveurs itératifs utilisant des itérations synchrones mais des communications asynchrones (SIAC) ont été développés avec pour objectif d'améliorer les performances sur des réseaux avec des inter-connexions lentes et/ou hétérogènes. Ils parviennent à réduire les temps d'inactivité des processeurs entre deux itérations. De manière identique aux algorithmes SISC, un processeur attend toujours la réception des données partagées calculées par tous ses voisins à l'itération précédente. Cependant, les communications synchrones globales sont remplacées par des envois asynchrones et des réceptions bloquantes.

La figure 4.9 illustre l'exécution d'un algorithme parallèle SIAC. Les conditions de convergence sont les mêmes que pour un algorithme SISC ou un algorithme séquentiel. Cependant, les communications asynchrones permettent une superposition entre calculs et transferts de données. Un processeur peut envoyer des données partagées à ses voisins dès qu'elles sont prêtes à être utilisées dans la nouvelle itération, ce qui a pour conséquence de réduire les temps d'attente nécessaires pour que les données soient reçues entre deux itérations.

4.4.1.3 AIAC - *Asynchronous Iterations, Asynchronous Communications*

Les solveurs itératifs à itérations asynchrones et communications asynchrones (AIAC) ont été développés pour améliorer les performances sur des plateformes de calcul constituées de ressources hétérogènes et géographiquement distantes. En effet, les méthodes itératives parallèles SISC et SIAC sont relativement faciles à mettre en œuvre mais pénalisent les performances sur ce type de plateforme à cause du synchronisme de leurs itérations. Sur ce type d'architecture, les réseaux de communication sont souvent lents et/ou l'hétérogénéité des ressources engendre des temps de calcul pouvant être très différents suivant les processeurs.

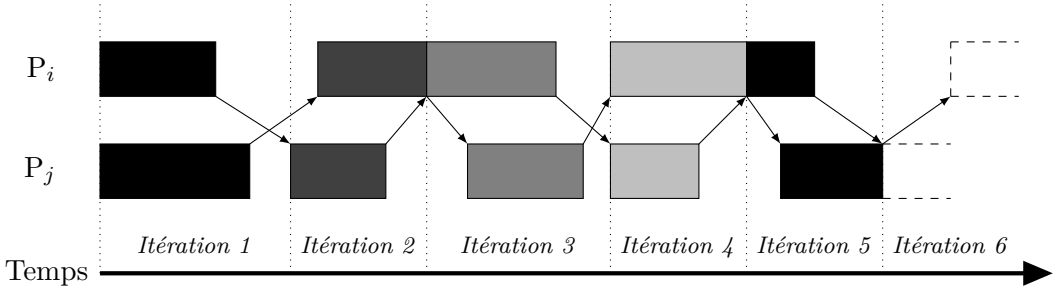


FIGURE 4.9 – Exemple d'un schéma SIAC sur deux processeurs P_i et P_j

Dans les méthodes parallèles AIAC, chaque processeur exécute ses propres itérations sans se préoccuper de la progression des autres processeurs ; il passe d'une itération à l'autre sans attendre la réception des données partagées par ses voisins à l'itération précédente et utilise des données locales et les versions les plus récentes des données partagées disponibles au début de chaque itération. Les solveurs AIAC utilisent des communications asynchrones de type envois et réceptions non bloquants.

La figure 4.10 illustre l'exécution d'un algorithme parallèle AIAC. Les itérations asynchrones permettent d'éliminer les temps d'inactivité des processeurs, des itérations avec des numéros différents peuvent alors être exécutées à un même moment ; certains processeurs peuvent donc être plus rapides et effectuer plus d'itérations que d'autres. En revanche, les solveurs itératifs de type AIAC effectuent plus d'itérations et nécessitent des conditions de convergence plus strictes que les solveurs à itérations synchrones SISC et SIAC. Une analyse plus complexe est alors nécessaire pour déterminer les bons critères de convergence.

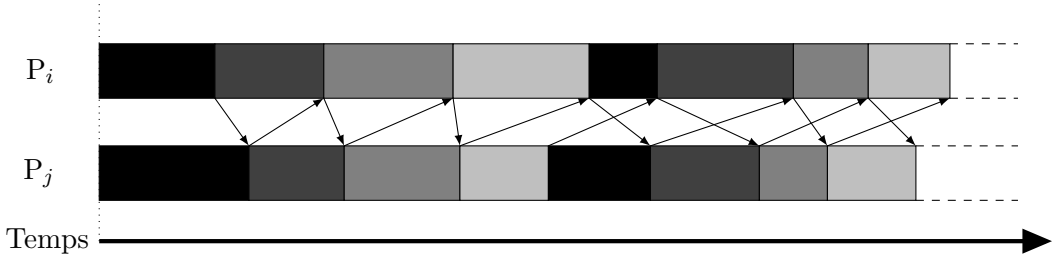


FIGURE 4.10 – Exemple d'un schéma AIAC sur deux processeurs P_i et P_j

Ces méthodes nécessitent l'utilisation de solveurs spécifiques tels que des solveurs basés sur des méthodes de relaxations à itérations asynchrones. Or, dans les applications qui font l'objet de ces travaux, il est préférable d'utiliser des méthodes de type gradient conjugué (cf. paragraphe 4.5) qui n'existent pas dans une variante asynchrone. C'est pourquoi, dans la suite, on se limitera à l'utilisation des méthodes SISC et SIAC.

4.4.2 Tests d'arrêt

Le critère d'arrêt des itérations parallèles asynchrones est un problème particulièrement difficile. Généralement, on distingue soit une approche informatique basée sur l'utilisation de *snapshots* [45] qui permettent de construire un vecteur résidu global, soit une approche purement numérique. Dans ce paragraphe, nous nous limiterons à ce dernier aspect, même si sur le plan pratique les deux approches peuvent être combinées.

Sur le plan numérique, la notion d'itérations, si elle est valable en séquentiel et en mode synchrone, perd tout son sens en mode asynchrone. Dans ce cas là, il conviendrait plutôt de parler de *macro-itération*, qui correspond au fait qu'on relaxe chaque composante au moins une fois. Par ailleurs, dans le chapitre 3, nous avons évoqué à la remarque 18, la possibilité de définir un critère d'arrêt, cependant coûteux, par encadrement de la solution, lorsque le principe du maximum discret est applicable. Une autre approche est également envisageable, qui utilise les ensembles emboîtés centrés en la solution dans lesquels sont contenus les itérés successifs (cf. [11, 73]). Dans ce type d'approche, on peut effectuer une estimation du diamètre de chaque ensemble emboîté (cf. [56, 57, 78]) et concevoir un test d'arrêt lorsque le diamètre de ces ensembles est inférieur à 2ε, ε étant la tolérance choisie.

Dans le cas des itérations synchrones que l'on utilise dans ce travail, les critères d'arrêt sont beaucoup plus faciles à déterminer puisqu'on peut se limiter à des tolérances sur les résidus des variables calculées.

4.5 Solveur FSI

Afin d'obtenir une bonne précision sur la solution du problème, il est nécessaire de considérer des maillages très fins. Cela amène à résoudre des systèmes algébriques de très grandes tailles de la forme $A\tilde{X} = \tilde{B}$ à chaque pas de temps. La résolution des systèmes linéaires provenant de la discrétisation des équations de Navier-Stokes et de Navier peut alors mener à des temps de calcul très importants. Afin de surmonter ces problèmes, nous sommes amenés à décomposer le problème en β grands blocs et, en utilisant l'une des méthodes décrites au paragraphe précédent, à paralléliser la résolution des systèmes algébriques grâce à la méthode du multi-splitting qui constitue alors l'itération externe.

La méthode de multi-splitting implémentée opère de façon itérative pour résoudre le système linéaire de telle façon que chaque sous-système soit de la forme

$$\hat{A}_{i,i} \cdot \hat{\tilde{X}}_i = \hat{\tilde{B}}_i - \sum_{j \neq i} \hat{A}_{i,j} \cdot \hat{\tilde{X}}_j,\ l = 1, \ldots, \beta, \tag{4.6}$$

où la matrice A du système linéaire est décomposée en blocs $\hat{A}_{i,j}$ avec $i, j = 1, \ldots, \beta$. La solution $\tilde{X}$ et le second membre $\tilde{B}$ sont décomposés en sous-vecteurs correspondant, respectivement notés $\hat{\tilde{X}}_i$ et $\hat{\tilde{B}}_i$. Chaque sous-système (4.6) est résolu indépendamment par un bloc de processeurs et des communications sont requises pour mettre à jour le second membre de chaque sous-domaine, de telle façon que les vecteurs mis à jour par les autres processeurs représentent les dépendances des données entre les blocs. Dans la

méthode de multi-splitting considérée, en raison de la structure creuse des sous-matrices, on a en fait une itération à deux niveaux ; une itération parallèle externe comprenant des communications bloquantes ou non bloquantes et une itération interne où chaque système du type (4.6) peut être résolu par une variante de la méthode du gradient conjugué ou par un autre algorithme. Il faut noter qu'en ce qui concerne l'itération parallèle externe, avec des communications bloquantes ou non bloquantes, la méthode de cacul considérée rentre bien dans le cadre du modèle d'algorithme (3.36) à (3.38) parce que les communications inter-processeurs peuvent être synchrones ou asynchrones.

Par ailleurs, en ce qui concerne les deux pas correcteurs de la méthode PISO :

- on calcule d'abord les corrections de pression par (3.15) et (3.18), ce qui nécessite l'utilisation de solveurs performants et on a choisit d'utiliser une méthode multigrille géométrique algébrique qui s'avère plus efficace ;
- on calcule ensuite les quantités de mouvement explicitement par (3.14) et (3.17).

Finalement, on constatera, dans le chapitre 5 présentant les résultats expérimentaux, l'efficacité de ce couplage de l'itération parallèle externe avec l'itération interne.

Pour la résolution des équations de Navier-Stokes, les matrices utilisées dans la méthode PISO sont non-symétriques pour les équations de quantité de mouvement et symétriques pour les équations de pression. Les matrices provenant de la discrétisation de l'équation de Navier sont symétriques ; par conséquent, pour chaque type de système, nous devons considérer des méthodes de résolution légèrement différentes compte tenu de ces propriétés.

4.5.1 Solveur structure

Pour le cas des matrices symétriques, une méthode classiquement considérée est celle du gradient conjugué préconditionné par une factorisation incomplète de Cholesky. Cependant, afin de réduire l'espace de stockage nécessaire, il est possible d'introduire une version simplifiée de cette méthode, appelée méthode de gradient conjugué préconditionné par une factorisation diagonale incomplète de Cholesky (*DICPCG : Diagonal Incomplete Cholesky Preconditioned Conjugate Gradient*). Dans cette méthode, le remplissage provenant de la factorisation, dû aux coefficients hors-diagonaux, est éliminé et seuls les coefficients diagonaux sont modifiés ; autrement dit, les parties supérieure et inférieure de la matrice à inverser restent inchangées. Un tel préconditonnement peut être utilisé pour résoudre l'équation de Navier mais aussi l'équation de pression. L'algorithme 3 résume l'implémentation de la méthode du gradient conjugué préconditionné où $\mathcal{P}$ est la matrice de préconditionnement.

4.5.2 Solveur fluide

Dans le cas des matrices non-symétriques, nous utilisons la méthode du gradient biconjugué préconditionné par la factorisation diagonale incomplète LU (*DILUBiPCG :*

Algorithme 3 Méthode PCG

1: $\mathbf{r}^{(0)} = \mathbf{b} - A\mathbf{\Phi}^{(0)}$
$\mathbf{d}^{(0)} = \mathcal{P}^{-1}\mathbf{r}^{(0)}$ *: choisir la direction de départ*
2: itérer jusqu'à convergence
3: $\mathtt{a}^{(n)} = \dfrac{\left(\mathbf{r}^{(n)}\right)^T \mathcal{P}^{-1}\mathbf{r}^{(n)}}{\left(\mathbf{d}^{(n)}\right)^T A\mathbf{d}^{(n)}}$ *: choisir le coefficient* $\mathtt{a}^{(n)}$ *dans la direction* $\mathbf{d}$
4: $\mathbf{\Phi}^{(n+1)} = \mathbf{\Phi}^{(n)} + \mathtt{a}^{(n)}\mathbf{d}^{(n)}$ *: obtenir le nouveau* $\mathbf{\Phi}$
5: $\mathbf{r}^{(n+1)} = \mathbf{r}^{(n)} - \mathtt{a}^{(n)}A\mathbf{d}^{(n)}$ *: calculer le nouveau résidu*
6: $\mathtt{b}^{(n+1)} = \dfrac{\left(\mathbf{r}^{(n+1)}\right)^T \mathcal{P}^{-1}\mathbf{r}^{(n+1)}}{\left(\mathbf{r}^{(n)}\right)^T \mathcal{P}^{-1}\mathbf{r}^{(n)}}$ *: calculer le coefficient du résidu conjugué*
7: $\mathbf{d}^{(n+1)} = \mathcal{P}^{-1}\mathbf{r}^{(n+1)} + \mathtt{b}^{(n+1)}\mathbf{d}^{(n)}$ *: obtenir la nouvelle direction conjuguée de recherche*

Diagonal Incomplete LU Preconditioned Bi-Conjugate Gradient). Le gradient bi-conjugué est une méthode utilisée pour résoudre le système suivant :

$$\begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} \hat{\Phi} \\ \Phi \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}, \tag{4.7}$$

où Φ est une variable auxiliaire qui n'est pas utilisée dans la pratique.

Cette méthode engendre des matrices globales symétriques et, par conséquent, permet une application directe de la méthode du gradient conjugué aux systèmes algébriques non-symétriques. L'avantage principal de la méthode du gradient bi-conjugué est sa facilité d'implémentation et l'espace relativement modéré qu'elle requiert pour stocker des informations. En revanche, cette méthode a le désavantage d'être instable. De plus, quand la méthode de gradient bi-conjugué est correctement préconditionnée, elle converge rapidement vers la solution du système linéaire. De la même façon que pour les matrices symétriques et pour les mêmes raisons, nous considérons un préconditionnement par la factorisation incomplète LU utilisant la forme par blocs des matrices, ce qui conduit ensuite à la méthode DILUPBiCG. La convergence de la méthode est alors fortement améliorée. L'algorithme 4 résume l'implémentation de la méthode du gradient bi-conjugué.

Cette méthode itérative est utilisée trois fois dans le première étape de l'algorithme PISO ; pour le calcul de chaque composante spatiale de la vitesse.

D'un point de vue théorique, il faut se demander si la factorisation incomplète est toujours possible et, si elle n'est pas toujours possible, dans quels cas l'est-t-elle ? En fait, une telle factorisation incomplète n'est pas possible pour n'importe quelle matrice et nous devons alors prendre en compte les propriétés des matrices de discrétisation présentées dans le paragraphe 3.2.1. En effet, dans les deux cas, les matrices provenant de la discrétisation ont une propriété commune ; en accord avec les expressions des coefficients de ces matrices ((3.5) et (3.29)), on peut noter que les coefficients diagonaux sont strictement positifs, les coefficients hors-diagonaux sont négatifs ou nuls et que de telles

Algorithme 4 Méthode PBiCG

1: $\mathbf{r}^{(0)} = \hat{\mathbf{r}}^{(0)} = \mathbf{b} - A\mathbf{\Phi}^{(0)}$
$\mathbf{d}^{(0)} = \mathcal{P}^{-1}\mathbf{r}^{(0)}$ *: choisir les directions de départ*
$\hat{\mathbf{d}}^{(0)} = \mathcal{P}^{-T}\hat{\mathbf{r}}^{(0)}$
2: itérer jusqu'à convergence
3: $\mathtt{a}^{(n)} = \dfrac{\left(\hat{\mathbf{r}}^{(n)}\right)^T \mathcal{P}^{-1}\mathbf{r}^{(n)}}{\left(\hat{\mathbf{d}}^{(n)}\right)^T A\mathbf{d}^{(n)}}$ *: choisir le coefficient* $\mathtt{a}^{(n)}$ *dans la direction* $\mathbf{d}$
4: $\mathbf{\Phi}^{(n+1)} = \mathbf{\Phi}^{(n)} + \mathtt{a}^{(n)}\mathbf{d}^{(n)}$ *: obtenir le nouveau* $\mathbf{\Phi}$
5: $\mathbf{r}^{(n+1)} = \mathbf{r}^{(n)} - \mathtt{a}^{(n)}A\mathbf{d}^{(n)}$ *: calculer le nouveau résidu* $\mathbf{r}$
6: $\hat{\mathbf{r}}^{(n+1)} = \hat{\mathbf{r}}^{(n)} - \mathtt{a}^{(n)}A^T\hat{\mathbf{d}}^{(n)}$ *: calculer le nouveau résidu* $\hat{\mathbf{r}}$
7: $\mathtt{b}^{(n+1)} = \dfrac{\left(\hat{\mathbf{r}}^{(n+1)}\right)^T \mathcal{P}^{-1}\mathbf{r}^{(n+1)}}{\left(\hat{\mathbf{r}}^{(n)}\right)^T \mathcal{P}^{-1}\mathbf{r}^{(n)}}$ *: calculer le coefficient du résidu conjugué*
8: $\mathbf{d}^{(n+1)} = \mathcal{P}^{-1}\mathbf{r}^{(n+1)} + \mathtt{b}^{(n+1)}\mathbf{d}^{(n)}$ *: obtenir la nouvelle direction conjuguée de recherche* $\mathbf{d}$
9: $\hat{\mathbf{d}}^{(n+1)} = \mathcal{P}^{-T}\hat{\mathbf{r}}^{(n+1)} + \mathtt{b}^{(n+1)}\hat{\mathbf{d}}^{(n)}$ *: obtenir la nouvelle direction conjuguée de recherche* $\hat{\mathbf{d}}$

matrices sont strictement diagonale dominante en raison de la discrétisation en temps. Comme déjà indiqué précédemment, ce sont des M-matrices [61].

Plusieurs travaux par des auteurs différents ont été menés sur ce type de matrices, et, pour plus de détails, le lecteur est renvoyé à [48, 72]. En prenant en compte la taille des systèmes à résoudre et en raison de la nature creuse des matrices, la méthode itérative du gradient conjugué et ses variantes préconditionnées sont très efficaces et plus précises que des méthodes directes. En effet, l'utilisation de méthodes itératives ne nécessite pas de modification des matrices ni de stockage imporant de ces dernières et comporte peu d'opérations arithmétiques lorsque les matrices sont creuses ; ce qui limite considérablement la propagation des erreurs d'arrondi, particulièrement lorsque les matrices sont mal conditionnées, comme c'est le cas pour celles intervenant dans les pas correcteurs de pression de la méthode PISO. Ainsi, on évite de dénaturer la solution calculée. Par contre, concernant l'utilisation de méthodes directes, si la matrice a une structure bande, de largeur $(2L+1)$, la méthode de Gauss nécessite de l'ordre de ML^2 opérations arithmétiques (M étant la dimension de la matrice) ce qui correspond à un coût de calcul important. Des pertes de précision importantes peuvent alors survenir et ce d'autant plus si la matrice est mal conditionnée. Par ailleurs, l'utilisation de la méthode de Gauss ne conserve pas le caractère creux des matrices dans la mesure où il se produit un phénomène de remplissage (*fill-in*) ce qui augmente le nombre d'opérations arithmétiques.

Afin d'accélérer la convergence de la méthode et en relation avec la propriété de M-matrices, des techniques de préconditionnement peuvent être utilisées. Parmi les préconditonneurs classiques, la factorisation incomplète LU, dérivée de la factorisation gaus-

sienne, a été démontrée comme étant l'une des plus efficaces. D'après un résultat de [72], nous savons que si la matrice provenant du système linéaire est une M-matrice, la factorisation incomplète est réalisable et, de plus, la méthode du gradient conjugué préconditionné par une factorisation incomplète fonctionne très bien. Notons aussi que, pour une M-matrice A quelconque, une propriété caractéristique est que la matrice bloc diagonal de $A_{l,l}$ est aussi une M-matrice [48, 61, 72], et par conséquent, le préconditionnement par une factorisation diagonale incomplète peut être appliqué avec succès. Dans ces conditions, la résolution des sous-systèmes (4.6) par des méthodes de type gradient conjugué est possible dans la mesure où la méthode DICPCG est utilisée pour la structure et où la méthode DILUPBiCG est utilisée pour résoudre les équations de Navier-Stokes. Par conséquent, pour la résolution des sous-problèmes (4.6) on peut aussi, dans l'itération externe SISC ou SIAC définie dans le paragraphe 4.4.1, utiliser une itération interne constituée des deux factorisations incomplètes précédentes.

Enfin, au lieu d'utiliser la méthode du gradient conjugué pour résoudre l'équation de correction de pression qui est mal conditionnée, il est préférable d'utiliser la méthode multi-grille géométrique algébrique (*GAMG : Geometric Algebraic Multi-Grid*) car cette méthode est plus efficace [58]. Cette approche peut être utilisée pour développer des solveurs linéaires très efficaces et robustes pour des maillages fortement anisotropes et/ou pour résoudre des problèmes avec d'importants changements des coefficients dans leurs équations. Pour plus de détails, le lecteur est renvoyé à [58].

4.5.3 Couplage

4.5.3.1 Forces à l'interface

Le couplage entre les solveurs fluide et structure s'effectue par un transfert de variables à chaque pas de temps, ce qui constitue un couplage fort. La résolution des équations de Navier-Stokes donne les vitesses et la pression du fluide à chaque nœud du maillage du fluide. Elles sont alors utilisées pour calculer les efforts qui s'appliquent sur la surface de la structure. Ces efforts sont alors utilisés comme conditions aux limites dans le solveur structure. Deux forces s'exercent sur l'interface fluide-structure :

- la **pression** due au fluide s'exerce selon la direction normale à la paroi. Elle est directement disponible après la résolution des équations régissant l'écoulement fluide, les valeurs de la pression à la surface du volume de contrôle qui constitue l'interface fluide-structure sont directement transférées entre le solveur fluide et le solveur solide ;

- la **contrainte de cisaillement** agit dans la direction tangentielle à la paroi. La condition de non glissement à la paroi impose que la vitesse relative du fluide par rapport à la frontière soit nulle. Cependant, à un certaine distance de la paroi, la vitesse de l'écoulement doit être égale à celle du reste du fluide ; la zone entre ces deux points est appelée la *couche limite*. Pour un fluide newtonien, la contrainte de cisaillement est proportionnelle au taux de déformation dans le fluide et la viscosité dynamique μ^f est la constante de proportionnalité. La contrainte de cisaillement

est transmise à la surface de la paroi en raison de la perte de vitesse. La contrainte de cisaillement à la paroi τ_{wall} est donnée par la dérivée normale de la vitesse tangente à la paroi :

$$\tau_{wall} = \mu^f \left. \frac{\partial u_t}{\partial x_n} \right|_{wall}, \tag{4.8}$$

où u_t est la vitesse tangente à la paroi et x_n est la coordonnée normale à la paroi.

4.5.3.2 Adaptation du maillage

La résolution de l'équation d'élasticité de Navier donne le vecteur déplacement **D** à chaque nœud du maillage de la structure et par conséquent, ces déplacements peuvent être ajoutés aux coordonnées des nœuds afin de déterminer la géométrie de la structure au prochain pas de temps. Afin que le solveur fluide puisse prendre en compte cette déformation, son sous-domaine doit être adapté. Pour cela, le maillage est alors considéré comme un corps déformable.

Une équation du laplacien de la vitesse de grille $\mathbf{u}^*$ donnée par :

$$\nabla^2 \left(\Gamma_f \mathbf{u}^* \right) = 0, \tag{4.9}$$

où Γ_f est un coefficient de diffusion, est alors résolue sur le sous-domaine fluide avec les déplacements calculés à l'interface fluide-structure comme conditions aux limites. Des conditions de déplacement nul sont appliquées aux autres frontières du sous-domaine. Les propriétés physiques du pseudo-matériau constituant le maillage sont en général proches de celles de la structure. Le déplacement calculé dans le sous-domaine fluide est alors ajouté aux coordonnées des nœuds afin de produire le maillage déformé. Le maillage est donc déformé au cours de la simulation fluide-structure tout en gardant une distribution des nœuds identique.

La figure 4.11 représente schématiquement le couplage fluide-structure avec les différentes valeurs échangées entre les sous-systèmes physiques.

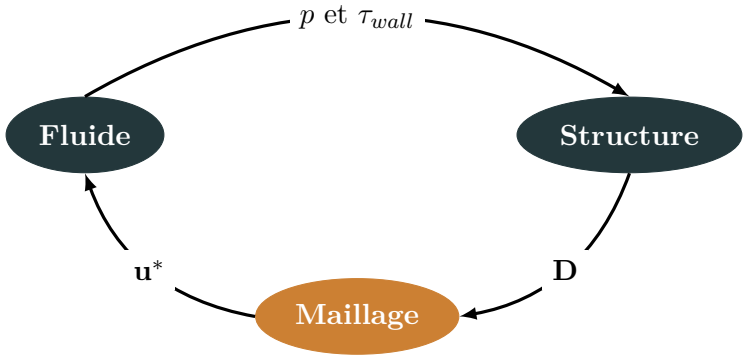


FIGURE 4.11 – Représentation du couplage fluide-structure

La figure 4.12 illustre la procédure FSI en parallèle. Les étapes de résolution des équations de Navier-Stokes, de l'équation de Navier ainsi que celle de résolution de

l'équation de type laplacien pour la déformation du maillage sont effectuées en parallèle sur un nombre P de processeurs.

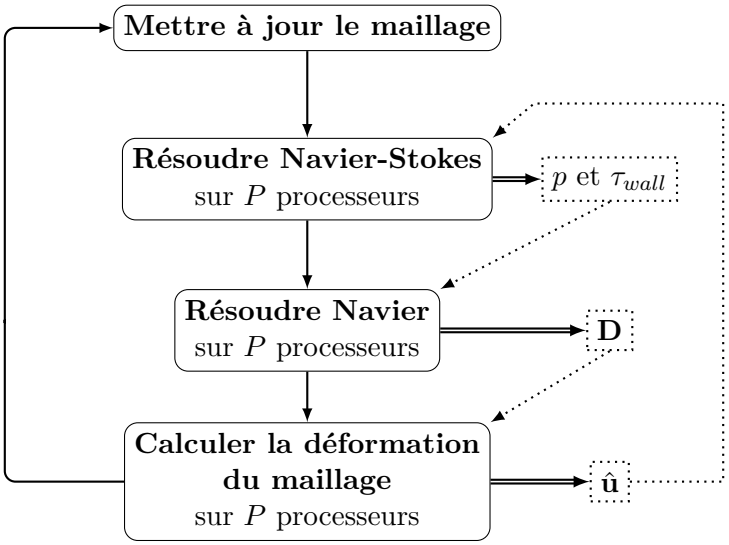


FIGURE 4.12 – Procédure FSI parallèle

4.5.4 Implémentation dans OpenFOAM

OpenFOAM (i.e. *Open Field Operation And Manipulation*) est un code de calcul CFD (i.e. *Computational Fluid Dynamic*) qui permet d'utiliser de larges ressources de calcul en n'imposant aucune limite sur la décomposition mise en œuvre. Son code source est écrit en langage C++ ce qui permet une très grande modularité d'utilisation. De plus, son code est *Open Source* c'est-à-dire qu'il permet un accès au code source afin de modifier les fonctionnalités existantes mais aussi de créer ou d'implanter des fonctionnalités complémentaires. Cette capacité conduit à une certaine difficulté dans la prise en main du code mais permet une liberté de développement en fonction des besoins. Plusieurs fork (nouveau logiciel créé à partir du code source) d'OpenFOAM ont ainsi vu le jour et la version la plus adaptée a été utilisée pour le traitement de nos simulations sur des processeurs distribués avec les méthodes SISC et SIAC.

La méthode de calcul parallèle utilisée par OpenFOAM est une décomposition de domaine associée à un partitionnement en grands blocs des matrices, dans laquelle les blocs associés sont partagés et répartis sur les différents processeurs. Le processus de calcul parallèle implique donc la décomposition du maillage et des domaines.

La version parallèle utilisée est basée sur Open MPI une bibliothèque de fonctions basées sur MPI (i.e. *Message Passing Interface*), utilisable avec les langages C, C++ et Fortran et qui permet d'utiliser des processeurs distants au moyen de passage de messages quelle que soit la technologie réseau (Ethernet Gigabit, InfiniBand, etc.).

Pour communiquer et passer des messages, Open MPI utilise des primitives d'envoi et de réception. Ces primitives sont bloquantes si l'espace mémoire servant à la commu-

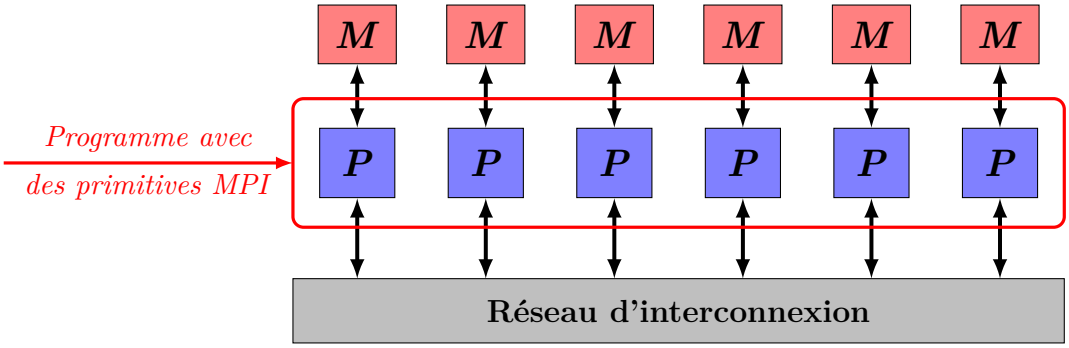


FIGURE 4.13 – Exemple de fonctionnement MPI (architecture SIMD)

nication peut être réutilisé immédiatement à la fin de son utilisation et non bloquantes si elles rendent la main immédiatement à la fin de leur utilisation. Dans ce cas, pour réutiliser l'espace mémoire servant à la communication, il faut finaliser la communication. Les communications non bloquantes laissent le programme poursuivre son exécution, que la communication soit réellement réalisée ou non et présentent un avantage indéniable au niveau du temps d'exécution puisqu'un processeur peut travailler tout en envoyant des données.

Les primitives bloquantes sont *MPI_Send* et *MPI_Recv* et les non bloquantes sont *MPI_Isend* et *MPI_Irecv*.

OpenFOAM utilise Open MPI d'une manière classique : initialisation de l'environnement MPI, échange de messages bloquants ou non bloquants pendant l'exécution et finalisation de l'environnement MPI (libéralisation des ressources). Toutes les applications OpenFOAM utilisent une des librairies C++ du code source : *Pstream*.

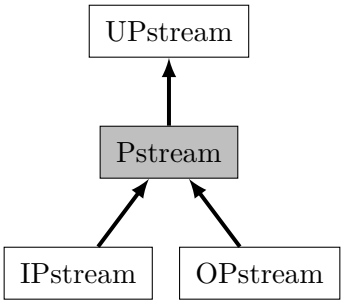


FIGURE 4.14 – Librairie *Pstream* d'OpenFOAM

Cette librairie s'appuie sur une abstraction des primitives d'envoi et de réception représentées par les méthodes C++ read et write contenues dans *IPstream* et *OPstream*. Dans les grandes lignes, les méthodes `IPstream::read` et `OPstream::write` consistent à recevoir et envoyer un message et utilisent les codes (version simplifiée par rapport à la version originale) ci-dessous :

```
1 IPstream::read
```

```
  if (commsType == blocking)
    {
      MPI_Status status;
      MPI_Recv
        (
          buf,
          bufSize,
          MPI_BYTE,
          fromProcNo,
          tag,
          PstreamGlobals::MPICommunicators_[comm],
          &status
        )
      return 0;
    }
  else if (commsType == nonBlocking)
    {
      MPI_Request request;
      MPI_Irecv
        (
          buf,
          bufSize,
          MPI_BYTE,
          fromProcNo,
          tag,
          PstreamGlobals::MPICommunicators_[comm],
          &request
        )
      return 0;
    }
```

```
OPstream::write
  if (commsType == blocking)
    {
      MPI_Send
        (
          buf,
          bufSize,
          MPI_BYTE,
          toProcNo,
          tag,
          PstreamGlobals::MPICommunicators_[comm],
          &status
```

```
        )
      return 0;
    }
  else if (commsType == nonBlocking)
    {
      MPI_Request request;
      MPI_Isend
        (
          buf,
          bufSize,
          MPI_BYTE,
          toProcNo,
          tag,
          PstreamGlobals::MPICommunicators_[comm],
          &request
        )
      return 0;
    }
```

L'implémentation des méthodes à itérations et communications synchrones (SISC) n'a pas posé de problème. En revanche, celle des méthodes avec itérations synchrones mais communications asynchrones (SIAC) où les réceptions sont non bloquantes a posé une réelle difficulté qui a finalement pu être surmontée.

En effet, une erreur irrécupérable survenait lors de l'exécution en non bloquant. C'est par une analyse fine du code source d'OpenFOAM qu'il a été mis en évidence que la mise à jour du maillage (cf. figure 4.12) provoquait cette erreur. En effet, chaque processeur P effectue cette mise à jour qui requiert la terminaison des envois et réceptions non bloquants de ses voisins afin d'avoir accès à toutes les données nécessaires. L'implémentation de la méthode « *mesh update* » du code parallèle appelait une méthode qui était incorrectement implémentée en non bloquant. Plus précisément, cette méthode était programmée dans le fichier `ProcessorPointPatchField.C` qui permet l'interpolation de point.

Le correctif ci-dessous a permis de résoudre ce problème en permettant aux communications non bloquantes de se terminer avant d'effectuer l'interpolation de point.

```
ProcessorPointPatchField
<PatchField, Mesh, PointPatch, ProcessorPointPatch, MatrixType, Type>::
receivePointField
  (
    const Pstream::commsTypes commsType
  ) const
  {
    tmp<Field<Type2> > tf(new Field<Type2>(this->size()));
```

```
9
10      label startOfRequests = Pstream::nRequests();
11      // récupération des messages envoyés par les voisins
12
13      IPstream::read
14        (
15          commsType,
16          procPatch_.neighbProcNo(),
17          reinterpret_cast<char*>(tf().begin()),
18          tf().byteSize()
19        );
20
21      Pstream::waitRequests(startOfRequests);
22      // accusé de réception des messages envoyés par les voisins
23
24      return tf;
25    }
```

Les messages non bloquants sont reçus dans un ordre chaotique car chaque processeur travaille à son propre rythme sans aucune synchronisation entre eux. Dans la configuration originale du code, quel que soit le mode de communication, la mise à jour du maillage était lancée avant que tous les processeurs aient terminé leurs processus de réception des messages de leurs voisins ce qui provoquait une erreur irrécupérable. En effet, les données échangées par les voisins doivent être utilisées pour la mise à jour du maillage. La solution adoptée pour les communications non bloquantes a donc été d'utiliser la méthode `Pstream::waitRequests` qui permet d'attendre la terminaison d'une ou plusieurs émissions ou réceptions non bloquantes entre les voisins d'un processeur avant de poursuivre son exécution. La figure 4.15 présente le principe général de fonctionnement de cette solution. Par contre, dans une configuration bloquante, les processeurs se synchronisant le problème ne se posait pas.

4.5.5 Récapitulatif

Compte-tenu des développements précédents, l'algorithme à implémenter, et conforme à (3.168), peut se résumer comme suit. D'une part en ce qui concerne le schéma de principe, on a l'algorithme 5 et, d'autre part, en ce qui concerne l'implémentation parallèle de la méthode, on a l'algorithme 6.

Références

[1] G. M. Amdahl. "Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities". In : *Proceedings of the April 18-20, 1967, Spring Joint Computer Conference*. New York, USA : ACM, 1967, p. 483–485.

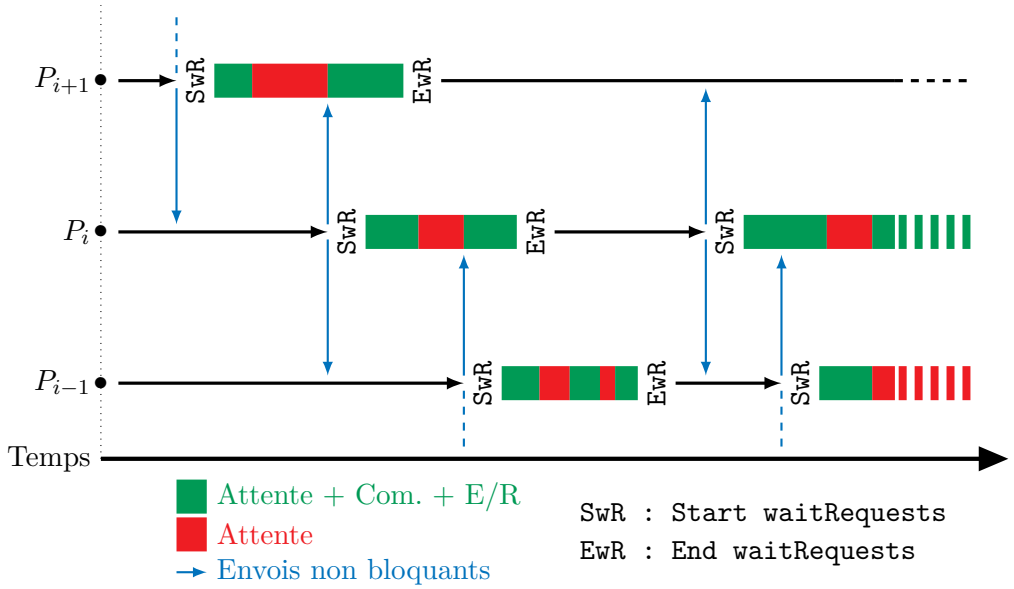


FIGURE 4.15 – Principe général de fonctionnement de la méthode `waitRequests`

Algorithme 5 Schéma de principe du multi-splitting

1: **pour** $r = 0, 1, \ldots$ jusqu'à convergence **faire**
2: **pour** $l = 1$ à m **faire**
3: $M^l \tilde{X}^l = N^l \tilde{Y}^r + \tilde{B}$
4: $X^{r+1} = \sum_{l=1}^{m} \tilde{W}_l \tilde{X}^l$
5: **fin**
6: **fin**

[2] N. ANTONOPOULOS et L. GILLAM. *Cloud Computing : Principles, Systems and Applications*. Computer Communications and Networks. Berlin, Heidelberg : Springer, 2010.

[4] J. BAHI, S. CONTASSOT-VIVIER et R. COUTURIER. *Parallel Iterative Algorithms : From Sequential to Grid Computing*. Numerical Analysis and Scientific Computing. Chapman & Hall/CRC, 2007.

[9] G. BAUDET. "Asynchronous Iterative Methods for Multiprocessors". In : *Journal of the Association for Computing Machinery* 25 (1978), p. 226–244.

[11] D. P. BERTSEKAS et J. N. TSITSIKLIS. "Some Aspects of Parallel and Distributed Iterative Algorithms – A Survey". In : *Automatica* 27 (1991), p. 3–21.

[12] R. BOLZE, F. CAPPELLO, E. CARON, M. DAYDE, F. DESPREZ, E. JEANNOT, Y. JÉGOU, S. LANTERI, J. LEDUC, N. MELAB, G. MORNET, R. NAMYST, P. PRIMET, B. QUÉTIER, O. RICHARD, T. EL-GHAZALI et I. TOUCHE. "Grid'5000 : A Large

Algorithme 6 Multi-splitting parallèle

pour $r = 0, 1, \ldots$ jusqu'à convergence **faire**
  décomposition du problème en α blocs
  pour $k = 1$ à α **faire**
    résolution des sous-problèmes (par exemple par gradient conjugué)
    pour $l = 1$ à m **faire**

$$\begin{cases} \left[M^l \tilde{X}^{l,r+1}\right]_k = \left[N^l \tilde{Y}^r + \tilde{B}\right]_k & \text{si } k \in s(r) \\ \tilde{X}_k^{l,r+1} = \tilde{X}_k^{l,r} & \text{si } k \notin s(r) \end{cases}$$

$$X_k^{r+1} = \sum_{l=1}^{m} \left[\tilde{W}_l\right]_k \tilde{X}_k^{l,r+1}$$

    fin
  fin
fin

Scale And Highly Reconfigurable Experimental Grid Testbed". In : *International Journal of High Performance Computing Applications* 20 (2006), p. 481–494.

[17] R. Buyya, J. Broberg et A. M. Goscinski. *Cloud Computing : Principles and Paradigms*. Wiley, 2011.

[18] M. Chau. "Algorithmes Parallèles Asynchrones pour la Simulation Numérique". Thèse de doct. Institut National Polytechnique de Toulouse, 2005.

[19] M. Chau, T. Garcia et P. Spiteri. "Asynchronous Schwarz Methods Applied to Constrained Mechanical Structures in Grid Environment". In : *Advances in Engineering Software* 74 (2014), p. 1–15.

[21] *Cloud computing, Wikipedia*. `www.wikipedia.org/wiki/Cloud_computing`.

[23] R. Couturier. *Designing Scientific Applications on GPUs*. Numerical Analysis and Scientific Computing. Chapman & Hall/CRC, 2013.

[24] R. Couturier, C. Denis et F. Jézéquel. "GREMLINS : a Large Sparse Linear Solver for Grid Environment". In : *Parallel Computing* 34 (2008), p. 380–391.

[25] R. Couturier et L. Ziane Khodja. "A Scalable Multisplitting Algorithm to Solve Large Sparse Linear Systems". In : *The Journal of Supercomputing* 71 (2015), p. 1345–1356.

[30] M. J. Flynn. "Very High-Speed Computing Systems". In : *Proceedings of the IEEE 54*. 1966, p. 1901–1909.

[34] M. Gengler, S. Ubéda et F. Desprez. *Initiation au Parallélisme : Concept, Architectures et Algorithmes*. Masson, 1996.

[37] *Grid'5000*. `www.grid5000.fr`.

[40] R. W. Hockney et C. R. Jesshope. *Parallel Computers : Architecture, Programming and Algorithms*. Adam Hilger Ltd, 1981.

[43] F. Jézéquel, R. Couturier et C. Denis. “Solving Large Sparse Linear Systems in a Grid Environment : the GREMLINS Code Versus the PETSc Library”. In : *The Journal of Supercomputing* 59 (2012), p. 1517–1532.

[45] F. Magoulès et G. Gbikpi-Benissan. “Distributed Computation of Convergence Residual Under Asynchronous Iterations”. In : *IEEE Transactions on Parallel and Distributed Systems* (to be published).

[47] F. Magoulès, F.-X. Roux et G. Houzeaux. *Parallel Scientific Computing*. Wiley, 2015.

[48] G. Meurant. *Computer Solution of Large Linear Systems*. Studies in Mathematics and its Applications. North Holland, 1999.

[56] J.-C. Miellou et P. Spiteri. “Stopping Criteria for Parallel Asynchronous Iterations for Fixed Point Methods”. In : *Developments in Parallel, Distributed, Grid and Cloud Computing for Engineering*. Sous la dir. de B. H. V. Topping et P. Iványi. Stirlingshire, UK : Saxe-Coburg Publications, 2013. Chap. 12, p. 277–308.

[57] J.-C. Miellou, P. Spiteri et D. El Baz. “A New Stopping Criterion for Linear Perturbed Asynchronous Iterations”. In : *Journal of Computational and Applied Mathematics* 219 (2008), p. 471–483.

[58] F. Moukalled, L. Mangani et M. Darwish. *The Finite Volume Method in Computational Fluid Dynamics*. Fluid Mechanics and its Applications. Berlin, Heidelberg : Springer, 2015.

[59] *National Institute of Standards and Technology (NIST)*. `www.nist.gov`.

[61] J. M. Ortega et W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Computer Science and Applied Mathematics. New-York : Academic Press, 1970.

[72] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Other Titles in Applied Mathematics. SIAM, 2003.

[73] S. A. Savari et D. P. Bertsekas. “Finite Termination of Asynchronous Iterative Algorithms”. In : *Parallel Computing* 22 (1996), p. 39–56.

[78] P. Spiteri. “Finite Precision Computation for Linear Fixed Point Methods of Parallel Asynchronous Iterations”. In : *Techniques for Parallel, Distributed and Cloud Computing in Engineering*. Sous la dir. de P. Iványi et B. H. V. Topping. Stirlingshire, UK : Saxe-Coburg Publications, 2015. Chap. 8, p. 163–196.

[81] D. Tromeur-Dervout, G. Brenner, D. R. Emerson et J. Erhel, éds. *Parallel Computational Fluid Dynamics 2008. Parallel Numerical Methods, Software Development and Applications*. Computational Science and Engineering. Berlin, Heidelberg : Springer, 2008.

[86] L. Ziane Khodja. “Résolution de systèmes linéaires et non linéaires creux sur grappes de GPUs”. Thèse de doct. Université de Franche-Comté, 2013.

[87] L. Ziane Khodja, M. Chau, R. Couturier, J. Bahi et P. Spiteri. “Parallel Solution of American Option Derivatives on GPU Clusters”. In : *Computers & Mathematics with Applications* 65 (2013), p. 1830–1848.

CHAPITRE

5

MISE EN ŒUVRE, APPLICATIONS

Sommaire

5.1 Introduction

On a mis en œuvre, sur des cas tests ainsi que sur un cas de type industriel, la parallélisation de la résolution de problèmes d'interaction fluide-structure. On a essentiellement utilisé un cluster local comportant un processeur Intel Xeon E5-2630 v4 @ 2.2 GHz de 10 cœurs et on également effectué quelques tests sur Grid'5000 où l'exploitation s'est avérée problématique compte tenu des difficultés d'accès aux ressources.

Lors du choix des méthodes numériques, on a systématiquement utilisé des schémas décentrés (afin de limiter les phénomènes de couche limite [69]). On a pu constater que le mode de calcul séquentiel nécessitait des temps de restitution très important. Par conséquent, la parallélisation des méthodes de calcul présente un grand intérêt.

Dans le paragraphe 5.2, nous présentons une première application simple dont le but est de tester les deux méthodes de parallélisation. Dans le paragraphe 5.3, on s'intéresse à une première application d'interaction fluide-structure constituée par un mur plongé dans un canal et qui subit les forces d'un courant. Dans le paragraphe 5.4, une deuxième application fluide-structure est présentée ; il s'agit d'un tube rempli d'eau dans lequel une onde de pression se propage. Enfin, dans le paragraphe 5.5, nous présentons une dernière application constituée par un conteneur de train de marchandise soumis à un vent de travers.

5.2 Cas test : résolution de l'équation de la chaleur

5.2.1 Description de l'application

Afin de tester les méthodes de parallélisation SISC et SIAC, on les applique dans un premier temps à la résolution d'une simple équation de la chaleur instationnaire [62]. Ce cas test consiste donc à résoudre l'équation suivante sur une domaine carré $\Omega = [0,1] \times [0,1]$:

$$\begin{cases} \dfrac{\partial T}{\partial t} - \nu \nabla^2 T = 0, \\ T(0,x,y) = 273\,\mathrm{K}, \\ T|_{\partial\Omega} = 573\,\mathrm{K}, \end{cases} \tag{5.1}$$

où T est la température, ν est le coefficient de diffusion thermique et $\partial\Omega$ est la frontière du domaine Ω.

5.2.2 Paramètres numériques et physiques

Le domaine est discrétisé par la méthode des volumes finis afin d'obtenir un maillage constitué de 25 millions de volumes de contrôle. La simulation dure 0.5 secondes avec un pas de temps de 0.01 seconde. On considère une méthode de Gauss-Seidel pour résoudre le système linéaire.

Les calculs parallèles sont menés sur un cluster local constitué d'un processeur Intel Xeon E5-2630 v4 @ 2.2 GHz sur un nombre de cœurs allant de 2 à 8 cœurs en utilisant les méthodes SISC et SIAC. La décomposition du domaine est réalisée en utilisant une méthode de Scotch [75], la figure 5.1 représente différentes décompositions.

(a) 5 sous-domaines (b) 7 sous-domaines (c) 8 sous-domaines

FIGURE 5.1 – Cas test - Décompositions du domaine

5.2.3 Résultats numériques

Les temps de restitution ainsi que les accélérations et les efficacités pour les deux méthodes sont donnés dans la table 5.1. La figure 5.2 représente les temps de restitution en fonction du nombre de cœurs utilisés, la figure 5.3 représente l'accélération correspondante et la figure 5.4 représente l'efficacité.

On peut noter que les résultats d'accélération et d'efficacité sont satisfaisants, on remarque en particulier que pour la méthode SISC, le poids des communications synchrones pénalise les performances. Cependant, la méthode SIAC réduit les temps d'inactivité inhérents aux méthodes à communications synchrones et les temps de restitution diminuent d'environ 4 % sur 2 cœurs jusqu'à 45.6 % sur 4 cœurs. Des calculs parallèles ont aussi été menés pour le même problème mais sur un maillage ne comportant qu'un million de volumes de contrôle et, lorsqu'on compare les résultats obtenus à la table 5.1 et à la figure 5.2, on note que l'augmentation du nombre de volumes de contrôle améliore les performances des solveurs parallèles.

5.3 Mur dans un courant

5.3.1 Description de l'application

La première application fluide-structure consiste en un simple mur soumis à un écoulement fluide dans une conduite (cf. figure 5.5). Le mur fait 20 cm de haut, 40 cm de long et 10 cm d'épaisseur. Il est encastré à sa base dans une conduite de dimensions deux fois supérieures à celles du mur, soit : 40 cm de haut et 80 cm de large. La conduite est

Nb. de cœurs	SISC	Accélération	Efficacité	SIAC	Accélération	Efficacité
1	20 543			20 543		
2	10 845	1.89	0.95	10 412	1.97	0.99
3	11 593	1.77	0.59	7546	2.72	0.91
4	11 192	1.84	0.46	6089	3.37	0.84
5	9128	2.25	0.45	5300	3.88	0.78
6	8656	2.37	0.40	4655	4.41	0.74
7	6872	2.99	0.43	4462	4.60	0.66
8	7244	2.84	0.35	4303	4.77	0.60

TABLE 5.1 – Cas test - Temps de restitution (s), accélérations et efficacités pour les méthodes SISC et SIAC sur le cluster local

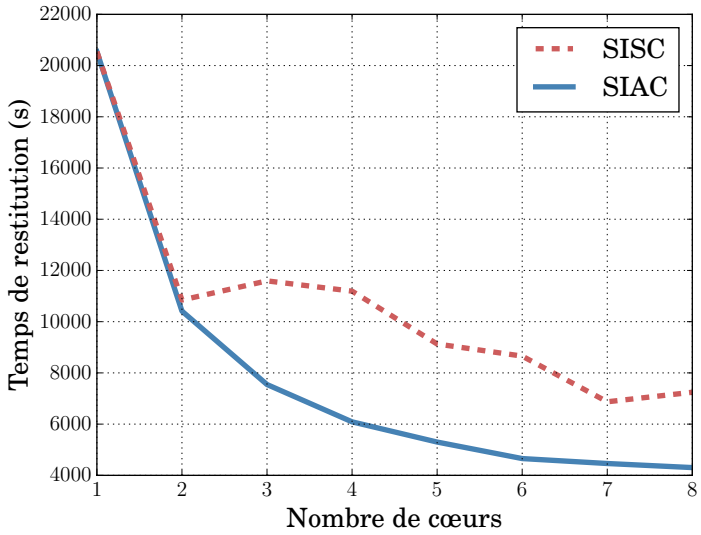


FIGURE 5.2 – Cas test - Temps de restitution (s) sur le cluster local

remplie d'eau et le courant est dirigé dans le sens perpendiculaire à la largeur du mur. C'est un cas test d'interaction fluide-structure présenté en séquentiel dans [65], mais sa géométrie simple permet d'aisément raffiner le maillage afin d'obtenir de grandes matrices de discrétisation et de tester les performances du solveur fluide-structure parallèle.

5.3.2 Paramètres numériques et physiques

Les essais ont été réalisés sur un cluster local ainsi que sur un cluster de la plateforme Grid'5000 (cf. paragraphe 4.2.3.2) sur le site de Nancy. Le cluster local possède 10 cœurs et le cluster situé à Nancy possède 51 nœuds de calcul ; leurs caractéristiques sont présentées dans la table 5.2.

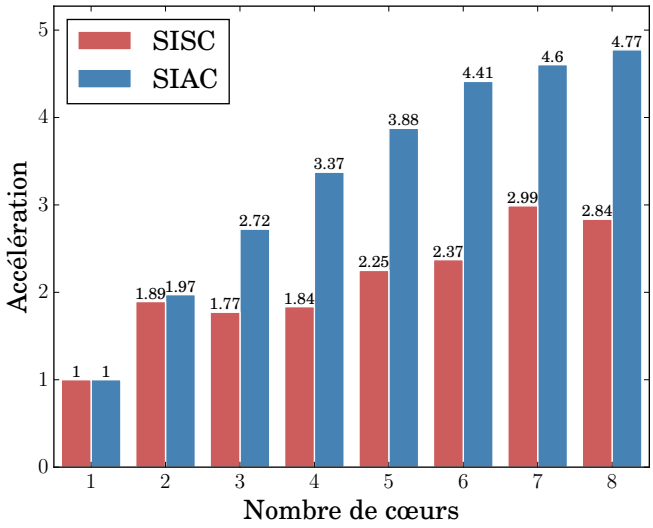


FIGURE 5.3 – Cas test - Accélération sur le cluster local

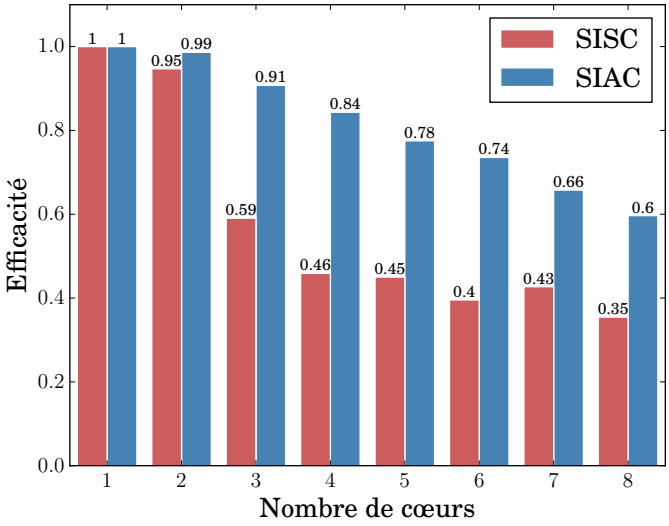


FIGURE 5.4 – Cas test - Efficacité sur le cluster local

Cluster	CPUs	Cœurs	RAM
Local	Intel Xeon E5-2630 v4	10 cœurs	64GB
Nancy	2x Intel Xeon E5-2630 v3	8 cœurs/CPU	126GB

TABLE 5.2 – Mur - Caractéristiques des clusters

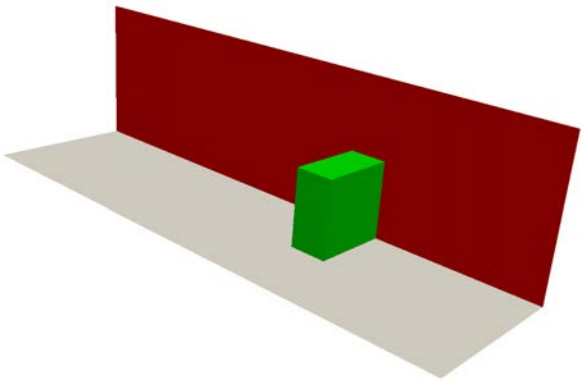


FIGURE 5.5 – Mur (en vert) et plan de symétrie (en rouge)

Les paramètres physiques et numériques du fluide et de la structure sont résumés dans la table 5.3. Dans le but de satisfaire la condition de Courant - Friedrichs - Lewy : $CFL = \Delta t \sum_{i=1}^{3}(u_i/\Delta x_i) < CFL_{max}$, nous avons choisi de prendre une pas de temps de 0.005 s.

Paramètres du fluide

Nombre de CVs	ν^f ($m^2\,s^{-1}$)	ρ^f ($kg\,m^{-3}$)	U_{max} ($m\,s^{-1}$)
3 151 872	0.001	1000	0.2

Paramètres de la structure

Nombre de CVs	ν	ρ^s ($kg\,m^{-3}$)	E (Pa)
55 296	0.4	1000	10 000

Paramètres de calcul

Itérations FSI max.	Temps final (s)	Δt (s)
20	0.4	0.005

TABLE 5.3 – Mur - Paramètres physiques et numériques

5.3.3 Résultats numériques

Les temps de restitution des deux méthodes SISC et SIAC des simulations parallèles sur le cluster local ainsi que les accélérations et les efficacités correspondantes, pour 1 à 8 cœurs utilisés, sont présentés dans la table 5.4. Les résultats pour le cluster situé à Nancy, en utilisant 8, 10, 12, 14 et 16 nœuds sont présentés dans la table 5.5. Notons qu'à cause de certaines contraintes d'utilisation de Grid'5000, nous n'avons pas eu l'opportunité de réaliser la simulation en séquentiel sur de cluster de Nancy ; notons aussi que les essais réalisés sur cette architecture étaient limités du fait des temps de restitution importants. De plus, des parties du système étaient en maintenance lors des essais, ce qui a rendu le cluster difficile d'accès.

Sur le cluster local, pour cette application fluide-structure, nous obtenons de bonnes efficacités de 2 à 6 cœurs, cependant, au-delà de 6 cœurs, l'efficacité devient moyenne à cause de la taille des systèmes à résoudre. Néanmoins, malgré des efficacités moyennes, la différence entre les temps de restitution pour les méthodes de parallélisation SISC et SIAC est significative.

Sur le cluster de Nancy, les améliorations de la méthode SIAC produisent un gain sur les temps de restitution de 22.5 % sur 8 nœuds. Le gain est plus faible avec plus de nœuds à cause de la taille insuffisante du problème mais aussi car l'architecture de Grid'5000 peut être ralentie pendant certaines périodes. Le problème d'interaction fluide-structure est très dépendant du nombre et de la valeur du pas de temps afin d'obtenir des résultats cohérents. En effet, à chaque pas de temps, il est nécessaire de mettre en place une barrière de synchronisation ainsi que pour le traitement de Navier et à chaque phase de la méthode PISO. Ces barrières permettent d'assurer la cohérence du traitement pénalisant cependant les performances.

Nb. de cœurs	SISC	Accélération	Efficacité	SIAC	Accélération	Efficacité
1	85 440			85 440		
2	47 595	1.80	0.90	46 440	1.84	0.92
3	50 443	1.69	0.56	35 648	2.40	0.80
4	41 115	2.08	0.52	28 826	2.96	0.74
5	35 097	2.43	0.49	25 192	3.39	0.68
6	32 246	2.65	0.44	22 022	3.88	0.65
7	29 800	2.87	0.41	21 005	4.07	0.58
8	27 176	3.14	0.39	20 941	4.08	0.51

TABLE 5.4 – Mur - Temps de restitution (s), accélérations et efficacités pour les méthodes SISC et SIAC sur le cluster local

Nb. de nœuds	SISC	SIAC	Gain
8	27 952	21 648	22.5 %
10	23 324	20 513	12.1 %
12	21 232	19 596	7.7 %
14	21 040	18 062	14.2 %
16	24 240	23 520	3 %

TABLE 5.5 – Mur - Temps de restitution (s) et gain entre les méthodes SISC et SIAC sur le cluster de Nancy

Les figures 5.6, 5.7 et 5.8 représentent respectivement les temps de restitution, les accélérations et les efficacités sur le cluster local pour les méthodes SISC et SIAC.

La figure 5.9 représente les temps de restitution en fonction du nombre de nœuds utilisé sur le cluster situé à Nancy. On note qu'avec 16 nœuds, les temps de restitution sont en réalité plus importants qu'avec 10 nœuds. Ceci est dû à la taille du problème,

en effet, il n'y a pas assez de CVs et trop de communications au regard des temps de calcul.

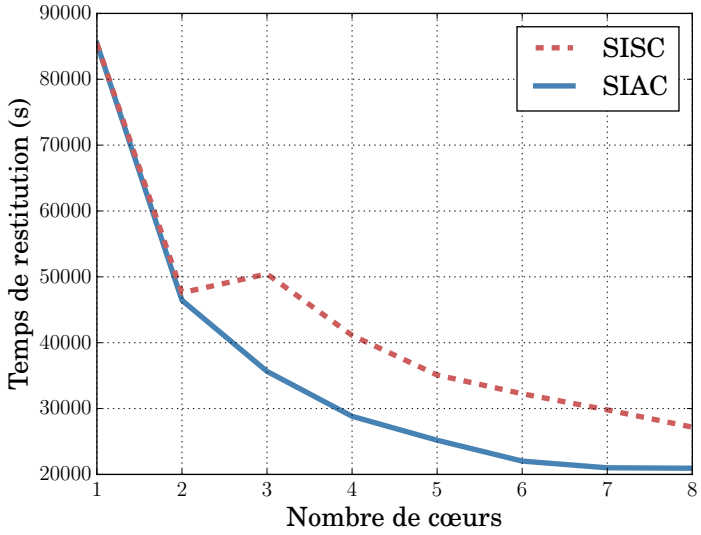


FIGURE 5.6 – Mur - Temps de restitution (s) sur le cluster local

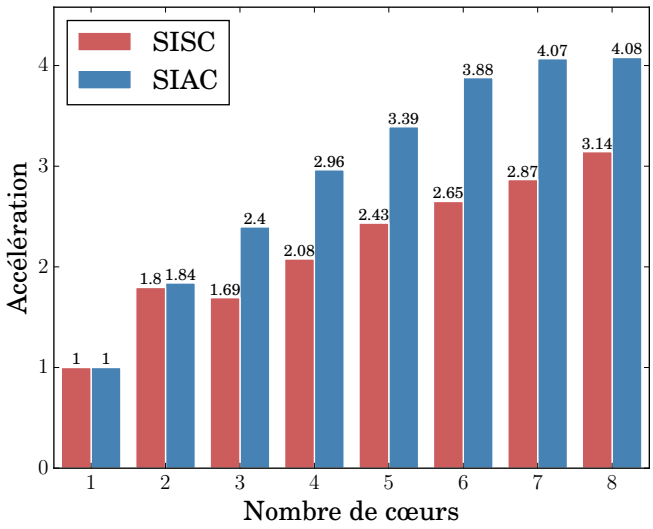


FIGURE 5.7 – Mur - Accélération sur le cluster local

Remarque 22. *Nous avons aussi effectué des essais sur deux sites géographiquement distants de la plateforme Grid'5000 (Nancy et Lyon), mais du fait de la lenteur du*

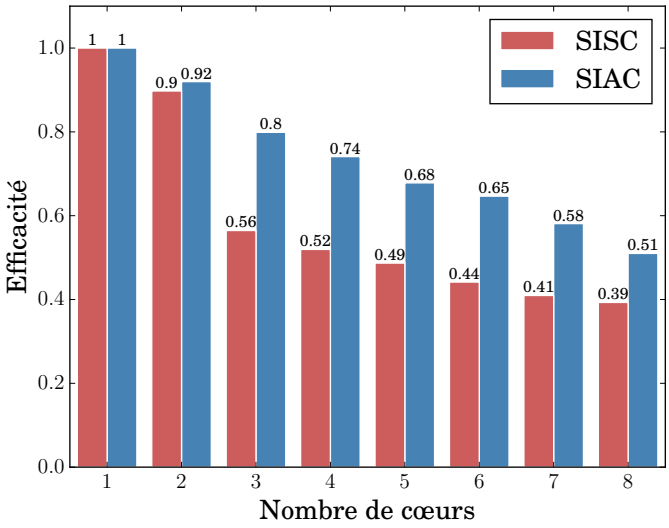


FIGURE 5.8 – Mur - Efficacité sur le cluster local

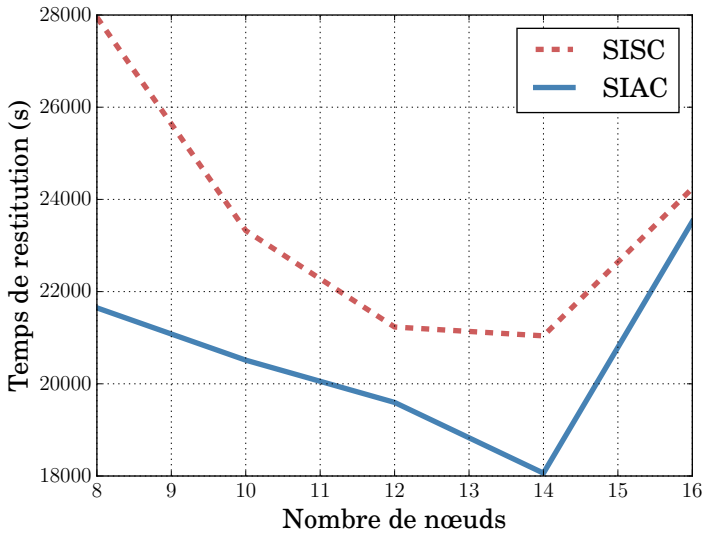


FIGURE 5.9 – Mur - Temps de restitution (s) sur le cluster de Nancy

réseau, les gains de la méthode SIAC (bien qu'existants) ne sont pas assez significatifs pour être présentés.

5.3.4 Résultats physiques

Des simulations sur un maillage plus grossier nous permettent de visualiser la déformation du mur et, par conséquent, du maillage comme on peut le voir sur la figure 5.10.

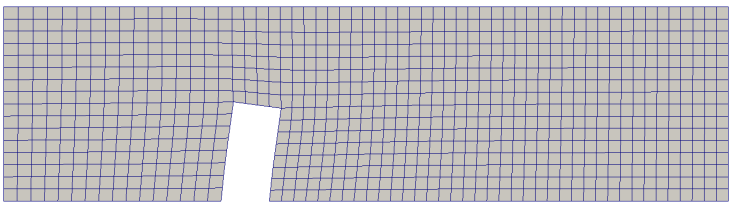


FIGURE 5.10 – Mur - Déformation du maillage à $t = 0.9\,\mathrm{s}$

La figure 5.11 représente le déplacement d'un point au sommet du mur au cours du temps. On peut voir que le déplacement maximum est atteint à $t = 0.9\,\mathrm{s}$ et l'état stationnaire intervient à partir de $t = 4\,\mathrm{s}$.

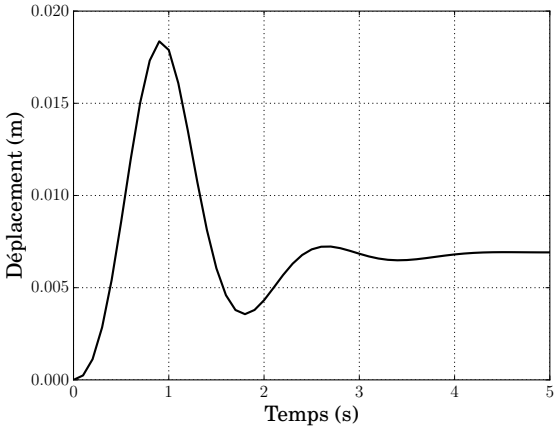


FIGURE 5.11 – Mur - Déplacement au sommet

5.4 Tube en 3D

5.4.1 Description de l'application

Il s'agit ici de simuler le comportement d'un tube rempli d'eau dans lequel se propage une onde de surpression. Le tube, représenté en figure 5.12, fait 5 cm de long, il a un rayon interne de 3.5 mm et un rayon externe de 5 mm et peut, par exemple, représenter une artère ou un vaisseau sanguin. Le tube est encastré à ses deux extrémités.

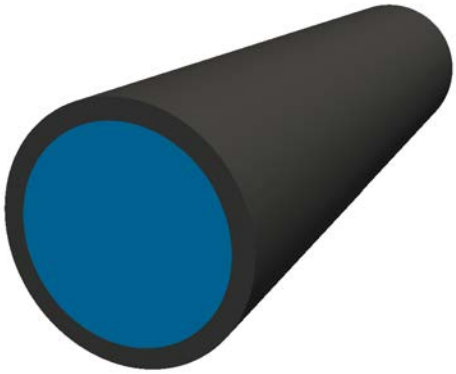


FIGURE 5.12 – Tube (en gris foncé) et fluide (en bleu)

5.4.2 Paramètres physiques et numériques

La table 5.6 résume les différents paramètres physiques et numériques de cette application. Les deux maillages totalisent environ 1.5 millions de CVs (cf. table 5.6), c'est donc un problème moins important que l'application précédente mais on peut noter que la distribution du nombre de CVs entre les deux domaines est plus équilibrée.

La surpression est modélisée par une valeur de pression non nulle en entrée pendant 0.003 s, soit l'équivalent de 30 pas de temps. Cette onde de surpression va se propager vers la sortie du tube où une pression nulle est appliquée.

Paramètres du fluide

Nombre de CVs	ν^f ($\mathrm{m^2\,s^{-1}}$)	ρ^f ($\mathrm{kg\,m^{-3}}$)
1 024 000	3×10^{-6}	1000

Paramètres de la structure

Nombre de CVs	ν	ρ^s ($\mathrm{kg\,m^{-3}}$)	E (Pa)
409 600	0.3	1200	3×10^5

Paramètres de calcul

Itérations FSI max.	Temps final (s)	Δt (s)
30	0.005	1×10^{-4}

TABLE 5.6 – Tube - Paramètres physiques et numériques

5.4.3 Résultats numériques

Les temps de restitution des deux méthodes SISC et SIAC des simulations parallèles sur le cluster local ainsi que les accélérations et les efficacités correspondantes, pour 1 à 8 cœurs utilisés, sont présentés dans la table 5.7.

Les figures 5.13, 5.14 et 5.15 représentent respectivement les temps de restitution, les accélérations et les efficacités sur le cluster local pour les méthodes SISC et SIAC.

Nb. de cœurs	SISC	Accélération	Efficacité	SIAC	Accélération	Efficacité
1	30 282			30 282		
2	18 495	1.64	0.82	17 723	1.71	0.85
3	21 290	1.42	0.47	15 831	1.91	0.64
4	18 131	1.67	0.42	13 668	2.22	0.55
5	15 874	1.91	0.38	12 645	2.39	0.48
6	14 757	2.05	0.34	11 604	2.61	0.43
7	14 123	2.14	0.31	11 063	2.74	0.39
8	14 039	2.16	0.27	11 138	2.72	0.34

TABLE 5.7 – Tube - Temps de restitution (s), accélérations et efficacités pour les méthodes SISC et SIAC sur le cluster local

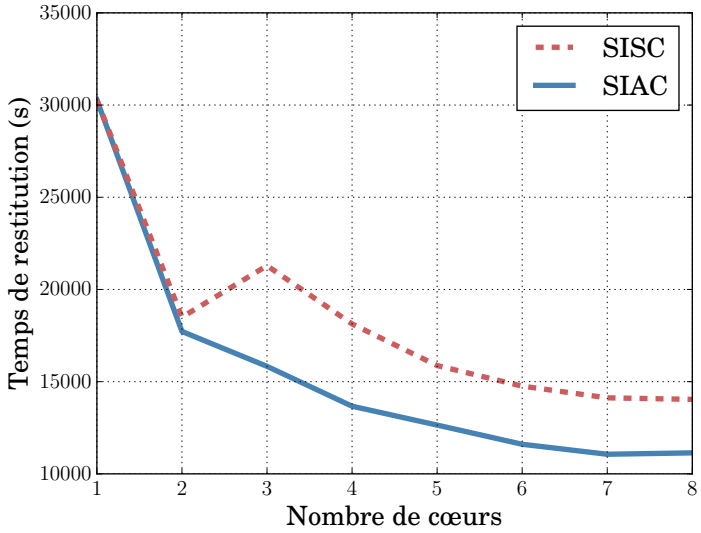


FIGURE 5.13 – Tube - Temps de restitution (s) sur le cluster local

On observe les mêmes tendances que pour les applications précédentes. Cependant, on voit clairement que l'efficacité diminue plus vite que pour l'application du paragraphe 5.3, ceci est dû à la taille moins importante des systèmes à résoudre dans le cas présent.

5.4.4 Résultats physiques

Le résultat intéressant pour ce type d'application est la déformation subie par le tube à cause de la surpression. La figure 5.16 représente donc le déplacement radial dans le tube à différents pas de temps. On observe un déplacement maximal de l'ordre de 0.17 mm.

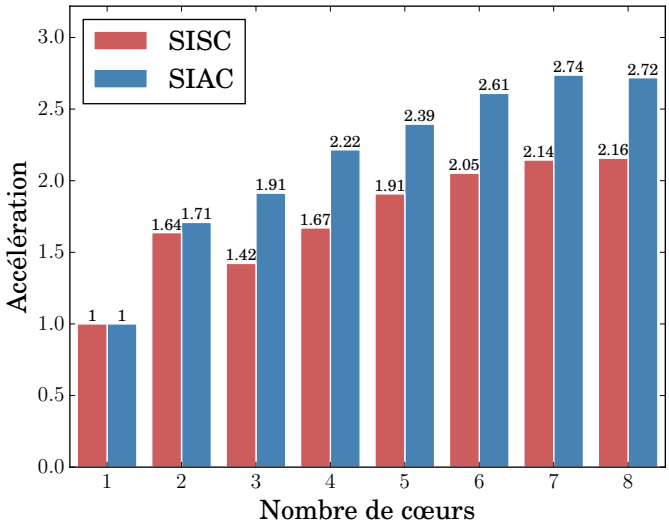


FIGURE 5.14 – Tube - Accélération sur le cluster local

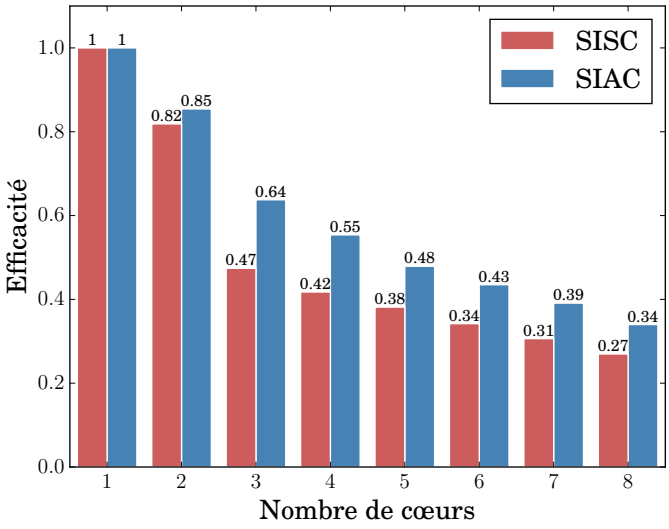


FIGURE 5.15 – Tube - Efficacité sur le cluster local

5.5 Conteneur soumis à un vent de travers

5.5.1 Description de l'application

Dans cette application, on s'intéresse à un conteneur de transport de marchandises soumis à un vent de travers (cf. figure 5.17). Il est de dimensions similaires à un conteneur

(a) $t = 0.001\,\mathrm{s}$

(b) $t = 0.002\,\mathrm{s}$

(c) $t = 0.003\,\mathrm{s}$

(d) $t = 0.004\,\mathrm{s}$

(e) $t = 0.005\,\mathrm{s}$

FIGURE 5.16 – Tube - Déplacement radial du tube (m)

de 20 pieds (cf. figure 5.18), c'est-à-dire 6 m de longueur, 2.5 m de largeur et 2.5 m de hauteur (sans prendre en compte les 4 supports).

Le conteneur repose dans le sous-domaine fluide (cf. figure 5.19) et est soumis à un vent de travers perpendiculaire à la paroi latérale. Cette application en trois dimensions est une extension de l'application bi-dimensionnelle présentée dans [65].

5.5.2 Paramètres numériques et physiques

Le maillage généré pour cette application est plutôt grossier afin d'obtenir des résultats réalistes avec des temps de restitution raisonnables. On s'est donc limité à des vitesse de

(a) Vue de face (b) Vue de côté

FIGURE 5.17 – Conteneur - Modèle 3D

(a) Train de marchandises (b) Conteneur de 20 pieds

FIGURE 5.18 – Conteneur - Un train de marchandises transportant des conteneurs

fluide faible car les volumes de contrôle dans le fluide ne sont pas assez fin pour obtenir de bons résultats à des vitesses plus élevées. Par conséquent, pour pouvoir constater des déformations notables du conteneur, les propriétés physiques prises en compte sont celles d'un matériau plus souple que celui d'un conteneur réel. La figure 5.20 représente le maillage de la structure et la table 5.8 résume les paramètres physiques et numériques de cette application.

Le conteneur est encastré au niveau de ses 4 supports, l'intérieur est vide et donc sans sollicitation et toutes les surfaces restantes (extérieur du conteneur) constituent l'interface fluide-structure et sont soumises aux efforts du fluide.

Le sous-domaine fluide est constitué :

– d'une surface représentant le sol, sur laquelle repose le conteneur, considérée comme une surface solide où des conditions de non glissement et de non pénétration sont appliquées ;

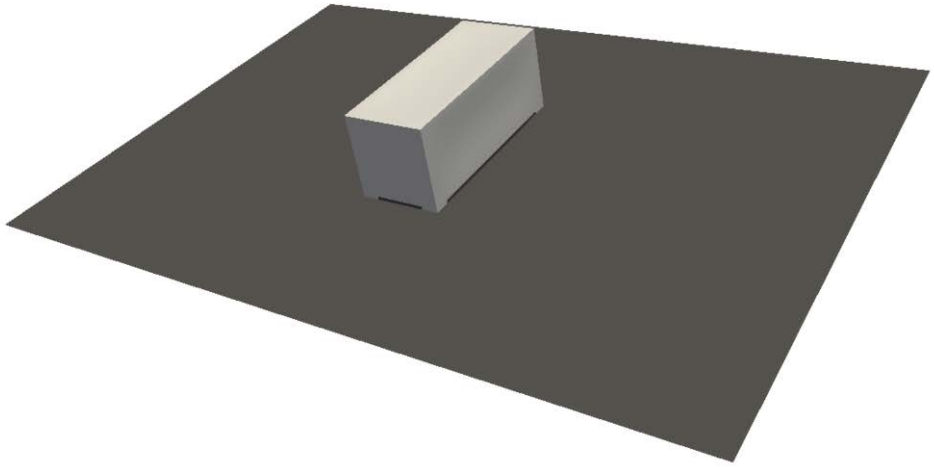


FIGURE 5.19 – Conteneur - Conteneur (gris clair) et sol du domaine fluide (gris foncé)

Paramètres du fluide

Nombre de CVs	ν^f ($\mathrm{m^2\,s^{-1}}$)	ρ^f ($\mathrm{kg\,m^{-3}}$)	U_{max} ($\mathrm{m\,s^{-1}}$)
512 309	15.6×10^{-6}	1.204	0.6

Paramètres de la structure

Nombre de CVs	ν	ρ^s ($\mathrm{kg\,m^{-3}}$)	E (Pa)
439 765	0.4	2000	1×10^{12}

Paramètres de calcul

Itérations FSI max.	Temps final (s)	Δt (s)
10	3	0.01

TABLE 5.8 – Conteneur - Paramètres physiques et numériques

- d'une arrivée d'air, avec une vitesse d'entrée croissante linéaire, ce qui signifie que $U_1 = 0\,\mathrm{m\,s^{-1}}$ à $t = 0\,\mathrm{s}$ et $U_1 = U_{max}$ à $t = t_{max}$;
- d'une sortie ;
- de 3 surfaces latérales ;
- d'une interface avec la structure considérée comme une surface solide avec une vitesse déterminée par le solveur structure.

Les calculs sont réalisés sur le cluster local dont les caractéristiques sont résumées dans la table 5.2.

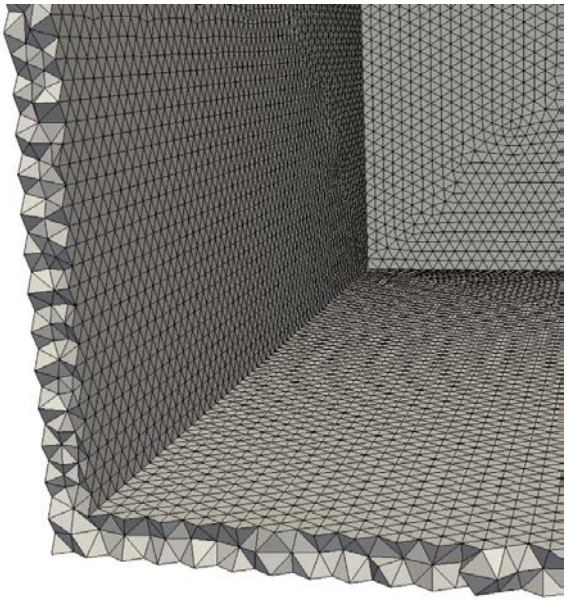


FIGURE 5.20 – Conteneur - Maillage du conteneur

5.5.3 Résultats numériques

Les temps de restitution des deux méthodes SISC et SIAC des simulations parallèles sur le cluster local ainsi que les accélérations et les efficacités correspondantes, pour 1 à 8 cœurs utilisés, sont présentés dans la table 5.9.

Nb. de cœurs	SISC	Accélération	Efficacité	SIAC	Accélération	Efficacité
1	7896			7896		
2	3989	1.98	0.99	3990	1.98	0.99
3	4100	1.93	0.64	2928	2.70	0.90
4	3186	2.48	0.62	2256	3.50	0.88
5	2793	2.83	0.57	1927	4.10	0.82
6	2528	3.12	0.52	1767	4.47	0.74
7	2183	3.62	0.52	1645	4.80	0.69
8	2084	3.79	0.47	1546	5.11	0.64

TABLE 5.9 – Conteneur - Temps de restitution (s), accélérations et efficacités pour les méthodes SISC et SIAC

Les figures 5.21, 5.22 et 5.23 représentent respectivement les temps de restitution, les accélérations et les efficacités sur le cluster local pour les méthodes SISC et SIAC.

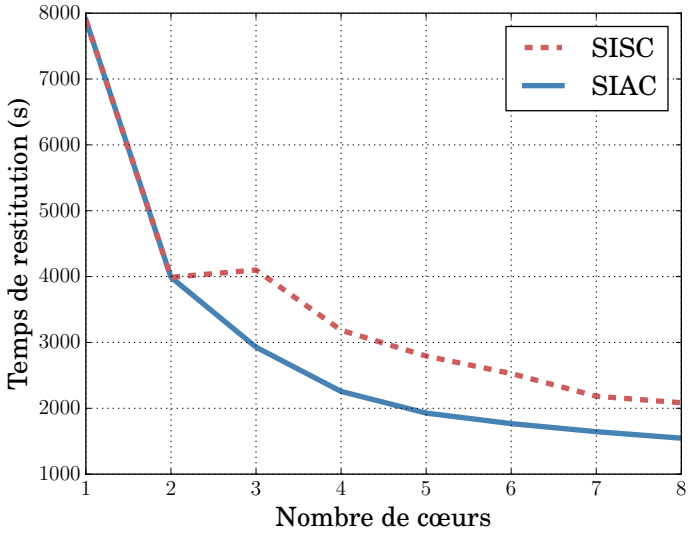


FIGURE 5.21 – Conteneur - Temps de restitution (s)

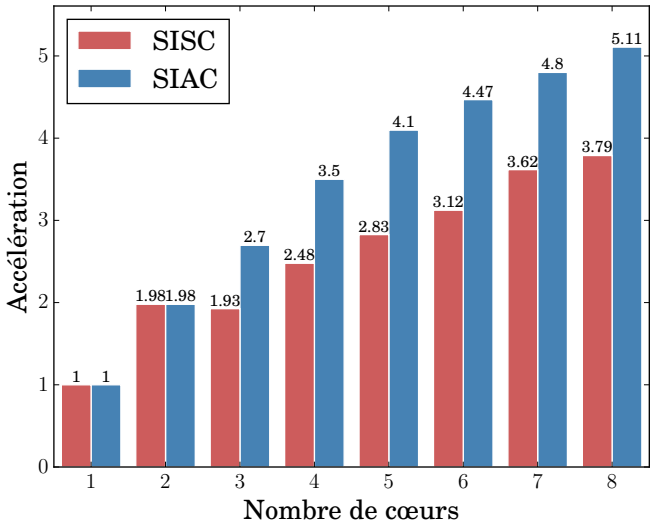


FIGURE 5.22 – Conteneur - Accélération

5.5.4 Résultats physiques

La résolution de l'équation de Navier pour l'élasticité linéaire donne les champs de déplacement. À partir de ces déplacements, on peut calculer les déformations et les contraintes correspondantes.

Les différents niveaux de gris pour les figures 5.24, 5.25 et 5.26 représentent des

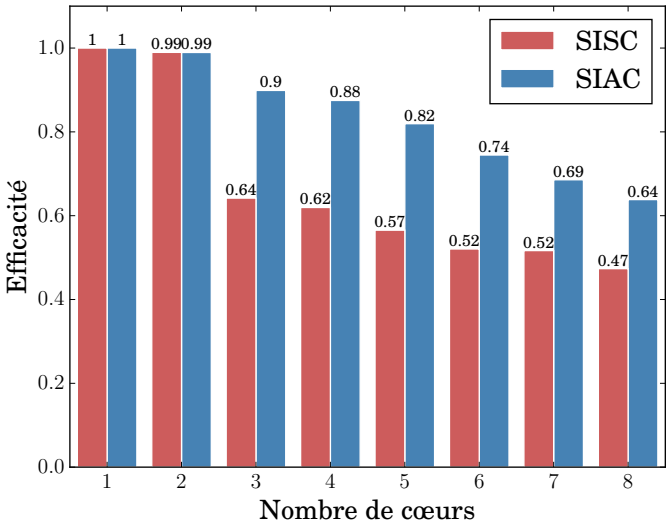


FIGURE 5.23 – Conteneur - Efficacité

valeurs faibles pour le gris clair et des valeurs élevées pour le gris foncé et le noir.

La figure 5.24 représente le déplacement moyen du conteneur à $t = 3\,\mathrm{s}$. Comme on pouvait le prévoir, le déplacement le plus important se produit sur le côté faisant directement face à l'entrée d'air. La paroi supérieure s'effondre légèrement du fait de la présence de tourbillons créés par les angles du conteneur.

La figure 5.25 représente toujours le déplacement moyen mais seulement sur le côté le plus affecté, faisant face à l'entrée d'air. On peut noter la forte similarité avec le premier mode d'une plaque rectangulaire dont les 4 côtés sont encastrés.

Les figures 5.26a et 5.26b représentent la contrainte moyenne et le déplacement moyen respectivement dans une coupe transversale du conteneur à mi-longueur. Les valeurs de contraintes élevées sont situées dans les angles et le déplacement correspond à celui visible sur la figure 5.24.

Les figures 5.27a et 5.27b représentent les lignes de courant dans le fluide autour du conteneur.

5.6 Synthèse

En résumé, on a pu constater plusieurs choses durant cette mise en œuvre des méthodes de résolution parallèles développées dans le chapitre 4. Premièrement, on constate que la parallélisation permet une diminution des temps de calculs comparés aux temps séquentiels. Deuxièmement, les temps de restitution obtenus lorsque l'on utilise des communications asynchrones (SIAC) sont toujours inférieurs à ceux obtenus avec des communications synchrones (SISC) ; ce gain de performance est dû au poids des synchronisations. Cela provient principalement de la relative lenteur de convergence des méthodes numé-

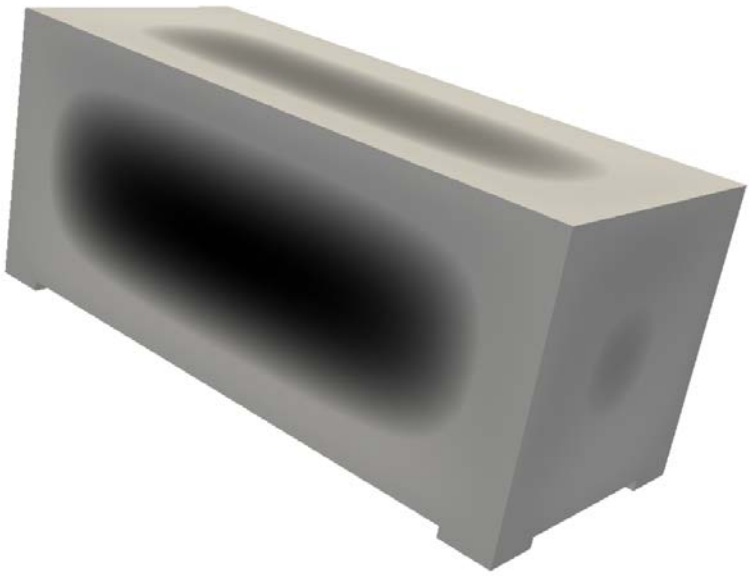


FIGURE 5.24 – Conteneur - Déplacement moyen

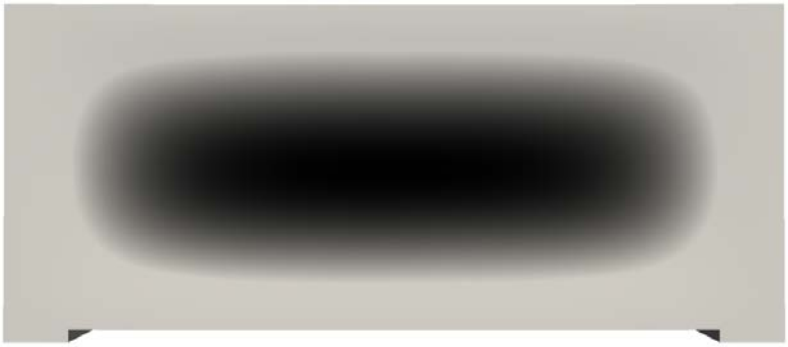
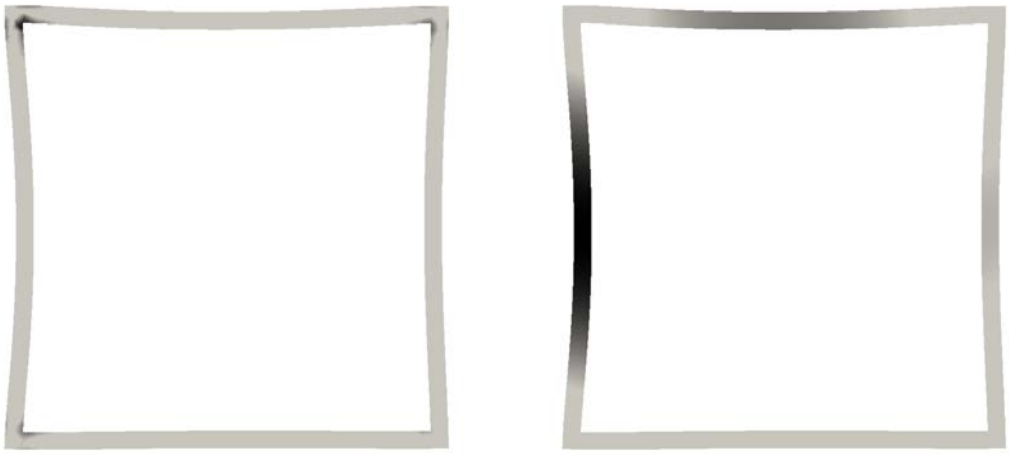


FIGURE 5.25 – Conteneur - Déplacement moyen sur le côté faisant face à l'écoulement

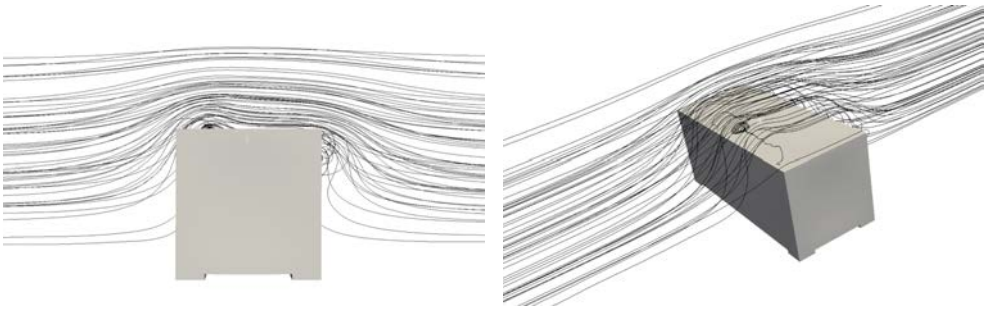
riques, cette faible vitesse de convergence a pour effet d'augmenter le nombre de synchronisations qui pénalise les méthodes synchrones. Il faut également noter que compte tenu du type d'application, il n'est pas nécessaire d'utiliser beaucoup de plus de processeurs.

De plus, même si les accélérations et les efficacité restent modérées par rapport à des applications parallèles plus classiques, on remarquera que l'on obtient très souvent des gains en temps significatifs, allant jusqu'à 30 %, des méthodes à communications



(a) Contrainte moyenne (b) Déplacement moyen

FIGURE 5.26 – Conteneur - Visualisations dans une coupe transversale



(a) Vue de face (b) Vue en perspective

FIGURE 5.27 – Conteneur - Lignes de courant autour du conteneur

asynchrones sur les méthodes à communications synchrones.

Cette dernière remarque permet aussi de noter que les notions d'accélérations et d'efficacités perdent ici de leur intérêt comme mesures de performance. En effet, les nombreuses barrières de synchronisation nécessaires pour respecter la cohérence du traitement numérique pénalisent les performances des solveurs parallèles.

En ce qui concerne les vitesses de convergence, il serait beaucoup trop lourd d'indiquer le détail du nombre d'itérations à chaque pas de temps et pour chaque système résolu. En revanche, on peut noter que l'algorithme de calcul des vitesses converge assez rapidement (de l'ordre de 10 itérations) alors que pour le calcul des corrections pression, la convergence est plus lente (de l'ordre de 30 itérations). Côté structure, le calcul du déplacement est en général assez lent (nécessite de l'ordre de 50 itérations pour le mur)

mais dépend de la complexité de la structure étudiée. Le calcul de la déformation du maillage est quant à lui très rapide puisqu'on ne résout qu'un laplacien avec une méthode multi-grille très efficace dans ce cas.

Il est alors à noter que lorsque la convergence est lent, le nombre de synchronisations est important et les méthodes parallèles asynchrones présentent dans ce cas un grand intérêt pour diminuer les temps de restitution.

Références

[62] V. Partimbene, T. Garcia, P. Spiteri, P. Marthon et L. Ratsifandrihana. "A Parallel Method for the Solution of Fluid-Structure Interaction Problems". In : *Proceedings of the Fifth International Conference on Parallel, Distributed, Grid and Cloud Computing for Engineering*. Sous la dir. de P. Iványi, B. H. V. Topping et G. Várady. Stirlingshire, UK : Civil-Comp Press, 2017. doi : 10.4203/ccp.111.20.

[65] V. Partimbene, P. Spiteri, P. Marthon et L. Ratsifandrihana. "A Two-Dimensional Numerical Case Study of Fluid-Structure Interaction". In : *Proceedings of the Third International Conference on Railway Technology : Research, Development and Maintenance*. Sous la dir. de J. Pombo. Stirlingshire, UK : Civil-Comp Press, 2016. doi : 10.4203/ccp.110.43.

[69] P.-A. Raviart. *Les Méthodes d'Éléments Finis en Mécanique des Fluides*. Eyrolles, 1981.

[75] *Scotch Project*. www.gforge.inria.fr/projects/scotch/.

CHAPITRE

6

CONCLUSION

Dans ce travail, on s'est intéressé à la résolution parallèle par des méthodes de relaxation synchrones et asynchrones des problèmes d'interaction fluide-structure. Ces méthodes présentent un intérêt dans le cadre du parallélisme massif, compte-tenu de leur souplesse.

Après une présentation des aspects physiques sous-jacents, nous avons abouti à un modèle mathématique représentant les phénomènes d'interaction fluide-structure. Nous avons alors développé une méthode algorithmique qui permet, de manière unifiée, de séparer en sous-problèmes le problème global décomposé en grands blocs. D'autre part, nous avons aussi proposé une amélioration d'implémentation dans le code OpenFOAM qui permet des communications non bloquantes entre les processeurs.

Cette étude a été appliquée à des problèmes d'interaction fluide-structure pouvant intervenir dans des applications industrielles. Il est cependant à noter que les essais numériques ont été principalement effectués, dans un laps de temps bref, sur un cluster local dans la mesure où l'accès à un réseau plus performant tel que Grid'5000 a été rendu difficile. L'utilisation d'un cluster local n'a pas permis de considérer des tailles de problèmes discrétisés très importantes. Cependant, il est à noter que les communications non bloquantes permettent d'avoir des gains allant jusqu'à 30 % pour les problèmes considérés de tailles modestes. Dans la mesure où il serait possible de considérer des tailles de problèmes plus importantes, ce qui augmenterait la granularité, on peut, compte-tenu de l'expérience acquise au laboratoire, espérer des gains notables lors de l'utilisation des méthodes avec communications non bloquantes comparées aux méthodes avec communications bloquantes. Par ailleurs, pour des problèmes de couplages forts dans lesquels la structure vibre à des fréquences élevées, il est nécessaire de choisir des pas de temps petits ce qui, compte-tenu de l'utilisation de schémas implicites en temps, nécessite l'introduction d'une barrière de synchronisation à chaque pas de temps. Cette barrière,

additionnée à d'autres barrières assurant la cohérence des algorithmes utilisés, contribue à la dégradation des performances des algorithmes parallèles. Cependant, on a pu constater une diminution notable et régulière des temps de restitution lorsque le nombre de processeurs augmente.

Les objectifs énumérés dans l'introduction générale ont donc été atteints. En effet, les différents modèles mathématiques ont pu être résolus avec une méthode de discrétisation identique (i.e. volumes finis). Le solveur fluide-structure parallélisé possède de bonnes performances permettant d'obtenir de réels gains sur les temps de restitution et ce, notamment, grace à une méthode de parallélisation faisant intervenir des communications non bloquantes. Enfin, le solveur a pu être utilisé avec succès sur des applications faisant intervenir divers phénomènes physiques.

Cette étude est une première étape que nous comptons poursuivre en considérant d'une part des modèles mathématiques modélisant des aspects physiques plus complexes comme, par exemple, des matériaux aux comportements non linéaires (composites, etc.), des structures beaucoup plus souples (dirigeables, ballons, etc.) ou encore en introduisant des contraintes sur le déplacement de la structure. D'autre part, on peut envisager de mettre en œuvre les méthodes considérées dans des problèmes de type industriel à beaucoup plus grande échelle, intervenant notamment dans l'aéronautique ou l'aérospatiale.

ANNEXE

A

RAPPELS MATHÉMATIQUES

On rappelle dans cette annexe quelques définitions et résultats utiles.

A.1 Sous-différentiabilité et différentiabilité

A.1.1 Sous-différentiablilité

On considère à présent une notion importante applicable aux fonctions convexes non différentiables. Il s'agit de la notion de sous-différentiabilité.

Définition 11. (Sous-gradient et sous-différentiel) Soit ϕ une fonction convexe sur E à valeurs dans $\bar{\mathbb{R}}$, un vecteur $\mathbf{x}^* \in E^*$ (dual de E) est appelé sous-gradient de ϕ au point $\mathbf{x} \in E$ si

$$\phi(\mathbf{y}) \geq \phi(\mathbf{x}) + < \mathbf{x}^*, \mathbf{y} - \mathbf{x} > , \forall \mathbf{y} \in E. \tag{A.1}$$

L'ensemble de tous les sous-gradients en $\mathbf{x}$ est appelé sous-différentiel de ϕ. Il est noté $\partial\phi(\mathbf{x})$.

L'interprétation géométrique du sous-différentiel est la suivante. Il est formé par toutes les directions des hyperplans qui passent par le point $(\mathbf{x}, \phi(\mathbf{x}))$ et restent « sous » le graphe de la fonction ϕ. Ces hyperplans sont appelés *hyperplans support* ou *hyperplans d'appui* au graphe de ϕ en $\mathbf{x}$.

On a la propriété importante suivante.

Proposition 6. *Le sous-différentiel de ϕ est un opérateur monotone éventuellement multivoque de E dans E^*.*

Démonstration. Soit $\mathbf{y}$ et $\mathbf{z}$ éléments de E, on a par définition

$$\begin{array}{l} \phi(\mathbf{z}) \geq \phi(\mathbf{y}) + < \mathbf{z}-\mathbf{y}, \mathbf{y}^* > , \\ \phi(\mathbf{y}) \geq \phi(\mathbf{z}) + < \mathbf{y}-\mathbf{z}, \mathbf{z}^* > , \end{array} \tag{A.2}$$

alors

$$0 \geq < \mathbf{z}-\mathbf{y}, \mathbf{y}^* > + < \mathbf{y}-\mathbf{z}, \mathbf{z}^* > , \tag{A.3}$$

donc

$$< \mathbf{z}-\mathbf{y}, \mathbf{z}^*-\mathbf{y}^* > \geq 0. \tag{A.4}$$

□

A.1.2 Différentiabilité au sens de Gâteaux

Définition 12. Soient E et F deux espaces vectoriels normés. Soit J une application définie sur le domaine $D \subset E$ et à valeurs dans F. L'application J est dite *Gâteaux-différentiable* (ou différentiable au sens Gâteaux) en un point $\mathbf{x}$ de l'intérieur du domaine D, s'il existe un opérateur linéaire continu $\mathbf{A}$ de E dans F ($\mathbf{A} \in \mathcal{L}(E,F)$), tel que

$$\forall \mathbf{h} \in E, \lim_{\substack{\tau \to 0 \\ \tau \in \mathbb{R}}} \frac{\|J(\mathbf{x}+\tau\mathbf{h}) - J(\mathbf{x}) - \tau\mathbf{A}\mathbf{h}\|_F}{\tau} = 0. \tag{A.5}$$

Si elle existe, la G-dérivée de J au point $\mathbf{x}$ est unique, et on la notera $J'(\mathbf{x}) = \mathbf{A}$. De plus, on a le résultat suivant, utile lors du calcul explicite de l'opérateur $J'(\mathbf{x})$:

$$\forall \mathbf{h} \in E, \; J'(\mathbf{x}) \cdot \mathbf{h} = \lim_{\substack{\theta \to 0 \\ \theta \in \mathbb{R}}} \frac{J(\mathbf{x}+\theta\mathbf{h}) - J(\mathbf{x})}{\theta}. \tag{A.6}$$

A.1.3 Différentiabilité au sens de Fréchet

Définition 13. Soient E et F deux espaces vectoriels normés. Soit J une application définie sur le domaine $D \subset E$ et à valeurs dans F. L'application J est dite *Fréchet-différentiable* (ou différentiable au sens Fréchet, ou encore différentiable au sens fort) en un point $\mathbf{x}$ de l'intérieur du domaine D, s'il existe un opérateur linéaire continu $\mathbf{A}$ de E dans F ($\mathbf{A} \in \mathcal{L}(E,F)$), tel que

$$\forall \mathbf{h} \in E, \; J(\mathbf{x}+\mathbf{h}) = J(\mathbf{x}) + \mathbf{A}\mathbf{h} + \| \mathbf{h} \|_E \varepsilon(\mathbf{h}), \text{ avec } \lim_{\|\mathbf{h}\|_E \to 0} \|\varepsilon(\mathbf{h})\|_F = 0. \tag{A.7}$$

Proposition 7. *Soit ϕ une fonction Gâteaux-différentiable, alors $\partial\phi(\mathbf{x})$ se réduit à un simple élément, la différentielle au sens de Gâteaux [8].*

Démonstration. En effet, ϕ étant convexe, on a classiquement

$$\phi(\mathbf{x} + \tau(\mathbf{y}-\mathbf{x})) \leq \phi(\mathbf{x}) + \tau(\phi(\mathbf{y}) - \phi(\mathbf{x})), \tag{A.8}$$

où $0 < \tau \leq 1$; donc

$$\frac{\phi(\mathbf{x} + \tau(\mathbf{y} - \mathbf{x})) - \phi(\mathbf{x})}{\tau} \leq \phi(\mathbf{y}) - \phi(\mathbf{x}), \tag{A.9}$$

et en prenant la limite quand $\tau \to 0$, on a bien $\nabla\phi(\mathbf{x}) \in \partial\phi(\mathbf{x})$.

Réciproquement, si au point $\mathbf{x}$, ϕ est finie-continue, soit $\mathbf{x}^* \in \partial\phi(\mathbf{x})$; par définition on a

$$\phi(\mathbf{x}) - \phi(\mathbf{y}) \leq < \mathbf{x} - \mathbf{y}, \mathbf{x}^* > , \forall \mathbf{y} \in E, \tag{A.10}$$

en changeant $\mathbf{y}$ en $\mathbf{x} + \tau\mathbf{z}$,

$$\begin{aligned} < \mathbf{y} - \mathbf{x}, \mathbf{x}^* > &\equiv < \mathbf{x} + \tau\mathbf{z} - \mathbf{x}, \mathbf{x}^* > , \\ &\equiv \tau < \mathbf{z}, \mathbf{x}^* > , \end{aligned} \tag{A.11}$$

donc

$$< \mathbf{z}, \mathbf{x}^* > \leq \frac{\phi(\mathbf{x} + \tau\mathbf{z}) - \phi(\mathbf{x})}{\tau}. \tag{A.12}$$

On passe à la limite quand $\tau \to 0^+$ et on a bien sûr la Gâteaux-dérivée ; donc le membre de droite est

$$< \mathbf{z}, \nabla\phi(\mathbf{x}) - \mathbf{x}^* > \geq 0, \forall \mathbf{z} \in E, \tag{A.13}$$

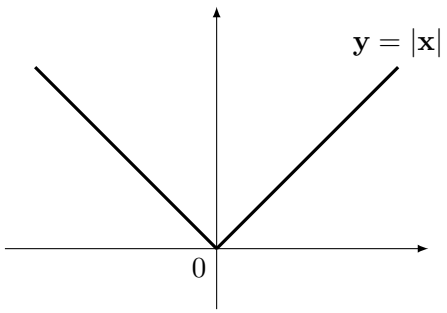
et donc $\mathbf{x}^* = \nabla\phi(\mathbf{x})$. □

Remarque 23. *On a le même type de résultat si ϕ est Fréchet-différentiable.*

Remarque 24. *Si l'application J est Fréchet-différentiable au point $\mathbf{x}$, elle est aussi Gâteaux-différentiable au point $\mathbf{x}$, et les deux dérivées coïncident. Par conséquent, la Fréchet-dérivée de J au point $\mathbf{x}$, si elle existe, est unique, et sera aussi notée $J'(\mathbf{x})$.*

Remarque 25. *Si l'application J est Fréchet-différentiable au point $\mathbf{x}$, elle est alors continue au point $\mathbf{x}$.*

Exemple 5. Soit $\phi : \mathbf{x} \in \mathbb{R} \mapsto |\mathbf{x}|$. Calculons les sous-gradients de ϕ en tout point $\mathbf{x}$ de $\mathbb{R}$.



On sait que ϕ est Gâteaux-différentiable pour $\mathbf{x} \neq 0$; donc

$$\partial\phi(\mathbf{x}) = \begin{cases} -1 & \text{si } \mathbf{x} < 0, \\ +1 & \text{si } \mathbf{x} > 0. \end{cases} \tag{A.14}$$

En 0 il vient

$$\begin{aligned} \mathbf{x}^* \in \partial\phi(0) &\Leftrightarrow |\mathbf{y}| \geq \mathbf{x}^*\mathbf{y}, \forall \mathbf{y} \in \mathbb{R} \setminus \{0\}, \\ &\Leftrightarrow \mathbf{x}^* \frac{\mathbf{y}}{|\mathbf{y}|} \leq 1, \\ &\Leftrightarrow -1 \leq \mathbf{x}^* \leq 1, \\ &\Leftrightarrow \partial\phi(0) = [-1, 1]. \end{aligned} \tag{A.15}$$

Dans $\mathbb{R}^2$, les hyperplans d'appui sont des droites et les sous-gradients associés leurs coefficients directeurs.

$$\partial\phi(\mathbf{x}) = \begin{cases} -1 & \text{si } \mathbf{x} < 0, \\ +1 & \text{si } \mathbf{x} > 0, \\ [-1,1] & \text{si } \mathbf{x} = 0. \end{cases} \tag{A.16}$$

A.2 Norme matricielle

On appelle norme matricielle induite par une norme vectorielle $\|\cdot\|$, l'application de l'espace des matrices dans $\mathbb{R}^M$ définie pour toute matrice A par

$$\|A\| = \max_{X \neq 0} \frac{\|AX\|}{\|X\|}. \tag{A.17}$$

Pour les normes vectorielles usuelles, les normes subordonnées correspondantes sont données par le résultat suivant.

Lemme 4. *Soit A une matrice $M \times M$ réelle ou complexe. Les normes d'opérateur $\|A\|_r$ subordonnées à la norme l_r pour $r = 1, 2, \infty$ sont données par*

$$\|A\|_1 = \max_{1 \leq j \leq M} \sum_{i=1}^{M} |a_{i,j}|, \tag{A.18}$$

$$\|A\|_\infty = \max_{1 \leq i \leq M} \sum_{j=1}^{M} |a_{i,j}|, \tag{A.19}$$

$$\|A\|_2 = \sqrt{\delta(A^T A)} = \sqrt{\delta(A A^T)}, \tag{A.20}$$

où $\delta(\cdot)$ est le rayon spectral de la matrice.

A.3 Rappels d'algèbre linéaire

Définition 14. (Matrice à diagonale strictement dominante) Soit $A \in \mathcal{L}(\mathbb{R}^M)$ une matrice, elle est dite à diagonale strictement dominante lorsque

$$|a_{i,i}| > \sum_{\substack{j=1 \\ j \neq i}}^{M} |a_{i,j}|, \; i = 1, \ldots, M. \tag{A.21}$$

Définition 15. (Z-matrice) Toute matrice dont les coefficients diagonaux sont strictement positifs et dont les coefficients hors diagonaux sont négatifs ou nuls est appelée une Z-matrice.

Définition 16. (M-matrice et matrice de Stieltjes) Soit $A \in \mathcal{L}(\mathbb{R}^M)$ une matrice à diagonale strictement dominante et supposons que $a_{i,j} \leq 0$, $i \neq j$ et que $a_{i,i} > 0$, $i = 1, \ldots, M$; alors A est appelée une M-matrice. Si A est aussi symétrique, alors A est appelée une matrice de Stieltjes.

Théorème 1. *Toute Z-matrice A telle que l'inverse de A existe et est non négative (i.e. $A^{-1} \geq 0$) est une M-matrice.*

Théorème 2. *Si $A \in \mathcal{L}(\mathbb{R}^M)$ est une Z-matrice, les propriétés suivantes sont équivalentes :*

- *A est une M-matrice ;*
- *tous les mineurs principaux de A sont positifs.*

Théorème 3. *Une matrice A avec $a_{i,j} \leq 0$, $i \neq j$, est une M-matrice si, et seulement si il existe un vecteur X positif tel que AX est positif.*

On cite à présent quelques résultats utiles sur les propriétés des M-matrices.

Théorème 4. *Soit $A \in \mathcal{L}(\mathbb{R}^M)$, avec $a_{i,j} \leq 0$, $i \neq j$, alors A est une M-matrice si, et seulement si les coefficients diagonaux de A sont positifs et la matrice $B = \mathbb{I} - D^{-1}A$, où $D = diag(a_{1,1}, \ldots, a_{M,M})$, satisfait $\delta(B) < 1$.*

Théorème 5. *Soit $A \in \mathcal{L}(\mathbb{R}^M)$ une M-matrice et soit $D \in \mathcal{L}(\mathbb{R}^M)$ une matrice diagonale non négative quelconque, alors $A + D$ est une M-matrice, et $(A + D)^{-1} \leq A^{-1}$.*

Théorème 6. *Soit $A \in \mathcal{L}(\mathbb{R}^M)$ une matrice de Stieltjes, alors A est définie positive.*

Théorème 7. *Soit $A, B \in \mathcal{L}(\mathbb{R}^M)$, alors $A = B - C$ est un **partitionnement régulier** de A si B est inversible, $B^{-1} \geq 0$ et $C \geq 0$; c'est un **partitionnement faiblement régulier** si la condition $C \geq 0$ est remplacée par $B^{-1}C \geq 0$ et $CB^{-1} \geq 0$.*

Théorème 8. *Soit $A \in \mathcal{L}(\mathbb{R}^M)$ et supposons que $A = B - C$ est un partitionnement faiblement régulier, alors $\delta(B^{-1}C) < 1$ si, et seulement si A^{-1} existe et est non négative.*

Lemme 5. *Soit $A \in \mathcal{L}(\mathbb{R}^M)$ une M-matrice admettant un partitionnement régulier $A = B - C$, $\hat{J}$ la matrice de Jacobi de $N(A)$ et Θ le vecteur propre associé au rayon spectral $\hat{\delta}$ de $\hat{J}$; alors il existe un vecteur $\vartheta \in \mathbb{R}^M$ positif et un scalaire $v \in [0, 1[$ telle que l'inégalité suivante est vérifiée :*

$$B^{-1}C\Theta = \hat{J}\vartheta \leq v\vartheta. \tag{A.22}$$

Démonstration. A étant une M-matrice, il existe alors un vecteur ϑ positif tel que $r = A\vartheta = (B - C)\vartheta > 0$. Notons que $B^{-1}r \geq 0$. De plus, si $B^{-1}r = 0$, pour tout $k \in \{1, \ldots, M\}$, tel que $\sum_{i=1}^{M}(B^{-1})_{k,i}r_i = 0$, alors toutes les lignes de B sont nulles, ce qui est en contradiction avec la non-unicité de B. Donc, $B^{-1}r > 0$, alors

$$B^{-1}(B - C)\vartheta > 0 \Rightarrow B^{-1}C\vartheta < \vartheta, \tag{A.23}$$

et donc A est une M-matrice, il existe à la fois un vecteur ϑ positif et un nombre réel positif $v < 1$ tels que l'inégalité (A.22) soit vérifiée. □

A.4 M-fonctions

On aborde à présent une classe d'applications non linéaires généralisant la notion de M-matrice.

Définition 17. (Isotone) Une application $\mathcal{A}$ de $\mathbb{R}^M$ dans $\mathbb{R}^M$ est diagonale monotone croissante (ou isotone) si pour tout $x \in \mathbb{R}^M$ et $i \in \{1, \ldots, M\}$, les fonctions $\mathcal{A}_i^i$ définies comme suit :

$$\begin{cases} \mathcal{A}_i^i : \{t \in \mathbb{R} \mid x + te^i \in \mathbb{R}^M\} \mapsto \mathbb{R}, \\ \mathcal{A}_i^i(t) = \mathcal{A}_i(x + te^i), \end{cases} \tag{A.24}$$

sont monotones croissantes, $\mathcal{A}_i$ étant la i-ème composante de l'application $\mathcal{A}$ et $e^i \in \mathbb{R}^M$, $i = 1, \ldots, M$, le i-ème vecteur unitaire de la base canonique.

Définition 18. Une application $\mathcal{A}$ de $\mathbb{R}^M$ dans $\mathbb{R}^M$ est diagonale monotone strictement croissante si pour tout $x \in \mathbb{R}^M$ et $i \in \{1, \ldots, M\}$, les fonctions $\mathcal{A}_i^i$ définies comme suit :

$$\begin{cases} \mathcal{A}_i^i : \{t \in \mathbb{R} \mid x + te^i \in \mathbb{R}^M\} \mapsto \mathbb{R}, \\ \mathcal{A}_i^i(t) = \mathcal{A}_i(x + te^i), \end{cases} \tag{A.25}$$

sont strictement monotones croissantes.

Définition 19. (Antitone) Une application $\mathcal{A}$ de $\mathbb{R}^M$ dans $\mathbb{R}^M$ est hors-diagonale monotone décroissante (ou antitone) si pour tout $x \in \mathbb{R}^M$ et $i, l \in \{1, \ldots, M\}$, $l \neq i$ les fonctions $\mathcal{A}_i^i$ définies comme suit :

$$\begin{cases} \mathcal{A}_i^l : \{t \in \mathbb{R} \mid x + te^l \in \mathbb{R}^M\} \mapsto \mathbb{R}, \\ \mathcal{A}_i^l(t) = \mathcal{A}_i(x + te^l), \end{cases} \tag{A.26}$$

sont monotones décroissantes.

Définition 20. Une application $\mathcal{A}$ de $\mathbb{R}^M$ dans $\mathbb{R}^M$ est inverse monotone croissante si $\mathcal{A}(x) \leq \mathcal{A}(y)$ pour tout $x, y \in \mathbb{R}^M$ implique que $x \leq y$.

Définition 21. Une application $\mathcal{A}$ de $\mathbb{R}^M$ dans $\mathbb{R}^M$ est une M-fonction au sens de Rheinbolt (cf. [61] et [70]) si $\mathcal{A}$ est hors-diagonale monotone décroissante et inverse monotone croissante.

Rheinboldt a donné une caractérisation des M-fonctions (théorème 3.4 de [70]).

Théorème 9. *Soit $\mathcal{A}$ de $\mathbb{R}^M$ dans $\mathbb{R}^M$ continue et hors-diagonale décroissante, alors $\mathcal{A}$ est une M-fonction surjective.*

BIBLIOGRAPHIE

[1] G. M. Amdahl. "Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities". In : *Proceedings of the April 18-20, 1967, Spring Joint Computer Conference*. New York, USA : ACM, 1967, p. 483–485.

[2] N. Antonopoulos et L. Gillam. *Cloud Computing : Principles, Systems and Applications*. Computer Communications and Networks. Berlin, Heidelberg : Springer, 2010.

[3] J. Arnal, V. Migallón et J. Penadés. "Non-Stationary Parallel Multisplitting Algorithms for Almost Linear Systems". In : *Numerical Linear Algebra with Applications* 6 (1999), p. 79–92.

[4] J. Bahi, S. Contassot-Vivier et R. Couturier. *Parallel Iterative Algorithms : From Sequential to Grid Computing*. Numerical Analysis and Scientific Computing. Chapman & Hall/CRC, 2007.

[5] J. Bahi, J.-C. Miellou et K. Rhofir. "Asynchronous Multisplitting Methods for Nonlinear Fixed Point Problems". In : *Numerical Algorithms* 15 (1997), p. 315–345.

[6] Z.-Z. Bai. "The Monotone Convergence Rate of the Parallel Nonlinear AOR Method". In : *Computers & Mathematics with Applications* 31 (1996), p. 1–8.

[7] Z.-Z. Bai, V. Migallón, J. Penadés et D. B. Szyld. "Block and Asynchronous Two-stage Methods for Mildly Nonlinear Systems". In : *Numerische Mathematik* 82 (1999), p. 1–20.

[8] V. Barbu. *Nonlinear Semigroups and Differential Equations in Banach Spaces*. Noordhoff International Publishing, 1976.

[9] G. Baudet. “Asynchronous Iterative Methods for Multiprocessors”. In : *Journal of the Association for Computing Machinery* 25 (1978), p. 226–244.

[10] Y. Bazilevs, K. Takizawa et T. E. Tezduyar. *Computational Fluid-Structure Interaction : Methods and Applications*. Wiley, 2013.

[11] D. P. Bertsekas et J. N. Tsitsiklis. “Some Aspects of Parallel and Distributed Iterative Algorithms – A Survey”. In : *Automatica* 27 (1991), p. 3–21.

[12] R. Bolze, F. Cappello, E. Caron, M. Dayde, F. Desprez, E. Jeannot, Y. Jégou, S. Lanteri, J. Leduc, N. Melab, G. Mornet, R. Namyst, P. Primet, B. Quétier, O. Richard, T. El-Ghazali et I. Touche. “Grid’5000 : A Large Scale And Highly Reconfigurable Experimental Grid Testbed”. In : *International Journal of High Performance Computing Applications* 20 (2006), p. 481–494.

[13] R. Boman. “Développement d’un formalisme Arbitraire Lagrangien Eulérien tridimensionel en dynamique implicite. Application aux opérations de mise à forme”. Thèse de doct. Université de Liège, 2010.

[14] W. L. Briggs, V. E. Henson et S. F. McCormick, éds. *A Multigrid Tutorial, Second Edition*. Other Titles in Applied Mathematics. SIAM, 2000.

[15] H.-J. Bungartz, M. Mehl et M. Schafer, éds. *Fluid-Structure Interaction II : Modelling, Simulation, Optimization*. Berlin, Heidelberg : Springer, 2010.

[16] H.-J. Bungartz et M. Schafer, éds. *Fluid-Structure Interaction : Modelling, Simulation, Optimization*. Berlin, Heidelberg : Springer, 2006.

[17] R. Buyya, J. Broberg et A. M. Goscinski. *Cloud Computing : Principles and Paradigms*. Wiley, 2011.

[18] M. Chau. “Algorithmes Parallèles Asynchrones pour la Simulation Numérique”. Thèse de doct. Institut National Polytechnique de Toulouse, 2005.

[19] M. Chau, T. Garcia et P. Spiteri. “Asynchronous Schwarz Methods Applied to Constrained Mechanical Structures in Grid Environment”. In : *Advances in Engineering Software* 74 (2014), p. 1–15.

[20] D. Chazan et W. Miranker. “Chaotic Relaxation”. In : *Linear Algebra and its Applications* 2 (1969), p. 199–222.

[21] *Cloud computing, Wikipedia*. `www.wikipedia.org/wiki/Cloud_computing`.

[22] P. Comte, J.-C. Miellou et P. Spiteri. “La Notion de H-accrétivité, Applications”. In : *CRAS* 283 (1976), p. 655–658.

[23] R. Couturier. *Designing Scientific Applications on GPUs*. Numerical Analysis and Scientific Computing. Chapman & Hall/CRC, 2013.

[24] R. Couturier, C. Denis et F. Jézéquel. “GREMLINS : a Large Sparse Linear Solver for Grid Environment”. In : *Parallel Computing* 34 (2008), p. 380–391.

[25] R. Couturier et L. Ziane Khodja. “A Scalable Multisplitting Algorithm to Solve Large Sparse Linear Systems”. In : *The Journal of Supercomputing* 71 (2015), p. 1345–1356.

[26] E. de Langre. *Fluide et Solide*. Ellipses, 2001.

[27] G. Duvaut et J.-L. Lions. *Les Inéquations en Mécanique et Physique*. Dunod, 1972.

[28] M. N. El Tarazi. "Some Convergence Results for Asynchronous Algorithms". In : *Numerische Mathematik* 39 (1982), p. 325–240.

[29] D. G. Feingold et R. S. Varga. "Block Diagonally Dominant Matrices and Generalizations of the Gerschgorin Circle Theorem". In : *Pacific Journal of Mathematics* 12 (1962), p. 1241–1250.

[30] M. J. Flynn. "Very High-Speed Computing Systems". In : *Proceedings of the IEEE 54*. 1966, p. 1901–1909.

[31] A. Frommer. "On Asynchronous Iterations in Partially Ordered Spaces". In : *Numerical Functional Analysis and Optimization* 12 (1991), p. 315–325.

[32] A. Frommer et G. Mayer. "Convergence of Relaxed Parallel Multisplitting Methods". In : *Linear Algebra and its Applications* 119 (1989), p. 141–152.

[33] A. Frommer et G. Mayer. "On the Theory and Practice of Multisplitting Methods in Parallel Computation". In : *Computing* 49 (1992), p. 63–74.

[34] M. Gengler, S. Ubéda et F. Desprez. *Initiation au Parallélisme : Concept, Architectures et Algorithmes*. Masson, 1996.

[35] L. Giraud et P. Spiteri. "Résolution Parallèle de Problèmes aux Limites Non Linéaires". In : *ESAIM : Mathematical Modelling and Numerical Analysis* 25 (1991), p. 579–606.

[36] R. Glowinski, J.-L. Lions et R. Tremolieres. *Analyse Numérique des Inéquations Variationnelles*. T. 1 – 2. Dunod, 1976.

[37] *Grid'5000*. `www.grid5000.fr`.

[38] W. Hackbusch et U. Trottenberg, éds. *Multigrid Methods - Proceedings of the Conference Held at Köln-Porz*. Lecture Notes in Mathematics. Berlin, Heidelberg : Springer, 1981.

[39] P. Hémon. *Vibrations des Structures Couplées avec le Vent*. Ellipses, 2006.

[40] R. W. Hockney et C. R. Jesshope. *Parallel Computers : Architecture, Programming and Algorithms*. Adam Hilger Ltd, 1981.

[41] R. I. Issa. "Solution of the Implicitly Discretised Fluid Flow Equation by Operator-Splitting". In : *Journal of Computational Physics* 62 (1986), p. 40–65.

[42] H. Jasak et H. G. Weller. "Application of the Finite Volume Method and Unstructured Meshes to Linear Elasticity". In : *International Journal for Numerical Methods in Engineering* 48 (2000), p. 267–287.

[43] F. Jézéquel, R. Couturier et C. Denis. "Solving Large Sparse Linear Systems in a Grid Environment : the GREMLINS Code Versus the PETSc Library". In : *The Journal of Supercomputing* 59 (2012), p. 1517–1532.

[44] L. S. Lai, C. H. Lai, A.-K. Cheik Ahamed et F. Magoulès. "Coupling and Simulation of Fluid-Structure Interaction Problems for Automotive Sun-Roof on Graphics Processing Unit". In : *2014 IEEE International Conference on High Performance Computing and Communications*. 2014, p. 137–144.

[45] F. Magoulès et G. Gbikpi-Benissan. "Distributed Computation of Convergence Residual Under Asynchronous Iterations". In : *IEEE Transactions on Parallel and Distributed Systems* (to be published).

[46] F. Magoulès et F.-X. Roux. *Calcul Scientifique Parallèle - 2e Édition*. Sciences Sup. Dunod, 2017.

[47] F. Magoulès, F.-X. Roux et G. Houzeaux. *Parallel Scientific Computing*. Wiley, 2015.

[48] G. Meurant. *Computer Solution of Large Linear Systems*. Studies in Mathematics and its Applications. North Holland, 1999.

[49] J.-C. Miellou. "Méthodes de Jacobi, Gauss-Seidel, Sur-Relaxation par Blocs, appliquées à une Classe de Problèmes Non Linéaires". In : *CRAS* 273 (1971), p. 1257–1260.

[50] J.-C. Miellou. "Sur une Variante de la Méthode de Relaxation, Appliquée à des Problèmes Comportant un Opérateur Somme d'un Opérateur Différentiable et d'un Opérateur Maximal Monotone Diagonal". In : *CRAS* 275 (1972), p. 1107–1110.

[51] J.-C. Miellou. "Algorithmes de Relaxation Chaotique à Retards". In : *ESAIM : Mathematical Modelling and Numerical Analysis* 9 (1975), p. 55–82.

[52] J.-C. Miellou. "Itérations Chaotiques à Retards, Étude de la Convergence dans le Cas d'Espaces Partiellement Ordonnés". In : *CRAS* 280 (1975), p. 233–236.

[53] J.-C. Miellou. "Asynchronous Iterations and Order Intervals". In : *Parallel Algorithms and Architectures*. Sous la dir. de M. Cosnard et D. Trystram. North Holland, 1986, p. 85–96.

[54] J.-C. Miellou, D. El Baz et P. Spiteri. "A New Class of Asynchronous Iterative Algorithms with Order Intervals". In : *Mathematics of Computation* 67 (1998), p. 237–255.

[55] J.-C. Miellou et P. Spiteri. "Un Critère de Convergence pour des Méthodes Générales de Point Fixe". In : *ESAIM : Mathematical Modelling and Numerical Analysis* 19 (1985), p. 645–669.

[56] J.-C. Miellou et P. Spiteri. "Stopping Criteria for Parallel Asynchronous Iterations for Fixed Point Methods". In : *Developments in Parallel, Distributed, Grid and Cloud Computing for Engineering*. Sous la dir. de B. H. V. Topping et P. Iványi. Stirlingshire, UK : Saxe-Coburg Publications, 2013. Chap. 12, p. 277–308.

[57] J.-C. Miellou, P. Spiteri et D. El Baz. "A New Stopping Criterion for Linear Perturbed Asynchronous Iterations". In : *Journal of Computational and Applied Mathematics* 219 (2008), p. 471–483.

[58] F. Moukalled, L. Mangani et M. Darwish. *The Finite Volume Method in Computational Fluid Dynamics*. Fluid Mechanics and its Applications. Berlin, Heidelberg : Springer, 2015.

[59] *National Institute of Standards and Technology (NIST)*. `www.nist.gov`.

[60] D. P. O'Leary et R. E. White. "Multi-Splittings of Matrices and Parallel Solution of Linear Systems". In : *SIAM : Journal on Algebraic Discrete Methods* 6 (1985), p. 630–640.

[61] J. M. Ortega et W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Computer Science and Applied Mathematics. New-York : Academic Press, 1970.

[62] V. Partimbene, T. Garcia, P. Spiteri, P. Marthon et L. Ratsifandrihana. "A Parallel Method for the Solution of Fluid-Structure Interaction Problems". In : *Proceedings of the Fifth International Conference on Parallel, Distributed, Grid and Cloud Computing for Engineering*. Sous la dir. de P. Iványi, B. H. V. Topping et G. Várady. Stirlingshire, UK : Civil-Comp Press, 2017. doi : `10.4203/ccp.111.20`.

[63] V. Partimbene, T. Garcia, P. Spiteri, P. Marthon et L. Ratsifandrihana. "Asynchronous Multi-Splitting Method for the Solution of Fluid-Structure Interaction Problems". In : *Advances in Engineering Software* (soumis).

[64] V. Partimbene, P. Spiteri, P. Marthon et L. Ratsifandrihana. *A Fluid-Structure Interaction Study on 3-D Structures Under Crosswind*. Rapport de Recherche - Institut de Recherche en Informatique de Toulouse. 2016.

[65] V. Partimbene, P. Spiteri, P. Marthon et L. Ratsifandrihana. "A Two-Dimensional Numerical Case Study of Fluid-Structure Interaction". In : *Proceedings of the Third International Conference on Railway Technology : Research, Development and Maintenance*. Sous la dir. de J. Pombo. Stirlingshire, UK : Civil-Comp Press, 2016. doi : `10.4203/ccp.110.43`.

[66] S. V. Patankar. *Numerical Heat Transfer and Fluid Flow*. Series in Computational Methods in Mechanics and Thermal Science. New-York : Hemisphere Publishing Corporation, 1980.

[67] S. V. Patankar et D. B. Spalding. "A Calculation Procedure for Heat, Mass and Momentum Transfer in Three-Dimensional Parabolic Flows". In : *International Journal of Heat and Mass Transfer* 15 (1972), p. 1787–1806.

[68] S. Piperno. *Interactions Fluide-Structure*. Mastère de Mécanique Numérique - École Nationale Supérieure des Mines de Paris. 2005–2006.

[69] P.-A. Raviart. *Les Méthodes d'Éléments Finis en Mécanique des Fluides*. Eyrolles, 1981.

[70] W. Rheinboldt. "On M-functions and Their Application to Nonlinear Gauss-Seidel Iterations and to Network Flows". In : *Journal of Mathematical Analysis and Applications* 32 (1970), p. 274–307.

[71] F. Robert. “Recherche d’une Matrice Parmi les Minorants d’un Opérateur Linéaire”. In : *Numerische Mathematik* 9 (1966), p. 189–199.

[72] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Other Titles in Applied Mathematics. SIAM, 2003.

[73] S. A. Savari et D. P. Bertsekas. “Finite Termination of Asynchronous Iterative Algorithms”. In : *Parallel Computing* 22 (1996), p. 39–56.

[74] M. Schafer. *Computational Engineering - Introduction to Numerical Methods*. Berlin, Heidelberg : Springer, 2006.

[75] *Scotch Project*. `www.gforge.inria.fr/projects/scotch/`.

[76] P. Spiteri. “Contribution à l’étude de grands systèmes non-linéaires, comportement d’algorithmes itératifs, stabilité de systèmes continus”. Thèse de doct. Université de Besançon, 1984.

[77] P. Spiteri. “A New Characterization of M-matrices and H-matrices”. In : *BIT Numerical Mathematics* 43 (2003), p. 1019–1032.

[78] P. Spiteri. “Finite Precision Computation for Linear Fixed Point Methods of Parallel Asynchronous Iterations”. In : *Techniques for Parallel, Distributed and Cloud Computing in Engineering*. Sous la dir. de P. Iványi et B. H. V. Topping. Stirlingshire, UK : Saxe-Coburg Publications, 2015. Chap. 8, p. 163–196.

[79] P. Spiteri, J.-C. Miellou et D. El Baz. “Parallel Asynchronous Schwarz and Multisplitting Methods for a Nonlinear Diffusion Problem”. In : *Numerical Algorithms* 33 (2003), p. 461–474.

[80] D. B. Szyld. “Different Models Of Parallel Asynchronous Iterations With Overlapping Blocks”. In : *Computational and Applied Mathematics* 17 (1998), p. 101–115.

[81] D. Tromeur-Dervout, G. Brenner, D. R. Emerson et J. Erhel, éds. *Parallel Computational Fluid Dynamics 2008. Parallel Numerical Methods, Software Development and Applications*. Computational Science and Engineering. Berlin, Heidelberg : Springer, 2008.

[82] R. S. Varga. *Matrix Iterative Analysis*. Computational Mathematics. Berlin, Heidelberg : Springer, 2000.

[83] H. K. Versteeg et W. Malalasekera. *An Introduction to Computational Fluid Dynamics : The Finite Volume Method (2nd Edition)*. Pearson, 2007.

[84] D.-R. Wang, Z.-Z. Bai et D. J. Evans. “On the Monotone Convergence of Multisplitting Method for a Class of Systems of Weakly Nonlinear Equations”. In : *International Journal of Computer Mathematics* 60 (1996), p. 229–242.

[85] R. E. White. “Parallel Algorithms for Nonlinear Problems”. In : *SIAM : Journal on Algebraic Discrete Methods* 7 (1986), p. 137–149.

[86] L. Ziane Khodja. “Résolution de systèmes linéaires et non linéaires creux sur grappes de GPUs”. Thèse de doct. Université de Franche-Comté, 2013.

[87] L. Ziane Khodja, M. Chau, R. Couturier, J. Bahi et P. Spiteri. “Parallel Solution of American Option Derivatives on GPU Clusters”. In : *Computers & Mathematics with Applications* 65 (2013), p. 1830–1848.