Méthodes de décomposition de domaine "Duale" et "Duale-Primale" pour les simulations magnétostatiques

Mohamed Ghenai

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ GRENOBLE ALPES

École doctorale : EEATS - Electronique, Electrotechnique, Automatique, Traitement du Signal (EEATS)
Spécialité : Génie électrique
Unité de recherche : Laboratoire de Génie Electrique

Méthodes de décomposition de domaine "Duale" et "Duale-Primale" pour les simulations magnétostatiques.

"Dual" and "Dual-Primal" domain decomposition methods for magnetostatic simulations.

Présentée par :

MOHAMED GHENAI

Direction de thèse :

Olivier CHADEBEC Directeur de recherche CNRS, Université Grenoble Alpes	Directeur de thèse
Ronan PERRUSSEL CHARGE DE RECHERCHE CNRS, Institut National Polytechnique de Toulouse	Co-encadrant de thèse
Jean-Michel GUICHON MAITRE DE CONFERENCE, Université Grenoble Alpes	Co-encadrant de thèse

Rapporteurs :

Yvonnick LE MENACH
PROFESSEUR DES UNIVERSITES, Université de Lille
François LOUF
PROFESSEUR DES UNIVERSITES, ENS Paris-Saclay

Thèse soutenue publiquement le **3 juillet 2024**, devant le jury composé de :

Pierre GOSSELET, DIRECTEUR DE RECHERCHE, CNRS DELEGATION HAUTS-DE-FRANCE	Président
Olivier CHADEBEC, DIRECTEUR DE RECHERCHE, CNRS DELEGATION ALPES	Directeur de thèse
Yvonnick LE MENACH, PROFESSEUR DES UNIVERSITES, Université de Lille	Rapporteur
François LOUF, PROFESSEUR DES UNIVERSITES, ENS Paris-Saclay	Rapporteur
Christophe PICARD, MAITRE DE CONFERENCES, Grenoble INP	Examinateur
Victorita DOLEAN-MAINI, FULL PROFESSOR, Technische Universiteit Eindhoven	Examinatrice
Laurent GERBAUD, PROFESSEUR DES UNIVERSITES, Université Grenoble Alpes	Examinateur

Invités :

Ronan PERRUSSEL
CHARGE DE RECHERCHE, CNRS
Frederic Vi
INGENIEUR DOCTEUR, Altair Engineering Inc

Remerciements

Un remerciement, on le fait autant pour les autres que pour soi. Ce sera donc, pour moi, un rappel de la chance que j'ai eue pendant cette thèse, de rencontrer plein de gens, ce qui m'a aidé à grandir sur le plan scientifique et, plus important, sur le plan humain.

Après une longue réflexion, j'ai décidé de commencer par remercier le jury qui a pris le temps et l'effort de lire ce manuscrit et qui m'a posé beaucoup de questions que j'ai trouvées très intéressantes. Je remercie aussi l'encadrement de ma thèse, qui a été là chaque semaine, même quand je n'avançais pas, pour donner leurs avis et m'aider à aller plus loin dans ce travail. Je tiens à nommer deux personnes en particulier : Ronan Perussel, sans qui je n'aurais jamais réalisé ce travail. Je n'oublierai jamais les heures que tu as passées à revoir ce que j'ai fait pour essayer de trouver ce qui manquait, les nombreuses relectures du manuscrit et ta manière d'expliquer les choses sans jamais vexer. Ronan est tout simplement l'encadrant que tout thésard rêve d'avoir. Je remercie également Frédéric, mon encadrant chez Altair, mon manager, et après cinq ans de collaboration, mon ami. Tu es un modèle pour moi, professionnellement parlant, et je garderai à jamais les moments que nous avons passés ensemble : les nombreuses fois où je t'ai battu en pause perf, les gossip, les problèmes de parking, les KFC, les 5-0 sur FIFA et les boissons chelou. N'oublie pas, il faut toujours choisir les bonnes portes et avoir un briquet sur soi.

Je vais passer à la partie un peu plus fun, et je commencerai par les gens que j'ai rencontrés à Grenoble. D'abord, les anciens collègues d'Altair, chacun avec sa personnalité mais tous aimables : Christian, Christophe, Enrico, Lucas, Evelyne, Nicolas, Souleymane qui n'a pas oublié, Sreekanth (I'll hunt you down if you ever forget to thank me), Yann aka la douce terreur adorable et Guy « merci pour le Japon ». Il y a ceux de ma team : Jon l'incorruptible, Koceila le fameux docteur K, Popo heureusement que tu existes (un jour je finirai l'histoire, promis) et Olivier, mon camarade de guerre. Tellement de choses que nous avons vécues ensemble, toi qui n'écoutes jamais, et des conneries que nous avons tout le temps assumées. Pour toute ma team, j'espère que notre amitié durera aussi longtemps que les daily TMC (c'est-à-dire vraiment beaucoup). Je remercie l'ensemble des gens que j'ai rencontrés grâce à Altair, qu'ils soient encore là ou partis. Je n'oublierai pas non plus les gens que j'ai pu connaître au G2Elab : les collègues, Mayra, Antoine, Gauthier, Alberto, et ceux qui étaient présents à ma soutenance ainsi que beaucoup d'autres encore.

Je vais ensuite remercier ma deuxième famille, les Toulousains. Je vous aime. Je commencerai par Ayoub, merci pour tout et spécialement pour les écrans, ton pot de thèse était magnifique, dommage qu'ils ont volé la pastilla de Fred. Sophie, avec toi on ne s'ennuie jamais hein, j'espère que tu ne seras pas perdue sans mon coaching. Les Grem3 : Marion, Leonard, Amine, Tom, Thomas, François, Eric, Arnaud, Fred et Olivier, merci pour les pauses café, les repas de midi, les bars, les raclettes et le match de rugby. Merci de m'avoir adopté, moi, petit GRE clandestin. Et enfin, Zak, mon pote, le mec le plus perché du labo. Tu as été là quand j'en avais besoin et tu m'as

écouté me plaindre des dizaines de fois, t'embêter est la seule chose qui m'a fait tenir pendant cette thèse, je te dirais encore une fois "he, he, he he...".

Je finirai par remercier mes proches, ceux avec qui il y a eu de « nombreux bronzages nocturnes », ceux avec qui j'ai commencé cette nouvelle vie en France, ceux qui étaient là pour moi dans le meilleur et dans le pire, en France ou en Algérie, vous serez toujours dans mon cœur. Enfin, mes parents, pour qui avoir un fils docteur a longtemps été un rêve, j'espère que je vous rends fiers et sachez que je vous aime énormément.

Je suis certain que j'en oublie beaucoup, chaque rencontre que j'ai faite dans ma vie a participé de près ou de loin à ma réussite présente et j'espère future. Je vous remercie tous pour cela.

Chi va piano va sano e va lontano.

Proverbe italien

Résumé

Les logiciels de simulation doivent intégrer de nouvelles méthodes de calcul scientifique pour améliorer leurs performances et répondre aux besoins croissants en efficacité. Cette thèse se concentre sur le développement des méthodes de décomposition de domaine dans le but de réduire les temps de calcul des simulations par éléments finis, en particulier dans le domaine de l'électromagnétisme et plus précisément de la magnétostatique en 3D. Ces méthodes sont particulièrement adaptées au calcul parallèle, mais leur adaptation aux problèmes magnétostatiques, caractérisés par l'utilisation de formulations en potentiel scalaire et potentiel vecteur, a nécessité une approche spécifique. Ces potentiels ne sont pas uniques et nécessitent des conditions de jauge. Dans le cas scalaire, la décomposition entraîne la création de sous-domaines flottants où la matrice devient singulière. Dans le cas du potentiel vecteur, la décomposition empêche l'utilisation de la jauge par arbre d'arête. Deux méthodes ont été retenues, FETI-1 et FETI-DP, toutes deux sans recouvrement, et ont été implémentées dans le logiciel Altair Flux. Nous avons comparé leurs performances à celles des méthodes standard, directes ou itératives. Nous avons testé les deux cas, linéaire et non linéaire, en utilisant deux approches différentes pour ce dernier : la décomposition de domaine après linéarisation et la localisation des problèmes non linéaires sur des sous-domaines. Pour résoudre les problèmes de jauge, une méthode de régularisation a été développée. Les résultats montrent que si des conditions relatives à la taille des sous-domaines et à l'équilibre de la charge de travail entre les différents processeurs sont respectées, la décomposition de domaine permet de réduire significativement le temps de simulation, par exemple d'un facteur 3.8, tout en assurant une bonne scalabilité. De plus, la consommation de mémoire est moindre par rapport à une résolution avec un solveur direct, tout en offrant des résultats très similaires.

Mots clés : Décomposition de domaine, Magnétostatique, Méthode FETI, Calcul parallèle, Éléments finis, Solveur linéaire, Solveur non-linéaire.

Abstract

Simulation software needs to incorporate new scientific computing methods to enhance performance and meet growing efficiency demands. This thesis focuses on developing domain decomposition methods to reduce computation times for finite element simulations, particularly in the field of electromagnetism and specifically 3D magnetostatics. While these methods are well-suited for parallel computing, adapting them to magnetostatic problems, characterized by the use of scalar and vector potential formulations, required a specific approach. These potentials are not unique and require gauge conditions. In the scalar case, decomposition leads to the creation of floating subdomains where the matrix becomes singular. In the vector potential case, decomposition precludes the use of edge tree gauge. Two methods, FETI-1 and FETI-DP, both non-overlapping, were selected and implemented in the Altair Flux software. Their performances were compared to standard, direct or iterative methods. Both linear and nonlinear cases were tested using two different approaches for the latter : domain decomposition after linearization and localization of nonlinear problems on subdomains. A regularization method was developed to address gauge issues. Results show that if conditions regarding subdomain size and workload balance among processors are met, domain decomposition significantly reduces simulation time, for example, by a factor of 3.8, while ensuring good scalability. Additionally, memory consumption is lower compared to direct solver resolution, while providing very similar results.

Keywords : Domain decomposition, Magnetostatic, FETI method, Parallel computing, Finite Element, Linear solver, Nonlinear solver.

Table des matières

Liste des Symboles

Acronyme

EDP	Équation aux dérivées partielles
DDM	Domain decomposition method : Méthode de décomposition de domaine
FEM	Finite element method : Méthode des éléments finis
RHS	Right-hand side : second membre
FETI	Finite Element Tearing and Interconnecting : Décomposition et Interconnexion des Éléments Finis
BDDC	Balancing Domain Decomposition by Constraints : Décomposition de domaine équilibrée par contraintes
PCPG	Preconditioned Conjugate Projected Gradient : gradient conjugué projeté et préconditionné
MPI	Message Passing Interface : Interface de Passage de Messages

Espaces fonctionnels sur Ω

$L^2(\Omega)$	Fonction réelle de carré intégrable sur Ω, $\Omega \ni x \to f(x) \in \mathcal{R}$ s.t. $\int_\Omega f^2(x)dx < \infty$
$\boldsymbol{L}^2(\Omega)$	Fonctions vectorielles réelles avec un carré intégrable sur Ω, $\Omega \ni x \to \boldsymbol{f}(x) \in \mathcal{R}^3$ s.t. $\int_\Omega \boldsymbol{f}(x) \cdot \boldsymbol{f}(x)dx < \infty$
$H^1(\Omega)$	$\left\{u \in L^2(\Omega) \text{ tel que } \frac{\partial u}{\partial x_i} \in L^2(\Omega) \text{ avec } i = 1, \cdots, m\right\}$

Introduction

Depuis plusieurs années, l'utilisation de la simulation numérique est devenue un atout majeur pour le développement industriel et la conception de nouveaux dispositifs. Avec l'essor de l'électrification, les simulations électromagnétiques ont suscité un regain d'intérêt en raison des possibilités qu'elles offrent aux ingénieurs pour répondre à une forte demande dans des délais beaucoup plus courts.

Les logiciels de simulation doivent intégrer de nouvelles méthodes de calcul scientifique pour gagner en performances et répondre à ce besoin d'efficacité. La réduction des temps de calcul tout en gardant la meilleure précision possible est un problème apparu simultanément avec le développement de l'informatique. Étant donné les ressources de calcul limitées, il incombe aux développeurs de concevoir des méthodes et des algorithmes nécessitant moins de ressources de la manière la plus optimale possible.

Différentes approches peuvent être envisagées, telles que le choix des formulations, les techniques de réduction de modèle ou encore l'utilisation du calcul parallèle. Une simulation se compose généralement de plusieurs étapes, parmi lesquelles les plus importantes sont le maillage, l'assemblage des matrices et la résolution des systèmes linéaires. Réduire le temps de calcul implique inévitablement la réduction du temps nécessaire pour réaliser ces étapes. Parmi celles-ci, la résolution des systèmes linéaires est de loin la plus chronophage en raison du grand nombre d'opérations nécessaires pour son exécution, ce qui est particulièrement vrai dans le cadre des simulations 3D où le nombre de variables est élevé.

Il existe deux grandes familles de solveurs de systèmes linéaires. Les solveurs directs reposent sur des méthodes visant à transformer les problèmes de type $Ax = B$ en une résolution de systèmes avec des matrices triangulaires. Ces méthodes sont peu adaptées au calcul parallèle du fait qu'elles sont souvent séquentielles, nécessitant des étapes consécutives dépendant des résultats précédents. Par exemple, la factorisation LU, la factorisation QR ou la méthode de Gauss-Jordan requièrent des étapes ordonnées. Par conséquent, il est difficile de les paralléliser sans introduire des dépendances limitant l'efficacité du calcul

parallèle. De plus, ces méthodes ont une consommation de mémoire très importante. En revanche, les méthodes directes offrent une grande robustesse.

La deuxième famille, les méthodes itératives, permettent une plus grande parallélisation en raison de leur nature itérative et de la répartition plus claire des tâches entre les processeurs. Des algorithmes comme la méthode du gradient conjugué, la méthode de Jacobi ou Gauss-Seidel sont mieux adaptés au calcul parallèle du fait de leur capacité à effectuer des calculs indépendants à chaque itération. Elles ont aussi l'avantage de consommer moins de mémoire. Contrairement aux méthodes directes, les méthodes itératives sont sensibles aux erreurs numériques, dépendent fortement du préconditionnement et parfois présentent une convergence difficile à prédire.

De nouvelles méthodes, telles que la décomposition de domaine, offrent en théorie les avantages des deux méthodes sans leurs inconvénients. Elles sont très adaptées au calcul parallèle et sont considérées comme plus robustes. C'est pour ces raisons que nous avons étudié dans cette thèse l'utilisation de la méthode de décomposition de domaines dans le cadre des problèmes magnétostatiques en 3D. La méthode de décomposition de domaine vise à diviser un problème complexe en sous-problèmes plus simples, à les résoudre individuellement, puis à combiner les solutions pour obtenir la solution globale. Cela permet de réduire la complexité computationnelle et de faciliter la résolution de problèmes de grande taille.

La complexité algorithmique d'une méthode de résolution représente le nombre d'opérations à effectuer pour résoudre un système de taille n. Dans le cas d'un système plein, le nombre d'opérations nécessaire pour effectuer une factorisation LU d'un système de taille $N \times N$ est proportionnel à n^3. Ainsi, la complexité de ces opérations est d'ordre cubique, notée $o(n^3)$.

La méthode du gradient conjugué converge généralement vers la solution exacte (erreur nul) en $o(n)$ itérations. À chaque itération de la méthode du gradient conjugué, les opérations dominantes impliquent des produits scalaires de vecteurs, des additions de vecteurs et des multiplications matrice-vecteur. Le coût computationnel par itération est approximativement en $o(n^2)$ en raison de ces opérations. Le coût total de la méthode du gradient conjugué s'élève généralement à $o(n^3)$ en raison de $o(n)$ itérations, chacune nécessitant $o(n^2)$ opérations. Le conditionnement de la méthode de décomposition de domaine est de l'ordre de $(1 + \log n)^b$ où $b = 2$ ou $b = 3$.

Dans le cas des matrices creuses, la convergence avec un solveur itératif est de l'ordre de $o(n^{3/2})$. La figure 0.1 montre à quel point une augmentation dans le nombre de variables peut entraîner une très grande augmentation dans le nombre d'opérations à effectuer, ce qui arrive souvent lors d'optimisation de forme, d'optimisation topologique ou de calcul de perte fer, où le maillage doit être fin. Par conséquent, le temps de calcul est augmenté et il

est nécessaire de s'orienter vers des méthodes plus performantes.

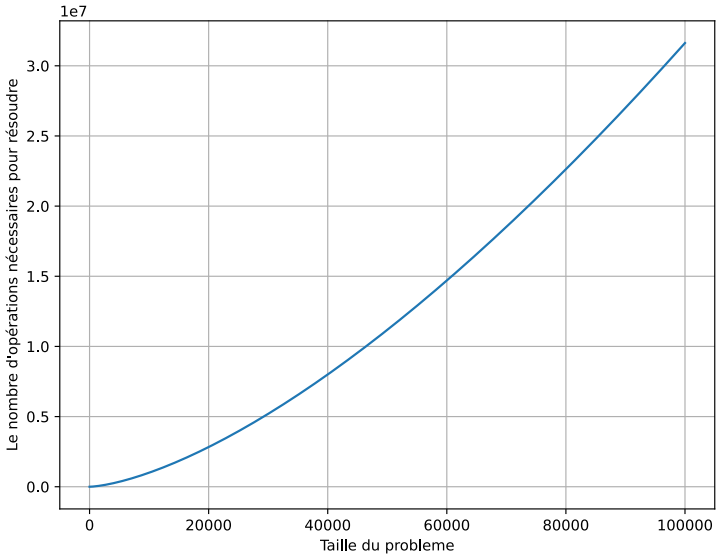

Figure 0.1 – Complexité algorithmique pour une résolution avec un solveur itératif

Ce travail est le fruit d'une collaboration entre deux laboratoires : le G2Elab à Grenoble et le LAPLACE à Toulouse et l'entreprise ALTAIR, acteur mondial de l'édition de logiciels scientifiques pour les ingénieur, au travers d'un financement CIFRE. Nous avons travaillé dans le logiciel Altair Flux^{TM}, utilisé pour les simulations électromagnétiques à basses fréquences. Flux est utilisé pour des simulations de plus en plus coûteuses en termes de ressources ce qui pose de nouveaux défis aux développeurs qui doivent avoir une pensée plus orientée vers la performance. Sur un code maintenu depuis plusieurs dizaines d'années, rajouter une approche parallèle est difficilement applicable et nécessite d'aborder le problème d'une manière innovante.

Dans ce contexte, un projet pour séparer le processus de résolution du reste du processus a été proposé. C'est dans ce contexte que se place cette thèse : réduire les temps de calcul en améliorant l'efficacité des méthodes de résolution.

Cette thèse a pour objectif le développement d'une méthode de décomposition de domaine dans ce contexte industriel sur des problèmes de simulation magnétostatique. Afin d'être un réel atout pour la réduction des temps de calcul, cette méthode devra être compatible avec les éléments déjà présents dans le logiciel Flux. L'objectif de ce travail de recherche est ainsi de proposer une alternative efficace aux solveurs directs et itératifs actuels, en permettant l'accélération de la résolution complète de simulations et en prenant en compte l'ensemble des processus de simulation, maillage, assemblage et résolution de systèmes

linéaires. Afin d'y parvenir, la méthode de décomposition de domaine développée devra être compatible avec les multiples particularités des problèmes à traiter, telles que les non-linéarités, l'hétérogénéité des matériaux mais aussi les différentes formulations utilisées.

Dans un premier temps, nous présentons, dans le chapitre 1, un état de l'art des méthodes de décomposition de domaine, ainsi que les deux méthodes que nous avons utilisées dans cette thèse. Ces méthodes ont été développées initialement pour la mécanique. Nous établissons également des liens avec d'autres méthodes récemment développées.

Le chapitre 2 se concentre sur la physique traitée, à savoir la magnétostatique. Nous décrivons le processus permettant d'obtenir le système à résoudre en partant des équations de Maxwell jusqu'à la discrétisation.

Dans le troisième chapitre, nous présentons le logiciel Flux, notre environnement de travail, ainsi que les différents outils utilisés pour la mise en place de la méthode de décomposition de domaine.

Les résultats relatifs au comportement et aux performances de la méthode pour la résolution des cas tests, sur lesquels la méthode a été appliquée, sont présentés pour le problème en formulation potentiel scalaire dans le chapitre 4. Nous abordons à la fois le cas linéaire et le cas non-linéaire.

Enfin, le chapitre 5 expose d'autres approches explorées pour résoudre les problèmes en éléments finis d'arête avec les formulations en potentiel vecteur, qui présentent leur lot de difficultés.

Chapitre 1

Etat de l'art des méthodes de décomposition de domaine

Ce chapitre présente un état de l'art de la méthode de décomposition de domaine (ou DDM pour Domain Decomposition Method), qui est une technique numérique permettant de résoudre des problèmes aux limites de grande taille en les divisant en sous-problèmes plus petits et plus faciles à traiter. Nous commençons par introduire les notions générales et la motivation de la DDM puis nous passons en revue les principales familles de méthodes de décomposition de domaine, en mettant l'accent sur les méthodes sans recouvrement. Ensuite, nous nous intéressons à deux méthodes particulières de la famille des méthodes FETI (Finite Element Tearing and Interconnecting), qui sont la méthode FETI-1 et la méthode FETI-DP. Nous présentons les principes, les algorithmes, les propriétés et les applications de ces méthodes, ainsi que quelques résultats numériques obtenus dans la littérature.

1.1 Généralités

La décomposition de domaine fait généralement référence à la division d'une équation aux dérivées partielles, ou d'une approximation de celle-ci, en problèmes couplés sur des sous-domaines plus petits formant une partition du domaine original. Cette décomposition peut intervenir :

- au niveau continu où différents modèles physiques peuvent être utilisés dans différentes régions ;
- au niveau de la discrétisation où il peut être pratique d'employer différentes méthodes d'approximation dans différentes régions ;
- au niveau de la résolution des systèmes algébriques résultant de l'approximation de l'équation aux dérivées partielles.

Ces aspects sont souvent liés.

1.2 Motivation

L'intérêt accru pour la décomposition de domaine est lié notamment à la croissance du calcul hautes performances (ou HPC pour High Performance Computing). Le développement de méthodes numériques pour les grands systèmes algébriques est une étape essentielle dans le développement de codes efficaces pour simuler différents phénomènes physiques et résoudre des problèmes de mécanique, des calculs thermiques ou bien des problèmes d'optimisation. De nombreuses autres tâches dans de tels codes se parallélisent relativement facilement.

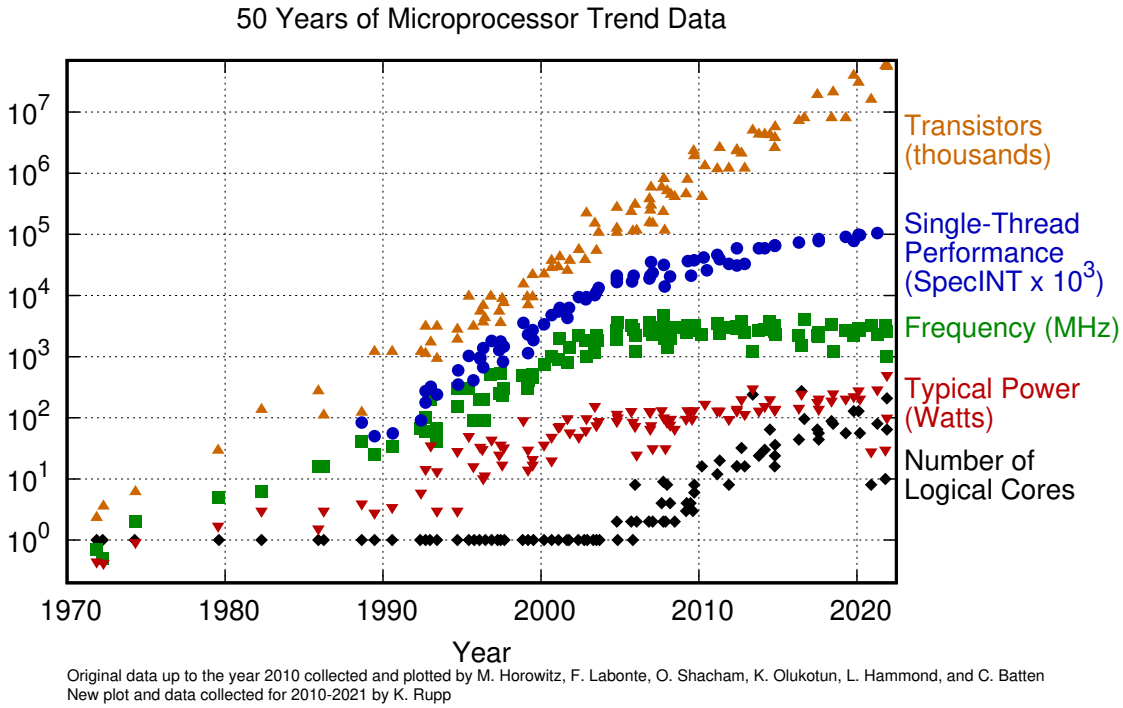

Figure 1.1 – Loi de Moore

L'importance des solveurs de systèmes algébriques parallèles tient aussi à l'architecture des machines et des processeurs. La loi de Moore, en figure 1.1, stipule ainsi que la puissance de calcul double tous les deux ans. Néanmoins, depuis les années 2000, le recours au traitement parallèle est devenu indispensable pour maintenir la validité de cette loi. Pendant longtemps, les fabricants de processeurs ont pu offrir des améliorations de fréquence d'horloge et de parallélisme au niveau des jeux d'instructions, permettant ainsi aux applications séquentielles de s'exécuter considérablement plus rapidement sur une nouvelle génération de processeurs sans nécessiter de modifications. Cependant, afin de mieux gérer la puissance dissipée (étroitement liée à la fréquence d'horloge), les fabricants

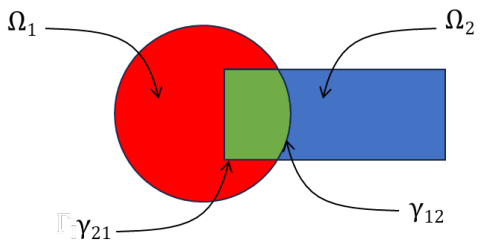

Figure 1.2 – Décomposition de domaine avec recouvrement (Schwarz 1869 [1]).

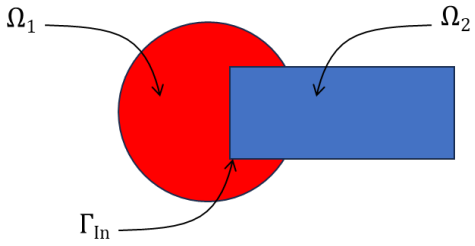

Figure 1.3 – Décomposition de domaine sans recouvrement (Przemieniecki 1963 [1]).

privilégient désormais les architectures multi-cœurs. Cette transition a un impact significatif sur les logiciels qui doivent être spécifiquement adaptés pour exploiter au mieux le matériel disponible.

Dans les applications pratiques, la discrétisation par la méthode des éléments finis ou d'autres techniques de discrétisation réduit le problème à la résolution d'un système algébrique souvent de grande taille. La factorisation directe de tels systèmes peut alors ne pas être une option viable (à la fois en consommation mémoire et en temps de calcul) et l'utilisation de méthodes itératives, comme l'algorithme du gradient conjugué, peut entraîner une convergence très lente.

La DDM offre une alternative efficace aux approches conventionnelles en permettant une parallélisation naturelle du processus de résolution.

1.3 Familles de décomposition de domaine

Il existe principalement deux types de méthodes : les méthodes avec recouvrement (voir la figure 1.2) et les méthodes sans recouvrement (voir la figure 1.3).

Prenons comme exemple un problème de type Poisson. Pour un domaine global Ω, de frontière $\partial\Omega$, le problème s'écrit comme suit [2] :

$$\begin{cases} -\Delta u = f & \text{dans } \Omega\,, \\ u = 0 & \text{sur } \partial\Omega\,, \end{cases} \tag{1.1}$$

avec u la solution recherchée et f un second membre donné.

Ce type de problème permet d'illustrer de manière simple l'application de la méthode et nous le retrouvons en magnétostatique par exemple. Dans les chapitres suivants, nous examinerons l'application de la méthode pour les différentes formulations de la magnétostatique. Cependant, dans un premier temps, nous présentons l'état de l'art tel qu'il est décrit dans de nombreuses références, pour des problèmes mécaniques.

Le problème (1.1), dans le cas "sans recouvrement", pour deux sous domaines Ω_1 et Ω_2 partageant une interface Γ_{In}, avec ∂n_1 et ∂n_2 les normales sortantes, est équivalent au système suivant :

$$\begin{aligned} -\Delta u_1 &= f && \text{dans } \Omega_1 \\ u_1 &= 0 && \text{sur } \partial\Omega_1 \backslash \Gamma_{In} \\ u_1 &= u_2 && \text{sur } \Gamma_{In} \\ \frac{\partial u_1}{\partial n_1} &= -\frac{\partial u_2}{\partial n_2} && \text{sur } \Gamma_{In} \\ -\Delta u_2 &= f && \text{dans } \Omega_2 \\ u_2 &= 0 && \text{sur } \partial\Omega_2 \backslash \Gamma_{In} \end{aligned} \quad . \tag{1.2}$$

Il est équivalent au système (1.3) si un recouvrement existe.

$$\begin{aligned} -\Delta u_1 &= f && \text{dans } \Omega_1 \\ u_1 &= 0 && \text{sur } \partial\Omega_1 \cap \partial\Omega \\ u_1 &= u_2 && \text{sur } \gamma_{12} \\ -\Delta u_2 &= f && \text{dans } \Omega_2 \\ u_2 &= 0 && \text{sur } \partial\Omega_2 \cap \partial\Omega \\ u_2 &= u_1 && \text{sur } \gamma_{21} \end{aligned} \quad . \tag{1.3}$$

Les méthodes de décomposition de domaine avec recouvrement peuvent avoir une meilleure convergence mais la gestion du recouvrement ajoute de la complexité à la mise en œuvre, nécessitant une synchronisation appropriée entre les sous-domaines. Ceci se traduit par une augmentation des communications dans un contexte parallèle et un découpage plus "complexe" à réaliser. Nous n'avons pas considéré cette approche dans nos travaux.

Les méthodes de décomposition de domaine sans recouvrement ont comme avantage la séparation totale du travail à effectuer par sous-domaine, offrant une grande adaptabilité au parallélisme. La mise en place est simple et des méthodes existent pour améliorer sa convergence. Ceci s'accorde avec le but de la thèse ce qui nous a mené à orienter nos travaux vers ce type de méthodes.

Dans le cas des méthodes sans recouvrement, les conditions sur l'interface sont appelées les conditions de transmission. En fonction de ces conditions de transmissions, nous aurons différentes approches de décomposition de domaine (Dirichlet-Neumann, Neumann-Neumann, Dirichlet-Dirichlet). Par exemple, la méthode FETI est dite de type Dirichlet-Dirichlet : Sur chaque sous domaine, nous partons d'une estimation initiale du Flux sur l'interface que nous utilisons tout d'abord pour résoudre un problème de Neumann. Ces solutions locales sont ensuite utilisées comme condition de Dirichlet par la méthode FETI pour trouver la bonne valeur de u à l'interface. En revanche, dans une approche de type

Neumann-Neumann, comme dans les travaux de [3] et [4], nous résolvons d'abord deux problèmes de Dirichlet avec une valeur de u donnée sur l'interface avant de résoudre deux problèmes de Neumann et utiliser la solution pour corriger les valeurs de u (voir [1]).

1.4 FETI-1

La méthode FETI-1 a été introduite par Farhat et Roux [2]. Le domaine de calcul éléments finis est décomposé en sous-domaines et chaque sous domaine est traité indépendamment sur un processeur. Cette décomposition donne lieu à des problèmes locaux et à un problème d'interface. En général, un solveur direct est utilisé sur chaque sous-domaine pour résoudre le problème local tandis qu'un solveur itératif est utilisé pour résoudre le problème sur l'interface. Elle est dite de niveau 1, pour indiquer l'absence d'un problème grossier [5] ; problème grossier qui est souvent utilisé dans les méthodes hybrides pour améliorer la convergence et réduire la coût du calcul. Cette méthode est également dite itérative et duale [1]. Elle est itérative car elle résout le problème par des itérations successives et elle est dite duale en raison de l'introduction des multiplicateurs de Lagrange pour imposer les conditions de continuité entre les sous-domaines. Les itérations donnent des solutions discontinues et cette discontinuité disparaît seulement à la convergence.

Cette méthode présente plusieurs avantages :

1. C'est une approche bien connue et elle est à la base de techniques plus avancées, notamment FETI-DP (DP pour Dual-Primal [6]), que nous avons identifiée comme intéressante pour la suite de nos travaux.
2. Elle offre de bonnes performances pour des problèmes simples.
3. Avec un préconditionnement adapté par sous-domaine, il a été montré que c'est une méthode optimale pour un problème de diffusion scalaire [7].

Nous allons commencer par donner la méthode pour traiter un problème de Poisson, obtenue suite à une modélisation d'un problème d'élasticité dans le cas linéaire, nous utiliserons ainsi un vocabulaire de la mécanique. L'adaptation pour un problème magnétostatique est donnée dans le chapitre 4.

1.4.1 Cas avec 2 sous-domaines

Pour un domaine Ω divisé en deux sous-domaines Ω_1 et Ω_2 avec deux interfaces Γ_1 et Γ_2 et qui partagent Γ_{In}, voir la figure 1.4, le problème en 3D revient à minimiser la fonctionnelle [8] :

$$J(\mathbf{v}) = \frac{1}{2}a(\mathbf{v},\mathbf{v}) - (\mathbf{v},\mathbf{f}) - (\mathbf{v},\mathbf{h})_{\Gamma_{In}}\,, \tag{1.4}$$

1. En réalité, il existe un niveau grossier implicite avec les potentiels flottants, ce qui rend tout de même le solveur optimal.

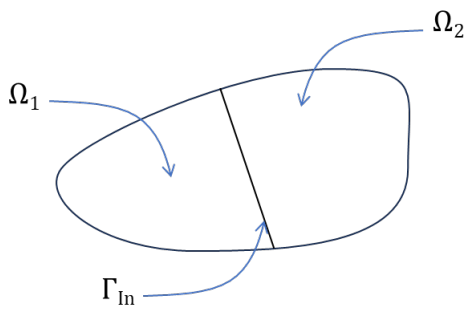

Figure 1.4 – Décomposition en deux sous-domaines

avec

$$a(\mathbf{v},\mathbf{w}) = \int_\Omega \sum_{i,j,k,l} c\, v_{(i,j)} w_{(k,l)}\, \mathrm{d}\Omega\,,$$
$$(\mathbf{v},\mathbf{f}) = \int_\Omega \sum_{i} v_i f_i\, \mathrm{d}\Omega\,,$$
$$(\mathbf{v},\mathbf{h})_{\Gamma_{In}} = \int_{\Gamma_h} \sum_{i} v_i h_i\, \mathrm{d}\Gamma_{In}\,,$$

où, i,j,k,l prennent les valeurs de 1 à 3, $v_{(i,j)} = \frac{v_{ij}+v_{ji}}{2}$ et v_{ij} désigne la dérivée partielle de la i-ème composante de $\mathbf{v}$ par rapport à la j-ème variable spatiale, c est le coefficient d'élasticité, nous le considérons constant. Γ_h est une portion de la frontière, Γ_{In} exclue, sur laquelle les forces de tractions h_i sont considéré comme imposées.

Ce problème est équivalent à trouver les fonctions $\mathbf{u}_1$ et $\mathbf{u}_2$ qui représentent des points stationnaires de :

$$J_1(\mathbf{v}_1) = \frac{1}{2}a(\mathbf{v}_1,\mathbf{v}_1) - (\mathbf{v}_1,\mathbf{f}) - (\mathbf{v}_1,\mathbf{h})_{\Gamma_1}\,,$$
$$J_2(\mathbf{v}_2) = \frac{1}{2}a(\mathbf{v}_2,\mathbf{v}_2) - (\mathbf{v}_2,\mathbf{f}) - (\mathbf{v}_2,\mathbf{h})_{\Gamma_2}\,,$$

et qui satisfont la condition de continuité sur la frontière partagée Γ_{In}, $\mathbf{u}_1 = \mathbf{u}_2$.

Le problème variationnel représenté par ces équations équivaut à trouver le point stationnaire de la fonctionnelle J^* :

$$J^*(\mathbf{v}_1,\mathbf{v}_2,\mu) = J_1(\mathbf{v}_1) + J_2(\mathbf{v}_2) + (\mathbf{v}_1 - \mathbf{v}_2, \mu)_{\Gamma_{In}}\,. \tag{1.5}$$

Cela revient à trouver les potentiels $\mathbf{u}_1$ et $\mathbf{u}_2$ ainsi que le vecteur des multiplicateurs de Lagrange λ qui satisfont pour toute valeur de $\mathbf{v}_1$, $\mathbf{v}_2$ et μ (la solution est un point-selle de la fonctionnelle) :

$$J^*(\mathbf{u}_1,\mathbf{u}_2,\mu) \leqslant J^*(\mathbf{u}_1,\mathbf{u}_2,\lambda) \leqslant J^*(\mathbf{v}_1,\mathbf{v}_2,\lambda)\,. \tag{1.6}$$

Nous pouvons prouver que $\mathbf{u}_1$ et $\mathbf{u}_2$ sont la restriction de la solution $\mathbf{u}$ du problème non partitionné et que, en effet, les équations (1.5) et (1.6) correspondent à un principe variationnel hybride dans lequel la contrainte de continuité inter-sous-domaine $\mathbf{u}_1 = \mathbf{u}_2$ est imposée par le multiplicateur de Lagrange [2].

Si maintenant les champs de déplacement $\mathbf{u}_1$ et $\mathbf{u}_2$ sont exprimés avec des fonctions de base éléments finis appropriées (nous reviendrons sur ces notions dans la section 2.2.3)

$$\begin{aligned} \mathbf{u}_1 &= N u_1 \,, \\ \mathbf{u}_2 &= N u_2 \,, \end{aligned} \tag{1.7}$$

et que, par abus de notation, nous ne faisons pas de distinction entre la notation de λ en continu et celle après discrétisation, le système algébrique obtenu a la forme suivante :

$$\begin{aligned} K_1 u_1 &= f_1 + B_1^t \lambda \,, \\ K_2 u_2 &= f_2 - B_2^t \lambda \,, \\ B_1 u_1 &= B_2 u_2 \,, \end{aligned} \tag{1.8}$$

où K_i, u_i, f_i sont respectivement la matrice de rigidité (de taille $n_i \times n_i$), la solution (de taille n_i) et le second membre (de taille n_i) associés à la discrétisation par éléments finis de Ω_i. Le vecteur λ (de taille n^s est le vecteur des multiplicateurs de Lagrange et B_i (de taille $n_i^s \times n_i$) la matrice de restriction sur l'interface. Le vecteur des multiplicateurs de Lagrange assure les échanges de flux entre les sous-domaines voisins sur l'interface commune ; nous pouvons le voir comme imposer des forces de traction artificielle pour coupler les sous-domaines. La matrice de restriction permet de restreindre le problème sur l'interface entre sous-domaine en gardant les parties de la matrice qui représentent l'interface. En pratique c'est une matrice composée de 0 et de 1 que nous ne construisons pas explicitement. Nous utilisons les indices (i, j) des degrés de liberté concernés pour faire l'extraction et la projection.

Par exemple, prenons le sous-domaine 2 de la figure 1.5 :

L'interface entre les deux sous-domaines est définie par les points 2 et 3. En conséquence la matrice B_2 est définie par

$$B_2 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{1.9}$$

1.4.2 Cas de 2 sous domaines dont 1 flottant

Un sous-domaine est dit flottant si sa matrice de rigidité est singulière. Dans le cas où une condition aux limites de Dirichlet est fixée sur un seul sous-domaine Γ_1, la matrice K_2 du

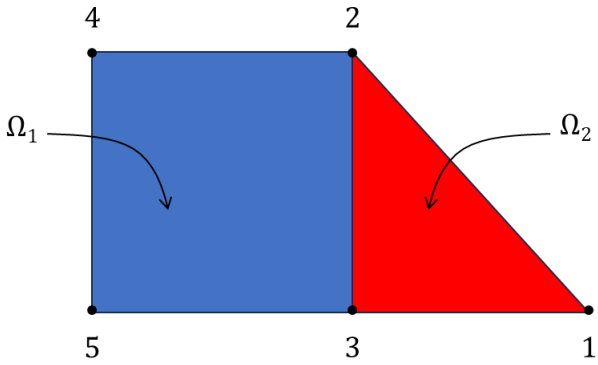

Figure 1.5 – Schéma deux-sous domaines et 5 degrés de liberté

deuxième sous-domaine est nécessairement singulière [2] d'où la nécessité de gérer les sous-domaines flottants. Nous verrons dans le chapitre suivant que cette problématique, gérer les possibles singularités des problèmes locaux, est l'un des points clés pour l'application de la décomposition de domaine en magnétostatique. Les problèmes en magnétostatique nécessitent la résolution de formulations en potentiel scalaire ou en potentiel vecteur, dont l'unicité requiert l'imposition de conditions supplémentaires. Garantir en magnétostatique l'unicité du potentiel est l'équivalent en mécanique à garantir l'unicité du déplacement.

En pratique, cela consiste à calculer la pseudo-inverse de K_2. La pseudo-inverse possède certaines propriétés d'une matrice inverse, mais pas nécessairement toutes (pas d'unicité notamment), elle généralise donc la notion d'inverse aux matrices non inversibles. Plusieurs méthodes existent pour la calculer (voir [9]). La solution pour le sous-domaine flottant est :

$$u_2 = K_2^+ (f_2 - B_2^t \lambda) + E_2 \alpha \,, \tag{1.10}$$

avec E_2 une matrice rectangulaire (de taille $n_2 \times n^r$, n^r est le nombre de modes de corps rigides) dont les colonnes forment une base du noyau de K_2, un vecteur constant pour notre cas. Pour rappel, les modes de corps rigides sont des modes qui ne nécessitent aucune énergie de déformation. Le nombre de ces modes correspond au nombre de valeurs propres nulles de la matrice de rigidité globale, sans tenir compte des conditions aux limites de type Dirichlet. En imposant un nombre suffisant (ou minimal) de conditions aux limites, on peut éliminer ces modes de corps rigides et obtenir ainsi une solution unique. Le terme α est un vecteur (de taille n^r). Pour un problème scalaire le nombre de modes de corps rigides vaut 1 (donc $n^r = 1$, et en élasticité $n^r = 3$). Nous rappelons qu'imposer une condition de Dirichlet en fixant le potentiel en un point permet d'éliminer ces modes de corps rigides et de calculer ainsi une solution unique. Le vecteur α spécifie une combinaison linéaire de ces modes de corps rigides.

2. Sans considérer le cas des conditions mixtes

Le système (1.8) qui décrit le problème en décomposition de domaine devient :

$$\begin{aligned}(B_1K_1^{-1}B_1^t + B_2K_2^+B_2^t)\lambda &= -B_1K_1^{-1}f_1 + B_2(K_2^+f_2 + E_2\alpha)\,,\\ u_1 &= K_1^{-1}(f_1 + B_1^t\lambda)\,,\\ u_2 &= K_2^+(f_2 + B_2^t\lambda) + E_2\alpha.\end{aligned} \tag{1.11}$$

Le calcul de la pseudo-inverse se fait en utilisant les méthodes de décomposition de Cholesky avec fixation des nœuds et régularisation [10]. En termes d'implémentation, cela se résume à la suppression d'une ligne et d'une colonne, ou à l'utilisation des options de MUMPS pour détecter les pivots nuls. Nous avons d'abord utilisé la première méthode, puis nous avons opté pour les options de MUMPS.

1.4.3 Cas général

Pour un cas général avec N_{sd} sous-domaines, le système d'équations s'écrit

$$\begin{aligned}K_ju_j + B_j^t\lambda &= f_j\,, \quad \forall j \in [\![1; N_{\text{sd}}]\!]\,,\\ \sum_{j=1}^{N_{\text{sd}}} B_ju_j &= 0\,,\end{aligned} \tag{1.12}$$

avec B_j la matrice de restriction du sous-domaine j à son interface où cette fois la valeur, si elle est non nulle, peut être de ± 1 pour tenir compte du sens du flux opposé entre deux sous-domaines voisins (dans les cas précédents avec 2 sous-domaines, le signe n'était pas intégré).

Pour les sous-domaines flottants, la solution s'écrit

$$u_j = K_j^+\left(f_j - B_j^t\lambda\right) + E_j\alpha_j\,, \tag{1.13}$$

avec E_j la colonne de la matrice E, voir (1.14), associée au sous-domaine j et α_j le coefficient correspondant dans le vecteur α, l'annexe A.1 donne un exemple.

Sachant que pour un nombre N_f de sous-domaines flottants

$$E = \begin{bmatrix} E_1 & 0 & 0 \\ 0 & ... & 0 \\ 0 & 0 & E_{N_f} \end{bmatrix}. \tag{1.14}$$

La condition d'orthogonalité, qui impose que le second membre soit compatible en (1.12), permet d'ajouter l'équation manquante au système :

$$(E^B)^t\lambda = E^tf\,, \tag{1.15}$$

avec $E_j^B = B_j E_j$, l'exposant B désignant l'interface. En éliminant les inconnues u_j, le système à résoudre avec N_{sd} sous-domaines devient

$$\begin{bmatrix} F & E^B \\ (E^B)^t & 0 \end{bmatrix} \begin{bmatrix} \lambda \\ \alpha \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{N_{\text{sd}}} B_j K_j^+ f_j \\ E^t f \end{bmatrix}, \tag{1.16}$$

avec $F = \sum_{j=1}^{N_{\text{sd}}} B_j K_j^+ B_j^t$.

Après le calcul de λ, nous calculons α avec :

$$\alpha = ((E^B)^t E^B)^{-1} (E^B)^t \left(-F\lambda + \sum_{j=1}^{N_{sd}} B_j K_j^+ f_j \right). \tag{1.17}$$

L'annexe A.1 donne les équations pour l'exemple de la figure 1.6.

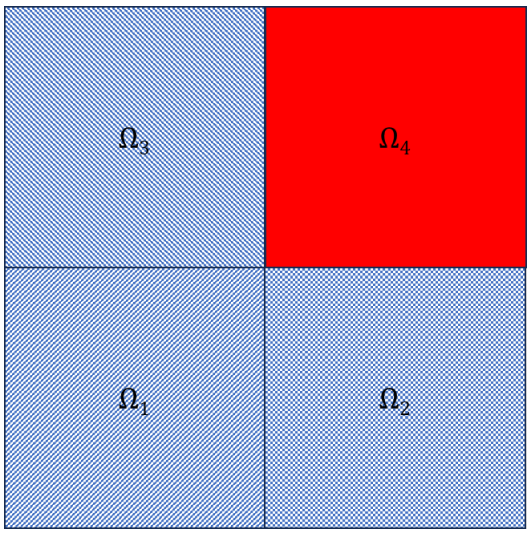

Figure 1.6 – Domaine coupé en 4 sous-domaines dont 3 flottants.

1.4.4 Résolution avec la DDM

La résolution de ce système est faite en utilisant un gradient conjugué projeté et préconditionné [7], voir l'Algorithme 1. Un rappel de la méthode du gradient conjugué ainsi que d'autre méthode itératives sera fait dans le chapitre 3.

Dans cet algorithme, la matrice P représente le préconditionneur et la tolérance *tol* est définie par l'utilisateur.

Algorithm 1 PCPG

Require : $\lambda^{(0)} = E^B((E^B)^t E^B)^{-1} E^t f, r^{(0)} = (\sum_{j=1}^{N_{\text{sd}}} B_j K_j^+ f_j) - F\lambda^{(0)}$
  while $r^t Pr > tol$ **do**
    $w^{(k-1)} \leftarrow [I - E^B((E^B)^t E^B)^{-1}(E^B)^t] r^{(k-1)}$ ▷ Projette
    $z^{(k-1)} \leftarrow Pw^{(k-1)}$ ▷ Préconditionne
    $y^{(k-1)} \leftarrow [I - E^B((E^B)^t E^B)^{-1}(E^B)^t] z^{(k-1)}$ ▷ Projette
    $\gamma^{(k)} \leftarrow y^{(k-1)} w^{(k-1)} / y^{(k-2)} w^{(k-2)}$ ▷ $\gamma^{(1)} \leftarrow 0$
    $s^{(k)} \leftarrow y^{(k-1)} + \gamma^{(k)} s^{(k-1)}$ ▷ $s^{(1)} \leftarrow 0$
    $\beta^{(k)} \leftarrow y^{k-1^t} w^{(k-1)} / s^{k^t} F s^{(k)}$
    $\lambda^{(k)} \leftarrow \lambda^{(k-1)} + \beta^{(k)} s^{(k)}$
    $r^{(k)} \leftarrow r^{(k-1)} - \beta^{(k)} F s^{(k)}$
    $k \leftarrow k + 1$

Conditionnement :

Le conditionnement mesure la sensibilité de la solution d'un problème aux perturbations dans les données. Il fournit une borne supérieure approximative sur l'erreur dans une solution calculée et il peut également être utilisé pour prédire la convergence des méthodes itératives [11, 12]. Pour une matrice A, le conditionnement est défini par :

$$\kappa(A) = \|A\| \|A^{-1}\| ,$$

où $\|\cdot\|$ est une norme matricielle. Pour la norme euclidienne, le conditionnement d'une matrice symétrique définie positive est équivalent au rapport entre la plus grande et la plus petite valeur propre. Un conditionnement proche de 1 indique que la matrice est bien conditionnée et une matrice dont le conditionnement est élevé est dite mal conditionnée.

Préconditionnement :

Un préconditionneur P d'une matrice K est une matrice telle que le conditionnement de $P^{-1}K$ est plus petit que celui de K. Nous rappelons que, dans le cas de matrices symétriques définies positives, la réduction du conditionnement se traduit par une convergence plus rapide des méthodes itératives. Dans le cas du gradient conjugué notamment, l'analyse de convergence donne [13] :

Pour $e_k = x_k - x_*$, l'erreur entre la solution obtenue à l'iteration k et la solution exacte du problème $Ax = b$, nous avons la majoration suivante

$$\|e_k\|_A \leqslant 2 \left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1} \right)^k \|e_0\|_A . \tag{1.18}$$

La convergence du gradient conjugué dépend donc directement de $\sqrt{\kappa(A)}$.

Si les éléments et les sous-domaines ont des formes et des tailles régulières, alors la matrice de la méthode FETI sans précondionnement appliquée à un problème de Poisson a un conditionnement κ majoré de la manière suivante [5] :

$$\kappa \leqslant C\frac{H}{h}, \tag{1.19}$$

avec C une constante, H la "taille" du sous-domaine et h la "taille" des éléments.

Deux préconditionneurs "classiques" peuvent être utilisés : le préconditionneur de Dirichlet ou des approximations de celui-ci de type "lumped preconditioner" [5].

Dans un premier temps, nous nous focalisons sur le préconditionneur de Dirichlet. Ce préconditionneur permet de limiter le conditionnement de la matrice à :

$$\kappa \leqslant \left(1 + \log\frac{H}{h}\right)^b, \tag{1.20}$$

où, en général, $b = 3$. Dans certaines situation, $b = 2$ (comme l'absence de nœuds partagés par plus de deux sous-domaines ou si la matrice A présente certaines propriétés en 2D [14]). Dans les deux cas, on obtient un préconditionnement (quasi-)optimal [5], *i.e.* que le conditionnement du problème à résoudre est (quasi-)indépendant du pas de discrétisation et du nombre de sous-domaines.

Le préconditionneur de Dirichlet peut s'écrire :

$$D_I^{-1} = \sum_{s=1}^{N_{\text{sd}}} B_s \begin{bmatrix} 0 & 0 \\ 0 & S_s \end{bmatrix} B_s^t \quad , \tag{1.21}$$

avec S_s le *complément de Schur* local obtenu en éliminant les degrés de liberté internes au sous-domaine Ω_s et en ne conservant que ceux sur le bord de Ω_s.

Pour être plus explicite : on discrétise sur un domaine Ω une équation aux dérivées partielles par la méthode des éléments finis et on obtient une matrice de rigidité K et un second membre f. On nomme u la solution du système linéaire $Ku = f$. On distingue les degrés de liberté internes à Ω avec un indice I de ceux sur le bord avec un indice B. Avec une numérotation appropriée, K, u et f auront la forme par blocs suivante

$$K = \begin{bmatrix} K_{II} & K_{IB} \\ K_{IB}^t & K_{BB} \end{bmatrix}, \quad u = \begin{bmatrix} u_I \\ u_B \end{bmatrix}, \quad f = \begin{bmatrix} f_I \\ f_B \end{bmatrix}. \tag{1.22}$$

En éliminant les degrés de liberté internes, on obtient un système linéaire de solution u_B avec pour matrice le *complément de Schur*

$$S = K_{BB} - K_{IB}^t K_{II}^{-1} K_{IB}, \tag{1.23}$$

et pour second membre

$$g = f_B - K_{IB}^t K_{II}^{-1} f_I. \tag{1.24}$$

On peut donc d'abord résoudre le système linéaire $Su_B = g$ pour déterminer u_B et revenir ensuite à u_I :

$$u_I = K_{II}^{-1}(f_I - K_{IB}u_B). \tag{1.25}$$

En pratique, nous n'utilisons pas la matrice B_s mais un système d'extraction/projection en utilisons les indices, d'une manière similaire à B_s et directement à partir du complément de Schur [3].

Pondération :

Pour les équations du type $-\nabla \cdot \rho \nabla u = f$, le coefficient de diffusion ρ peut jouer un rôle sur l'ordre de grandeur des coefficients de la matrice. L'analyse faite par Klawonn et Widlund [15] montre que FETI avec le "préconditionneur de Dirichlet avec pondération" permet d'obtenir $b = 2$ dans (1.20) [5]. La pondération utilisée est fondée sur la définition de poids scalaire associé à chaque nœud de l'interface.

Pour un sous domaine Ω_i, a_i sous-domaines voisins, et pour tout nœud x_j appartenant a l'interface, la fonction de poids $W_i(x_j)$ est définie comme :

$$W_i(x_j) = \frac{\omega_i(x_j)}{\sum_{k=1}^{k=a_i} \omega_k(x_j)} \text{tel que } \omega_i(x_j) \text{ est un scalaire.} \tag{1.26}$$

Ces poids seront incorporés dans la matrice de passage interface locale/globale pondérée $\widetilde{B}_s$ telle que :

$$\widetilde{B}_s = W_s B_s \,, \tag{1.27}$$

avec W_s la matrice diagonale par sous-domaine.

Il existe plusieurs méthodes de pondération [5]. Nous en considérerons trois :

1. Pondération avec la multiplicité : si le coefficient de diffusion (la perméabilité magnétique dans notre cas) ne varie pas ou varie peu, alors $\omega_i(x_j) = 1$, la fonction de poids sera donc l'inverse de la multiplicité du nœud.
2. Pondération avec coefficient : si le coefficient de diffusion varie, $\omega_i(x_j)$ est choisi ressemblant au coefficient de diffusion.
3. Pondération avec rigidité : $\omega_i(x_j)$ peut être choisi comme l'entrée diagonale de la matrice de rigidité associée au nœud x_j.

3. Sans construire la matrice avec les zéros

Effet de la perméabilité :

En magnétostatique, $\mu_{r,i}$ est la perméabilité relative du sous-domaine i (voir 2.1.2). Lors de la pondération, la prise en considération de cette perméabilité doit être faite. Ceci est essentiel quand nous avons des sous-domaines avec une perméabilité différente entre eux. La fonction de poids $W_i(x_j)$ devient :

$$W_i(x_j) = \frac{\omega_i(x_j)}{\sum_{k=1}^{k=N} \omega_k(x_j)} \, \mu_{r,i}^{-1} \, . \tag{1.28}$$

Facteur Q :

Nous pouvons introduire une matrice Q symétrique définie positive, que nous appelons matrice de pondération. Cette matrice intervient dans la projection M tel que :

$$M = I - QE^B((E^B)^t Q E^B)^{-1}(E^B)^t \, . \tag{1.29}$$

Une analyse du choix de la matrice Q se trouve dans [15]. Si la variation des coefficients est "petite", nous pouvons poser $Q = I$. Les différentes analyses montrent que pour les grandes variations, nous pourrons choisir $Q = D_I^{-1}$, voir (1.21). Pour notre cas, nous avons choisi de prendre la matrice diagonale du préconditionneur D_I^{-1} comme matrice de pondération. Un choix peu coûteux car il simplifie l'implémentation tout en gardant certaines caractéristiques du préconditionneur [5].

1.5 FETI-DP

Les méthodes FETI dual-primal (FETI-DP) ont été introduites pour la première fois par Farhat, Lesoinne, Le Tallec, Pierson et Rixen dans [16] comme un développement ultérieur d'une méthode FETI adaptée aux problèmes du quatrième ordre tels que des problèmes de coques ou de plaques en mécanique des structures. L'idée principale est de maintenir les inconnues aux sommets des sous-domaines (coins, points de croisement ou "crosspoints") en tant que variables *primales*, c'est-à-dire de ne pas introduire de multiplicateurs de Lagrange sur ces points. De cette manière, après l'élimination de ces inconnues primales qui définissent un problème grossier, les matrices par sous-domaine résultantes sont toujours inversibles. La motivation d'utiliser des contraintes aux coins provient du fait que la méthode FETI-1 ne convergeait pas bien pour les problèmes de plaques et de coques. Le choix des "variables primales" a été le sujet de plusieurs études pour situer le juste milieu entre la taille de problème grossier et le conditionnement. Dans un premier temps nous allons expliquer le cas simple avec les "crosspoints" définis comme les points partagés par plus de 2 sous-domaines. Nous verrons après comment les remplacer par des contraintes exprimées en termes de moyennes sur les arêtes (et les faces). De telles contraintes primales

sur les arêtes offrent généralement de meilleures performances en termes de convergence des algorithmes FETI-DP que lorsque seules des contraintes primales aux sommets sont utilisées [1].

Nous allons voir comment écrire le système à résoudre pour la méthode FETI-DP. Pour un domaine Ω découpé en N_{sd} sous-domaines (voir la figure 1.7), nous notons toujours K_s, u_s et f_s les matrice de rigidité, solution et second membre associés au problème dans le sous-domaine s. Dans les équations, nous préciserons également par un indice c les degrés de liberté associés aux variables primales (ou aux "cross points", voir la figure 1.7) et r pour le reste des degrés de liberté internes *et* sur l'interface.

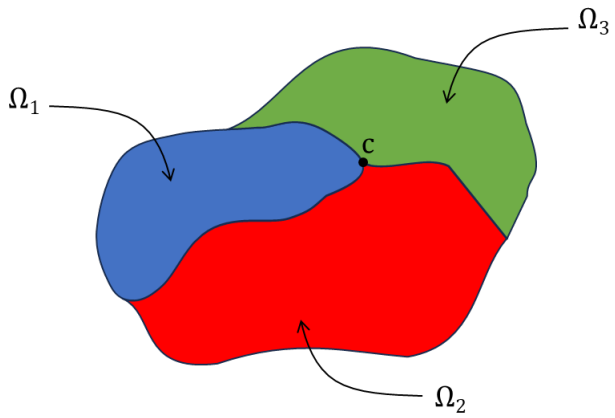

Figure 1.7 – Définition de point coin (point marqué C) comme variable Primale, les autres points sur l'interface seront associés aux variables Duales.

Nous allons ainsi distinguer plusieurs sous-ensemble d'indices dans la matrice K_s et les vecteurs u_s et f_s :

$$K_s = \begin{bmatrix} K_{s,rr} & K_{s,rc} \\ K_{s,rc}^t & K_{s,cc} \end{bmatrix}, \quad u_s = \begin{bmatrix} u_{s,r} \\ B_{s,c}u_c \end{bmatrix}, \quad f_s = \begin{bmatrix} f_{s,r} \\ f_{s,c} \end{bmatrix}. \tag{1.30}$$

Le vecteur u_c regroupe l'ensemble des degrés de liberté attachés aux variables primales et la matrice booléenne $B_{s,c}$ permet de récupérer les composantes concernant le sous-domaine Ω_s. On notera également $B_{s,r}$ l'équivalent de la matrice B_s de la méthode FETI-1 mais restreinte aux variables pour lesquelles il est encore nécessaire d'imposer une contrainte.

En utilisant ces notations, le système d'équations s'écrit :

$$K_{s,rr}u_{s,r} + K_{s,rc}B_{s,c}u_c = f_{s,r} - B_{s,r}^t\lambda, \quad \forall s \in [\![1; N_{\text{sd}}]\!], \tag{1.31}$$

$$\sum_{s=1}^{s=N_{\text{sd}}} B_{s,c}^t K_{s,rc}^t u_{s,r} + \sum_{s=1}^{s=N_{\text{sd}}} B_{s,c}^t K_{s,cc} B_{s,c} u_c = \sum_{s=1}^{s=N_{\text{sd}}} B_{s,c}^t f_{s,c}, \tag{1.32}$$

avec la condition de continuité sur l'interface :

$$\sum_{s=1}^{s=N_{\text{sd}}} B_{s,r} u_{s,r} = 0. \tag{1.33}$$

Le vecteur de multiplicateurs λ est utilisé pour toutes les variables d'interface qui n'ont pas été retenues comme primales et pour lesquelles il est nécessaire d'imposer une contrainte.

Le problème d'interface équivalent au problème (1.16) est :

$$\begin{bmatrix} F_{rr} & F_{rc} \\ F_{rc}^t & -K_{cc} \end{bmatrix} \begin{bmatrix} \lambda \\ u_c \end{bmatrix} = \begin{bmatrix} d_r \\ f_c \end{bmatrix}, \tag{1.34}$$

avec F_{rr}, F_{rc}, K_{cc}, d_r et f_c définis comme suit [16] :

$$F_{rr} = \sum_{s=1}^{s=N_{\text{sd}}} B_{s,r} K_{s,rr}^{-1} B_{s,r}^t \,, \tag{1.35}$$

$$F_{rc} = \sum_{s=1}^{s=N_{\text{sd}}} B_{s,r} K_{s,rr}^{-1} K_{s,rc} B_{s,c} \,, \tag{1.36}$$

$$K_{cc} = \sum_{s=1}^{s=N_{\text{sd}}} B_{s,c}^t K_{s,cc} B_{s,c} - \sum_{s=1}^{s=N_{\text{sd}}} \left(K_{s,rc} B_{s,c}\right)^{\text{T}} K_{s,rr}^{-1} \left(K_{s,rc} B_{s,c}\right) , \tag{1.37}$$

$$d_r = \sum_{s=1}^{s=N_{\text{sd}}} B_{s,r} K_{s,rr}^{-1} f_{s,r} \,, \tag{1.38}$$

$$f_c = \sum_{s=1}^{s=N_{\text{sd}}} B_{s,c}^t f_{b_c}^s - \sum_{s=1}^{s=N_x} B_{s,c}^t K_{s,rc}^t K_{s,rr}^{-1} f_{s,r} \,. \tag{1.39}$$

Le problème ci-dessus est un problème Dual-Primal car il lie les inconnues du multiplicateur de Lagrange λ ''dual'' aux degrés de liberté ''primaux'' u_c. En éliminant u_c, il peut cependant être transformé en un problème dual d'interface dont la matrice est symétrique définie positive.

$$(F_{rr} + F_{rc} K_{cc}^{-1} F_{rc}^t)\lambda = d_r - F_{rc} K_{cc}^{-1} f_c \,. \tag{1.40}$$

La méthode FETI-DP consiste à transformer le problème dans un premier temps sous la forme Dual-Primal et de résoudre ensuite (1.40) avec une méthode de gradient conjugué.

Préconditionneur :

Le préconditionneur utilisé pour la méthode FETI-DP est le même que pour la méthode FETI-1, il s'écrit comme :

$$D_I^{-1} = \sum_{s=1}^{s=N_{\text{sd}}} B_{s,r} \begin{bmatrix} 0 & 0 \\ 0 & S_{s,r} \end{bmatrix} B_{s,r}^T \,. \tag{1.41}$$

La matrice $S_{s,r}$ est le complément de Schur où ont été éliminés à la fois les degrés de liberté internes *et* les variables primales.

De la même manière que pour FETI, nous appliquons une pondération au préconditionneur qui tient compte du coefficient de diffusion.

FETI-1 vs FETI-DP :

La différence entre FETI-1 et FETI-DP peut se résumer en quelques points :

- L'application des contraintes supplémentaires à chaque itération rend toujours les problèmes locaux non singuliers et fournit simultanément un problème global grossier sous-jacent pour FETI-DP. La gestion des sous-domaines flottants, nécessaire pour les méthodes FETI-1, peut ne pas être une tâche triviale pour certains problèmes plus complexes et ce qui représente un avantage pour la méthode FETI-DP.
- Les méthodes FETI-DP ne nécessitent pas l'introduction d'une matrice de pondération Q, qui intervient dans la projection pour les algorithmes FETI-1.
- Les méthodes FETI-1 sont des algorithmes de gradient conjugué projeté qui ne peuvent pas commencer à partir d'un vecteur initial arbitraire. Ce choix initial peut être éloigné de la solution exacte. En revanche, les méthodes FETI-DP sont des algorithmes de gradient conjugué préconditionnés standard et peuvent donc utiliser un vecteur initial arbitraire.
- Le problème grossier construit pour FETI-DP a besoin des contributions des différents sous domaines, ceci ajoute de la complexité lors de l'implémentation.

1.6 Autre écriture de FETI-DP

Dans cette partie, nous allons présenter une autre approche pour formuler le problème FETI-DP. L'objectif est de mettre en évidence plus clairement la relation entre cette méthode et d'autres méthodes similaires. De nombreux auteurs [5, 17, 18, 19] présentent la méthode FETI-DP comme un moyen de résoudre le problème écrit avec le complément de Schur, en ayant effectué auparavant une opération de *condensation statique*. Le principe est le suivant :

Au lieu de résoudre

$$Ku = f \,, \tag{1.42}$$

avec K la matrice éléments finis sur le domaine global Ω, u la solution correspondante et f le second membre, la condensation statique est mise en œuvre sur les différents

sous-domaines pour obtenir un système uniquement à l'interface entre les sous-domaines

$$\sum_{i=1}^{i=N_{\text{sd}}} (R_i^{\text{T}} S_i R_i) u_B = \sum_{i=1}^{i=N_{\text{sd}}} R_i^{\text{T}} g_i \,, \tag{1.43}$$

avec R_i la matrice de restriction du vecteur sur l'interface complète vers un vecteur sur le bord de Ω_i uniquement. Le complément de Schur S_i est le même que celui défini pour le préconditionneur de Dirichlet (1.21), de même que les seconds membres locaux g_i définis en (1.24). Le vecteur u_B regroupe l'ensemble des degrés de liberté sur l'interface. Une fois u_B déterminé, on peut résoudre dans chaque sous-domaine Ω_i

$$u_{i,I} = K_{i,II}(f_{i,I} - K_{i,IB} R_i u_B) \,, \tag{1.44}$$

avec l'indice I pour désigner les degrés de liberté intérieurs et B pour ceux sur le bord.

On pourra également écrire (1.43) de manière plus concise

$$R^T S R u_B = R^T g \,, \tag{1.45}$$

avec

$$S = \begin{bmatrix} S_1 & & 0 \\ & \ddots & \\ 0 & & S_{N_{\text{sd}}} \end{bmatrix}, \quad g = \begin{bmatrix} g_1 \\ \vdots \\ g_{N_{\text{sd}}} \end{bmatrix}, \quad R = \begin{bmatrix} R_1 \\ \vdots \\ R_{N_{\text{sd}}} \end{bmatrix}. \tag{1.46}$$

Les auteurs de [5, 17, 18, 19] définissent également les espaces W, $\widehat{W}$ et $\widetilde{W}$ comme illustré à la figure 1.8 et $\widehat{W} \subset \widetilde{W} \subset W$. Si W_i désigne l'espace des degrés de liberté sur le bord du sous-domaine Ω_i, l'espace W désigne l'ensemble des degrés de liberté sur l'interface considérés de manière indépendante $W = W_1 \times \ldots \times W_{N_{\text{sd}}}$. L'espace $\widehat{W}$ désigne l'espace des degrés de liberté sur l'interface mais avec toutes les contraintes de continuité vérifiés ; c'est l'espace dans lequel est défini le vecteur u_B. Enfin l'espace $\widetilde{W}$ est un espace intermédiaire où seul un nombre réduit de contraintes de continuité est assuré.

Si on note $\widetilde{I}$ l'injection naturelle de $\widetilde{W}$ dans W, l'opérateur S considéré sur $\widetilde{W}$ pourra s'écrire

$$\widetilde{S} = \widetilde{I}^T S \widetilde{I} \,. \tag{1.47}$$

De même l'opérateur pour les contraintes restant à imposer sur l'interface entre les sous-domaines aura la forme

$$\widetilde{B} = B \widetilde{I} \,, \tag{1.48}$$

où B est l'opérateur de saut qui peut se décliner localement dans chaque sous-domaine i en matrice B_i, voir notamment le système (1.16). Le problème à résoudre en intégrant les

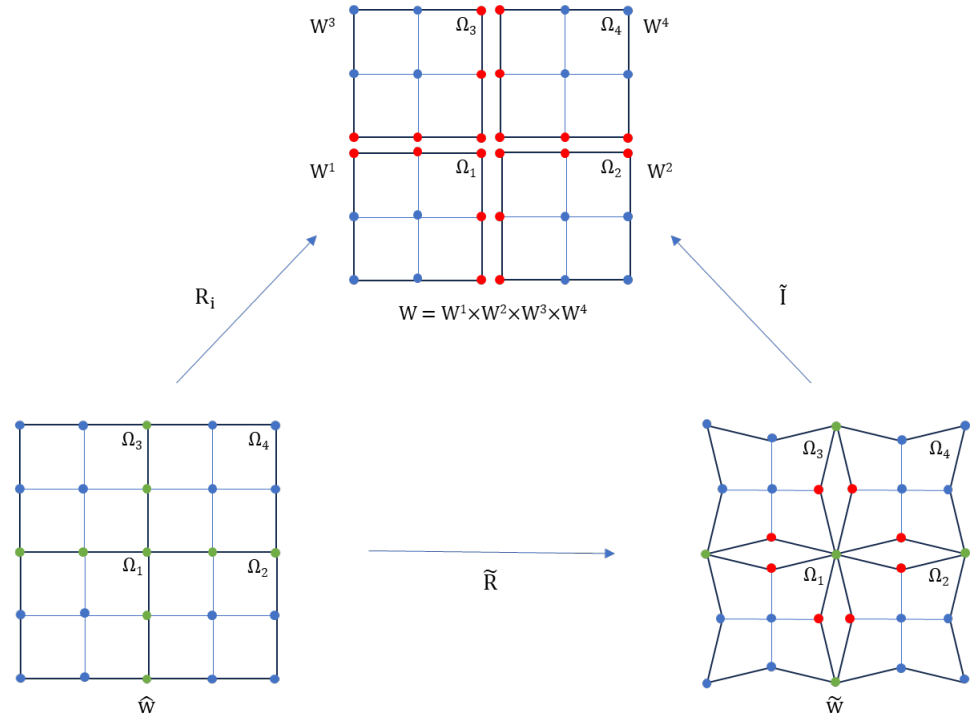

Figure 1.8 – Illustration des différents espaces W, $\widehat{W}$, $\widetilde{W}$ et des opérateurs de passage.

contraintes restantes sera de trouver $\widetilde{u}_B$ et λ vérifiant :

$$\begin{aligned} \widetilde{S}\widetilde{u}_B + \widetilde{B}^T\lambda &= \widetilde{g} \\ \widetilde{B}\widetilde{u}_B &= 0 \end{aligned}, \tag{1.49}$$

avec $\widetilde{g} = \widetilde{I}^t g$. En éliminant $\widetilde{u}_B$ grâce à la première équation de (1.49), on obtient le système dual

$$\widetilde{B}\widetilde{S}^{-1}\widetilde{B}^t\lambda = \widetilde{B}\widetilde{S}^{-1}\widetilde{g}\,. \tag{1.50}$$

Si nous utilisons un gradient conjugué preconditionné sur le système (1.50), le préconditionneur de Dirichlet FETI-DP s'écrira sous la forme

$$P_{FETI-DP} = \widetilde{B}\widetilde{S}\widetilde{B}^t\,. \tag{1.51}$$

Remarque : En réalité, le préconditionneur s'écrit avec $P_{FETI-DP} = B_D\widetilde{S}B_D^t$ et B_D un opérateur voisin de $\widetilde{B}$ avec des poids.

Lorsque ce préconditionneur est appliqué au système (1.50), l'opérateur preconditionné est donnée par :

$$\left(\widetilde{B}\widetilde{S}\widetilde{B}^t\right)\left(\widetilde{B}\widetilde{S}^{-1}\widetilde{B}^t\right). \tag{1.52}$$

1.7 BDDC

La méthode BDDC, pour Balancing Domain Decomposition by Constraints [20] peut être vu comme l'approche "primale" correspond à la méthode FETI-DP. On cherche ici à résoudre "directement" le système (1.43) sans recourir aux multiplicateurs de Lagrange. On s'appuie alors, pour cette résolution, sur un préconditionneur de la forme suivante

$$P_{BDDC} = E\widetilde{S}^{-1}E^t\,, \tag{1.53}$$

où $\widetilde{S}$ est définie en (1.47) et E est une matrice pondérant les valeurs de différents sous-domaines voisins pour obtenir une unique valeur sur l'interface. On vérifie notamment que

$$ER = I\,, \tag{1.54}$$

où R est définie en (1.46) et I est la matrice identité. On observe également que RE est un opérateur de projection sur l'espace des fonctions continues à l'interface.

L'opérateur préconditionné pour la methode BDDC est :

$$\left(\widetilde{E}\widetilde{S}^{-1}E^t\right)\left(R^t\widetilde{S}R\right). \tag{1.55}$$

On retrouve ainsi des éléments communs à la méthode FETI-DP, notamment la matrice $\widetilde{S}^{-1}$ dont l'implémentation en pratique nécessite quelques précisions, et la même forme :

$$\left(LA^{-1}L^t\right)\left(T^tAT\right)\,, \tag{1.56}$$

avec T et L des opérateurs et $A = \widetilde{S}$.

Implémentation de $\widetilde{S}^{-1}$ Pour le i-ème sous-domaine, la matrice C_i contient l'ensemble des contraintes définissant les degrés de libertés associés aux variables primales sur ce sous-domaine (par exemple, ce pourrait être la valeur en un nœud particulier ou la moyenne sur un ensemble de nœuds). Ainsi, si $\varphi_{i,k}$ désigne un vecteur local au sous-domaine associé au k-ième degré de liberté primal local, on aura :

$$C_i\varphi_{i,k} = e_{i,k}\,, \tag{1.57}$$

avec $e_{i,k}$ le k-ième vecteur de la base canonique de $\mathbb{R}^{N_{c,i}}$ et $N_{c,i}$ le nombre de contraintes locales (ou nombre de degrés de liberté primaux).

Si en outre, on impose à la fonction $\varphi_{i,k}$ une "condition d'énergie minimale", le problème

suivant permet de déterminer localement $\varphi_{i,k}$:

$$\begin{bmatrix} S_i & C_i^t \\ C_i & 0 \end{bmatrix} \begin{bmatrix} \varphi_{i,k} \\ \mu_{i,k} \end{bmatrix} = \begin{bmatrix} 0 \\ e_{i,k} \end{bmatrix}. \tag{1.58}$$

On peut ainsi construire sous-domaine par sous-domaine une base grossière globale à support local Φ et qui appartient à $\widetilde{W}$.

La référence [20] donne plus de détails sur la construction de la matrice C_i. On peut aussi trouver des explications complémentaires dans [5, Chapitre 5].

Le choix d'une base qui minimise l'énergie permet notamment d'écrire $\widetilde{W}$ comme la somme de deux sous-espaces complémentaires et S-orthogonaux sur lesquels des corrections vont pouvoir être construites :

$$\widetilde{W} = \text{vect}(\Phi) \oplus_S W_\Delta \,, \tag{1.59}$$

où $\text{vect}(\Phi)$ est le sous-espace engendrée par la base grossière Φ et

$$W_\Delta = \{w \in W \,|\, \text{les degrés de liberté primaux de } w \text{ s'annulent}\} \,. \tag{1.60}$$

Une fois la base grossière globale construite, on peut ainsi implémenter l'action de $\widetilde{S}^{-1}$ en deux étapes :

1. une correction au niveau du problème grossier ;
2. une correction sur chacun des sous-domaines.

L'algorithme 2 décrit plus précisément l'implémentation de $\widetilde{S}^{-1}$.

Algorithm 2 Action de $\widetilde{S}^{-1}$

Entrée : $r \in W$

1. Trouver u_c telle que $\Phi^t S \Phi u_c = \Phi^t r$.
2. Pour tout i, trouver $z_i \in W_i$ telle que

$$\begin{bmatrix} S_i & C_i^t \\ C_i & 0 \end{bmatrix} \begin{bmatrix} z_i \\ \zeta_i \end{bmatrix} = \begin{bmatrix} r_i \\ 0 \end{bmatrix}.$$

Sortie : $\widetilde{S}^{-1} r = \Phi u_c + z$

1.8 Autres méthodes de décomposition de domaine

Nous allons brièvement aborder deux autres méthodes présentées dans des travaux récents : la méthode LATIN [21] et la méthode FETI hybride [22].

La méthode LATIN (Large Time INcremental method) est une technique de décomposition de domaine qui a été initialement développée pour résoudre des problèmes en mécanique. Elle est particulièrement bien adaptée aux problèmes non linéaires, comme le démontrent les travaux dans [21], qui soulignent également son adaptation réussie aux problèmes en magnétostatique. Un point commun entre la méthode LATIN et la méthode FETI hybride est l'utilisation de conditions aux limites mixtes à l'interface entre les sous-domaines. Étant donné que notre travail se concentre principalement sur les méthodes FETI, nous avons fourni davantage de détails sur l'utilisation de ces conditions dans la méthode FETI hybride dans l'annexe A.2.

1.9 Conclusion

Le but de cette thèse est de développer une méthode de décomposition de domaine exploitant la programmation parallèle et spécifiquement adaptée aux défis rencontrés dans la simulation magnétostatique. Dans ce chapitre, nous avons examiné différentes méthodes de décomposition de domaine et exploré leurs interconnexions. Nous avons fourni une description détaillée des différents composants de la méthode FETI-1, que nous allons expérimenter en premier lieu.

La méthode FETI-1 a été choisie car elle répondait le mieux à notre objectif d'avoir une stratégie de résolution entièrement parallèle, ce qui est précisément le principe fondateur de cette méthode. Elle permet de paralléliser l'ensemble du processus de résolution, y compris ses différentes étapes, en une seule fois. De plus, elle constitue une base solide pour des méthodes telles que FETI-DP, que nous avons également explorée et pour lesquelles nous pouvons trouver des implémentations pour comparaison, dans la bibliothèque PETSc par exemple. Les dernières sections de ce chapitre explorent la possibilité d'utiliser les développements effectués pour mettre en œuvre d'autres méthodes telles que BDDC, également largement utilisée. Elles offrent des solutions aux diverses difficultés que nous pouvons rencontrer en magnétostatique, telles que la non-linéarité, les éléments finis d'arête et même le maillage non conforme.

La référence [1] offre une présentation assez complète des méthodes de décomposition de domaine les plus utilisées. La référence [5] se concentre davantage sur la méthode FETI. FETI-1 est décrite dans [2], et dans [7], nous trouvons une étude des propriétés de convergence de la méthode. Le travail [10] traite la gestion des sous-domaines flottants. La méthode FETI-DP est détaillée dans [16] et élargie dans [23]. Dans [24], des éclaircissements sur l'implémentation de la méthode dans PETSc sont fournis.

Notre travail s'appuie sur les travaux de [25], [26], [27], et [6] pour FETI-DP dans des contextes non linéaires. FETI-DP pour les éléments finis d'arête en 2D est présenté dans [28] et pour le 3D dans [29].

La méthode BDD est décrite dans [30], et le préconditionneur BDDC est expliqué dans [20]. La référence [31] aborde le problème en électromagnétisme.

Dans le prochain chapitre, nous allons poser le problème magnétostatique traité dans cette thèse.

Chapitre 2

Formulations et discrétisations éléments finis pour la magnétostatique

La simulation électromagnétique repose sur la résolution des équations de Maxwell, qui décrivent le comportement électromagnétique d'un système, en utilisant des méthodes de résolution numériques couplées à des techniques de discrétisation. Ces méthodes transforment les équations aux dérivées partielles en systèmes d'équations algébriques, dont la solution fournit une approximation des champs électromagnétiques. La méthode numérique la plus couramment utilisée en électromagnétisme basses fréquences est la méthode des éléments finis.

Dans ce deuxième chapitre, nous présentons le principe de la modélisation des problèmes en magnétostatique. Nous énonçons les hypothèses de départ qui permettent de formuler le problème dans le régime statique. De plus, nous effectuons une étude des différentes formulations du problème de magnétostatique, toutes équivalentes au modèle initial, mais se distinguant par l'utilisation de potentiels scalaire ou vectoriel. Nous exposons également les particularités de chacune des formulations qui auront des répercussions sur la méthode de décomposition de domaine. Enfin, nous allons traiter la transition vers les équations à résoudre en utilisant la méthode des éléments finis.

2.1 Équations de Maxwell et formulations en magnétostatique

2.1.1 Équations de Maxwell

Nous allons commencer par un rappel des équations de Maxwell qui décrivent les comportements électromagnétiques, en particulier magnétostatiques, des systèmes. Elles constituent un système d'équations aux dérivées partielles qui lient les phénomènes magnétiques aux phénomènes électriques, et qui unifient les principes de l'électromagnétisme. Ces équations sont les suivantes :

1. La loi de Maxwell-Ampère énonce que les champs magnétiques peuvent être engendrés de deux manières : par les courants électriques et par la variation temporelle d'un champ électrique.
2. La loi de Maxwell-Faraday traduit le phénomène fondamental d'induction électromagnétique.
3. La loi de Maxwell-Thomson exprime que le flux du champ magnétique à travers la frontière délimitant un volume est toujours nul.
4. La loi de Maxwell-Gauss donne la divergence du champ électrique en fonction de la densité de la charge électrique.

Les équations de Maxwell dans les milieux continus [32, 33] s'écrit telles que :

$$\operatorname{rot} \mathbf{h} = \mathbf{j} + \partial_t \mathbf{d}\,, \tag{2.1}$$

$$\operatorname{rot} \mathbf{e} = -\partial_t \mathbf{b}\,, \tag{2.2}$$

$$\operatorname{div} \mathbf{b} = 0\,, \tag{2.3}$$

$$\operatorname{div} \mathbf{d} = \rho_v\,. \tag{2.4}$$

Où les champs $\mathbf{h}$ (en $\mathrm{A\,m^{-1}}$) et $\mathbf{d}$ (en $\mathrm{C\,m^{-2}}$), appelés champ magnétique et déplacement électrique. $\mathbf{b}$ est l'induction magnétique (en T, *i.e* tesla), $\mathbf{e}$ est le champ électrique (en $\mathrm{V\,m^{-1}}$), $\mathbf{j}$ est la densité de courant de conduction (en $\mathrm{A\,m^{-2}}$), et ρ_v est la densité volumique de charge électrique (en $\mathrm{C\,m^{-3}}$).

2.1.2 Lois de comportement

Les lois de comportement des matériaux décrivent comment les matériaux réagissent aux champs électrique et magnétique. Ces lois varient en fonction des propriétés des matériaux et des conditions environnementales. Ces trois lois sont :

1. la loi de comportement magnétique, qui relie le champ magnétique à l'induction magnétique.
2. la loi de comportement électrique, qui relie le déplacement électrique au champ électrique.
3. la loi d'Ohm, qui relie le champ électrique à la densité de courant.

Pour des matériaux isotropes, elles s'écrivent [32, 34] :

$$\mathbf{b} = \mu \mathbf{h} + \mathbf{b_r}\,, \tag{2.5}$$

$$\mathbf{d} = \varepsilon \mathbf{e}\,, \tag{2.6}$$

$$\mathbf{j} = \sigma \mathbf{e}\,, \tag{2.7}$$

où μ est la perméabilité magnétique (en $\mathrm{H\,m^{-1}}$), ε est la permittivité électrique (en $\mathrm{F\,m^{-1}}$), $\mathbf{b_r}$ est l'induction magnétique rémanente due aux aimants permanents (en T) et σ est la conductivité électrique (en $\mathrm{S\,m^{-1}}$).

Pour les matériaux non linéaires (saturables), la perméabilité magnétique est une fonction de h, le module de $\mathbf{h}$. La forme généralisée de la loi de comportement magnétique est donc

$$\mathbf{b} = \mu(h)\mathbf{h} + \mathbf{b_r}\,. \tag{2.8}$$

2.1.3 Conditions de transmission

Nous définissons les conditions de transmission (ou conditions d'interface) [35], *i.e.* les conditions qui assurent la continuité des champs électromagnétiques lors du passage d'un milieu à un autre.

Soient deux milieux Ω_1 et Ω_2 séparés par une interface Γ, voir la figure 2.1.

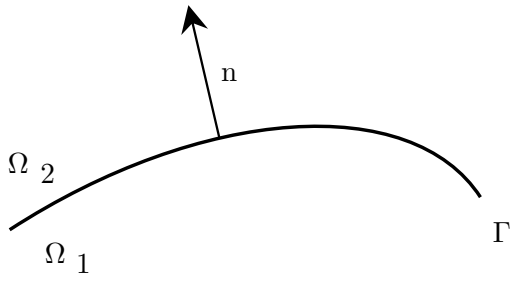

Figure 2.1 – Interface entre deux milieux Ω_1 et Ω_2

Les équations de Maxwell (2.1), (2.2), (2.3) et (2.4) peuvent être intégrées sur des volumes incluant des portions de la surface Γ. L'application du théorème de la divergence ou du théorème de Stokes [32] implique alors les conditions suivantes :

$$\mathbf{n} \times (\mathbf{h}_2 - \mathbf{h}_1)|_\Gamma = \mathbf{j_{surf}}\,, \tag{2.9}$$

$$\mathbf{n} \times (\mathbf{e}_2 - \mathbf{e}_1)|_\Gamma = 0\,, \tag{2.10}$$

$$\mathbf{n} \cdot (\mathbf{b}_2 - \mathbf{b}_1)|_\Gamma = 0\,, \tag{2.11}$$

$$\mathbf{n} \cdot (\mathbf{d}_2 - \mathbf{d}_1)|_\Gamma = \rho_{surf}\,, \tag{2.12}$$

avec $\mathbf{j_{surf}}$ et ρ_{surf} respectivement les densités surfaciques de courant (en $\mathrm{A\,m^{-1}}$) et de charge (en $\mathrm{C\,m^{-2}}$). La notation "$\times$" désigne le produit vectoriel, "$\cdot$" le produit scalaire et $\mathbf{n}$ est un vecteur normal à Γ (voir la figure 2.1).

Les conditions sont relatives soit à la composante normale, soit à la composante tangentielle des champs. Par conséquent, les composantes normales de $\mathbf{b}$ et $\mathbf{d}$ ainsi que les composantes

tangentielles de $\mathbf{e}$ et $\mathbf{h}$ sont continues à travers l'interface Γ en l'absence des densités de charge ρ_{surf} et de courant $\mathbf{j_{surf}}$.

2.1.4 Formulations magnétostatiques

La magnétostatique consiste à étudier les phénomènes magnétiques lorsque ils sont invariants dans le temps. Dans ce régime statique, le champ est dû uniquement aux courants continus imposés $\mathbf{j_s}$ ou à des aimants permanents. Comme les dérivées temporelles dans les équations de Maxwell disparaissent, l'équation (2.1) devient :

$$\operatorname{rot} \mathbf{h} = \mathbf{j_s}\,, \tag{2.13}$$

et nous gardons

$$\operatorname{div} \mathbf{b} = 0, \tag{2.14}$$

$$\mathbf{b} = \mu \mathbf{h} + \mathbf{b_r}\,. \tag{2.15}$$

À ces équations s'ajoutent les conditions aux limites définies sur la frontière du domaine. Ces conditions sont liées soit à la composante tangentielle de $\mathbf{h}$, soit à la composante normale de $\mathbf{b}$, garantissant ainsi l'unicité de la solution [33]. Sur une portion de la surface Γ_h de la frontière Γ du domaine Ω, éventuellement non connexe, la première condition est définie par

$$\mathbf{n} \times \mathbf{h}|_{\Gamma_h} = 0\,, \tag{2.16}$$

où $\mathbf{n}$ est la normale unitaire sortante de Ω. Sur la portion de surface Γ_b, complémentaire de Γ_h dans Γ, à nouveau éventuellement non connexe, la deuxième condition est définie par

$$\mathbf{n} \cdot \mathbf{b}|_{\Gamma_b} = 0\,. \tag{2.17}$$

Les concepts de champ magnétique $\mathbf{h}$ ou d'induction $\mathbf{b}$ sont, bien entendu, adéquats pour décrire un état magnétique dans l'espace. Cependant, l'introduction d'un potentiel magnétique peut présenter un intérêt. Ce potentiel magnétique est souvent utilisé pour simplifier la formulation des équations de Maxwell.

Divers potentiels peuvent être envisagés, ouvrant ainsi la voie à différentes formulations. Elles sont groupées en deux familles : la famille de formulations conformes en $\mathbf{h}$, dites en potentiel scalaire magnétique qui assure fortement l'équation (2.13) et faiblement l'équation(2.14), et la famille de formulations conformes en $\mathbf{b}$, dites en potentiel vecteur magnétique, formulation duale, qui assure fortement l'équation (2.14) et faiblement l'équa-

tion(2.13). Dans ce travail, nous avons considéré les deux formulations et nous allons voir dans la suite l'écriture du problème magnétostatique en utilisant les formulations en potentiel scalaire et en potentiel vecteur.

Formulation magnétostatique en potentiel scalaire

Si dans une région de l'espace les courants sont nuls, l'équation (2.13) devient

$$\operatorname{rot} \mathbf{h} = 0\,. \tag{2.18}$$

Nous pouvons alors dériver $\mathbf{h}$ d'un potentiel scalaire magnétique Φ, défini à une constante près tel que

$$\mathbf{h} = -\operatorname{grad} \Phi\,. \tag{2.19}$$

Il suffit de remplacer l'équation (2.19) dans (2.5) et ensuite dans (2.3) pour obtenir l'équation

$$\operatorname{div}(\mu(-\operatorname{grad} \Phi) + \mathbf{b_r}) = 0\,. \tag{2.20}$$

Cette équation constitue une formulation magnétostatique en potentiel scalaire total [32]. Résoudre cette équation permet de trouver le champ magnétique et le champ d'induction. Dans le cas où le domaine est parcouru par un courant source $\mathbf{j_s}$, il suffit de décomposer le champ magnétique en deux parties dont l'une, $\mathbf{h_s}$, est définie telle que

$$\operatorname{rot} \mathbf{h_s} = \mathbf{j_s}\,. \tag{2.21}$$

Cela permet d'écrire une formulation en potentiel scalaire partiel (ou réduit) de la forme

$$\operatorname{div}(\mu(\mathbf{h_s} - \operatorname{grad} \Phi_r) + \mathbf{b_r}) = 0\,. \tag{2.22}$$

La continuité aux interfaces de la composante tangentielle du champ magnétique est assurée si le potentiel scalaire est continu. La condition de continuité de la composante normale de l'induction est, quant à elle, implicite dans la formulation.

Nous notons que dans le cas où le domaine Ω est multiplement connexe (figure 2.2), où le potentiel est multivoque [36] [37] et que nous voulons utiliser le potentiel scalaire total, nous devons introduire des coupures pour rendre le problème univoque.

Pour des raisons historiques d'implémentation, seuls les aimants ont été utilisés dans les cas tests, ce qui a également simplifié la mise en place de la DDM. Pour traiter les problèmes de magnétostatique impliquant des bobines, deux difficultés majeures sont à

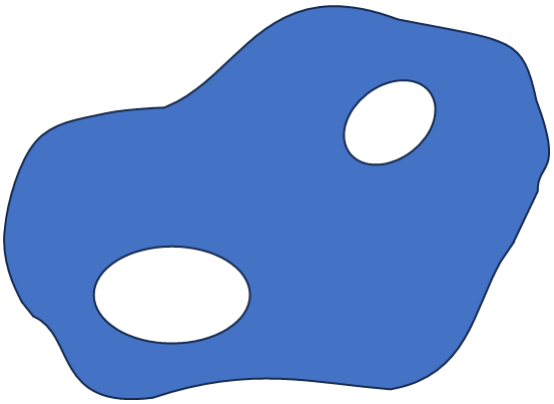

Figure 2.2 – Domaine Ω multiplement connexe

noter. Tout d'abord, il y a la question du partitionnement des sous-domaines. Ensuite, il y a le calcul des termes sources. Dans Flux, nous utilisons le champ magnétique source projeté $\mathbf{h_j}$ pour les bobines maillées et le potentiel vecteur magnétique source projeté $\mathbf{a_j}$ pour les bobines non maillées, nécessitant une pré-résolution [38]. La pré-résolution pour $\mathbf{h_j}$ consiste à calculer : $R_{\mathrm{rot\,rot}}\mathbf{h_j} = M_{\mathrm{rot}\,\mathbf{j_s}}$, avec une matrice M dépendant de $\mathbf{j_s}$, donnée de départ (l'utilisateur définit le courant dans les bobines). $R_{\mathrm{rot\,rot}}$ est un terme en rot − rot. La pré-résolution pour $\mathbf{a_j}$ consiste à calculer : $R_{\mathrm{rot\,rot}}\mathbf{a_j} = M_{\mathrm{rot}\,\mathbf{h_s}\mu}$, avec M calculé à partir du champ magnétique source $\mathbf{h_s}$, lui-même calculé analytiquement à partir de la forme de la bobine. D'autres méthodes de calcul du terme source existent, mais elles ne sont pas implémentées dans Flux car elles nécessitent d'être recalculées et stockées. Nous abordons dans les perspectives quelques réflexions sur l'utilisation de la décomposition de domaine avec les bobines.

Formulation magnétostatique en potentiel vecteur

Le potentiel vecteur magnétique $\mathbf{a}$ est introduit, à partir de l'équation (2.3), tel que [38] :

$$\mathbf{b} = \mathrm{rot}\,\mathbf{a}\,. \tag{2.23}$$

Ce potentiel vecteur n'est pas unique et il est déterminé à un "gradient" près. Nous remplaçons (2.23) dans (2.5) et ensuite dans (2.13) pour obtenir la formulation magnétostatique en potentiel vecteur

$$\mathrm{rot}(\mu^{-1}\,\mathrm{rot}\,\mathbf{a}) = \mathbf{j_s} + \mathrm{rot}(\mu^{-1}\mathbf{b_r})\,. \tag{2.24}$$

La continuité aux interfaces de la composante normale du champ d'induction est assurée si la composante tangentielle du potentiel vecteur est continue. La condition de continuité de la composante tangentielle du champ magnétique est, quant à elle, implicite dans la

formulation.

Pour assurer l'unicité du potentiel vecteur, il faut rajouter une condition dite de jauge. Plusieurs jauges peuvent être envisagées [39, 40] et plusieurs méthodes peuvent être mises en place pour résoudre ce problème. Ceci est important pour résoudre les systèmes assemblés à partir de ces formulations et peut affecter la convergence du solveur et/ou la solution obtenue. Nous abordons maintenant deux méthodes pour appréhender les conditions de jauge. Nous présenterons une troisième, la méthode de la régularisation pour la résolution des problèmes magnétostatique en éléments finis d'arête dans le dernier chapitre de ce document. Nous verrons que cette dernière peut-être utilisée plus simplement dans le cadre de la DDM que les méthodes plus classique présentées ci-dessous..

Conditions de jauge

1. Jauge de Coulomb : cette jauge se présente sous la forme $\operatorname{div} \mathbf{a} = 0$. La composante normale du potentiel vecteur est continue.
2. Jauge d'arête : la condition s'écrit sous la forme $\mathbf{a} \cdot \mathbf{w} = 0$, avec $\mathbf{w}$ un champ de vecteurs qui ne forme pas de boucle. Cette condition consiste à annuler la composante du potentiel vecteur selon la direction $\mathbf{w}$.

Le problème d'unicité du potentiel et, par extension, l'utilisation de la condition de jauge, représentent l'une des principales difficultés rencontrées lors de l'application de la méthode de décomposition de domaine. Pour l'instant, nous nous sommes limités à une brève présentation des deux conditions de jauge. Nous reviendrons plus en détail sur l'utilisation de ces conditions dans le chapitre 5, en les intégrant à la discrétisation la plus appropriée.

2.2 Discrétisation avec la méthodes des éléments finis

2.2.1 Espaces fonctionnels

Les formulations précédemment définies utilisent des opérateurs différentiels appliqués à des champs vectoriels et scalaires dans le domaine considéré Ω. Lors de l'intégration sur ce domaine Ω, nous devons nous assurer que toutes les intégrales sont bien définies. Par conséquent, une structure mathématique doit être spécifiée. Les domaines de définition sont appelés espaces fonctionnels. Les espaces fonctionnels qui vont être utilisés sont :

1. Les espaces des champs scalaires

$$L^2(\Omega) = \left\{ u : \Omega \to \mathbb{R} \;\middle|\; \int_\Omega (u(\mathbf{x}))^2 d\Omega < +\infty \right\}, \tag{2.25}$$

$$H^1(\Omega) = \left\{ u \in L^2(\Omega) \;\middle|\; \frac{\partial u}{\partial x_i} \in L^2(\Omega) \text{ avec } i = 1, \cdots, 3 \right\}, \tag{2.26}$$

$$H^1(\Omega) = H(\mathrm{grad}, \Omega) = \left\{ u \in L^2(\Omega) \;\middle|\; \mathrm{grad}\, u \in \mathbf{L}^2(\Omega) \right\}. \tag{2.27}$$

2. Les espaces des champs de vecteurs

$$\mathbf{L}^2(\Omega) = \left\{ \mathbf{u} : \Omega \to \mathbb{R}^3 \;\middle|\; \int_\Omega \|\mathbf{u}(\mathbf{x})\|^2_{\mathbb{R}} d\Omega < +\infty \right\}, \tag{2.28}$$

$$\mathbf{H}(\mathrm{rot}, \Omega) = \left\{ \mathbf{u} \in \mathbf{L}^2(\Omega) \;\middle|\; \mathrm{rot}\, \mathbf{u} \in \mathbf{L}^2(\Omega) \right\}, \tag{2.29}$$

$$\mathbf{H}(\mathrm{div}, \Omega) = \left\{ \mathbf{u} \in \mathbf{L}^2(\Omega) \;\middle|\; \mathrm{div}\, \mathbf{u} \in L^2(\Omega) \right\}. \tag{2.30}$$

Les espaces $H(\mathrm{grad}, \Omega)$, $\mathbf{H}(\mathrm{rot}, \Omega)$ et $\mathbf{H}(\mathrm{div}, \Omega)$ sont munis des produits scalaires suivants :

$$(u, v)_{H(\mathrm{grad},\Omega)} = (u, v)_{L^2(\Omega)} + (\mathrm{grad}\, u, \mathrm{grad}\, v)_{\mathbf{L}^2(\Omega)}, \quad \forall u, v \in H^1(\Omega), \tag{2.31}$$

$$(\mathbf{a}, \mathbf{b})_{\mathbf{H}(\mathrm{rot},\Omega)} = (\mathbf{a}, \mathbf{b})_{\mathbf{L}^2(\Omega)} + (\mathrm{rot}\, \mathbf{a}, \mathrm{rot}\, \mathbf{b})_{\mathbf{L}^2(\Omega)}, \quad \forall \mathbf{a}, \mathbf{b} \in \mathbf{H^1}(\Omega), \tag{2.32}$$

$$(\mathbf{a}, \mathbf{b})_{\mathbf{H}(\mathrm{div},\Omega)} = (\mathbf{a}, \mathbf{b})_{\mathbf{L}^2(\Omega)} + (\mathrm{div}\, \mathbf{a}, \mathrm{div}\, \mathbf{b})_{L^2(\Omega)}, \quad \forall \mathbf{a}, \mathbf{b} \in \mathbf{H^1}(\Omega), \tag{2.33}$$

où

$$(u, v)_{L^2(\Omega)} = \int_\Omega uv \, d\Omega,$$
$$(\mathbf{a}, \mathbf{b})_{\mathbf{L}^2(\Omega)} = \int_\Omega \mathbf{a} \cdot \mathbf{b} \, d\Omega,$$

désignent les produits scalaires respectivement sur $L^2(\Omega)$ et $\mathbf{L}^2(\Omega)$. Pour alléger les notations et si cela est clair dans le contexte, on ne notera pas les indices $L^2(\Omega)$ et $\mathbf{L}^2(\Omega)$ dans la suite.

Quatre sous-espaces fonctionnels des opérateurs différentiels peuvent être définis et notés E^p_v, avec $0 \leqslant p \leqslant 3$ et v ou $\mathbf{v}$ un champ :

$$E^0_v(\Omega) = \left\{ u \in L^2(\Omega) \;\middle|\; \mathrm{grad}\, u \in \mathbf{L}^2(\Omega), u|_{\Gamma_v} = 0 \right\} \subset H(\mathrm{grad}, \Omega), \tag{2.34}$$

$$\mathbf{E^1_v}(\Omega) = \left\{ \mathbf{u} \in \mathbf{L}^2(\Omega) \;\middle|\; \mathrm{rot}\, \mathbf{u} \in \mathbf{L}^2(\Omega), \mathbf{n} \times \mathbf{u}|_{\Gamma_v} = \mathbf{0} \right\} \subset \mathbf{H}(\mathrm{rot}, \Omega), \tag{2.35}$$

$$\mathbf{E^2_v}(\Omega) = \left\{ \mathbf{u} \in \mathbf{L}^2(\Omega) \;\middle|\; \mathrm{div}\, \mathbf{u} \in L^2(\Omega), \mathbf{n} \cdot \mathbf{u}|_{\Gamma_v} = 0 \right\} \subset \mathbf{H}(\mathrm{div}, \Omega), \tag{2.36}$$

$$\mathbf{E^2_v}(\Omega) = \mathrm{cod}(\mathrm{div}), \tag{2.37}$$

avec cod l'espace d'arrivée ou le codomaine.

Ils forment une suite, dite de De Rham, qui lie les espaces fonctionnels entre eux tel que :

$$E_v^0 \xrightarrow{\text{grad}} \mathbf{E}_\mathbf{v}^1 \xrightarrow{\text{rot}} \mathbf{E}_\mathbf{v}^2 \xrightarrow{\text{div}} \mathbf{E}_\mathbf{v}^3 \,. \tag{2.38}$$

Cette suite symbolise les inclusions des espaces fonctionnels tel que $\operatorname{grad} E_v^0 \subset \mathbf{E}_\mathbf{v}^1$, $\operatorname{rot} \mathbf{E}_\mathbf{v}^1 \subset \mathbf{E}_\mathbf{v}^2$ et $\operatorname{div} \mathbf{E}_\mathbf{v}^2 \subset \mathbf{E}_\mathbf{v}^3$.

2.2.2 Formulations faibles

Les formulations (2.20) (2.22) (2.24) présentées dans 2.1.4, sont écrites sous forme forte. Lors de la discrétisation, nous travaillons avec la formulation faible. En partant de la formulation forte (2.20), en multipliant par une fonction test Φ', en intégrant sur le domaine Ω et en appliquant la formule de Green [41], la formulation faible en potentiel scalaire s'écrit :

$$(\mu(-\operatorname{grad}\Phi), \operatorname{grad}\Phi') = (\mathbf{b_r}, \operatorname{grad}\Phi'), \quad \forall \Phi' \in E_h^0 \,. \tag{2.39}$$

Pour le potentiel scalaire réduit (2.22), la formulation faible est :

$$(\mu(\mathbf{h_s} - \operatorname{grad}\Phi_r), \operatorname{grad}\Phi') = (\mathbf{b_r}, \operatorname{grad}\Phi'), \quad \forall \Phi' \in E_h^0 \,. \tag{2.40}$$

D'un autre côté, la formulation en potentiel vecteur s'écrit :

$$(\mu^{-1} \operatorname{rot}\mathbf{a}, \operatorname{rot}\mathbf{a}') = (\mathbf{j_s}, \mathbf{a}') + (\mu^{-1}\mathbf{b_r}, \operatorname{rot}\mathbf{a}'), \quad \forall \mathbf{a}' \in E_{b0}^1 \,. \tag{2.41}$$

Ces formulations faibles peuvent être discrétisées en utilisant la méthode des éléments finis à travers la méthode de Galerkin [42], c'est l'objet de la prochaine section.

2.2.3 Éléments finis et fonctions de forme

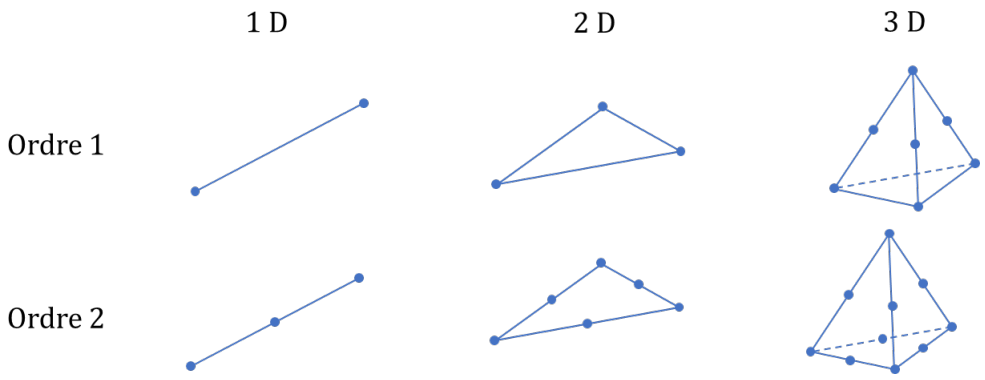

Figure 2.3 – Elements finis 1D, 2D et 3D, pour le premier et le second ordre en éléments finis nodaux

La méthode des éléments finis est une des méthodes les plus souvent utilisées pour résoudre numériquement des modèles à base d'équations aux dérivées partielles [43, 33]. Elle consiste à découper le domaine Ω en petits éléments simples appelés éléments finis ; voir la figure 2.3. Les éléments finis sont composés de nœuds, d'arêtes et de faces. L'ensemble des éléments constitue le maillage, qui reconstruit une représentation discrétisée du domaine continu d'origine. Sur chacun des éléments finis, les équations aux dérivées partielles peuvent être remplacée par un système d'équations linéaires par approximation. Cette approximation est obtenue grâce aux fonctions de forme et à la formulation variationnelle du problème. Le système d'équations linéaires peut être décrit par une matrice élémentaire (une par élément fini). Les matrices sont regroupées dans une matrice globale lors de l'assemblage.

En théorie, un élément fini est défini par un triplet (K, P_K, Σ_K) [32] :

1. K est un domaine géométrique, par exemple un triangle en 2D ou un tétraèdre en 3D ;
2. P_K est un espace de fonctions sur K, appelé espace des fonctions de base ;
3. Σ_K est un ensemble de formes linéaires sur P_K, appelé degrés de liberté.

Une approximation des variables à l'intérieur des éléments finis est réalisée au moyen de fonctions. Les fonctions utilisées pour représenter la solution dans chaque élément sont appelées fonctions de forme, fonctions d'interpolation ou fonctions de base. Selon la fonction de forme, les inconnues discrètes du problème peuvent être associées soit aux nœuds, aux arêtes, aux faces ou aux volumes des éléments finis. Cela dépend souvent des quantités que l'on souhaite conservées sur la variable à calculer. Par exemple, si nous cherchons à conserver la composante tangentielle, nous allons privilégier les éléments finis d'arête ; les éléments finis de facette seront considérés pour une conservation du flux entre éléments.

Nous nous limitons ici à des éléments finis d'ordre faible. Il existe quatre types de fonctions de forme suivant que nous travaillions avec les nœuds, les arêtes, les facettes ou les volumes du maillage [32]. Les fonctions de forme sont généralement définies sur des éléments de référence, qui sont des éléments de forme simple dans un espace de référence fixe facilitant l'analyse. Au moyen d'une transformation géométrique, les éléments de référence peuvent être transformés en tout élément réel. Les formulations en potentiel scalaire sont généralement discrétisées par des éléments finis dits "nodaux", cela est lié au fait que les potentiels sont continus dans l'espace. Pour le potentiel vecteur magnétique, l'utilisation des éléments finis dits "d'arête" est systématique, car elle permet d'assurer la conservation de la composante normale de $\mathbf{b}$ sans forcer la continuité normale de $\mathbf{a}$ qui peut poser problème.

1. **Fonctions de forme nodales** : elles sont associées à chaque nœud. Elles valent 1 en un nœud et zéro pour tous les autres nœud en étant continus. Pour x_i les coordonnées

du nœud n_i appartenant à l'ensemble de nœuds N_n, la fonction de forme w_i s'écrit

$$w_i(x_j) = \delta_{i,j}\,, \quad \forall i,j \in N_n\,, \tag{2.42}$$

avec $\delta_{i,j}$ le symbole de Kronecker qui vaut 1 quand $i = j$ et 0 sinon. L'ensemble des fonctions de forme nodales appartient à l'espace de fonctions discret W^0 et assure la continuité d'un champ scalaire à l'interface entre deux éléments du maillage. Leur gradient possède une composante tangentielle continue aux interfaces ce qui permet de représenter le champ $\mathbf{h}$, cette continuité tangentielle a déjà été évoquée pour le problème continue en sous-section 2.1.4.

2. **Fonctions de forme d'arête** : elles sont associées aux arêtes des éléments. Elles servent à représenter des champs vectoriels. Elles assurent une continuité de la composante tangentielle du champ vectoriel à l'interface entre deux éléments du maillage et permettent une discontinuité de sa composante normale. Pour w_i et w_j les fonctions de forme nodales linéaires associées aux nœuds i et j, la fonction de forme d'arête $\mathbf{w}_{ij}$ associée à l'arête ij reliant le nœud i au nœud j s'écrit

$$\mathbf{w}_{ij} = w_i \operatorname{grad} w_j - w_j \operatorname{grad} w_i\,, \quad \forall i,j \in N_n\,. \tag{2.43}$$

La fonction de forme d'arête $\mathbf{w}_{ij}$ est nulle dans tous les éléments non adjacents à l'arête. L'ensemble des fonctions de forme d'arête appartient à l'espace de fonctions discret W^1.

Au niveau discret, pour N nœuds et A arêtes internes, le potentiel scalaire appartenant à E_h^0 s'exprime sous la forme :

$$\Phi_r = \sum_{i=1}^{N} \Phi_{r,i} w_i\,, \tag{2.44}$$

et son gradient :

$$\operatorname{grad} \Phi_r = \sum_{i=1}^{N} \Phi_{r,i} \operatorname{grad} w_i\,. \tag{2.45}$$

Le champs source $\mathbf{h_s}$ appartenant à E_h^1 s'exprime comme :

$$\mathbf{h_s} = \sum_{i=1}^{A} \mathrm{h}_{s_i} \mathbf{w}_i\,. \tag{2.46}$$

En utilisant les expressions (2.44), (2.45) et (2.46) dans la formulation (2.40) testée avec

chacune des fonctions w_i, on obtient

$$\sum_{j=1}^{N}(\mu \operatorname{grad} w_j, \operatorname{grad} w_i)\Phi_{\mathrm{r},j} - \sum_{j=1}^{A}(\mu \mathbf{w_j}, \operatorname{grad} w_i)\mathrm{h}_{\mathrm{s},j} = -(\mathbf{b_r}, \operatorname{grad} w_i)\,, \quad \forall i \in [\![1; N]\!]\,. \tag{2.47}$$

Il en découle ainsi un système algébrique :

$$Ku - K_s u_s = -b_r\,, \tag{2.48}$$

avec les matrices K et K_s définis de la manière suivante :

$$K_{i,j} = (\mu \operatorname{grad} w_j, \operatorname{grad} w_i)\,, \tag{2.49}$$

$$(K_s)_{i,j} = (\mu \mathbf{w}_j, \operatorname{grad} w_i)\,, \tag{2.50}$$

et les vecteurs u, u_s et b_r définis ainsi

$$(b_r)_i = (\mathbf{b_r}, \operatorname{grad} w_i)\,, \quad u = \begin{pmatrix} \Phi_{\mathrm{r},1} \\ \vdots \\ \Phi_{\mathrm{r},N} \end{pmatrix}, \quad u_{\mathrm{s}} = \begin{pmatrix} \mathrm{h}_{\mathrm{s},1} \\ \vdots \\ \mathrm{h}_{s,A} \end{pmatrix}. \tag{2.51}$$

Les systèmes obtenues à partir de la discrétisation de formulation la formulation en potentiel scalaire seront donc de la forme

$$K(u)u = b\,, \tag{2.52}$$

avec u le vecteur inconnu, K la matrice éléments finis et b le terme source. Les matériaux saturables, dans lesquels la perméabilité magnétique μ est une fonction du module du champ magnétique h, donne le caractère non linéaire au système d'équations. Si μ est constante, le système peut être réécrit sous la forme :

$$Ku = b\,, \tag{2.53}$$

avec K une matrice indépendante de la solution.

De manière similaire, nous pouvons exprimer le système en utilisant la formulation en potentiel vecteur [32, 38]. Pour cela, le potentiel vecteur appartenant à E_h^1 s'écrit

$$\mathbf{a} = \sum_{i=1}^{A} \mathbf{a}_i \mathbf{w}_i\,. \tag{2.54}$$

En injectant la relation (2.54) dans la formulation (2.41) testée pour chacune des fonctions

$\mathbf{w}_i$, on obtient

$$\sum_{i=1}^{A}(\mu^{-1}\operatorname{rot}\mathbf{w}_j,\operatorname{rot}\mathbf{w}_i)\mathbf{a}_j=(\mathbf{j_s},\mathbf{w}_i)+(\mu^{-1}\mathbf{b_r},\operatorname{rot}\mathbf{w}_i)\,,\quad\forall i\in[\![1;A]\!]\,. \tag{2.55}$$

Nous obtenons ainsi un système d'équations de la forme :

$$K(u)u=b\,, \tag{2.56}$$

avec :

$$K_{i,j}=(\mu^{-1}\operatorname{rot}\mathbf{w}_j,\operatorname{rot}\mathbf{w}_i)\,, \tag{2.57}$$

$$b=b_j+b_M=(\mathbf{j_s},\mathbf{w}_i)+(\mu^{-1}\mathbf{b_r},\operatorname{rot}\mathbf{w}_i)\,, \tag{2.58}$$

$$u=\begin{pmatrix}\mathbf{a}_1 & \dots & \mathbf{a}_A\end{pmatrix}^t\,. \tag{2.59}$$

2.3 Conclusion

Dans ce chapitre, nous avons examiné le problème spécifique que nous avons choisi de traiter en électromagnétisme. Nous avons présenté les équations de Maxwell ainsi que les différentes formulations qui en découlent, et nous avons également discuté de leur discrétisation à l'aide de la méthode des éléments finis. Nous avons démontré comment obtenir le système d'équations qui sera résolu par la méthode de décomposition de domaine. Tout cela a été réalisé dans le cadre du logiciel Altair Flux, et nous aborderons l'ensemble des outils utilisés dans le chapitre suivant.

Chapitre 3

Outils de mise en œuvre de la DDM

Le présent chapitre explore divers outils indispensables à la mise en œuvre de la méthode de décomposition de domaine, traitant des concepts tels que le calcul parallèle, les solveurs directs et itératifs, l'optimisation des algorithmes, la mesure de la performance, ainsi que le partitionnement de maillage. Ces éléments revêtent une grande importance pour appréhender la mise en place de la méthode et les présenter vise à clarifier l'implémentation.

3.1 Altair Flux

Le logiciel Altair Flux [44] est un logiciel de simulation électromagnétique basses fréquences 2D et 3D. Il permet de construire une géométrie, définir une physique, générer un maillage et résoudre le problème. Il permet aussi de faire du post-traitement avec les résultats obtenus. Altair Flux est un outil utilisé depuis plusieurs décennies pour simuler des dispositifs électromagnétiques en utilisant les formulations de la sous-section 2.1.4, ainsi que d'autres telles que les formulations en magnéto-transitoire ou en magnéto-harmonique [33]. Il est ainsi optimisé pour donner des bonnes performances en terme de précision, de temps de calcul et de consommation mémoire.

Le développement de la méthode de décomposition de domaine doit prendre en compte les spécificités [1] que présente un logiciel commercial utilisé à grande échelle. L'étude de performance de la méthode sera donc réalisée par comparaison à une utilisation standard du logiciel.

L'implémentation numérique a été entièrement réalisée dans le logiciel Altair Flux, plus particulièrement au sein du projet Flux Solver. Ce dernier vise à séparer le processus de résolution des autres étapes : géométrie, maillage et physique. Cette approche permet

1. Tel que le format de fichier, le stockage des valeurs, l'accès à certaine bibliothèque ou outils voir la physique proposée.

d'optimiser la mise en œuvre de la méthode de décomposition de domaine [2] et, dans une certaine mesure, simplifie les contraintes liées au travail sur un code industriel de grande envergure.

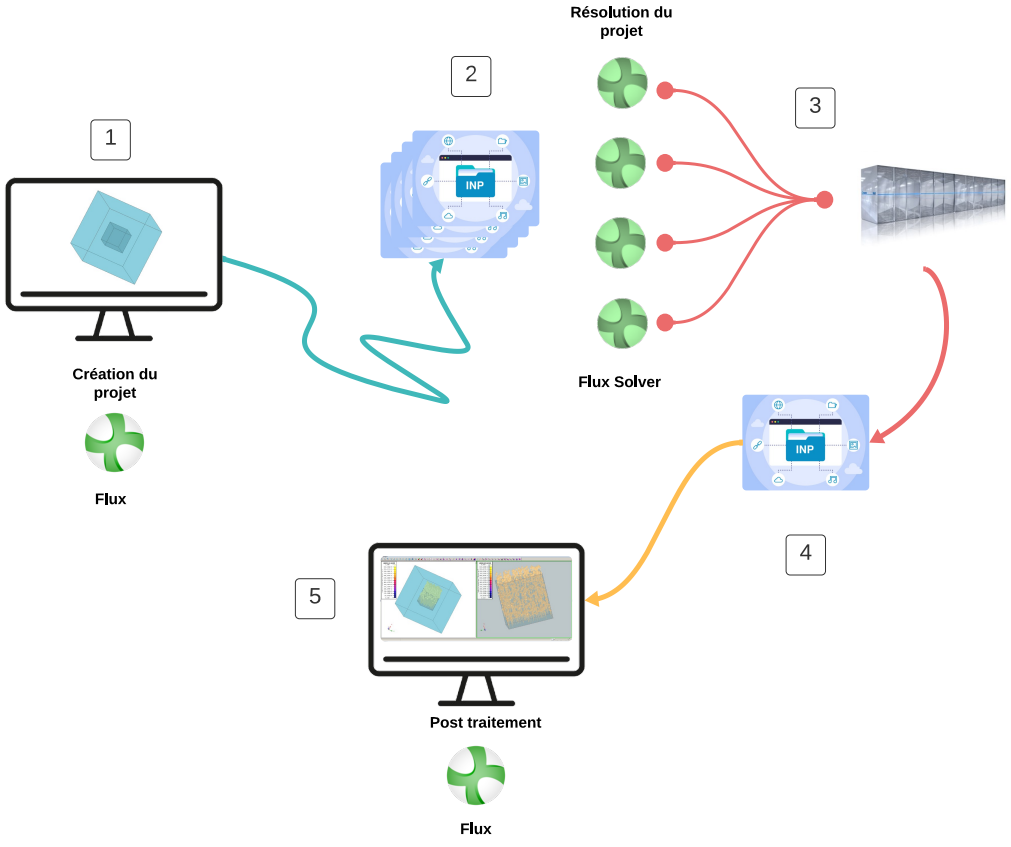

Figure 3.1 – Architecture du code de Flux Solver

La Figure 3.1 illustre les étapes d'une simulation en utilisant le projet Flux Solver, dans lequel la DDM a été intégrée. Ces étapes sont :

2. Pour des raisons d'implementation, la séparation du processus du résolution des autres étapes permet l'utilisation de certain outils de débogage et de profilage ainsi que la mise en place du code parallèle.

1. L'utilisateur définit le projet en choisissant la géométrie, les matériaux, la physique et le maillage.
2. La communication entre Flux et Flux Solver se fait à travers un système de fichiers qui contient les informations du projet. Ces fichiers sont au format Abaqus ".inp" [45].
3. L'étape 3 consiste à lire les fichiers par Flux Solver, qui peuvent être exécutés sur une machine différente (un cluster de calcul par exemple). Cette étape contient la partie intégration et construction du système matriciel que nous avons vu dans 2.2.3.
4. Le résultat (valeurs des potentiels par exemple) est écrit dans un nouveau fichier au même format ".inp".
5. L'utilisateur utilisera Flux pour lire le fichier de sortie et effectuer le post-traitement.

La Figure 3.2 présente, de manière non exhaustive, les apports réalisés au cours de la thèse pour intégrer la méthode de décomposition de domaine et donne le processus général de l'exécution de la résolution avec la DDM.

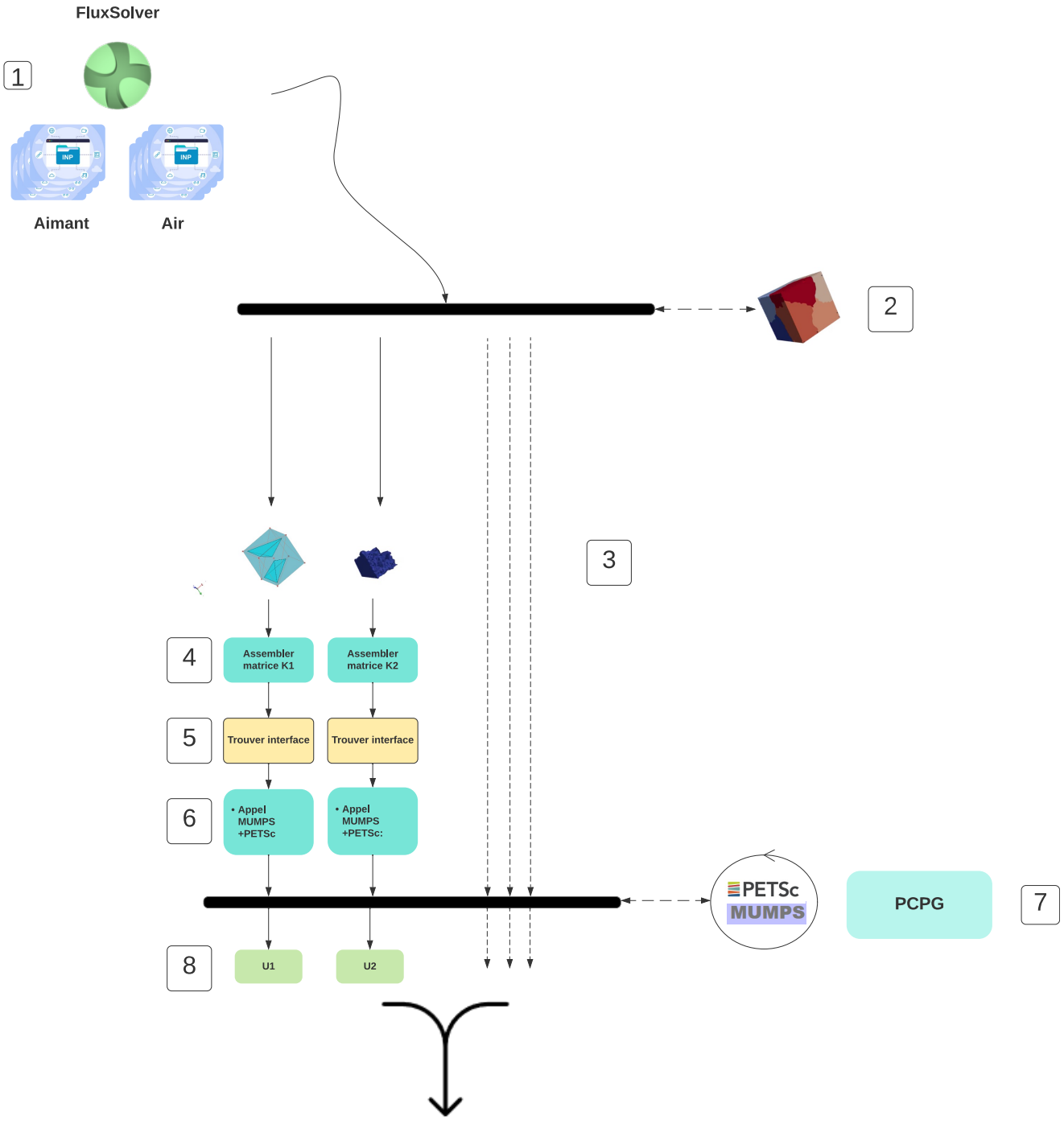

Figure 3.2 – Architecture du code de décomposition de domaine

Ce processus est constitué des étapes suivantes :

1. Nous effectuons déjà *un premier découpage par région physique*. Nous abordons dans

la section 3.5 l'intérêt de ce découpage.

2. Flux Solver récupère ces informations et lance la vérification de l'équilibre de la charge de travail. Si les différents domaines liés aux régions physiques montrent un grand déséquilibre en terme de nombre d'inconnues, un partitionnement du maillage est lancé. Nous récupérons ainsi des domaines aux propriétés physiques homogènes mais de tailles plus équilibrées (voir section 3.5).
3. Chaque partition est traité sur un processeur différent. L'ensemble s'exécute en parallèle (voir la section 3.2).
4. Nous procédons ensuite à l'assemblage de la matrice élément finis K et du second membre b pour chaque sous-domaine. La matrice est récupérée sous format CSR (Compressed Sparse Row [46]) (voir la sous-section 2.2.3).
5. Nous récupérerons aussi la liste des variables effectives. Une dichotomie est effectuée pour identifier l'interface entre deux voisins ainsi que l'interface globale. Nous construisons les vecteurs d'extraction qui contiennent les indices i et j de la matrice B_{ij} ; voir l'équation (1.8). Ce vecteur est utilisé pour récupérer la valeur recherchée sans construire la matrice B_{ij} (matrice de restriction/projection) (voir section 1.4).
6. Les bibliothèques MUMPS (sous-section 3.3.1) et PETSc (sous-section 3.3.2) sont utilisées pour factoriser les matrices, calculer le complément de Schur et construire le preconditionneur ; voir l'équation (1.21). La factorisation est sauvegardée en mémoire, elle est récupérée à chaque itération pour calculer le produit Fs dans l'algorithme 1. Nous utilisons les bibliothèques MUMPS pour effectuer la factorisation et la résolution des problèmes locaux beaucoup plus petit par rapport au problème de départ et PETSc pour effectuer les différentes opérations matricielles.
7. L'algorithme 1 est exécuté jusqu'à l'obtention d'une solution λ avec un résidu inférieure à la tolérance.
8. Nous utilisons ce λ pour calculer la solution finale sur l'ensemble des sous-domaines et fusionner le résultat en un seul fichier.

Ces étapes sont majoritairement communes aux deux méthodes FETI-1 et FETI-DP. Quelques différences peuvent être notées, telles que dans le calcul de l'interface, où pour FETI-DP nous nous intéressons également à la détection des points de croisement (cross points). Le calcul de la solution finale diffère également entre les deux méthodes ; dans FETI-DP, nous calculons la solution pour les variables primales avant les autres variables.

3.2 Calcul parallèle

Le calcul parallèle consiste à gérer de la manière la plus optimale possible les trois composantes principales d'une machine de calcul que sont les processeurs, les modules de mémoire et un réseau d'interconnexion. Il existe différentes architectures pour effectuer le

calcul parallèle regrouper dans la taxonomie de Flynn [47]. Les architectures sont classées en quatre catégories distinctes :

1. **SISD (Single Instruction stream, Single Data stream) :**
   — *Description :* Les architectures SISD sont des ordinateurs traditionnels à processeur unique, où une seule instruction est exécutée à la fois et traite un seul ensemble de données.
   — *Exemple :* Nous retrouvons dans ce type d'architecture des opérations comme le fetch [3] ainsi que les pipelines.
2. **SIMD (Single Instruction stream, Multiple Data streams) :**
   — *Description :* Dans les architectures SIMD, une seule instruction est diffusée à tous les processeurs simultanément, mais chaque processeur peut traiter un ensemble distinct de données.
   — *Exemple :* Les processeurs vectoriels ou les GPU modernes, où une opération est appliquée à plusieurs éléments de données en parallèle.
3. **MISD (Multiple Instruction streams, Single Data stream) :**
   — *Description :* Les architectures MISD sont moins courantes, avec plusieurs flux d'instructions agissant sur la même ensemble de données. Chaque flux d'instructions applique une opération différente.
   — *Exemple :* Bien que moins courantes dans la pratique, certains systèmes de surveillance et de diagnostic peuvent utiliser des architectures MISD.
4. **MIMD (Multiple Instruction streams, Multiple Data streams) :**
   — *Description :* Les architectures MIMD ont plusieurs flux d'instructions agissant sur des ensembles distincts de données. Chaque processeur peut exécuter des instructions indépendamment des autres.
   — *Exemple :* Les clusters de calcul, les grappes de serveurs parallèles et les supercalculateurs sont des exemples d'architectures MIMD où chaque processeur peut exécuter des instructions de manière autonome sur ses propres données. C'est le cas aussi de l'ensemble des ordinateurs portables voir des téléphones portables développés dans la dernière décennie.

Nous pouvons généraliser l'approche SIMD pour obtenir l'architecture SPMD (Single Program, Multiple Data) qui est un modèle d'exécution parallèle où un seul programme (au lieu d'une seule instruction) est exécuté simultanément par plusieurs processeurs indépendants. L'architecture SPMD est flexible car elle peut être implémentée sur différentes architectures matérielles, y compris des systèmes à mémoire partagée [4] ou distribuée [5]. Elle est utilisée dans un large éventail d'applications, notamment le calcul scientifique, la

3. Le fetch consiste à accéder à la mémoire pour récupérer une donnée.
4. Plusieurs processeurs ont accès à une seule et même mémoire.
5. Chaque processeur possède sa propre mémoire distincte et communique avec les autres processeurs via un réseau.

simulation, le rendu graphique et d'autres domaines nécessitant un traitement parallèle. C'est avec cette approche que la DDM a été implémentée, où les différents processeurs exécutent le même programme de résolution mais avec des partitions différentes.

Dans ce contexte, trois possibilités s'offrent à nous :

1. Le nombre de sous-domaines N_{sd} est le même que le nombre de processeurs N_{proc} disponibles : chaque processeur s'occupe d'un sous-domaine. Pour chaque opération sur l'interface globale, le processeur communique sa propre partie.
2. Le nombre de sous-domaines est plus grand que le nombre de processeurs disponibles : chaque processeur s'occupe d'un ou de plusieurs sous-domaines. Nous assurons l'équilibre de la charge en utilisant une distribution par bloc, définie dans la suite. Pour chaque opération sur l'interface globale, le processeur communique ses parties.
3. Le nombre de sous-domaines est plus petit que le nombre de processeurs disponibles : pour chaque sous-domaine, plusieurs processeurs sont affectés. Nous assurons l'équilibre de la charge en utilisant une distribution par bloc pour les processeurs cette fois. Chaque opération sur l'interface locale est divisée entre les processeurs à travers un système de sous-communicateur ; voir notamment la figure 3.4. Tous les processeurs au sein du même sous-communicateur s'échangent les informations pour les distribuer ensuite aux autres processeurs des autres domaines.

Pour tester les performances nous avons opté pour une distribution par bloc avec équilibre de charge. Le schéma de la figure 3.3 illustre comment 14 tâches sur 4 processeurs peuvent être distribuées.

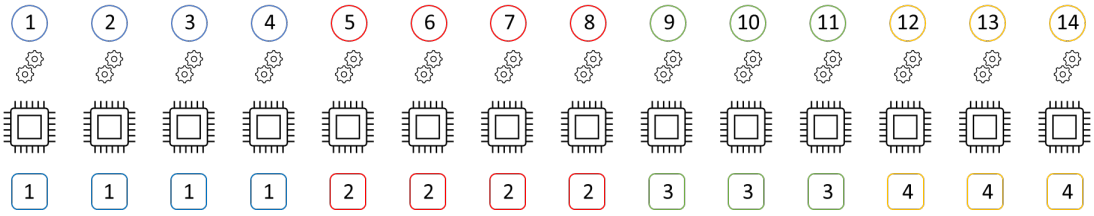

Figure 3.3 – Distribution par bloc avec équilibrage de la charge.

Dans le cas inverse (plus de processeurs que de sous-domaines), il suffit d'inverser la distribution (affecter des processeurs aux tâches au lieu des tâches aux processeurs).

En termes d'implementation nous utilisons la norme MPI, Message Passing Interface, qui est largement utilisée dans le domaine de la programmation parallèle et distribuée pour développer des applications qui s'exécutent sur des systèmes avec plusieurs processeurs ou nœuds.

Nous nous arrêtons sur un concept en particulier, qui est le communicateur. Il désigne un ensemble de processus pouvant communiquer ensemble, et deux processus ne pour-

ront communiquer que s'ils sont dans un même communicateur. MPI_COMM_WORLD englobe tous les processus. Il est subdivisé en sous-communicateurs avec la fonction MPI_COMM_SPLIT. Les sous-communicateurs seront groupés par couleur (figure 3.4), pour chaque couleur, ils auront des rangs. Nous utilisons ces rangs pour repartager l'information avec les autres communicateurs.

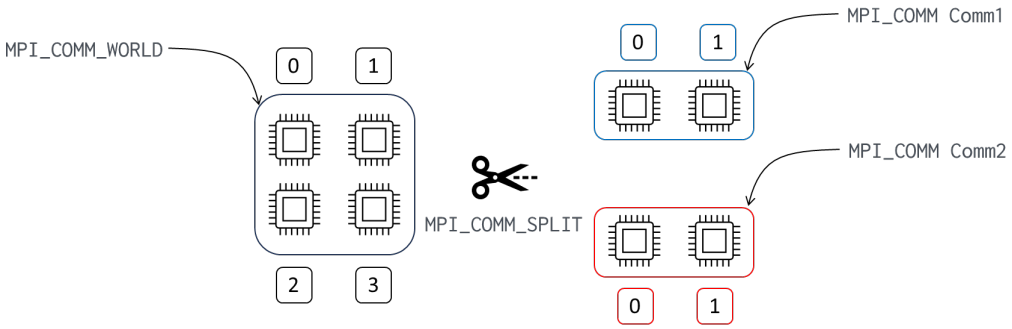

Figure 3.4 – Découpage en sous-communicateur, exemple sur une configuration à 4 processeurs.

Nous avons commencé par chercher la meilleure distribution de sous-domaines par processeurs. Pour effectuer cela, le même cas test est lancé avec différent nombre de processus MPI. Le nombre de sous-domaine étant fixe (4 sous-domaines), nous avons estimer le temps d'exécution avec 1, 4, et 8 processus MPI (figure 3.5). L'utilisation de 8 cœurs implique que chaque paire de processus MPI résoudra simultanément le problème local associé au même sous-domaine. Cela nécessite des communications entre ces deux processus, car seuls les rangs 0 de chaque groupe de couleur communiquent entre eux. L'ensemble des tests que nous avons effectué indique que la meilleure utilisation pour la décomposition de domaine est d'affecter un sous-domaine par processeur. L'exécution séquentielle est très lente[6]. Dans le cas où $N_{proc} > N_{sd}$, les communications entre processeurs de même couleur (figure 3.3) ralentissent l'exécution. Le gain sur la résolution en parallèle du problème local est négligeable.

6. Contrairement à ce à quoi nous pourrions nous attendre, en raison de la complexité algorithmique, résoudre plusieurs problèmes de petite taille devrait être plus rapide que résoudre le problème global.

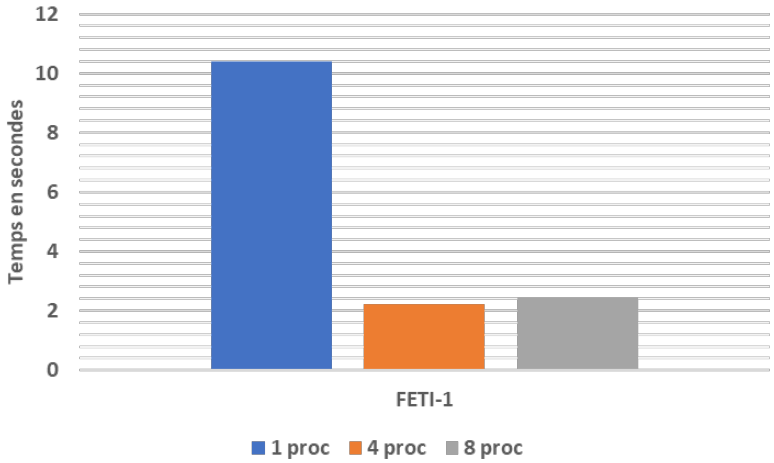

Figure 3.5 – Temps d'assemblage + résolution en utilisant FETI-1 avec 1, 4 et 8 processeurs (cas 1).

Pour l'utilisation de la DDM, nous conseillons de résoudre de la manière suivante :

1. Vérifier le nombre de cœurs sur la machine de calcul.
2. Diviser le domaine global par le nombre de cœurs en respectant la taille de problème minimale par sous-domaine. 4.2.1
3. Lancer la décomposition de domaine avec le nombre de processus MPI adéquat.

3.3 Solveurs locaux

La résolution des problèmes locaux $K_i u_i = b_i$ (voir la sous-section 1.4.1) nécessite d'utiliser des solveurs que nous pouvons classer en deux catégories principales : les solveurs directs et les solveurs itératifs.

Méthodes Directes

1. **Élimination de Gauss :** Cette méthode utilise des opérations élémentaires sur les lignes pour transformer la matrice augmentée [7] en une forme échelonnée réduite [48].
2. **Factorisation LU (Décomposition LU) :** La matrice est factorisée en un produit de deux matrices, une inférieure triangulaire (L) et une supérieure triangulaire (U). Les opérations effectuées lors de l'élimination de Gauss sont utilisées pour construire les matrices L et U [49].

7. En algèbre linéaire, une matrice augmentée $(A|B)$ est une matrice obtenue en ajoutant le vecteur second membre B à droite, comme une colonne supplémentaire, à la matrice A.

3. **Méthode de Cholesky :** Spécifique aux matrices symétriques définies positives, elle réalise une factorisation en un produit d'une matrice triangulaire inférieure par sa transposée conjuguée [49].
4. **Méthode de Factorisation QR :** La matrice est factorisée en un produit de deux matrices, l'une orthogonale (Q) et l'autre triangulaire supérieure (R) [49].

Méthodes Itératives

1. **Méthode de Jacobi :** Une approche itérative qui met à jour les variables du système en fonction des valeurs actuelles des autres variables. Lorsque la matrice n'est pas à diagonale strictement dominante, la convergence de la méthode de Jacobi peut être plus lente voire non garantie [46].
2. **Méthode de Gauss-Seidel :** Une amélioration de la méthode de Jacobi où les valeurs mises à jour des variables de l'itération en cours sont utilisées pour calculer les nouvelles valeurs. L'algorithme suppose que la diagonale de la matrice est formée d'éléments non nuls. Comme la méthode de Jacobi, la méthode de Gauss-Seidel peut converger plus rapidement lorsque la matrice symétrique définie positive est diagonale dominante [46].
3. **Méthode de SOR (Successive Over-Relaxation) :** Une amélioration de la méthode de Gauss-Seidel. Elle permet de résoudre les systèmes linéaires dont la matrice est symétrique définie positive. Elle introduit un paramètre de relaxation pour accélérer la convergence. Sous certaines hypothèses sur le paramètre de relaxation et la matrice, la convergence est obtenue [46].
4. **Méthode du Gradient Conjugé :** La plus connue des méthodes par sous-espaces de Krylov [46]. Elle permet de résoudre les systèmes linéaires dont la matrice est symétrique définie positive. C'est une méthode exacte, sous certaines conditions (propriétés de la matrice, arithmétique exacte...), qui permet de converger en un nombre fini d'itérations, inférieur ou égal à la taille du système linéaire. Cette méthode est employée pour résoudre le problème d'interface obtenu avec la DDM (système (1.16) avec matrice symétrique définie positive dans notre cas). C'est précisément pour cette raison que nous allons la détailler un peu plus. Décrite en une phrase, la méthode est une réalisation d'une technique de projection orthogonale sur le sous-espace de Krylov $K_m(r_0, K)$, où r_0 est le résidu initial [46]. Dans cette méthode le principe consiste à minimiser la fonctionnelle suivante [50] :

$$J(u) = \frac{1}{2}u^t K u - b^t u\,, \tag{3.1}$$

tel que le gradient de J est équivalent au résidu $Ku - b$. La méthode du gradient conjugué a la propriété de générer des directions de descente qui sont K-orthogonales

(ou K-conjuguées). Cette propriété est importante car nous pouvons minimiser la fonctionnelle J sur l'espace affine généré par ces directions. La méthode permet de générer ces vecteurs et, à partir d'une solution initiale u_0 choisie, de calculer à chaque itération k la solution u_{k+1}, et ceux jusqu'à l'obtention d'un résidu inférieur à une tolérance tol, avec l'algorithme 3 :

Algorithm 3 Gradient Conjugé

Require : $r_0 = Ku_0 - b, p_0 = r_0, k = 0$
  while $r_k^t r_k > tol$ **do**
    $\alpha_k \leftarrow -\frac{r_k^T p_k}{p_k^T K p_k}$
    $u_{k+1} \leftarrow u_k + \alpha_k p_k$
    $r_{k+1} \leftarrow K u_{k+1} - b$
    $\beta_{k+1} \leftarrow \frac{r_{k+1}^T K p_k}{p_k^T K p_k^T}$
    $p_{k+1} \leftarrow -r_{k+1} + \beta_{k+1} p_k$
    $k \leftarrow k + 1$

Le choix entre une méthode directe et une méthode itérative dépend des caractéristiques du système, de la taille de la matrice, et des ressources disponibles.

Les méthodes directes sont souvent préférées pour des systèmes de petite à moyenne taille, tandis que les méthodes itératives peuvent être plus efficaces pour des systèmes plus grands ou pour les systèmes creux (sparse). Le domaine d'application spécifique et les caractéristiques de la matrice influent également sur le choix de la méthode appropriée.

Dans Flux, l'utilisation de ces solveurs s'effectue à travers deux bibliothèques, MUMPS et PETSc. Ils peuvent être utilisés comme points de comparaison pour la DDM.

3.3.1 MUMPS

MUMPS (MUltifrontal Massively Parallel sparse direct Solver) [51] est une bibliothèque de solveurs directes pour des problèmes linéaires avec des matrices creuses développée pour être utilisée sur des architectures massivement parallèles. Elle est particulièrement efficace pour résoudre des systèmes linéaires provenant de problèmes complexes issus de la modélisation et de la simulation numérique.

MUMPS implémente une méthode directe basée sur une approche multifrontale (cette approche est expliquée dans la partie factorisation) qui effectue une décomposition :

$$A = LU\,,$$

où L est une matrice triangulaire inférieure et U une matrice triangulaire supérieure. Si la

matrice est symétrique, alors la factorisation est donnée par :

$$A = LDL^T\,,$$

où D est une matrice diagonale par blocs avec des blocs de taille 1 ou 2 sur la diagonale.

Le système $Ax = b$ est résolu en trois étapes principales :

1. **Analyse**
   Pendant l'analyse, un prétraitement est effectué, pour rendre la factorisation moins coûteuse, ainsi qu'une factorisation symbolique. Pendant cette dernière, une cartographie du graphe computationnel multifrontal [52], autrement dit l'arbre d'élimination, est calculée et utilisée pour estimer le nombre d'opérations et la mémoire nécessaires à la factorisation et à la résolution. Des implémentations parallèles et séquentielles de la phase d'analyse sont disponibles. Soit A_{pre} la matrice prétraitée.
2. **Factorisation**
   En fonction de la symétrie de la matrice prétraitée A_{pre} la factorisation, $A_{\text{pre}} = LU$ ou $A_{\text{pre}} = LDL^T$, est calculée. La matrice originale est d'abord distribuée (ou redistribuée) sur les processeurs en fonction de la cartographie calculée pendant l'analyse. Elle sera appelée matrice frontale. La factorisation numérique est ensuite une séquence de factorisations denses sur ces matrices frontales. En plus du pivotement standard avec seuil et du pivotement de bloc 2x2 (moins standard dans les codes à mémoire distribuée), il existe une option pour effectuer un pivotement statique [8] [53]. L'arbre d'élimination exprime également l'indépendance entre les tâches et permet le traitement simultané de plusieurs fronts. Cette approche est appelée approche multifrontale. Après la factorisation, les matrices factorisées sont conservées distribuées (en mémoire centrale ou sur disque) pour être utilisées lors de la phase de résolution.
3. **Solution**
   La solution x_{pre} de $LU\mathbf{x}_{\text{pre}} = b_{\text{pre}}$ ou $LDL^T x_{\text{pre}} = b_{\text{pre}}$, où x_{pre} et b_{pre} sont la solution transformée x et le second membre b associés à la matrice prétraitée A_{pre}. x_{pre} est obtenu à partir d'une descente :

   $$Ly = b_{\text{pre}} \text{ ou } LDy = b_{\text{pre}}\,,$$

   suivie d'une étape de remontée :

   $$Ux_{\text{pre}} = y \text{ ou } L^T x_{\text{pre}} = y\,.$$

   La solution x_{pre} est finalement post-traitée pour obtenir la solution x du système

8. Pivotement statique : un pivot plus petit qu'un seuil en valeur absolue est remplacé par ce seuil.

d'origine $Ax = b$, où x est soit assemblée sur un processeur identifié (l'hôte) soit conservée distribuée sur les processeurs en activité. Le raffinement itératif et l'analyse d'erreur rétrograde sont également des options de post-traitement de la phase de résolution [51].

3.3.2 PETSc

PETSc (Portable, Extensible Toolkit for Scientific Computation) [54] offre un ensemble riche d'outils pour résoudre des problèmes complexes de calcul scientifique sur une variété d'architectures matérielles. PETSc se distingue par sa capacité à traiter une gamme étendue de problèmes, des systèmes linéaires et non linéaires aux équations aux dérivées partielles. Elle est compatible avec divers langages de programmation tels que C, C++, et Fortran. Elle est présentée sous forme de module que nous pouvons voir dans la figure 3.6.

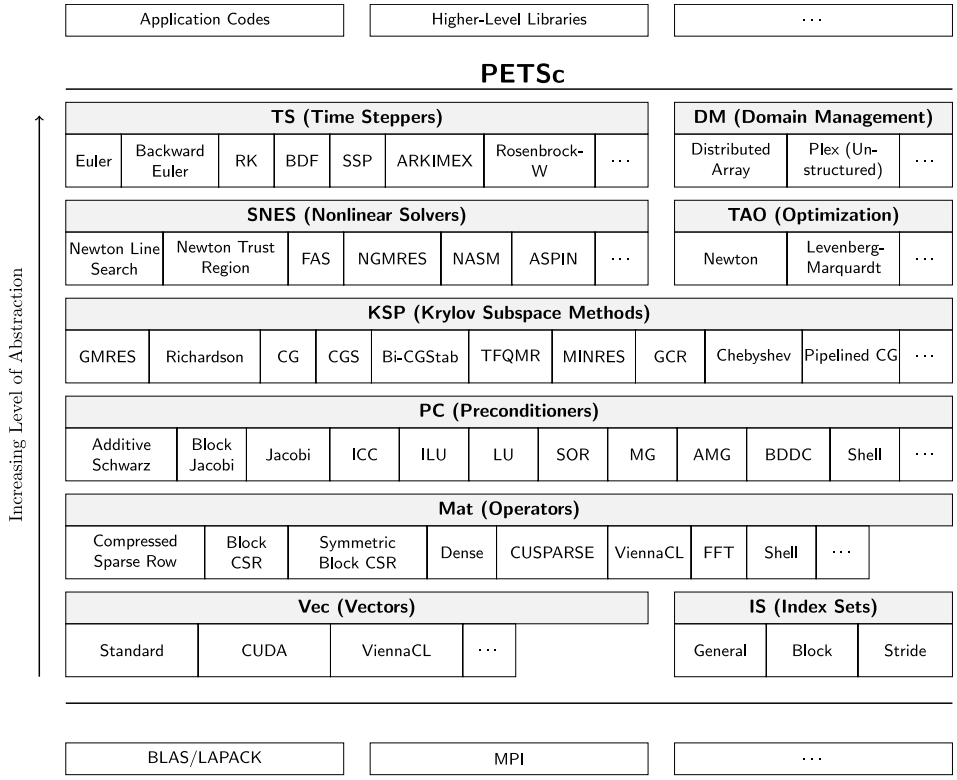

Figure 3.6 – Différents modules de la bibliothèque PETSc [54]

3.4 Optimisation VTune et mesure de performances

Intel VTune Profiler [55] est un logiciel de profilage de performances développé par Intel. Il est conçu pour aider les développeurs à optimiser les performances de leurs applications

en identifiant les goulots d'étranglement et en suggérant des améliorations.

Fonctionnalités Clés

— **Analyse de la Performance :** Le logiciel permet d'analyser la performance d'une application en identifiant les parties du code qui utilisent le plus de ressources CPU, de mémoire, et d'autres ressources système.
— **Localisation des Problèmes de Performances :** Il aide à localiser les problèmes de performances tels que les accès mémoire inefficaces, les boucles non optimisées, les goulots d'étranglement de CPU, etc.
— **Profiling Avancé :** Le profiler offre un profiling avancé qui prend en charge le profilage basé sur l'échantillonnage [9], le profilage de la consommation d'énergie, et d'autres mesures.
— **Analyse de la Parallélisation :** Il permet d'évaluer la parallélisation d'une application pour tirer parti des architectures multi-cœurs.

En résumé, Intel VTune Profiler est un outil puissant pour les développeurs cherchant à optimiser les performances de leurs applications, en particulier sur des architectures matérielles Intel. Il nous a été utile pour à la fois évaluer les performances de la méthode de décomposition de domaine, optimiser au mieux le code et détecter les possibles problèmes en terme d'implémentation.

3.4.1 Mesure de performances

L'analyse de performance du code de la DDM s'effectuera en quatre étapes et de la manière suivante :

1. Un premier profilage avec VTune sera effectué pour identifier les parties du code qui consomment le plus de ressources.
2. Une mesure du temps d'exécution total du code ainsi que des parties les plus importantes sera effectuée : lecture, assemblage et résolution.
3. Une mesure de la consommation de mémoire du code sera réalisée. Nous nous concentrerons principalement sur le pic de consommation lors de la résolution.
4. Une évaluation de l'efficacité de la parallélisation sera effectuée.

En combinant ces méthodes, nous pouvons obtenir une image complète des performances du code DDM.

9. Profilage par l'échantillonnage : collecte périodique des données sur l'état de l'application pendant son exécution au lieu de suivre chaque instruction ou chaque événement système.

3.5 Partitionnement de maillage

Le partitionnement du maillage représente une étape cruciale dans le processus de parallélisation, où la performance du calcul est directement influencée par la taille des interfaces entre les sous-domaines et par un équilibrage adéquat des charges. Pour ce travail, nous avons choisi de travailler sur un partitionnement à deux étapes : partitionner par région physique et partitionner par taille.

Ce choix a deux intérêt majeurs :

- Gain de performance à l'entrée : l'architecture du logiciel Altair permet d'avoir le maillage des régions physiques facilement et avec un très faible coût en termes de temps d'exécution.
- Gain de performance de la DDM : les matrices obtenues avec des régions homogènes ont des coefficients du même ordre de grandeur et donc plus facile à préconditionner.

Si nous reprenons l'exemple 1.3, la figure 3.7 illustre le partitionnement obtenu. Les régions en rouge clair et rouge foncé représentent la même région physique, de même pour les régions en bleu clair et en bleu foncé. Le résultat final est constitué de 4 sous-domaines homogènes.

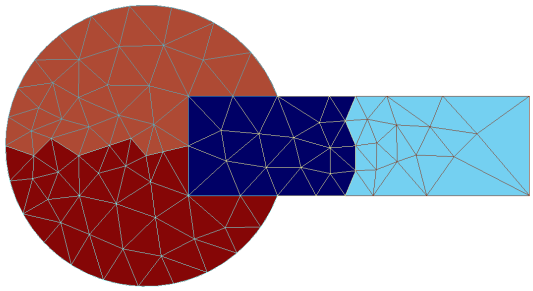

Figure 3.7 – Illustration d'un partitionnement de maillage basé sur la physique.

3.5.1 Concept

Le partitionnement de maillage vise à maximiser l'efficacité de la parallélisation en répartissant judicieusement les éléments du maillage entre les processeurs et à réduire les communications dues à l'interface. La plupart des partionneurs reposent sur des algorithmes

à deux étapes [56] :

1. Un algorithme *glouton* initie le processus en sélectionnant N_{proc} éléments de maillage, appelés *germes*, aussi éloignés les uns des autres que possible, où N_{proc} représente le nombre de processeurs impliqués dans le calcul. Chacun de ces germes devient le centre d'un sous-domaine dans la partition, autour duquel les éléments adjacents sont ajoutés pour former la partition.
2. Une amélioration du partitionnement obtenu avec l'algorithme *glouton* se réalise par la minimisation du nombre de nœuds aux interfaces. À cet effet, chaque élément se voit attribuer un critère tenant compte du nombre total d'éléments par processeur et du nombre de communications avec les éléments d'autres processeurs. Cela conduit à une réattribution ou non de l'élément à un autre processeur.

3.5.2 Méthodes

Il existe un certain nombre de bibliothèque de partitionnement de maillage. Nous proposons une brève explication des plus utilisées.

1. **Metis :** Metis [57] regroupe un ensemble de programmes séquentiels dédiés à la partition de graphes, à la division de maillages éléments finis, et à la création d'ordonnancements réduisant le remplissage pour les matrices creuses. Les algorithmes de Metis s'appuient sur des schémas de partitionnement récursif multiniveau, de partitionnement k-way multiniveau [10] et de partitionnement multi-contraintes [11] [58, 59, 60].
2. **Party :** La bibliothèque de partitionnement Party [61] offre diverses méthodes de partitionnement de manière simple et intuitive. Plutôt que d'implémenter directement ces méthodes, les utilisateurs peuvent exploiter celles déjà intégrées à la bibliothèque. Toutes les implémentations visent à améliorer les performances des heuristiques de partitionnement. La bibliothèque propose deux types d'interfaces permettant une utilisation autonome ainsi qu'une intégration aisée dans des codes d'application. Les structures de données utilisées comme interface pour Party sont simples et faciles à générer.
3. **Jostle :** Jostle [62] est un package logiciel spécialement conçu pour le partitionnement de maillages non structurés, tels que ceux utilisés dans les méthodes d'éléments finis

10. L'algorithme de partitionnement k-way à plusieurs niveaux réduit la taille du graphique en réduisant les sommets et les arêtes (phase de grossissement), trouve une partition k-way du graphique plus petit, puis construit une partition k-way pour le graphique original en projetant et en affinant la partition sur des graphiques de plus en plus fins (phase de dégrossissement).

11. Le partitionnement multi-contraintes est une fonctionnalité de la bibliothèque METIS qui permet de diviser un graphe en plusieurs sous-graphes tout en respectant plusieurs contraintes tel que l'équilibre de la taille des partitions, la minimisation du nombre d'arêtes coupées, ou d'autres critères spécifiques à l'application.

ou de volumes finis, destiné à être utilisé sur des ordinateurs parallèles à mémoire distribuée. Il offre également la possibilité de repartitionner et d'équilibrer des partitions existantes (par exemple, ceux issues de maillages adaptatifs raffinés). Le fonctionnement de Jostle repose sur la modélisation du maillage en tant que graphe non orienté et sur l'utilisation de techniques de partitionnement de graphe.

4. **Scotch :** La distribution Scotch [63] regroupe un ensemble de programmes et de bibliothèques mettant en œuvre un algorithme de static mapping [12] et de réorganisation de matrices creuses. Elle fournit des algorithmes de partitionnement de graphe et d'hypergraphe [13].
5. **Chaco :** Chaco [65] propose une variété d'algorithmes de partitionnement de graphes et les intègre dans un package. Ce code est utilisé pour simplifier le développement d'applications parallèles et garantir des performances élevées.
6. **Zoltan :** La bibliothèque Zoltan [66] rassemble une collection de services de gestion de données destinés à des applications parallèles, non structurées, adaptatives et dynamiques. Elle simplifie les défis liés à l'équilibrage de charge, au déplacement de données, à la communication non structurée, et à l'utilisation de la mémoire. On rencontres ces problèmes dans des applications dynamiques telles que les méthodes d'éléments finis adaptatifs, les méthodes de particules, et les simulations de collisions. La conception de Zoltan, neutre vis-à-vis de la structure des données, permet à un large éventail d'applications de l'utiliser sans imposer de restrictions sur les structures de données d'application.

3.5.3 Choix de l'outil de partitionnement

Pour les travaux de thèse, nous avons choisi de tester Metis dans un premier temps. Ce choix est motivé par la disponibilité de la bibliothèque, le caractère open source du code, et le fait que la bibliothèque est déjà compilée avec le logiciel Flux. De plus, différentes références bibliographiques indiquent l'utilisation courante de Metis en tant que partitionneur de maillage. La référence [67] souligne également que Metis est plus rapide que d'autres outils, les auteurs indiquent que : *des expériences sur un large éventail de graphes ont montré que METIS est une à deux ordres de grandeur plus rapide que d'autres algorithmes de partitionnement largement utilisés* [14].

12. Static mapping : le problème d'optimisation consistant à attribuer les processus de communication d'un programme parallèle à une machine parallèle afin de minimiser son temps d'exécution global [64]

13. Dans Scotch, il existe une compatibilité avec Metis : certaines fonctions de Metis peuvent être appelées dans Scotch, ce qui appellera leurs équivalents.

14. Experiments on a wide range of graphs have shown that METIS is one to two orders of magnitude faster than other widely used partitioning algorithms.

3.6 Conclusion

Dans ce chapitre, nous avons présenté un ensemble d'outils que nous avons utilisés tout au long de notre travail. Ces outils sont essentiels pour obtenir les meilleures performances de la méthode de décomposition de domaine. Les bibliothèques MUMPS et PETSc facilitent l'implémentation de la méthode, tandis que l'outil VTune permet un profilage précis. Le choix du partitionneur est crucial pour l'application de la décomposition de domaine. L'environnement de travail Flux Solver représente un autre défi que nous avons dû relever pour atteindre les résultats souhaités. Nous avons essayé de mettre en place la meilleure stratégie de parallélisme possible en tenant compte de tous ces paramètres.

Chapitre 4

Résultats de la DDM pour la formulation H-conforme

4.1 Signification des multiplicateurs de Lagrange pour FETI-1

Nous avons constaté que la résolution de problèmes de magnétostatique à l'aide de la méthode des éléments finis implique l'utilisation de formulations en potentiel scalaire ou en potentiel vecteur, discrétisées respectivement au moyen d'éléments finis nodaux ou d'arête. Chaque cas apporte son lot d'avantages et de complications. Le potentiel scalaire, par opposition au potentiel vecteur, a l'avantage d'être de nature scalaire plutôt que vectorielle, ce qui simplifie l'implémentation [1], peut diminuer [2] la dimension du problème à résoudre et nécessite une étape d'intégration moins lourde [3]. Cependant, il nécessite un traitement spécifique pour les problèmes multiplement connexes. En revanche, le potentiel vecteur ne présente pas ce problème, mais exige la définition d'une condition de jauge pour assurer son unicité, plus compliquée que celle pour le potentiel scalaire où il suffit juste de le fixer en un point. Les formulations en potentiel vecteur offre aussi une meilleure convergence en non-linéaire [69]. Malgré le fait que la DDM puisse être plus adaptée à une résolution avec les formulations en potentiel scalaire, les avantages des formulations en potentiel vecteur ont orienté notre approche pour explorer la possibilité d'adapter la méthode de décomposition de domaine aux deux formulations.

Comme nous l'avons expliqué au chapitre 1, nous avons testé dans un premier temps la méthode **FETI-1** [2, 7] qui peut fournir une base pour le développement des méthodes plus élaborées. Le but de cette section est d'écrire le système **FETI-1** en partant des

1. Le potentiel scalaire magnétique permet d'utiliser une seule inconnue dans l'air et de mettre en place un couplage avec les équations du circuit électrique. Son utilisation permet également de simplifier la mise en œuvre du raccordement de la solution éléments finis entre les parties fixes et mobiles au niveau d'un cylindre de glissement [68]
2. Il faut comparer le nombre d'arêtes au nombre de nœuds.
3. La matrice EF s'obtient par des produits scalaires entre vecteurs.

équations de la magnétostatique déjà présentées.

Considérons le cas avec deux sous-domaines Ω_1 et Ω_2, voir la figure 1.4. À partir de l'équation (2.40), le problème variationel avec formulation en potentiel scalaire revient à trouver u point stationnaire de la fonctionnelle d'énergie :

$$J(v) = \frac{1}{2}a(v,v) - (v, \mathbf{b_r}) \tag{4.1}$$

avec

$$a(v,\Phi) = \int_{\Omega_i} \mu \operatorname{grad} v \cdot (\mathbf{h_s} - \operatorname{grad} \Phi) \ \mathrm{d}\Omega_i \,, \tag{4.2}$$

$$(v, \mathbf{b_r}) = \int_{\Omega_i} \operatorname{grad} v \cdot \mathbf{b_r} \ \mathrm{d}\Omega_i \,. \tag{4.3}$$

La résolution du problème magnétostatique est également équivalent à trouver u_1 et u_2 points stationnaires des fonctionnelles d'énergie $J_1(v_1)$ et $J_2(v_2)$ de la même forme que (4.1) mais avec les intégrations respectivement restreintes à Ω_1 et Ω_2 et qui satisfont, sur l'interface entre les deux sous-domaines Γ_I (figure 1.4), la condition :

$$u_1 = u_2 \,. \tag{4.4}$$

Résoudre le problème variationnel exprimé sur les deux sous-domaines, en prenant en compte la condition de continuité (4.4) est équivalent à trouver le point stationnaire du Lagrangien :

$$J^*(v_1, v_2, \tau) = J_1(v_1) + J_2(v_2) + (v_1 - v_2, \tau)_{\Gamma_I} \tag{4.5}$$

avec

$$(v_1 - v_2, \tau)_{\Gamma_I} = \int_{\Gamma_I} \tau(v_1 - v_2) \ \mathrm{d}\Gamma \,. \tag{4.6}$$

Cela revient à trouver u_1 et u_2 ainsi que le multiplicateur de Lagrange λ qui vérifie pour tout v_1, v_2 et τ [70] :

$$J^*(u_1, u_2, \tau) \leqslant J^*(u_1, u_2, \lambda) \leqslant J^*(v_1, v_2, \lambda) \tag{4.7}$$

À partir de l'inégalité de gauche et l'équation (4.5) nous pouvons avoir :

$$(u_1 - u_2, \tau)_{\Gamma_I} \leqslant (u_1 - u_2, \lambda)_{\Gamma_I} \tag{4.8}$$

Nous posons $\tau = \widetilde{\tau} + \lambda$, ce qui donne :

$$(u_1 - u_2, \widetilde{\tau})_{\Gamma_I} + (u_1 - u_2, \lambda)_{\Gamma_I} \leqslant (u_1 - u_2, \lambda)_{\Gamma_I} \tag{4.9}$$

$$(u_1 - u_2, \widetilde{\tau})_{\Gamma_I} \leqslant 0 \tag{4.10}$$

Si $\tau = -\widetilde{\tau} + \lambda$ nous obtenons :

$$-(u_1 - u_2, \widetilde{\tau})_{\Gamma_I} + (u_1 - u_2, \lambda)_{\Gamma_I} \leqslant (u_1 - u_2, \lambda)_{\Gamma_I} \tag{4.11}$$

$$0 \leqslant (u_1 - u_2, \widetilde{\tau})_{\Gamma_I} \tag{4.12}$$

Ce qui impose que pour tout τ admissible, $(u_1 - u_2, \tau)_{\Gamma_I} = 0$ et donc $u_1 = u_2$ sur Γ_I. L'inégalité de droite implique que pour toutes les paires (v_1, v_2) admissibles, qui satisfont la condition de continuité, la paire (u_1, u_2) minimise la somme des fonctionnelles d'énergie J_1 et J_2 définies sur Ω_1 et Ω_2. En conséquence, u_1 et u_2 sont les restrictions de la solution u du problème non partitionné.

Dans un contexte d'un problème magnétostatique, nous allons voir le sens physique de la variable duale λ.

Commençons par le problème de type Poisson (1.1) déjà mentionné dans les chapitres précédents. Pour des raisons de simplicité, nous considérerons un milieu homogène. En utilisant la formule de Green sur le problème (1.1), nous obtenons :

$$\int_\Omega -\Delta u \cdot v = \int_\Omega fv$$
$$\int_\Omega -\Delta u \cdot v = \int_\Omega \nabla u \cdot \nabla v - \int_{\partial\Omega} v \nabla u \cdot \mathbf{n}.$$

Le dernier terme est nul du fait de la condition de Dirichlet ($v = 0$ sur $\partial\Omega$) et cela donne

$$\int_\Omega \nabla u \cdot \nabla v = \int_\Omega fv. \tag{4.13}$$

Nous appliquons la même approche sur les systèmes liés aux sous-domaines 1 et 2. Pour chaque sous-domaine, la condition de Dirichlet est valable sur $\partial\Omega_s \backslash \Gamma$, $s = 1, 2$. Ceci fait apparaître un terme en $\nabla u_s \cdot \mathbf{n}_s$ et la formulation variationnelle correspondante est :

$$\int_{\Omega_s} \nabla u_s \cdot \nabla v_s = \int_{\Omega_s} f_s v_s + \int_\Gamma (\nabla u_s \cdot \mathbf{n}_s) v_s \,. \tag{4.14}$$

Pour retrouver la solution sur le domaine global Ω, nous additionnons les deux problèmes

en prenant en compte que :

$$v = v_1 \text{ sur } \Omega_1 \text{ et } v_2 \text{ sur } \Omega_2.$$

Le problème variationnel global devient :

$$\int_\Omega \nabla u \cdot \nabla v = \int_\Omega fv + \int_\Gamma (\nabla u_1 \cdot \mathbf{n}_1 + \nabla u_2 \cdot \mathbf{n}_2)v \tag{4.15}$$

L'équation (4.15) indique que pour avoir une équivalence entre la solution issue des deux solutions locales et la solution du domaine global, nous avons besoin d'imposer une condition de continuité sur l'interface

$$u_1 = u_2 \text{ sur } \Gamma$$

et nous avons également besoin, par comparaison à (4.13), d'une condition sur $\nabla u_i \cdot \mathbf{n}_i$ telle que :

$$\nabla u_1 \cdot \mathbf{n}_1 = -\nabla u_2 \cdot \mathbf{n}_2 \text{ sur } \Gamma\,.$$

La méthode FETI cherche à satisfaire cette dernière condition avec un processus itératif pour calculer λ tel que :

$$\nabla u_1 \cdot \mathbf{n}_1 = -\nabla u_2 \cdot \mathbf{n}_2 = \lambda \text{ sur } \Gamma$$

C'est ce même λ qui vérifie la relation (4.7). Dans le cas des matériaux hétérogènes, nous aurons μ qui apparaît également dans cette condition.

Le vecteur de multiplicateurs de Lagrange représente donc exactement le flux pour le problème *continu*. Des travaux existent pour démontrer que nous pouvons approcher le flux au niveau discret de la même manière. Nous citons notamment les travaux [71], [72] et [73] qui montrent que cette approximation est suffisamment précise après une comparaison à un cas pour lequel le flux est calculé analytiquement.

Nous commençons par la propriété de conservation qui implique que le flux entrant ou sortant d'une quantité q à travers la frontière S qui englobe une région V est égal à la somme des sources f dans V tel que :

$$\int_{\mathbf{S}} \mathbf{q} \cdot \mathbf{n}\, \mathrm{d}S = \int_S \sigma \mathrm{d}S = \int_V f\ \mathrm{d}V\,, \tag{4.16}$$

avec $\sigma = \mathbf{q} \cdot \mathbf{n}$ qui représente la composante normal du flux.

À partir du théorème de flux-divergence, nous écrivons :

$$\int_S \mathbf{q} \cdot \mathbf{n}\, \mathrm{d}S = \int_V \nabla \cdot \mathbf{q}\, \mathrm{d}V \tag{4.17}$$

Nous pouvons déduire alors $\nabla \cdot \mathbf{q} = f$. Cette équation est utilisée pour écrire la première identité de Green :

$$\int_S v \nabla u \cdot \mathbf{n} \, \mathrm{d}S = \int_V \nabla u \cdot \nabla v \, \mathrm{d}V + \int_V v \Delta u \, \mathrm{d}V \,. \tag{4.18}$$

Si nous écrivons $\sigma = \nabla u \cdot \mathbf{n}$ alors

$$\int_S \sigma v \, \mathrm{d}S = \int_V \nabla u \cdot \nabla v \, \mathrm{d}V + \int_V v \Delta u \, \mathrm{d}V \,. \tag{4.19}$$

Nous pouvons remplacer $\Delta u = f$ dans l'équation précédente pour un problème de type Poisson, pour obtenir :

$$\int_S \sigma v \, \mathrm{d}S = \int_V \nabla u \cdot \nabla v \, \mathrm{d}V + \int_V f v \, \mathrm{d}V \,. \tag{4.20}$$

Dans [74], les auteurs expliquent comment calculer le flux à partir des multiplicateurs de Lagrange. Nous posons $v = \varphi_i$ avec φ_i une base et u_h l'approximation de u dans le sous-espace H^1_h, sous-espace de dimension finis de $H^1(\Omega)$ (voir liste des symboles), le système (4.20) devient :

$$\int_S \sigma_h \varphi_i \, \mathrm{d}S = \int_V \nabla u_h \cdot \nabla \varphi_i \, \mathrm{d}V + \int_V f \varphi_i \, \mathrm{d}V \,. \tag{4.21}$$

En potentiel scalaire, le terme de gauche peut s'écrire sous la forme d'une somme $\sum_{j=1}^{N} (w_j, w_i) \mathrm{h}_{s_j} \mathbf{w}_j$ avec w_j la fonction de base associée au nœud j [71].

Cette approximation du flux converge mieux que l'approximation utilisant directement $\nabla u_h \cdot \mathbf{n}$. Elle a aussi par construction une propriété de conservation qui permet d'avoir la même valeurs du flux des deux cotés de l'interface à un signe près.

Dans notre travail, le calcul du flux ne nous intéresse pas. Nous nous intéressons seulement à l'apport algébrique du vecteur de multiplicateurs de Lagrange exprimé dans le système discrétisé. Aussi, nous pouvons noter une conservation du flux sur l'interface, ce qui nous permet de généraliser pour N sous domaines et dire que le vecteur de multiplicateurs de Lagrange assure que la somme des flux est nulle sur l'interface.

Le système algébrique créé suite à une discrétisation du problème est de la forme [2] :

$$K_1 u_1 = b_1 + B_1^T \lambda \tag{4.22}$$

$$K_2 u_2 = b_2 + B_2^T \lambda \tag{4.23}$$

$$B_1 u_1 = B_2 u_2 \tag{4.24}$$

avec B_i, $i = 1, 2$ les matrices de restriction sur l'interface, voir la sous-section 1.4.1.

4.2 Cas linéaire 3D

Nous présenterons trois cas tests pour la simulation en magnétostatique avec des matériaux linéaires. Le premier, de nature plus académique, nous permettra d'étudier le comportement de la décomposition de domaine de manière approfondie. Le deuxième, plus industriel, illustrera à la fois l'application de la méthode à des problèmes concrets et mettra en évidence les limites de l'approche de la décomposition fondée sur les propriétés physiques. Un troisième cas test intégrera le partitionneur de maillage (section 3.5).

Pour l'ensemble des cas tests, les performances obtenues avec FETI-1 et FETI-DP sont similaires. Nous fournirons une comparaison des deux pour le premier cas test (figure 4.11), mais en général, la différence de temps de calcul est inférieure à 1%.

4.2.1 Test 1

Le premier cas est composé de 4 aimants en forme de cubes, de tailles égales. Nous déclinons ce cas test avec différents maillages pour changer la taille du problème.

Le cas test reprend l'exemple de la figure 1.6. Le partitionnement par région physique nous fournit 4 sous-domaines. Le nombre d'inconnues par sous-domaine varie en fonction du maillage, mais il est équilibré entre les sous-domaines.

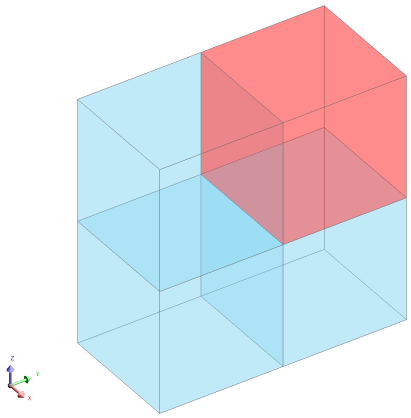

Figure 4.1 – Cas test 1 : géométrie

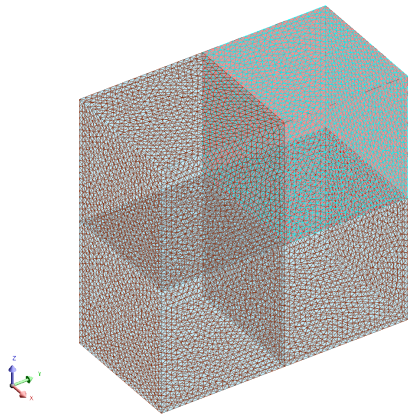

Figure 4.2 – Cas test 1 : maillage

La figure 4.1 illustre la géométrie utilisée. Le cube en rouge représente le sous-domaine non flottant et les autres sont flottants (voir sous-section 1.4.2). La figure 4.2 montre un maillage régulier sur l'ensemble des sous-domaines.

En général, les logiciels de simulation offrent la possibilité de paramétrer le maillage. Sur Flux, cela se traduit principalement par trois options :

— **La déflection :** le réglage de la déflection permet le découpage en éléments de maillage des objets courbes.

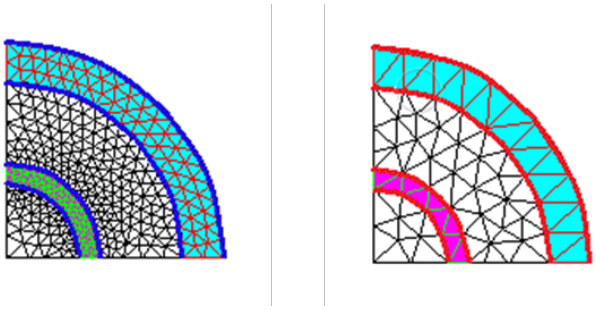

Figure 4.3 – Avec déflection (gauche) et sans déflection (droite)

— **La relaxation :** elle permet de lisser la répartition des mailles sur les lignes, les faces et les volumes.

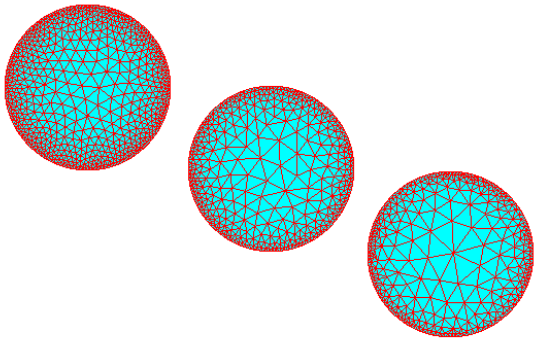

Figure 4.4 – Relaxation faible, moyenne et forte

— **L'ombrage :** il améliore le maillage des faces proches.

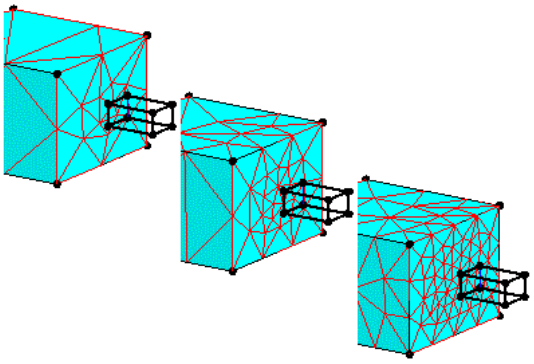

Figure 4.5 – Ombrage faible, moyen et fort

Ces paramètres génèrent souvent un maillage non homogène (des éléments de tailles différentes) qui peut être concentré sur l'interface entre les régions. La figure 4.6 montre le résultat visuel pour un tel problème. La DDM résout le problème réduit sur l'interface, le nombre d'inconnues sur cette interface influençant les performances de la méthode. Nous avons pris ces paramètres en compte dans les tests que nous avons effectués et nous avons conservé le maillage par défaut, malgré les inconvénients que cela peut présenter pour la DDM [4].

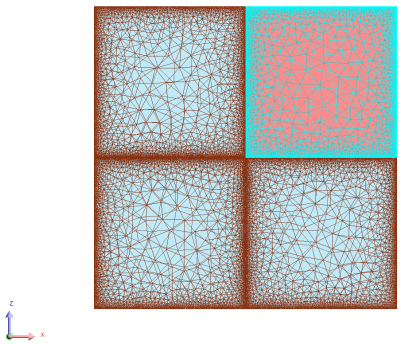

Figure 4.6 – Maillage concentré sur l'interface

Erreur

Nous prenons comme critère d'arrêt du gradient conjugué une tolérance à 10^{-7} en relatif sur le résidu, soit $\frac{r^t Pr}{r^{t(0)} r^{(0)}} \leqslant 10^{-7}$ (algorithme 1). L'évolution de ce résidu est montré dans la figure 4.7.

4. Les méthodes FETI résolvent un problème sur l'interface. Les options de maillage peuvent entraîner un maillage fin de cette interface, ce qui augmente le nombre de variables et donc la taille du problème résolu par la DDM.

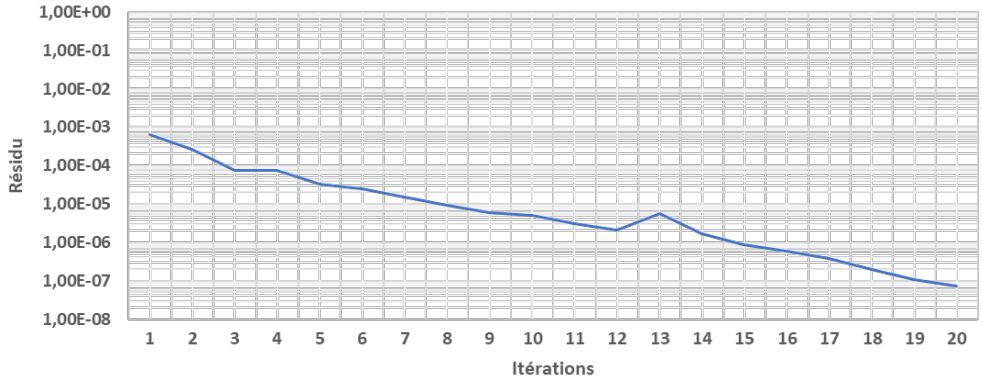

Figure 4.7 – Évolution du résidu en fonction des itérations du PCPG.

En utilisant le preconditionneur de Dirichlet (1.21), la convergence est obtenue après une vingtaine d'itérations.

Performance en temps

Nous allons commencer par identifier les étapes qui demandent le plus de temps d'exécution. Cela nous permettra d'évaluer précisément les performances de la méthode, en confirmant qu'une augmentation (le cas échéant, une diminution) est strictement liée à la méthode de résolution utilisée. Le profilage avec VTune nous offre cette certitude.

La figure 4.8 montre la proportion du temps consommé par rapport à l'exécution complète pour des différentes fonctions.

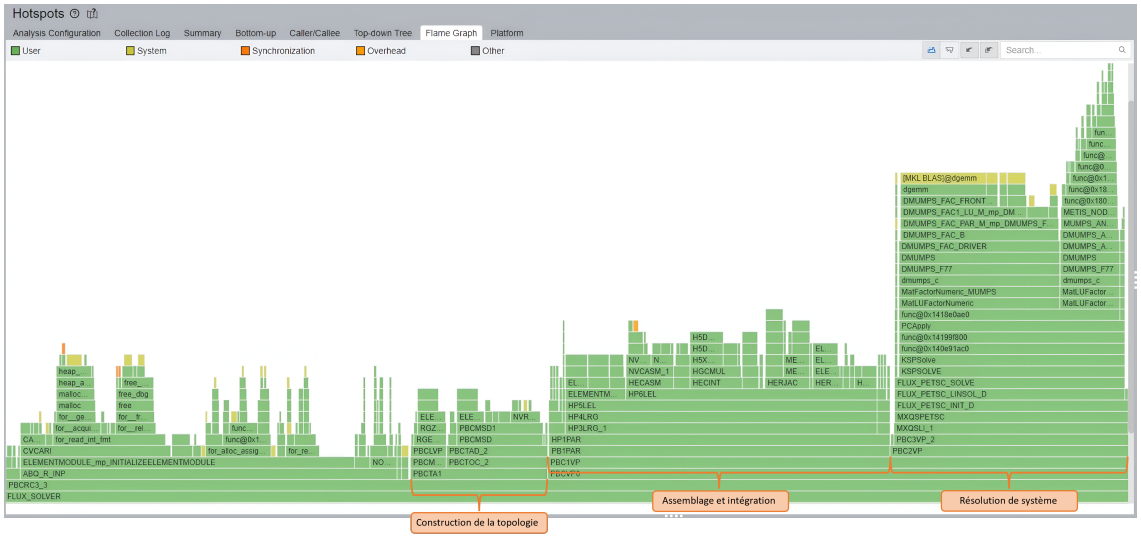

Figure 4.8 – Profilage avec VTune

Pour comparer les performances en termes de temps, nous avons mesuré le temps nécessaire à la construction de la matrice et à la résolution du problème. Ces deux étapes représentent la majeure partie du temps global. La phase de lecture du projet est également optimisée par le code de la DDM, mais cela résulte simplement d'un effet secondaire lié à la lecture parallèle des sous-domaines. Nous ne tiendrons pas compte des performances des entrées/sorties dans les résultats.

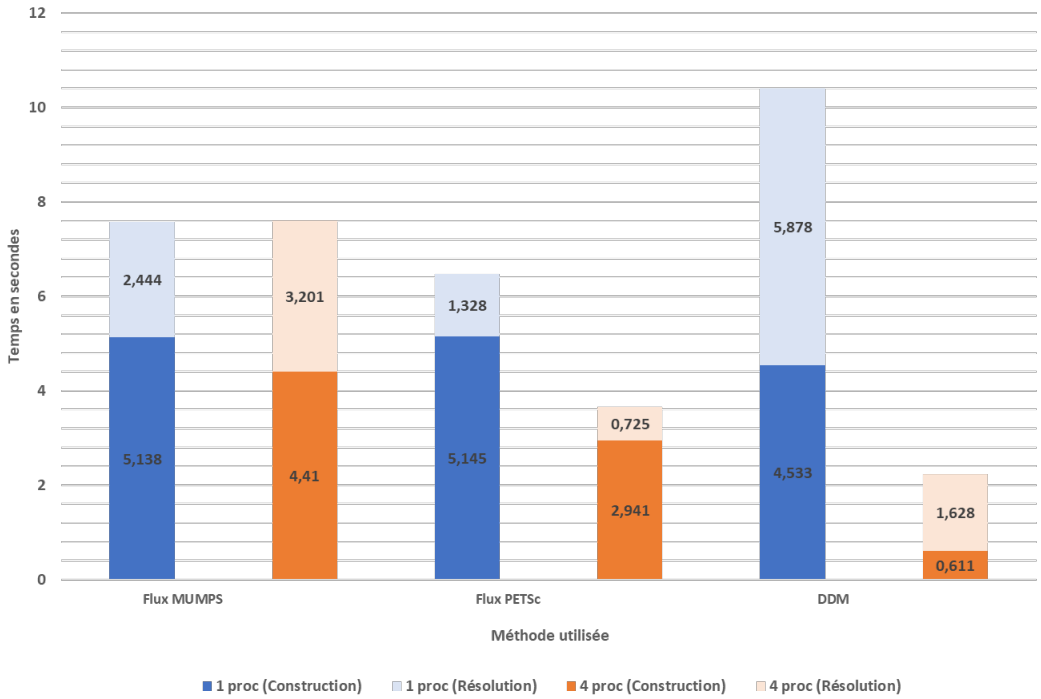

Figure 4.9 – Comparaison des temps de construction et de résolution (cas 1).

La figure 4.9 présente les temps pour un maillage de 105 637 nœuds, dont 16 744 sont situés sur l'interface partagée par les 4 sous-domaines, avec les tailles respectives suivantes : 30 463, 30 513, 30 292, 30 210.

Pour ce cas test avec des sous-domaines de tailles bien équilibrées, nous avons comparé les performances de la DDM en séquentiel et en parallèle avec 4 cœurs. Flux MUMPS effectue une résolution par une méthode directe (factorisation LU), tandis que Flux PETSc utilise une méthode itérative (GMRes) préconditionnée avec ILU. Hors DDM, les quatre cœurs sont exploités lors de la phase de résolution. La matrice est divisée en quatre parties, chacune étant attribuée à un cœur différent qui résout le système en parallèle.

Les résultats montrent que la DDM avec 4 cœurs est nettement plus performante en termes

de temps de calcul par rapport à ces deux méthodes, et qu'elle n'est pas adaptée à une exécution séquentielle. La figure 4.10 illustre l'effet de l'utilisation du préconditionneur de Dirichlet sur la résolution du problème réduit sur l'interface. Ceci est dû à la baisse du nombre d'itérations (20 contre 8 itérations).

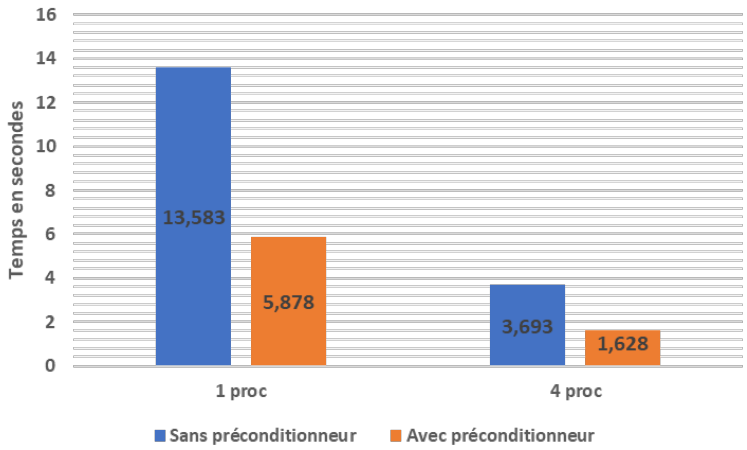

Figure 4.10 – Effet du préconditionneur sur la résolution du problème d'interface.

Nous terminons par une comparaison des temps de calcul entre FETI-1 et FETI-DP pour une exécution avec 4 processeurs. La taille du problème grossier compte 300 variables primales, comme défini dans la figure 1.7.Une variable est primale si elle est partagée par plus de deux sous-domaines. Si le nombre de sous-domaines est élevé, cette définition entraîne un grand nombre de variables primales et, par conséquent, une taille importante du problème grossier. Nous explorerons une alternative à cette définition dans le chapitre 5. Comme le montre la figure 4.11, notre implémentation donne des temps de calcul similaires pour les deux méthodes. Cette constatation est valide pour l'ensemble des cas tests ; par conséquent, nous n'indiquerons pas explicitement la méthode utilisée par la suite.

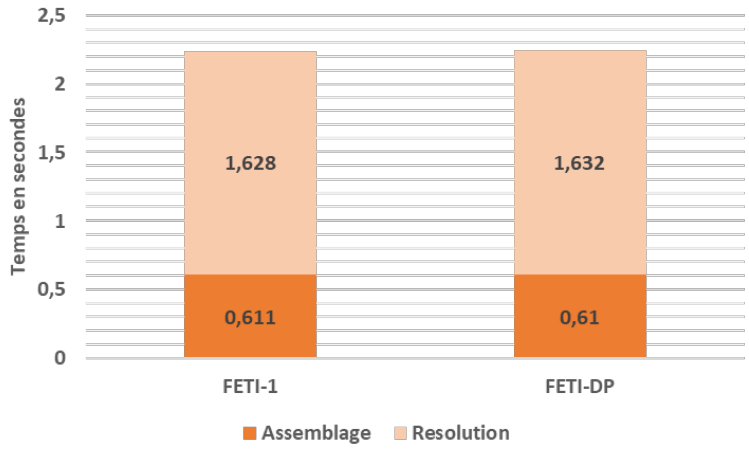

Figure 4.11 – Comparaison entre FETI-1 et FETI-DP pur le cas test 1.

Performance en mémoire

Nous pouvons aussi mesurer le pic de consommation mémoire. Lors de l'exécution avec 4 cœurs, nous prenons la somme sur les 4.

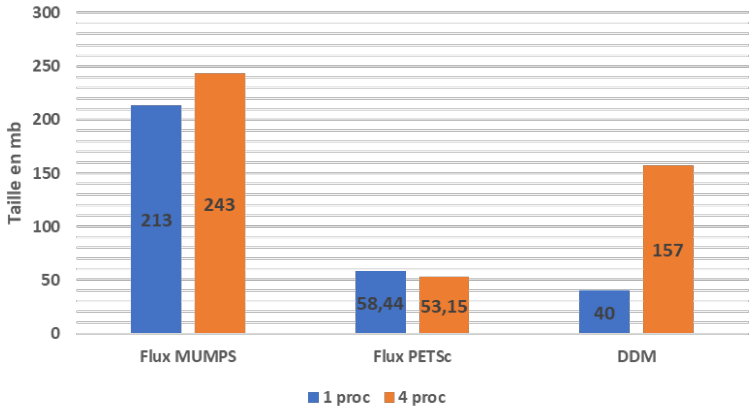

Figure 4.12 – Comparaison de la consommation mémoire entre les différents solveurs (cas 1).

La DDM se positionne entre le solveur itératif et le solveur direct. Ceci s'explique par l'utilisation d'un solveur direct sur chaque problème local.

Scalabilité

Pour la partie complexité, plusieurs maillages du cas test ont été générés allant de 5 000 à 500 000 nœuds. La figure 4.13 illustre l'évolution du temps de calcul en fonction de la taille du maillage. Le nombre de sous-domaines reste constant. La figure montre que pour une

taille globale comprise entre 100 000 et 140 000 nœuds, la DDM affiche des résultats jusqu'à 40 % meilleurs que le solveur itératif et jusqu'à 300 % meilleurs que le solveur direct. À mesure que la taille globale augmente, et donc la taille par sous-domaine, l'avantage de la DDM devient beaucoup moins significatif. Nous constatons que la configuration optimale pour la DDM, par rapport aux autres solveurs, est atteinte lorsque la taille par sous-domaine avoisine les 30 000 nœuds.

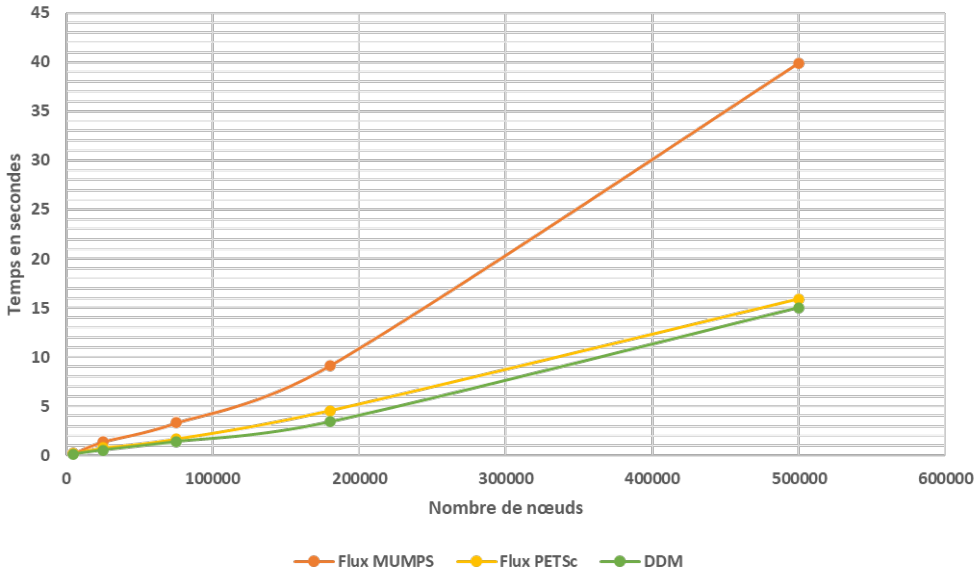

Figure 4.13 – Temps de calcul vs. taille du problème.

4.2.2 Test 2

Pour le deuxième cas, le dispositif étudié est un contacteur magnétique qui permettra d'établir ou couper la circulation du courant dans un circuit électrique. Il est constitué d'une partie fixe composée d'une armature métallique, d'un aimant permanent situé au dessus de la colonne centrale et de deux bobines entourant les colonnes latérales, ainsi que d'une partie mobile faite d'une lame ferromagnétique. La figure 4.14 montre les différents parties du dispositif simulé. Nous précisons que nous ne simulons que la moitié du dispositif (par symétrie).

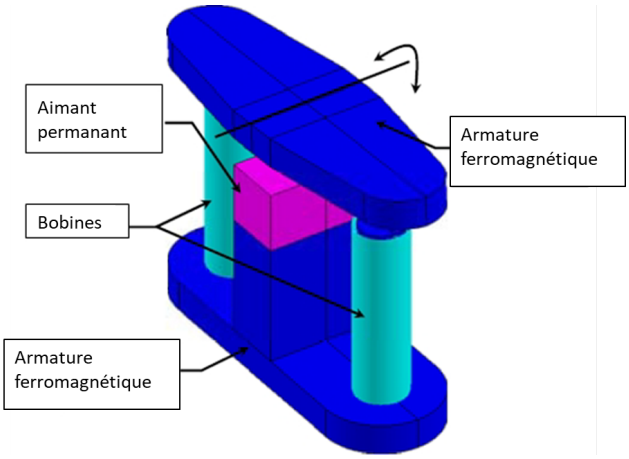

Figure 4.14 – Application magnétostatique contacteur

Les figures 4.15 et 4.16 montrent le projet dans Flux décomposé en 4 sous-domaines. Une région physique représente un sous-domaine.

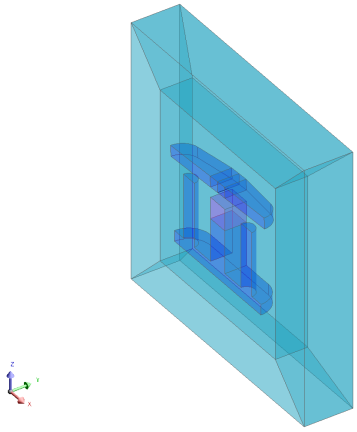

Figure 4.15 – Cas test 2 : géométrie

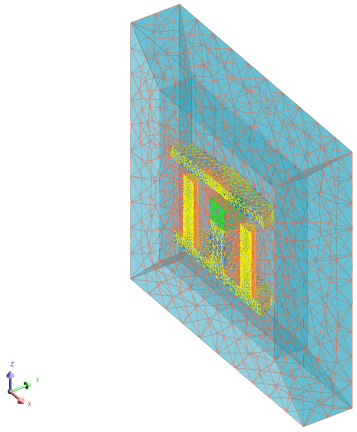

Figure 4.16 – Cas test 2 : maillage

Erreur

Les résultats de la figure 4.17 illustrent l'erreur obtenue avec une résolution en utilisant la DDM. Pour ce cas test ainsi que l'ensemble des cas mentionnés dans ce travail, l'erreur relative obtenue avec la DDM est inférieure à 10^{-7}.

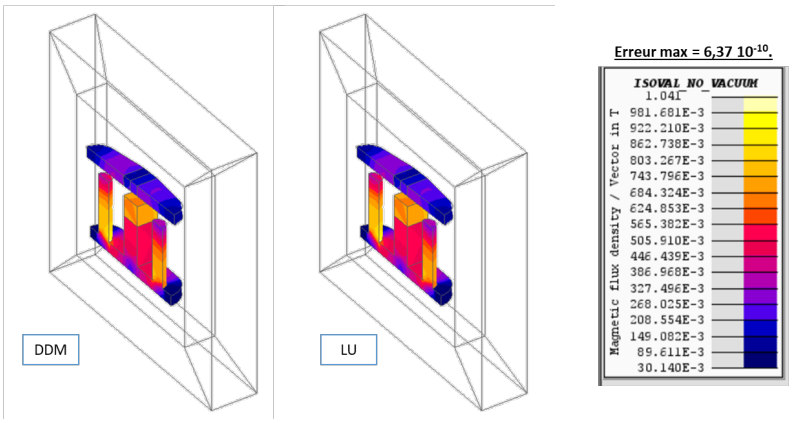

Figure 4.17 – Comparaison des densités de flux obtenues avec la DDM et un solveur direct LU.

Performance en temps

La figure 4.18 montre l'influence de la taille des sous-domaines sur les temps de calcul. La taille totale du problème est de 32 904 nœuds, dont 7 617 sur l'interface. Les tailles des différents sous-domaines sont les suivantes : 18 491 pour l'air, 15 494 pour l'armature ferromagnétique inférieure, 5 240 pour l'armature ferromagnétique supérieure et 1 165 pour l'aimant. Contrairement au cas test 1, la partie résolution du système d'interface prend plus de temps, à cause du déséquilibre entre les tailles des sous-domaines. La DDM est faiblement supérieure aux deux autres méthodes. La convergence est obtenue après 20 itérations du PCPG.

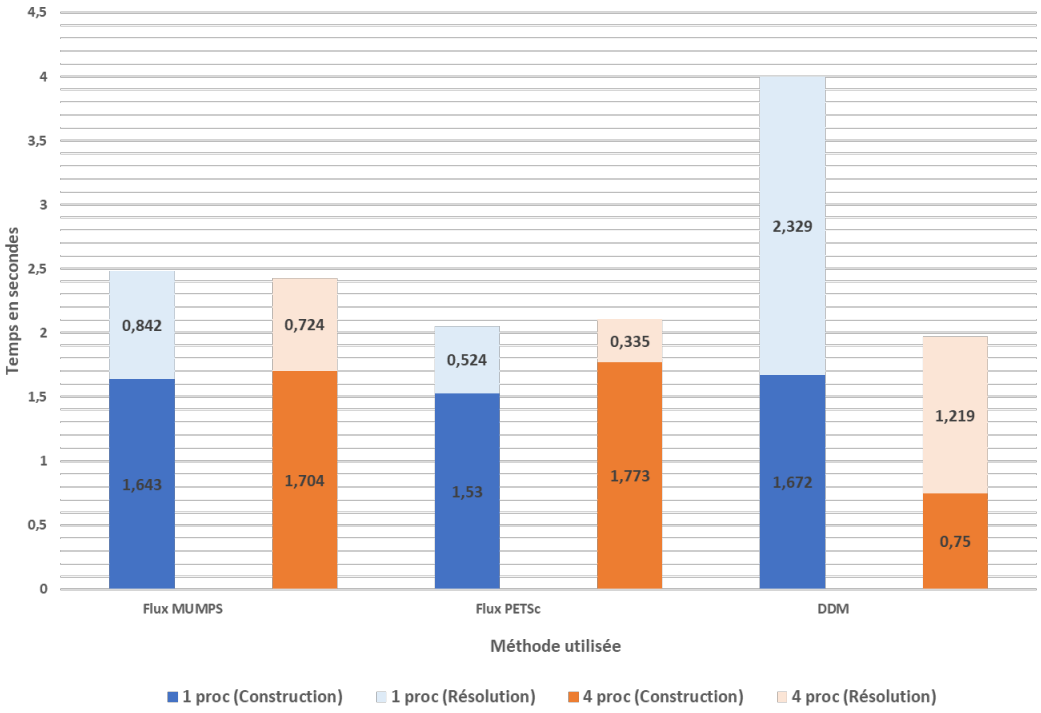

Figure 4.18 – Comparaison des temps de construction et de résolution (cas 2).

Performance en mémoire

En termes de consommation mémoire, nous observons une similarité avec le cas 1, comme illustré par la figure 4.19.

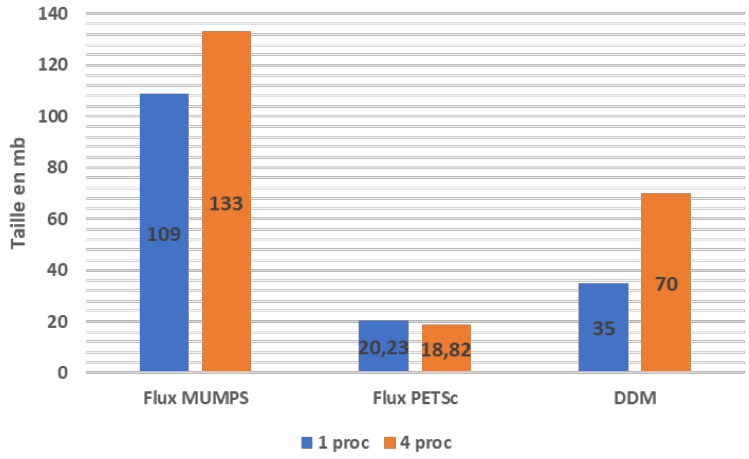

Figure 4.19 – Comparaison de la consommation mémoire entre les différents solveurs (cas 2).

4.2.3 Test 3

Nous utilisons un dernier cas test, composé d'un cube d'aimant entouré d'air (figure 4.20), afin de réaliser une étude de scalabilité en fonction du nombre de partitions, et par conséquent du nombre de cœurs utilisés.

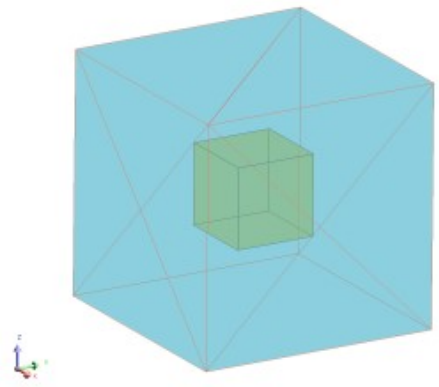

Figure 4.20 – Cas test 3 : géométrie.

Nous partitionnons la région "air" en plusieurs partitions comme montre la figure 4.21.

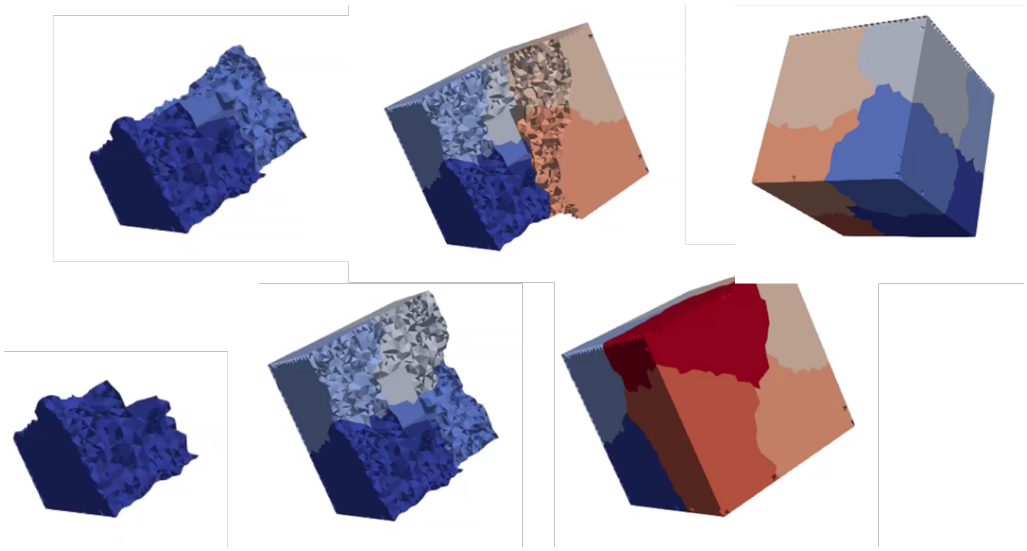

Figure 4.21 – Cas test 3 : partitionnement.

Scalabilité

La figure 4.22 donne les temps de résolution pour le cas test 3. Pour un cas test de 100 000 nœuds, nous pouvons voir dans l'interface graphique de Flux la recommandation d'utiliser 4 cœurs. La DDM en contrepartie arrive à donner une speedup de 2 avec 16 cœurs. Nous considérons ces performances satisfaisantes par rapport à la recommandation que nous avons donnée précédemment au client. La DDM permet d'exploiter davantage de ressources et continue à améliorer le temps de calcul. Bien évidemment, la taille des sous-domaines, d'environ 6 000 inconnues, représente un obstacle à une meilleure scalabilité. Les différents tableaux montrent le nombre de partition et la taille moyenne pour les deux régions. La comparaison est faite avec le meilleurs temps obtenu entre la méthode directe (MUMPS) et la méthode itérative (PETSc).

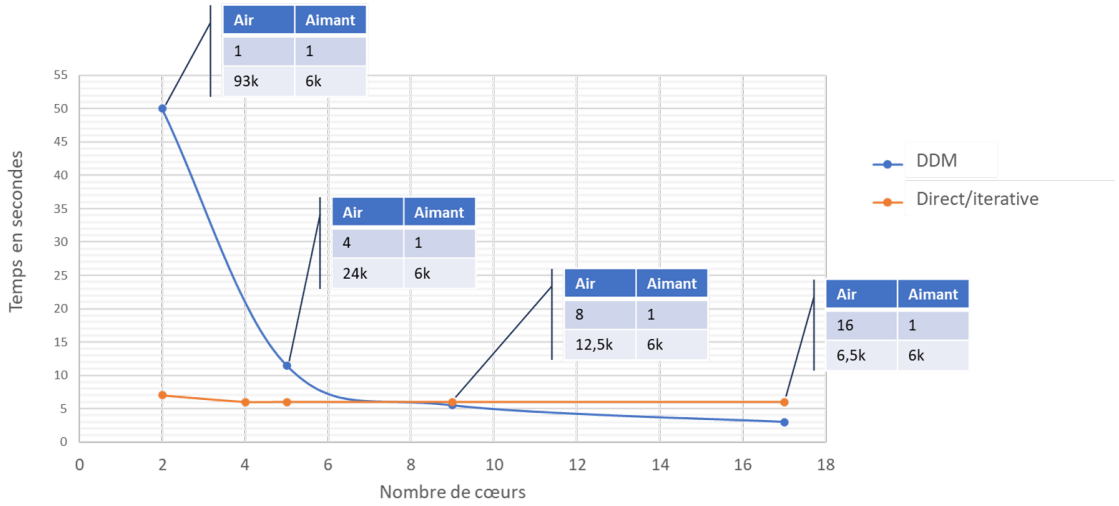

Figure 4.22 – Comparaison des temps de résolution pour différents nombres de partitions (cas 3).

L'utilisation de la DDM en potentiel scalaire avec des cas tests linéaires montre que les performances des deux méthodes utilisées, FETI-1 et FETI-DP, sont très intéressantes si nous respectons certaines conditions sur le parallélisme et l'équilibre de la charge de travail. Nous les avons comparées aux méthodes directes et itératives déjà utilisées dans Flux. Ces méthodes permettent d'avoir un processus de résolution parallèle et performant. Dans la prochaine section, nous explorons l'utilisation de la DDM pour les cas tests non linéaires.

4.3 Cas non linéaire 3D

4.3.1 Newton-Krylov-Schur

Nous allons présenter avec ce cas test une première approche de résolution des problèmes non linéaires en utilisant la DDM. Pour un problème 3D avec une formulation en potentiel scalaire, une source magnétique sous forme d'aimant, la présence de matériaux ferromagnétiques et l'ensemble entouré d'air, les équations sont

$$\operatorname{div}(\mu(\mathbf{h})(-\operatorname{grad}\Phi) + \mathbf{b_r}) = 0\,, \tag{4.25}$$

où φ est le potentiel scalaire magnétique et $\mathbf{b_r}$ est le terme source. Pour des matériaux non linéaires, μ est une fonction du champ magnétique $\mathbf{h} = -\operatorname{grad}(\Phi) + \mathbf{b_r}$, le système à résoudre après discrétisation par la méthode des éléments finis s'écrit

$$K(u)u = b, \tag{4.26}$$

où

- $K(u)$ est la matrice des éléments finis de la formulation en potentiel scalaire magnétique,
- u contient les valeurs du potentiel scalaire magnétique à chaque nœud du maillage,
- b est le vecteur source.

La méthode itérative de Newton-Raphson est généralement utilisée pour résoudre (4.26). Cette méthode consiste à calculer la matrice tangente DK ainsi que le résidu r_u à chaque itération n et à résoudre des systèmes linéaires successifs de la forme

$$DK(u^n)\delta u^n = -r_u(u^n). \tag{4.27}$$

Le résidu est donné par $r_u(u^n) = K(u^n)u^n - b(u^n)$. À chaque itération, un incrément sur la solution est calculé et la solution s'écrit $u^{n+1} = u^n + \delta u^n$. Le processus s'arrête lorsque le critère $\frac{|\delta u^n|}{|u^n|} \leqslant tol$, avec tol la précision de Newton-Raphson voulue.

Les méthodes Newton-Krylov-Schur (NKS) peuvent être mises en œuvre de manière directe. Pendant chaque itération de Newton-Raphson, les méthodes FETI-1 et FETI-DP sont utilisées pour résoudre le système linéaire tangent (4.27). Le terme (complément de) Schur dans le cadre de ces méthodes NKS caractérisent le problème d'interface. C'est important de noter que les méthodes NKS ne nécessitent pas l'assemblage du problème global, et le calcul du résidu est partitionné entre les sous-domaines [25, 27].

En adoptant cette approche, les méthodes NKS utilisent efficacement la méthode FETI pour résoudre le système linéaire tangent à chaque itération de Newton-Raphson. Cela contourne la nécessité d'assembler le problème global, entraînant une amélioration de

l'efficacité du calcul. De plus, en partitionnant le calcul du résidu entre les sous-domaines, le traitement parallèle peut être exploité pour améliorer les performances et réduire le temps de calcul. L'algorithme de la méthode NKS que nous avons utilisée est résumé dans l'algorithme 4.

Algorithm 4 Algorithme NKS avec FETI

Require : $p = 1; u^1 = 0$ ▷ Résoudre les problèmes NL globaux
  while $\delta u \leqslant tol_{NR}$ **do**
    Résoudre $DK(u^p)\delta^p u = -r_u^p(u^p)$ avec PCPG
    $u^{p+1} \leftarrow u^p + \delta^p u$
    $p \leftarrow p + 1$

4.3.2 Schur-Newton-Krylov

Les méthodes Schur-Newton-Krylov (SNK) ont été développées pour répondre à une observation cruciale : les méthodes existantes NKS n'exploitent pas pleinement le potentiel de la décomposition de domaine lors de la résolution de problèmes non linéaires. Alors que les méthodes NKS utilisent principalement la décomposition de domaine pour résoudre des systèmes linéaires tangents, les méthodes SNK vont plus loin. Dans les méthodes SNK, le problème non linéaire global est décomposé en problèmes non linéaires plus petits et localisés, accompagnés de l'imposition de conditions de continuité sur les inconnues d'interface à travers les sous-domaines [27].

Nous reprenons l'équation d'équilibre sur le sous-domaine s (1.12) :

$$K_s u_s + B_s^t \lambda = f_s \,, \tag{4.28}$$

avec la condition de continuité sur l'interface :

$$B_s u_s = 0 \,, \tag{4.29}$$

où $B_s = \sum_{i=1}^{i=a_s} B_{s_i}$ et a_s le nombre de sous-domaines voisins de Ω_s. Avec des matériaux non-linéaire [75], les équations (4.28) et (4.29) s'écrivent sous la forme :

$$\begin{aligned} K_s(u_s) - B_s^t \lambda &= f_s, \\ B_s u_s &= 0. \end{aligned} \tag{4.30}$$

La linearisation du problème donne [75] :

$$\begin{aligned} DK_s(u_{s,(k)})\delta u_{s,(k)} &= K(u_{s,(k)}) - f_s + B_s^t \lambda \\ B_s \delta u_{s,(k)} &= B_s u_{s,(k)} \end{aligned} \tag{4.31}$$

Après partitionnement de K_s, u_s et f_s en variables internes, interfaces et cross-points (1.30) et pour $r^{(k)} = K_s(u^{s,(k)}) - f_s$, $\lambda^{k+1} = \lambda^k - \delta\lambda^k$ et $u^{k+1} = u^k - \delta u^k$, l'équation d'équilibre et la condition de continuité à l'itération k deviennent (l'exposant $^{(k)}$ est omis par simplicité) :

$$\begin{aligned} DK_{s,rr}\delta u_{s,r} + DK_{s,rc}B_{s,c}\delta u_c &= r_r - B^t_{s,r}\delta\lambda + B^t_{s,r}\lambda \\ \sum_{s=1}^{s=N_s} B^t_{s,c}DK^t_{s,rc}\delta u_{s,r} + \sum_{s=1}^{s=N_s} B^t_{s,c}DK_{s,cc}B_{s,c}\delta u_{s,c} &= r_c \\ \sum_{s=1}^{s=N_s} B_{s,r}\delta u_{s,r} &= B_s u_s \end{aligned} \quad , \tag{4.32}$$

avec

$$u_c = \begin{bmatrix} u_{1,c} \\ \vdots \\ u_{j,c} \\ \vdots \\ u_{N_c,c} \end{bmatrix}. \tag{4.33}$$

En utilisant l'équation (4.32), nous pouvons écrire le problème dual-primal. À partir de la première ligne de (4.32) :

$$\delta u_{s,r} = DK^{-1}_{s,rr}(r_r - B^t_{s,r}\delta\lambda + B^t_{s,r}\lambda - DK_{s,rc}B_{s,c}\delta u_{s,c})\,. \tag{4.34}$$

En remplaçant (4.34) dans la deuxième ligne de (4.32) nous obtenons :

$$-F^t_{rc}\delta\lambda + DK_{cc}\delta u_c = f_c\,, \tag{4.35}$$

et (4.34) dans la troisième ligne de (4.32) :

$$F_{rr}\delta\lambda + F_{rc}\delta u_c = d_r - B_s u_s, \tag{4.36}$$

avec,

$$
\begin{aligned}
F_{rr} &= \sum_{s=1}^{s=N_{\mathrm{sd}}} B_{s,r} DK_{s,rr}^{-1} B_{s,r}^{t} \,, \\
F_{_{rc}} &= \sum_{s=1}^{s=N_{\mathrm{sd}}} B_{s,r} DK_{s,rr}^{-1} DK_{s,rc} B_{s,c} \,, \\
DK_{cc} &= \sum_{s=1}^{s=N_{\mathrm{sd}}} B_{s,c}^{t} DK_{s,cc} B_{s,c} - \sum_{s=1}^{s=N_{\mathrm{sd}}} \left(DK_{s,rc} B_{s,c}\right)^{\mathrm{T}} DK_{s,rr}^{-1} \left(DK_{s,rc} B_{s,c}\right) \,, \\
d_r &= \sum_{s=1}^{s=N_{\mathrm{sd}}} B_{s,r} DK_{s,rr}^{-1} (r_r + B_{s,r}^{t} \lambda) \\
f_c &= r_c - \sum_{s=1}^{s=N_x} B_{s,c}^{t} DK_{s,rc}^{t} DK_{s,rr}^{-1} (r_{s,r} + B_{s,r}^{t} \lambda) \,.
\end{aligned}
\tag{4.37}
$$

Ce processus de décomposition donne le système condensé linéarisé suivant :

$$
\begin{pmatrix} F_{rr} & F_{_{rc}} \\ -F_{_{rc}}^{T} & DK_{cc} \end{pmatrix} \begin{pmatrix} \delta\lambda \\ \delta u_c \end{pmatrix} = \begin{pmatrix} d_r - B_s u_s \\ f_c \end{pmatrix} . \tag{4.38}
$$

et dans le cas de FETI-1 :

$$
\begin{bmatrix} DF & E^{B} \\ (E^{B})^{t} & 0 \end{bmatrix} \begin{bmatrix} \lambda \\ \alpha \end{bmatrix} = \begin{bmatrix} -r \\ 0 \end{bmatrix} \tag{4.39}
$$

Avec $DF = \sum_{j=1}^{N_{\mathrm{sd}}} B_j DK_j^{+} B_j^{t}$.

Le problème (4.39), tout comme (4.38), est résolu en utilisant le PCPG comme dans l'algorithme 1. La méthode SNK avec FETI-1 est résumée dans l'algorithme 5.

Algorithm 5 Algorithme SNK avec FETI-1

Require : $n = 0, \lambda^0 = E^B((E^B)^t E^B)^{-1} E^t f$
  $P = I - E^B((E^B)^t E^B)^{-1}(E^B)^t$
  $K_s(u_s^n) = f_s(u_s^n) - B_s^t \lambda_s^n$
  while $\delta u \leqslant tol_{NR}$ **do**
    Solve $DK_s(u_s^{n^p})\delta^p u = f_s^p(u_s^{n^p}) - B_s^t \lambda_s^n - K_s(u_s^{n^p})$
    $u_s^{n^{p+1}} \leftarrow u_s^{n^p} + \delta^p u$
    $p \leftarrow p + 1$
  $r_\lambda^n \leftarrow \sum_{s=1}^{N_{sd}} B_s u_s^n$ ▷ Calcul du résidu
  while $\|\delta\lambda^n\| / \|\lambda^n\| \leqslant tol_g$ **do**
    Résoudre le probleme d'interface avec PCPG
    $\lambda^{n+1} \leftarrow \lambda^n + P\delta\lambda^n$
    $\alpha^{n+1} \leftarrow \alpha^n + ((E^B)^t E^B)^{-1}(E^B)^t(-DF(\lambda^n)\delta\lambda^n - r_\lambda^n)$
    $u_s^{n+1} \leftarrow u_s^n - DK_s^{-1} B_s^t P\delta\lambda^n - (E^B)^t \delta\alpha_s^n$
    $n \leftarrow n + 1$
    $K_s(u_s^n) = f_s(u_s^n) - B_s^t \lambda_s^n$
    $p = 1; u_s^{n^1} = u_s^n$
    while $\delta u \leqslant tol_{NR}$ **do**
      Résoudre $DK_s(u_s^{n^p})\delta^p u = f_s^p(u_s^{n^p}) - B_s^t \lambda_s^n - K_s(u_s^{n^p})$
      $u_s^{n^{p+1}} \leftarrow u_s^{n^p} + \delta^p u$
      $p \leftarrow p + 1$
    $r_\lambda^n \leftarrow \sum_{s=1}^{N_{sd}} B_s u_s^n$ ▷ Calcul du résidu

4.3.3 Test 4

Pour tester ces algorithmes, nous reprenons le cas test 2, mais cette fois avec des matériaux non linéaires dans les régions de l'armature ferromagnétique supérieure et inférieure. La figure 4.23 montre les résultats obtenus après résolution.

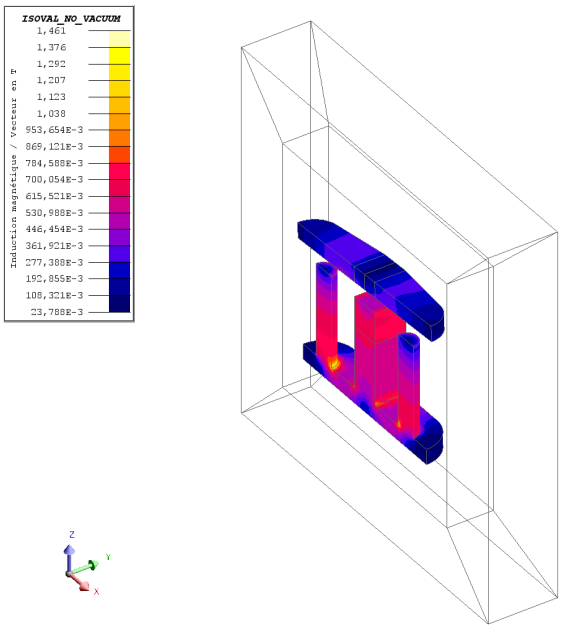

Figure 4.23 – Cas test 4 : resultats

Nous utilisons un modèle de loi de comportement $B(H)$ pour des matériaux doux avec une courbe de saturation analytique et ajustement du coude. Cela se traduit par une loi de la forme

$$\begin{aligned} \mathrm{B(H)} =& \mu_0 \mathrm{H} + \mathrm{J_s} \frac{\mathrm{H_a} + 1 - \sqrt{(\mathrm{H_a}+1)^2 - 4\mathrm{H_a}(1-\mathrm{a})}}{2(1-\mathrm{a})} \\ \mathrm{H_a} =& \mu_0 \mathrm{H} \frac{\mu_\mathrm{r} - 1}{\mathrm{J}_s}, \end{aligned} \tag{4.40}$$

Avec a cœfficient de courbure qui vaut 0.075 et $J_s = 2T$. La figure 4.24 montre la courbe $B(H)$.

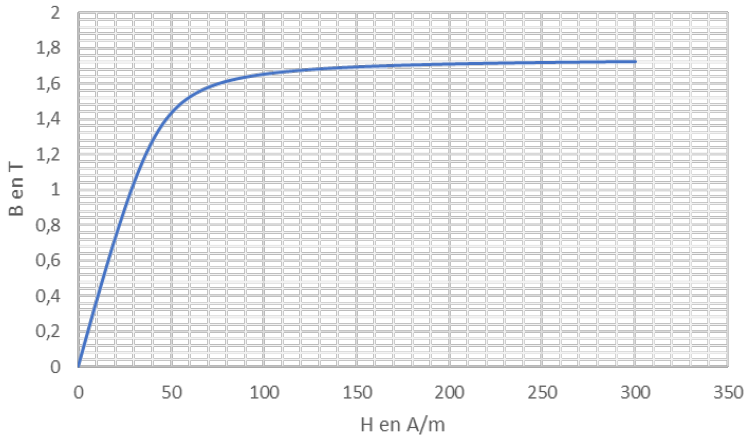

Figure 4.24 – Courbe B(H) (cas 4).

Résultats

Les algorithmes NKS et SNK ont tous deux été implémentés dans le logiciel Altair Flux^{TM} [44, 76]. Ils ont été testés et comparés à la méthode standard de Newton-Raphson généralement utilisée pour ce type de simulations. Le solveur linéaire utilisé par défaut dans Newton Raphson est un solveur direct (MUMPS). Le cas de test est illustré à la figure 4.23 et représente un contacteur magnétique 3D. Il a été décomposé en 4 sous-domaines : air (linéaire), palette supérieure (non linéaire), aimant (linéaire) et forme en E (non linéaire). La taille du cas de test est de 2 432 nœuds.

Table 4.1 – Comparaison des résultats pour le cas de test du contacteur

Méthode	NR	NKS	SNK
Nombre max d'itérations NL locales	/	/	5
Itérations NL globales	6	6	7
Speedup	1	1.3	0.4

Le tableau 4.1 résume les résultats. Un critère de convergence adaptatif a été utilisé pour obtenir moins d'itérations PCPG avec un préconditionneur de Dirichlet [7]. Le problème a été résolu en utilisant 4 cœurs. Pour la décomposition de domaine, cela signifie que nous résolvons un sous-domaine par processeur, ce qui est la situation optimale lorsque l'équilibrage de charge est bon. Le nombre maximal d'itérations non linéaires locales est la limite supérieure de la résolution non linéaire locale des deux sous-domaines non linéaires. L'accélération est le quotient du temps de résolution global de la méthode standard de Newton-Raphson de référence par celui des nouvelles méthodes de décomposition de domaine. Cela inclut les temps d'assemblage et de résolution. La tolérance sur l'erreur

relative est de 10^{-5}.

Performances

Le cas test révèle que la méthode SNK présente un grand nombre d'itérations non linéaires locales, entraînant un retard significatif de la convergence dans la boucle globale. Par conséquent, les performances de cette méthode sont affectées de manière défavorable, ce qui se traduit par un ralentissement notable. En revanche, la méthode NKS présente des performances favorables par rapport à la méthode standard de Newton-Raphson.

De plus, l'utilisation de la méthode de décomposition de domaine comme solveur linéaire donne des résultats prometteurs, avec une accélération de 30% observée. Ce résultat est conforme aux découvertes précédentes concernant le cas linéaire, ce qui valide davantage l'efficacité de l'approche de la décomposition de domaine pour ces problèmes en potentiel scalaire. Il convient de noter que la charge de travail entre les sous-domaines dans ce cas de test non linéaire spécifique n'était pas bien équilibrée, car le sous-domaine "air" concentre une grande partie du maillage global. La partition des grandes régions physiques améliorera certainement l'équilibrage pour obtenir des performances comparables à celles du cas linéaire.

Pour la méthode SNK, les résultats restent décevants pour notre cas. Cependant, l'idée de localiser les non-linéarités reste intéressante, et davantage d'investigations devraient être entreprises pour trouver des moyens d'améliorer les performances. Nous pouvons envisager la relaxation, par exemple, ou modifier le partitionnement, voir l'utilisation des formulations en potentiel vecteur.

4.4 Conclusion

Dans ce chapitre, nous avons présenté les résultats de l'utilisation de la méthode de décomposition de domaine avec la formulation en potentiel scalaire en 3D. Nous avons commencé par un premier cas test académique, au cours duquel nous avons pu constater les avantages de l'utilisation de cette méthode. Ensuite, nous avons étudié les différents facteurs influençant ses performances, notamment le préconditionnement et la répartition de la charge de travail, puis appliqué la DDM sur un cas test industriel.

Les deux méthodes FETI-1 et FETI-DP ont montré des performances équivalentes. En maintenant la taille des sous-domaines autour de 30 000 nœuds et équilibrés, la méthode de décomposition de domaine reste plus rapide que les méthodes directes et itératives. En ce qui concerne la consommation mémoire, elle se situe entre les deux.

Pour les cas non linéaires, nous pouvons utiliser la DDM comme solveur linéaire après linearisation, ce qui permet d'étendre les performances de la DDM aux résolutions non

linéaires. Dans le prochain chapitre, nous aborderons la DDM pour les formulation B-conforme que nous avons explorées.

Chapitre 5

Résultats de la DDM pour la formulation B-conforme

5.1 Modélisation avec des éléments finis d'arête

Dans ce chapitre nous allons aborder l'adaptation de la méthode de décomposition de domaine pour les formulations B-conforme. Pour cela, nous utiliserons les éléments finis d'arête pour discrétiser le problème en magnétostatique. Ce type d'éléments finis est adapté pour approcher les champs vectoriels qui sont dans $\mathbf{H}(\mathrm{rot}, \Omega)$ (voir sous-section 2.2.3).

Comme déjà indiqué dans le chapitre 2, la première difficulté sera de traiter le problème de jauge en décomposition de domaine. Nous allons, à travers ce chapitre, proposer un début de solution à cette question. Nous avons mis en œuvre la solution proposée, mais une implémentation complète et optimale spécifique aux éléments finis d'arête avec la DDM reste à réaliser et pourrait constituer le sujet de futurs travaux.

Tout d'abord, nous allons revenir sur l'utilisation des deux conditions de jauge présentées en sous-section 2.1.4 :

1. Jauge de Coulomb : Cette condition est généralement utilisée pour une discrétisation avec des éléments finis nodaux [38]. Les auteurs dans [77] proposent une adaptation pour les éléments finis d'arête mais cette jauge n'est pas disponible dans Flux.
2. Jauge par arbre d'arêtes : La condition de jauge par arbre est adaptée aux problèmes décrits avec des éléments finis d'arête.

Dans certain cas, nous pouvons nous passer de la condition de jauge. La résolution d'un système matriciel obtenue avec des formulations en potentiel vecteur peut se faire en utilisant un solveur itératif (sous-section 3.3) si les systèmes non-jaugés sont compatibles, c'est-à-dire lorsque le terme source du système (ou le second membre) est dans l'image de l'application linéaire associée à la matrice de discrétisation [78].

Nous allons commencer par expliquer la difficulté d'adapter la méthode de jauge par arbre pour la DDM avant de proposer une méthode de résolution.

5.1.1 Jauge par arbre

Dans la section 2.1.4, nous avons mentionné que, pour les problèmes traités avec les formulations en potentiel vecteur, nous pouvions utiliser une condition de jauge de la forme :

$$\mathbf{a} \cdot \mathbf{w} = 0\,. \tag{5.1}$$

Au niveau discret, nous utilisons une technique basée sur des concepts de la théorie des graphes pour sa mise en place, en raison de l'utilisation des éléments finis d'arête. Contrairement aux éléments finis nodaux (éléments de Lagrange), où la fonction de base est définie sur les nœuds, pour les éléments d'arête (éléments de Nédélec), la fonction de base est définie sur les arêtes [79]. Ainsi, les arêtes et les nœuds du maillage forment un graphe.

Considérons le graphe basé sur les nœuds et arêtes du maillage d'éléments finis qui discrétise le domaine d'intégration (figure 5.1) [80].

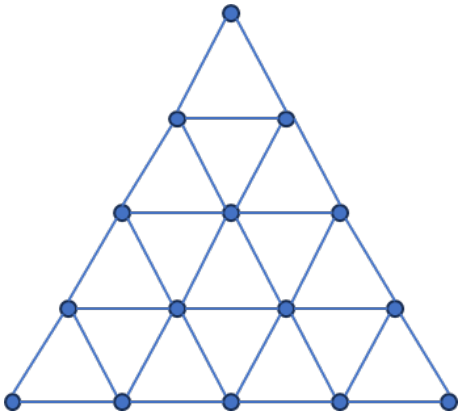

Figure 5.1 – Exemple Maillage/graphe.

Par définition, un arbre couvrant d'un graphe est un graphe partiel qui relie tous les nœuds du graphe sans former de cycle (ou boucle). La figure 5.2 montre un exemple d'arbre couvrant pour le graphe de la figure 5.1. Le co-arbre est son complémentaire dans le graphe initial.

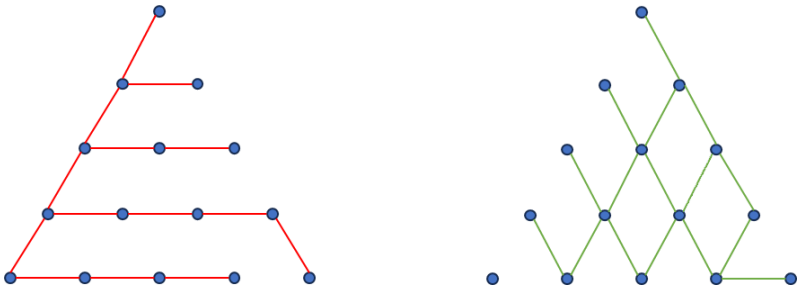

Figure 5.2 – Arbre à gauche (en rouge) et co-arbre à droite (en vert).

La condition de jauge (5.1) est obtenue en choisissant la direction de $\mathbf{w}$ [1] et à annulant la composante du potentiel vecteur selon cette direction. La jauge peut être représenté au niveau discret par un arbre d'arêtes tel que $\mathbf{w}$ correspond aux arêtes du maillage. Imposer la jauge revient alors à annuler les degrés de liberté correspondant aux arêtes de l'arbre [40] [2]. La construction de l'arbre peut être faite à travers l'attribution de poids aux arêtes du graphe. Les expériences numériques indiquent que la précision de la résolution et la vitesse de convergence du système dépendent du choix de l'arbre qui lui même dépend du choix des poids [38].

En raison des conditions limites, l'arbre est recherché à partir de la frontière du domaine, ce qui soulève une problématique pour la DDM. La décomposition globale du domaine implique que l'arbre obtenu sur les différents sous-domaines ne construit pas forcément le même arbre que sur le problème global. Nous pourrions envisager de modifier les poids sur les arêtes pour conserver le même arbre, mais nous n'avons pas retenu cette option par souci de simplicité, bien qu'elle mérite sans doute davantage d'investigations. Nous expliquerons par la suite l'approche adoptée dans ce travail.

5.1.2 Auto jauge

L'auto jauge est une première méthode pour résoudre le problème magnétostatique en potentiel vecteur sans utiliser une jauge par arbre. Cette méthode a été développée pour faire face à la difficulté de trouver un arbre couvrant approprié pour des maillages complexes (la construction de l'arbre peut être très lente [81]). La recherche de l'arbre dans ce cas peut s'avérer peu efficace.

L'auto jauge profite du fait qu'une grande partie des solveurs directs récents intègre des fonctions pour détecter les pivots nuls pour des matrices singulières [82]. C'est le cas pour la bibliothèque MUMPS que nous avons utilisés dans ce travail.

1. Rappel : $\mathbf{w}$ un champ de vecteurs arbitraire qui ne forme pas de boucle.
2. Dans ce travail, les domaines considérés sont simplement connexes. Dans le cas contraire, la mise en place de la jauge reste à préciser.

Comme expliqué dans la section 3.3, la résolution d'un système linéaire avec un solveur direct passe par plusieurs étapes : l'analyse, la factorisation et la solution. Dans la deuxième étape, la détection des pivots nuls et du rang incomplet est faite, ainsi que la recherche d'une base du noyau de la matrice [3]. De cette manière, le solveur peut traiter les équations de matrices symétriques singulières. Après que le noyau a été détecté, le solveur peut fournir une solution possible du système. Dans le processus de détection des pivots nuls, un pivot nul est identifié s'il est plus petit qu'une valeur de fixation donnée [4]. Pour garantir que la phase de solution/substitution fonctionne, les pivots nuls détectés seront remplacés par cette valeur de fixation. Il existe différentes stratégies de pivotage dans différents solveurs avec des décisions arbitraires similaires qui peuvent fournir une solution parmi les solutions infinies du système. Cette technique peut être présentée comme une "régularisation algébrique" de la matrice du système, notion reprise dans la section suivante.

Dans MUMPS, il existe une option pour activer la détection des pivots nuls [5] ainsi qu'un paramètre [6] pour régler le critère de détection. Lorsque cette option est activée, tous les éléments seront remplacés par zéro, à l'exception du coefficient diagonal, qui sera remplacé par la valeur spécifiée par l'utilisateur [7].

Pour notre cas, le choix de ce paramètre était difficile. Nous avons essayé plusieurs valeurs et nous n'avons pas réussi à obtenir l'unicité du potentiel vecteur entre une résolution sur le domaine global et une résolution par DDM avec cette méthode dans chaque sous-domaine.

5.1.3 Régularisation

La méthode de régularisation est celle que nous avons choisie pour ce travail, car elle offre une mise en œuvre plus simple et permet d'analyser la DDM ainsi que de comparer avec le résolution du problème global.

À partir de l'équation (2.41) (et en utilisant les rappels d'analyse numérique, voir [70]), le problème variationnel avec formulation en potentiel vecteur revient à trouver le point

3. C'est regrettable pour nous, car dans notre cas, nous disposons déjà de l'information sur le noyau, au moins dans le cas d'un domaine simplement connexe, car elle est donnée par la matrice d'incidence arêtes-nœuds du maillage.

4. La valeur de fixation est déterminée à partir du paramètre CNTL(3). Ainsi, pour :

$$\begin{aligned}
&\text{CNTL(3)} > 0.0 : \text{valeur de fixation} = \text{CNTL(3)} \times \|A_{\text{pre}}\| \\
&\text{CNTL(3)} = 0.0 : \text{valeur de fixation} = \varepsilon \times \|A_{\text{pre}}\| \times \sqrt{N_h} \\
&\text{CNTL(3)} < 0.0 : \text{valeur de fixation} = |\text{CNTL(3)}|
\end{aligned}$$

où A_{pre} est la matrice prétraitée à factoriser, N_h est le nombre de variables sur la branche la plus profonde de l'arbre d'élimination, ε est la précision machine et $\|\cdot\|$ est la norme infinie [51].

5. Le paramètre ICNTL(24).

6. Le paramètre CNTL(3).

7. Le paramètre CNTL(5).

stationnaire de la fonctionnelle d'énergie :

$$J(\mathbf{v}) = \frac{1}{2}a(\mathbf{v},\mathbf{v}) - (\mathbf{v},\mathbf{j}) - (\mathbf{v},\mathbf{b_r})\,, \tag{5.2}$$

avec

$$a(\mathbf{v},\mathbf{a}) = \int_{\Omega_i} \mu^{-1}\,\mathrm{rot}\,\mathbf{v}\cdot\mathrm{rot}\,\mathbf{a}\ \ \mathrm{d}\Omega_i\,, \tag{5.3}$$

$$(\mathbf{v},\mathbf{j}) = \int_{\Omega_i} \mathbf{v}\cdot\mathbf{j}\ \ \mathrm{d}\Omega_i\,, \tag{5.4}$$

$$(\mathbf{v},\mathbf{b_r}) = \int_{\Omega_i} \mu^{-1}\,\mathrm{rot}\,\mathbf{v}\cdot\mathbf{b_r}\ \ \mathrm{d}\Omega_i\,. \tag{5.5}$$

La méthode consiste à remplacer la forme bilinéaire dans (5.2) par :

$$a(\mathbf{v},\mathbf{a}) = \int_{\Omega_i} \mu^{-1}\,\mathrm{rot}\,\mathbf{v}\cdot\mathrm{rot}\,\mathbf{a} + \sigma\mathbf{v}\cdot\mathbf{a}\ \ \mathrm{d}\Omega_i\,, \tag{5.6}$$

avec σ un paramètre de régularisation strictement positif. Selon les auteurs de [83] et [84], la méthode de régularisation produit de bons résultats lorsque la valeur de σ est "petite" mais l'analyse fournie est plus précise. Nous pouvons voir notamment dans [84] des résultats avec des valeurs de σ qui sont liées à la taille du maillage. Pour un maillage de taille $h = 1/16$, les auteurs ont testé les valeurs de sigma suivantes : $\sigma = [1/10, 1/20, 1/30, 1/40, 1/50]$. Plus précisément, le résultat suivant est démontré dans [84] concernant l'erreur sur la solution u_σ obtenue avec la régularisation. Pour un pas de maillage régulier h, cette erreur est donnée par :

$$\|u - u_\sigma\| = C(\sigma + h)\|f\|\,. \tag{5.7}$$

En prenant $\sigma \leqslant h$, on constate que l'erreur de régularisation n'est plus prépondérante. Ceci est confirmé par les résultats des cas montrés dans [84].

Cette régularisation peut être implémenter en modifiant la formulation variationnelle dans le code.

Si l'accès à l'implémentation des formulations s'avère difficile, une méthode de régularisation plus simple peut être envisagée en ajoutant un terme à la matrice par sous-domaine comme suit :

$$K_s = K_s + \sigma Id\,.$$

Cette régularisation algébrique a des similitudes avec ce que nous avons vu dans la méthode par auto jauge.

5.2 FETI-DP pour les éléments finis d'arêtes

Dans le cas scalaire, la méthode FETI-1 est quasi-optimale, principalement grâce à la gestion des sous-domaines flottants, qui agit comme un problème grossier. Par conséquent, les performances de la méthode ne dépendent plus du nombre ni de la taille des sous-domaines. Cependant, en potentiel vecteur, cette propriété est perdue, et pour obtenir une méthode quasi-optimale, il est nécessaire d'introduire un problème grossier. C'est précisément ce que la méthode FETI-DP permet [16, 85]. La difficulté avec les éléments finis d'arête, où les degrés de liberté sont associées aux arêtes, est le choix de l'espace grossier qui permet de garder le conditionnement indépendant du nombre de sous-domaines.

Dans la littérature, on peut distinguer deux approches. La mise en œuvre de l'une ou l'autre aboutit à un espace grossier similaire. Dans [29], l'auteur préconise l'utilisation d'un changement de base et d'introduire cette base sur les arêtes grossières (une arête grossière est composé d'un ensemble d'arêtes du maillage initial). Le degré de liberté de l'arête grossière peut être prise comme la moyenne des valeurs sur les arêtes fines la constituant.

Dans [1], les auteurs proposent une autre approche qui ne nécessite pas de passer par un changement de base et que nous jugeons plus intuitive. Elle consiste à introduire des multiplicateurs de Lagrange pour vérifier la contrainte sur les variables primales :

$$\sum_{s=1}^{s=N_{\text{sd}}} Q_\Delta B_{s,\Delta} u_{s,\Delta} = 0\,, \tag{5.8}$$

avec Q_Δ une matrice où chaque ligne représente une nouvelle contrainte optionnelle. Les coefficients de la ligne représente le type de contrainte imposée, des 1 pour une moyenne par exemple. $B_{s,\Delta}$ est la matrice booléenne pour la restriction sur les variables d'interface $u_{s,\Delta}$.

Nous posons alors :

$$\widetilde{u}_c = \begin{bmatrix} u_c \\ u_{op} \end{bmatrix}, \tag{5.9}$$

u_c sont les cross-points et u_{op} les nouvelles contraintes rajoutées.

Le problème (1.34), de la section 1.5, s'écrira alors sous la forme :

$$\begin{bmatrix} F_{rr} & \widetilde{F}_{rc} \\ \widetilde{F}_{rc}^t & -\widetilde{K}_{cc} \end{bmatrix} \begin{bmatrix} \lambda \\ \widetilde{u}_c \end{bmatrix} = \begin{bmatrix} d_r \\ \widetilde{f}_c \end{bmatrix}, \tag{5.10}$$

avec

$$\widetilde{K}_{s,rc} = \left[K_{s,rc} B_{s,c} \quad B_{s,r} Q_\Delta \right], \tag{5.11}$$

$$\widetilde{F}_{rc} = \sum_{s=1}^{s=N_{\text{sd}}} B_{s,r} K_{s,rr}^{-1} \widetilde{K}_{s,rc} B_{s,c}\,, \tag{5.12}$$

$$\widetilde{K}_{cc} = \sum_{s=1}^{s=N_{\text{sd}}} B_{s,c}^t K_{s,cc} B_{s,c} - \sum_{s=1}^{s=N_{\text{sd}}} \left(\widetilde{K}_{s,rc} B_{s,c} \right)^{\text{T}} K_{s,rr}^{-1} \left(\widetilde{K}_{s,rc} B_{s,c} \right)\,, \tag{5.13}$$

$$\widetilde{f}_c = \sum_{s=1}^{s=N_{\text{sd}}} B_{s,c}^t f_{b_c}^s - \sum_{s=1}^{s=N_x} B_{s,c}^t \widetilde{K}_{s,rc}^t K_{s,rr}^{-1} f_{s,r}\,. \tag{5.14}$$

Il suffit de résoudre, avec le gradient conjugué, le problème (5.15) :

$$(F_{rr} + \widetilde{F}_{rc} \widetilde{K}_{cc}^{-1} \widetilde{F}_{rc}^t)\lambda = \sum_{s=1}^{s=N_{\text{sd}}} B_{s,r} K_{s,rr}^{-1} f_{s,r} - \widetilde{F}_{rc} \widetilde{K}_{cc}^{-1} \widetilde{f}_c\,. \tag{5.15}$$

Après calcul de u_c, nous pouvons calculer :

$$u_{s,r} = K_{s,rr}^{-1} (f_{s,r} - B_{s,r}^t \lambda - K_{s,rc} B_{s,c} u_c - B_{s,r}^t Q_\Delta u_{op})\,. \tag{5.16}$$

5.2.1 Résultats

Pour étudier la méthode FETI-DP en potentiel vecteur, nous avons choisit de travailler avec le cas test illustré à la figure 5.3.

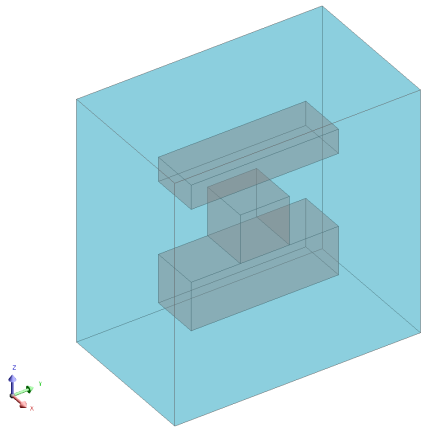

Figure 5.3 – Cas test 5 : géométrie

Ce cas test est une version simplifiée du cas test 2 (voir la figure 4.15), avec les deux régions ferromagnétiques supérieure et inférieure, le cube d'aimant ainsi que la boite d'air. Nous

allons l'utiliser pour analyser la DDM avec FETI-DP.

Nous allons étendre la définition donnée dans la figure 1.7, chapitre 1, section 1.5, pour les éléments finis d'arête. Nous allons prendre comme point le milieu d'arête. Un crosspoint sera donc une arête partagé par plus que deux sous-domaines.

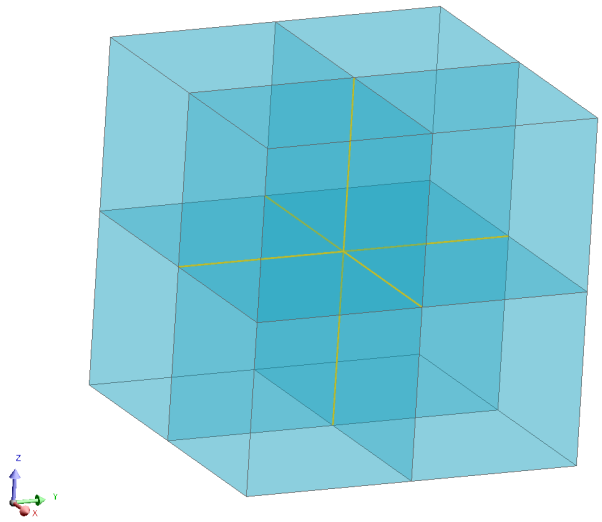

Figure 5.4 – Définition de point coin pour les éléments finis d'arête.

Dans la figure 5.4, qui illustre un domaine cubique divisé en 8 sous-domaines, les arêtes en jaune partagées par plus de deux sous-domaines sont considérées comme des points de croisement.

Les tests effectués consistent à essayer différentes valeurs de σ, dans l'équation (5.6). Pour chaque cas, le conditionnement effectif de la matrice du problème globale est noté.

Les valeurs de σ varient entre 10^{-6} et 10^{6}. Nous commençons par la valeur $\sigma = 10^{-6}$ et nous multiplions par 10 cette valeur pour chaque nouveau test. La figure 5.5 donne les résultats pour le conditionnement en fonction des valeurs de σ testées. Le rang "numérique" de la matrice complète avec 641 variables vaut 533 pour $10^{-6} \leqslant \sigma \leqslant 10^{-1}$ et 641 pour $1 \leqslant \sigma \leqslant 10^{6}$.

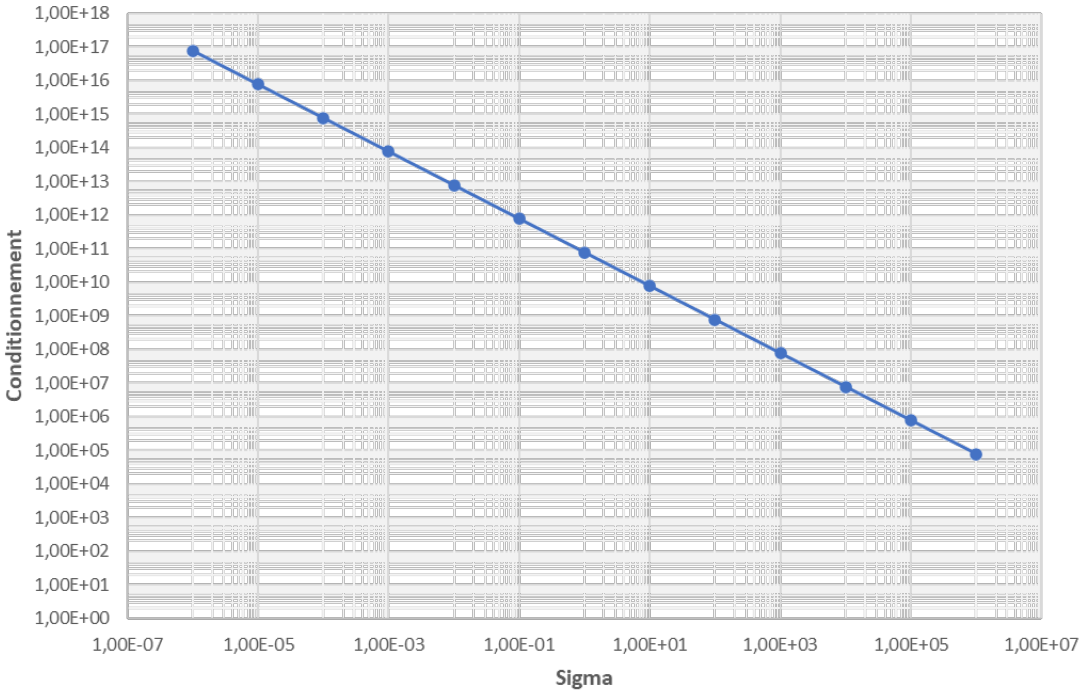

Figure 5.5 – Évolution du conditionnement de la matrice en fonction de la valeur de σ.

Les résultats montrent que plus la valeur de σ est grande, plus le conditionnement est faible. Ceci s'explique par le fait que la présence de la matrice de masse entraîne une augmentation de la plus petite valeur propre, ce qui améliore le conditionnement effectif, qui est directement proportionnel à σ. De plus, on observe qu'il existe une valeur minimale de σ en dessous de laquelle la matrice reste "numériquement" singulière, ne permettant pas d'obtenir un rang complet. En revanche, pour des valeurs de σ supérieures à 10^6, le problème est trop modifié et la solution n'est plus correcte. La figure 5.6 montre l'évolution de l'erreur en fonction de la valeur de σ. Pour les valeurs supérieures à 10^6, l'erreur relative (5.17) est supérieure à 2% en comparaison au problème de départ.

$$\text{erreur relative} = \frac{\max \|\text{solution avec Flux} - \text{solution après régularisation}\|}{\|\text{solution avec Flux}\|}\,. \qquad (5.17)$$

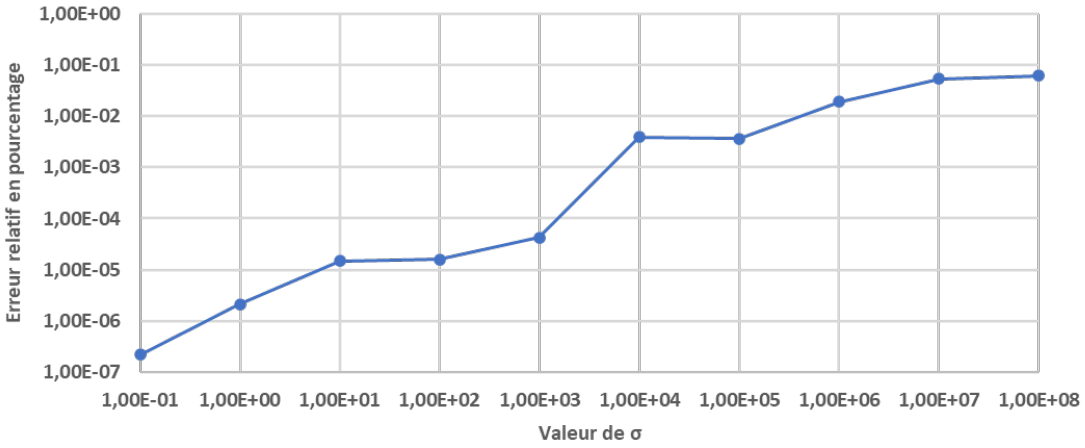

Figure 5.6 – Évolution de l'erreur en fonction de la valeur de σ.

Nous intéressons maintenant à la taille du problème grossier en prenant plusieurs tailles avec $\sigma = 10^6$. Le cas test a 207 degrés de liberté partagées par 2 sous-domaines et 4 variables partagées par 3 sous-domaines. Le test consiste à introduire de "nouvelles[8]" variables (duales) afin d'imposer la contrainte et calculer le conditionnement du problème d'interface $(F_{rr} + F_{rc}K_{cc}^{-1}F_{rc}^{t})$.

Nombre de variables primales	154	104	54	4
Nombre de variables duales	57	107	157	207
Conditionnement	86.457	114.357	367.297	426.457

La taille du problème grossier influence également les résultats : plus le nombre de contraintes primales est élevé, meilleur est le conditionnement.

Nous avons également testé l'implémentation du Q_Δ sur le cas test 3 de la sous-section 4.2.3, avec un petit maillage de 86 nœuds, pour pouvoir manipuler manuellement le choix des variables primales. Ce cas test contient 12 arêtes à l'interface entre l'aimant et l'air. Dans un premier temps, nous avons construit la matrice Q_Δ pour le cas scalaire, où chaque arête contient 3 variables. Nous avons pris la moyenne des 3 variables qui composent l'arête pour construire la matrice Q_Δ. Ensuite, nous avons testé la même implémentation avec un potentiel vecteur, même si dans cette configuration, il n'y a qu'une seule variable par arête. L'erreur sur la solution obtenue est inférieure à 10^{-12}. L'étape suivante serait de réaliser des tests avec la moyenne de plusieurs arêtes.

8. Prise au hasard dans la liste des variables duales.

5.3 Conclusion

Dans ce chapitre, nous avons examiné en détail les défis rencontrés lors de l'application de la décomposition de domaine aux formulations utilisant le potentiel vecteur. La première difficulté majeure réside dans la gestion de la condition de jauge. Nous avons commencé par explorer la méthode de la jauge par arbre, exposant ses principes et les raisons pour lesquelles son utilisation avec la décomposition de domaine s'avère problématique. Par ailleurs, une orientation de l'arbre en utilisant les poids peut être envisagée. L'auto jauge a été considérée mais elle a montré des limites pour la DDM. Ensuite, nous avons envisagé une approche alternative : la méthode de jauge automatique, que nous avons étudiée en détail.

Par la suite, nous avons présenté la méthode de régularisation retenue pour surmonter ces défis et nous avons donner des détails sur sa mise en œuvre.

Dans la seconde partie du chapitre, nous avons procédé à une analyse de l'impact du coefficient de régularisation sur le conditionnement du système. Nous avons observé une relation entre ces deux paramètres, où un choix judicieux du coefficient de régularisation peut grandement influencer la convergence du système. En particulier, nos résultats montrent qu'un coefficient de régularisation optimal doit être choisi, offrant un bon conditionnement sans altérer significativement la nature du problème initial.

En outre, nous avons examiné l'effet de la taille du problème grossier sur la qualité de la résolution. Nos observations ont révélé qu'une augmentation de la taille du problème grossier peut améliorer la résolution, mais il existe un équilibre délicat à trouver entre la taille du problème et l'efficacité de la résolution.

En somme, l'utilisation de la décomposition de domaine avec des formulations basées sur le potentiel vecteur requiert une approche plus minutieuse et présente des défis, notamment en ce qui concerne la gestion de la condition de jauge et le choix approprié du coefficient de régularisation.

Conclusion générale et perspectives

Conclusion

Dans le cadre de cette thèse, notre objectif principal était de réduire le temps nécessaire pour les simulations magnétostatiques par éléments finis. Nous avons développé une méthode de résolution visant à accélérer ces simulations tout en mettant en place une stratégie de calcul parallèle efficace. Ceci dans le but de faire face à l'augmentation de la taille des problèmes à simuler et à l'utilisation accrue de la 3D. Notre choix s'est porté sur la méthode de décomposition de domaine, réputée plus efficace que les méthodes directes et itératives couramment utilisées.

Dans l'optique d'avoir un solveur entièrement parallèle, nous avons orienté notre étude vers la famille des méthodes sans recouvrement. Nous avons ensuite commencé nos recherches en explorant la méthode FETI-1 comme base, avant de mettre en place une méthode plus avancée, la méthode FETI-DP. Lors de la résolution d'un problème magnétostatique, 2 familles de formulation peuvent être employées : Les formulations en potentiel scalaire (traité dans un premier temps cf. chapitre 2, 3 et 4) puis les formulations en potentiel vecteur (traité dans un second temps cf. chapitre 5). Notre objectif était de montrer comment adapter la décomposition de domaine à ces deux formulations en formulant correctement les équations.

L'un des principaux défis rencontrés était l'environnement de travail, un logiciel utilisé par de nombreux clients dans le monde. Nous avons dû tenir compte de ses spécificités pour mettre en place une stratégie de calcul parallèle appropriée. Nous avons tiré parti de bibliothèques performantes pour adapter notre implémentation et optimiser notre code.

Notre analyse nous a orientés vers un découpage des sous-domaines en fonction de la physique d'abord, puis en utilisant un partitionnement de maillage. Cette approche permet de réduire le temps nécessaire à la décomposition et d'obtenir des matrices homogènes en termes d'ordre de grandeur de leurs coefficients, ce qui facilite le préconditionnement.

Nous avons testé la méthode FETI-1 sur plusieurs cas tests et avons constaté son efficacité. Elle représente, ainsi que toutes les méthodes de décomposition de domaine, un moyen efficace d'implémenter une stratégie de calcul parallèle qui tire parti du code séquentiel préexistant. Nos résultats montrent une bonne scalabilité, avec une accélérations sur un cas test de 105 000 inconnues de 3.8 fois avec 4 processeurs et 4 sous-domaines, sous certaines conditions : une taille de problème local d'environ 25 000 inconnues, une parallélisation d'un processeur par sous-domaine et un bon équilibre de la charge de travail.

Nous avons ensuite optimisé le code FETI-DP pour obtenir des résultats similaires. Les performances en mémoire sont un autre avantage de la décomposition de domaine, qui surpasse un solveur direct grâce à la taille des problèmes locaux plus réduite.

Nous avons également abordé les problèmes non linéaires et testé deux méthodes. La première, la méthode NKS, plus simple à mettre en œuvre, améliore le temps de résolution à chaque itération du solveur Newton-Raphson, contribuant ainsi à améliorer les performances générales de la résolution de 30%. La seconde, la méthode SNK, plus novatrice, consiste à localiser la non-linéarité sur quelques sous-domaines. Le grand nombre d'itérations nécessaires à la convergence nous a empêchés d'obtenir de meilleurs résultats, mais cette méthode pourrait faire l'objet d'améliorations futures.

Enfin, nous avons abordé le cas des formulations en potentiel vecteur, mettant en lumière les défis liés à ce type de formulation, tels que les conditions de jauge et le choix du problème grossier. Nous avons présenté des solutions à ces problèmes et analysé les résultats obtenus.

Perspectives

Ce travail a posé les bases pour approfondir l'utilisation de la décomposition de domaine.

Tout d'abord, des améliorations peuvent être apportées à la partie "partitionnement de maillage". La stratégie basée sur la physique ouvre la voie à une implémentation parallèle de l'étape de décomposition qui reste à effectuer. Des études sur cet aspect sont en cours et d'autres outils que METIS sont envisagés.

Un autre aspect lié au maillage est l'utilisation de la décomposition de domaine comme moyen de résoudre des problèmes avec un maillage non conforme. Des travaux sur ce sujet existent, tels que [86], [87], et [88]. Ils permettent de mettre en place une interpolation des vecteurs des multiplicateurs de Lagrange pour traiter ce type de problème. Cela évite, par exemple, de devoir remailler des simulations impliquant du mouvement, ce qui est particulièrement utile dans la simulation des machines tournantes. À l'avenir, une adaptation des multiplicateurs de Lagrange, voire une combinaison avec la méthode mortar [89], pourrait être envisagée.

La méthode SNK mérite d'être étudiée de manière approfondie pour voir si une optimisation de la résolution non linéaire est possible. Des méthodes de relaxation peuvent être envisagées dans cette optique.

Pour ce qui est des formulations en potentiel vecteur, une optimisation des performances est nécessaire. Cela permettra à la décomposition de domaine d'être compétitive sur ces formulations et pourrait améliorer la convergence du non linéaire, qui se comporte mieux avec les formulations en potentiel vecteur. Concernant la problématique de la condition de jauge, nous pourrions essayer d'orienter l'arbre en ajustant les poids, afin d'obtenir le même résultat dans une résolution du problème globale ou une résolution avec la DDM.

Un autre aspect concerne le choix d'utiliser des aimants dans ce travail. Ce choix était motivé par les limitations de l'environnement de développement initial, qui ne permettait pas l'utilisation de bobines. Il serait pertinent d'envisager l'utilisation de la décomposition de domaine pour résoudre des problèmes avec des bobines. Deux défis se posent alors : comment gérer les pré-résolutions et comment partitionner le domaine global avec des bobines ? Pour commencer, nous pourrions considérer chaque bobine entière plus une couche comme un sous-domaine, et n'utiliser la DDM que pour la résolution principale. Il faut par contre faire attention aux fonctions de forme à la frontière du domaine "bobine" et à la précision du calcul.

Enfin, il serait intéressant d'utiliser la décomposition de domaine pour résoudre des problèmes magnétotransitoires, afin de simuler des phénomènes créés par un champ magnétique variable dans le temps, ou magnétoharmoniques, pour prendre en compte les courants induits et les effets de peau dans les régions conductrices.

En termes de calcul parallèle, des améliorations sont toujours possibles en explorant d'autres outils tels que le calcul sur GPU, en utilisant davantage d'OpenMP sur les problèmes locaux, et en augmentant l'échelle de parallélisme.

Bibliographie

[1] A. Toselli et O. Widlund, *Domain decomposition methods-algorithms and theory*. Springer Science & Business Media, 2004, t. 34.

[2] C. Farhat et F.-X. Roux, « A method of finite element tearing and interconnecting and its parallel solution algorithm, » *International journal for numerical methods in engineering*, t. 32, n° 6, p. 1205-1227, 1991.

[3] Q. V. Dihn, R. Glowinski et J. Périaux, « Solving elliptic problems by domain decomposition methods wtih applications, » in *Elliptic Problem Solvers*, Elsevier, 1984, p. 395-426.

[4] P. Le Tallec, Y.-H. De Roeck et M. Vidrascu, « Domain decomposition methods for large linearly elliptic three-dimensional problems, » *Journal of Computational and Applied Mathematics*, t. 34, n° 1, p. 93-117, 1991.

[5] C. Pechstein, *Finite and boundary element tearing and interconnecting solvers for multiscale problems*. Springer Science & Business Media, 2012, t. 90.

[6] A. Klawonn, M. Lanser, O. Rheinbach et M. Uran, « Nonlinear FETI-DP and BDDC methods : a unified framework and parallel results, » *SIAM Journal on Scientific Computing*, t. 39, p. C417-C451, 2017.

[7] C. Farhat, J. Mandel et F. X. Roux, « Optimal convergence properties of the FETI domain decomposition method, » *Computer methods in applied mechanics and engineering*, t. 115, n° 3-4, p. 365-385, 1994.

[8] B. A. Maury, « Méthode des éléments finis et optimisation sous contrainte, » 2021.

[9] A. Ben-Israel et T. N. Greville, *Generalized inverses : theory and applications*. Springer Science & Business Media, 2003, t. 15.

[10] A. Markopoulos, Z. Dostál, T. Kozubek, P. Kovář, T. Brzobohatỳ et R. Kučera, « Stable computations of generalized inverses of positive semidefinite matrices, » in *Domain Decomposition Methods in Science and Engineering XXI*, Springer, 2014, p. 909-916.

[11] A. K. Cline, C. B. Moler, G. W. Stewart et J. H. Wilkinson, « An estimate for the condition number of a matrix, » *SIAM Journal on Numerical Analysis*, t. 16, n° 2, p. 368-375, 1979.

[12] A. Pyzara, B. Bylina et J. Bylina, « The influence of a matrix condition number on iterative methods' convergence, » in *2011 Federated Conference on Computer Science and Information Systems (FedCSIS)*, IEEE, 2011, p. 459-464.

[13] B. Beckermann et A. B. Kuijlaars, « Superlinear convergence of conjugate gradients, » *SIAM Journal on Numerical Analysis*, t. 39, n° 1, p. 300-329, 2001.

[14] J. Mandel et R. Tezaur, « Convergence of a substructuring method with Lagrange multipliers, » *Numerische Mathematik*, t. 73, n° 4, p. 473-487, 1996.

[15] A. Klawonn et O. Widlund, « FETI and Neumann-Neumann iterative substructuring methods : connections and new results, » *Communications on Pure and Applied Mathematics : A Journal Issued by the Courant Institute of Mathematical Sciences*, t. 54, n° 1, p. 57-90, 2001.

[16] C. Farhat, M. Lesoinne, P. LeTallec, K. Pierson et D. Rixen, « FETI-DP : a dual–primal unified FETI method—part I : A faster alternative to the two-level FETI method, » *International journal for numerical methods in engineering*, t. 50, n° 7, p. 1523-1544, 2001.

[17] Y. Fragakis et M. Papadrakakis, « The mosaic of high performance domain decomposition methods for structural mechanics : Formulation, interrelation and numerical efficiency of primal and dual methods, » *Computer methods in applied mechanics and engineering*, t. 192, n° 35-36, p. 3799-3830, 2003.

[18] J. Mandel et B. Sousedík, « BDDC and FETI-DP under minimalist assumptions, » *Computing*, t. 81, n° 4, p. 269-280, 2007.

[19] A. Klawonn, O. B. Widlund et M. Dryja, « Dual-Primal FETI methods with face constraints, » in *Recent developments in domain decomposition methods*, Springer, 2002, p. 27-40.

[20] C. R. Dohrmann, « A preconditioner for substructuring based on constrained energy minimization, » *SIAM Journal on Scientific Computing*, t. 25, n° 1, p. 246-258, 2003.

[21] A. Ruda, « Méthode de Décomposition de Domaine Mixte pour la Simulation Magnétostatique de Machines Électriques, » thèse de doct., Paris-Saclay, 2023.

[22] R. Molina, « Hybridization of FETI Methods, » thèse de doct., Paris 6, 2017.

[23] A. Klawonn et O. Rheinbach, « Robust FETI-DP methods for heterogeneous three dimensional elasticity problems, » *Computer Methods in Applied Mechanics and Engineering*, t. 196, n° 8, p. 1400-1414, 2007.

[24] S. Zampini, « PCBDDC : a class of robust dual-primal methods in PETSc, » *SIAM Journal on Scientific Computing*, t. 38, n° 5, S282-S306, 2016.

[25] J. Pebrel, C. Rey et P. Gosselet, « A nonlinear dual-domain decomposition method : Application to structural problems with damage, » *International Journal for Multiscale Computational Engineering*, t. 6, n° 3, 2008.

[26] C. Negrello, P. Gosselet, C. Rey et J. Pebrel, « Substructured formulations of nonlinear structure problems–influence of the interface condition, » *International Journal for Numerical Methods in Engineering*, t. 107, n° 13, p. 1083-1105, 2016.

[27] C. Negrello, P. Gosselet et C. Rey, « Nonlinearly preconditioned FETI solver for substructured formulations of nonlinear problems, » *Mathematics*, t. 9, n° 24, p. 3165, 2021.

[28] A. Toselli et X. Vasseur, « Dual-primal FETI algorithms for edge element approximations : Two-dimensional h and p finite elements on shape-regular meshes, » *SIAM journal on numerical analysis*, t. 42, n° 6, p. 2590-2611, 2005.

[29] A. Toselli, « Dual-primal FETI algorithms for edge finite-element approximations in 3D, » *IMA journal of numerical analysis*, t. 26, n° 1, p. 96-130, 2006.

[30] J. Mandel, « Balancing domain decomposition, » *Communications in numerical methods in engineering*, t. 9, n° 3, p. 233-241, 1993.

[31] S. Badia, A. F. Martín et M. Olm, « Scalable solvers for complex electromagnetics problems, » *Finite elements in analysis and design*, t. 161, p. 16-31, 2019.

[32] P. Dular, « Modélisation du champ magnétique et des courants induits dans des systèmes tridimensionnels non linéaires, » 1994.

[33] G. Meunier, « The finite element method for electromagnetic modeling, » 2010.

[34] C. Vassallo, *Electromagnétisme classique dans la matière*. Dunod, 1980.

[35] R. Dautray et J.-L. Lions, « Analyse mathématique et calcul numérique pour les sciences et les techniques, » *Collection du Commissariat a l'Energie Atomique. Serie Scientifique*, 1985.

[36] A. Bossavit, *Computational electromagnetism : variational formulations, complementarity, edge elements*. Academic Press, 1998.

[37] A. Bossavit, « Eddy-currents and forces in deformable conductors, » *Mechanical Modellings of New Electromagnetic Materials*, t. 24, p. 235-242, 1990.

[38] P. Ferrouillat, « Développement de formulations éléments finis 3D en potentiel vecteur magnétique : application aux machines asynchrones en mouvement, » thèse de doct., Université Grenoble Alpes (ComUE), 2015.

[39] J.-L. Coulomb, F.-X. Zgainski et Y. Marechal, « A pyramidal element to link hexahedral, prismatic and tetrahedral edge finite elements, » *IEEE transactions on magnetics*, t. 33, n° 2, p. 1362-1365, 1997.

[40] R. Albanese et G. Rubinacci, « Magnetostatic field computations in terms of two-component vector potentials, » *International journal for numerical methods in engineering*, t. 29, n° 3, p. 515-532, 1990.

[41] W. Greiner, *Classical electrodynamics*. Springer Science & Business Media, 2012.

[42] H. Le Dret et H. Le Dret, « La méthode de Galerkin, » *Équations aux dérivées partielles elliptiques non linéaires*, p. 83-97, 2013.

[43] G. Dhatt, E. Lefrançois et G. Touzot, *Finite element method*. John Wiley & Sons, 2012.

[44] Altair Engineering, *Altair Flux^{TM}*. adresse : https://www.altair.com/flux.

[45] Simulia, *Getting started with Abaqus*, Version 6.8, Dassault Systemes, 2008.

[46] Y. Saad, *Iterative methods for sparse linear systems*. SIAM, 2003.

[47] M. J. Flynn, *Flynn's Taxonomy*. 2011.

[48] N. J. Higham, « Gaussian elimination, » *Wiley Interdisciplinary Reviews : Computational Statistics*, t. 3, n° 3, p. 230-238, 2011.

[49] T. A. Davis, S. Rajamanickam et W. M. Sid-Lakhdar, « A survey of direct methods for sparse linear systems, » *Acta Numerica*, t. 25, p. 383-566, 2016.

[50] J. Nocedal et S. J. Wright, « Conjugate gradient methods, » *Numerical optimization*, p. 101-134, 2006.

[51] P. Amestoy, I. S. Duff, J. Koster et J.-Y. L'Excellent, « A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling, » *SIAM Journal on Matrix Analysis and Applications*, t. 23, n° 1, p. 15-41, 2001.

[52] P. R. Amestoy et C. Puglisi, « An unsymmetrized multifrontal LU factorization, » *SIAM Journal on Matrix Analysis and Applications*, t. 24, n° 2, p. 553-569, 2002.

[53] P. R. Amestoy, I. S. Duff, J.-Y. L'excellent et X. S. Li, « Analysis and comparison of two general sparse solvers for distributed memory computers, » *ACM Transactions on Mathematical Software (TOMS)*, t. 27, n° 4, p. 388-421, 2001.

[54] S. Balay et al., *PETSc Web page*, https://petsc.org/, 2023. adresse : https://petsc.org/.

[55] Intel, *Intel® VTune™*. adresse : https://www.intel.com/content/www/us/en/developer/tools/oneapi/vtune-profiler.html.

[56] F. Vi, « Methode multigrilles parallèle pour les simulations 3D de mise en forme de matériaux, » thèse de doct., Paris Sciences et Lettres (ComUE), 2017.

[57] U. G. Partitioning, « GEORGE KARYPIS AND VIPIN KUMAR, » 1995.

[58] D. LaSalle, M. M. A. Patwary, N. Satish, N. Sundaram, P. Dubey et G. Karypis, « Improving graph partitioning for modern graphs and architectures, » in *Proceedings of the 5th Workshop on Irregular Applications : Architectures and Algorithms*, 2015, p. 1-4.

[59] G. Karypis et V. Kumar, « Multilevelk-way partitioning scheme for irregular graphs, » *Journal of Parallel and Distributed computing*, t. 48, n° 1, p. 96-129, 1998.

[60] G. Karypis et V. Kumar, « Multilevel algorithms for multi-constraint graph partitioning, » in *SC'98 : Proceedings of the 1998 ACM/IEEE Conference on Supercomputing*, IEEE, 1998, p. 28-28.

[61] R. Diekmann, B. Monien et R. Preis, « Using helpful sets to improve graph bisections, » *Interconnection networks and mapping and scheduling parallel computations*, t. 21, p. 57-73, 1994.

[62] C. Walshaw et M. Cross, « Mesh partitioning : a multilevel balancing and refinement algorithm, » *SIAM Journal on Scientific Computing*, t. 22, n° 1, p. 63-80, 2000.

[63] F. Pellegrini, « Scotch and PT-scotch graph partitioning software : an overview, » *Combinatorial Scientific Computing*, p. 373-406, 2012.

[64] F. Pellegrini, « Static mapping by dual recursive bipartitioning of process architecture graphs, » in *Proceedings of IEEE Scalable High Performance Computing Conference*, IEEE, 1994, p. 486-493.

[65] B. Hendrickson, R. W. Leland et al., « A Multi-Level Algorithm For Partitioning Graphs., » *SC*, t. 95, n° 28, p. 1-14, 1995.

[66] E. G. Boman, U. V. Catalyurek, C. Chevalier et K. D. Devine, « The Zoltan and Isorropia Parallel Toolkits for Combinatorial Scientific Computing : Partitioning, Ordering, and Coloring, » *Scientific Programming*, t. 20, n° 2, p. 129-150, 2012.

[67] D. Barrero Franquet, « Software for mesh partitioning, » mém. de mast., Universitat Politècnica de Catalunya, 2010.

[68] R. Perrin-Bit, « Modelisation des machines electriques tournantes par la methode des elements finis tridimensionnels : Calcul des grandeurs magnetiques avec prise en compte du mouvement, » thèse de doct., Institut National Polytechnique Grenoble (INPG), 1994.

[69] A. Chervyakov, « Comparison of magnetic vector and total scalar potential formulations for finite-element modeling of dipole magnet with COMSOL Multiphysics, » *arXiv preprint arXiv:2107.01957*, 2021.

[70] G. Allaire et M. Schoenauer, *Conception optimale de structures*. Springer, 2007, t. 58.

[71] T. J. Hughes, G. Engel, L. Mazzei et M. G. Larson, « The continuous Galerkin method is locally conservative, » *Journal of Computational Physics*, t. 163, n° 2, p. 467-488, 2000.

[72] T. Henneron, S. Clenet et F. Piriou, « Computation of the magnetic flux in the finite elements method, » *The European Physical Journal Applied Physics*, t. 39, n° 2, p. 119-128, 2007.

[73] P. Dular, W. Legros et A. Nicolet, « Coupling of local and global quantities in various finite element formulations and its application to electrostatics, magnetostatics and magnetodynamics, » *IEEE Transactions on Magnetics*, t. 34, n° 5, p. 3078-3081, 1998.

[74] G. F. Carey, « Some further properties of the superconvergent flux projection, » *Communications in numerical methods in engineering*, t. 18, n° 4, p. 241-250, 2002.

[75] A. Klawonn, M. Lanser et O. Rheinbach, « Nonlinear feti-dp and bddc methods, » *SIAM Journal on Scientific Computing*, t. 36, n° 2, A737-A765, 2014.

[76] M. I. Ghenai et al., « Domain decomposition for 3D nonlinear magnetostatic problems : Newton-Krylov-Schur vs. Schur-Newton-Krylov methods, » *IEEE Transactions on Magnetics*, 2023.

[77] Y. Zhao et W. Fu, « A novel formulation with coulomb gauge for 3-D magnetostatic problems using edge elements, » *IEEE Transactions on Magnetics*, t. 53, n° 6, p. 1-4, 2017.

[78] Z. Ren, « Influence of the RHS on the convergence behaviour of the curl-curl equation, » *IEEE transactions on magnetics*, t. 32, n° 3, p. 655-658, 1996.

[79] O. Castillo Reyes, J. de la Puente, V. Puzyrev et J. M. Cela, « Assessment of edge-based finite element technique for geophysical electromagnetic problems : efficiency, accuracy and reliability, » in *Proceedings of the 1st Pan-American Congress on Computational Mechanics*, International Center for Numerical Methods in Engineering (CIMNE), 2015, p. 984-995.

[80] F. Rapetti, A. Alonso Rodríguez et E. De Los Santos, « On the tree gauge in magnetostatics, » *J*, t. 5, n° 1, p. 52-63, 2022.

[81] Q. Zhang et S. Cen, *Multiphysics Modeling : Numerical Methods and Engineering Applications : Tsinghua University Press Computational Mechanics Series*. Elsevier, 2015.

[82] Z. Tang, Y. Zhao et Z. Ren, « Auto-gauging of vector potential by parallel sparse direct solvers—Numerical observations, » *IEEE Transactions on Magnetics*, t. 55, n° 6, p. 1-5, 2019.

[83] S. Reitzinger et J. Schöberl, « An algebraic multigrid method for finite element discretizations with edge elements, » *Numerical linear algebra with applications*, t. 9, n° 3, p. 223-238, 2002.

[84] H. Duan, S. Li, R. C. Tan et W. Zheng, « A delta-regularization finite element method for a double curl problem with divergence-free constraint, » *SIAM Journal on Numerical Analysis*, t. 50, n° 6, p. 3208-3230, 2012.

[85] J. Mandel et B. Sousedík, *Adaptive coarse space selection in the BDDC and the FETI-DP iterative substructuring methods : Optimal face degrees of freedom*. Springer, 2007.

[86] C. Lacour, « Two different approaches for matching nonconforming grids : the Mortar Element method and the FETI method, » *BIT*, t. 37, n° 3, p. 13-32, 1996.

[87] D. Stefanica, « A numerical study of FETI algorithms for mortar finite element methods, » *SIAM Journal on Scientific Computing*, t. 23, n° 4, p. 1135-1160, 2001.

[88] T. P. Mathew, *Domain decomposition methods for the numerical solution of partial differential equations*. Springer, 2008.

[89] A. Chauhan, « 3D modelling of movements in electric machines using edge finite elements, » thèse de doct., Université Grenoble Alpes, 2021.

Annexe A

Appendix A

A.1 Appendix A.1

Considérons un exemple simple pour illustrer le cas général. La figure 1.6 montre un domaine coupé en 4 sous domaines tous voisins les uns des autres.

Dans ce cas, seul le sous domaine 4 est non flottant, donc sa matrice K_4 est inversible.

Pour le sous domaine 1, les équations sont :

$$K_1 u_1 = f_1 + (B_{12}^t + B^t{}_{13} + B^t{}_{14})\lambda + E_1\alpha_1 \tag{A.1}$$

Les conditions sur l'interface :

$$\begin{aligned} B_{12}^t u_1 &= B_{21}^t u_2 \\ B_{13}^t u_1 &= B_{31}^t u_3 \\ B_{14}^t u_1 &= B_{41}^t u_4 \end{aligned} \tag{A.2}$$

Si nous utilisons la première équation du système précédent :

$$\begin{aligned} &(B_{12}K_1^+ B_{12}^t + B_{12}K_1^+ B_{13}^t + B_{12}K_1^+ B_{14}^t + B_{21}K_2^+ B_{21}^t - B_{21}K_2^+ B_{23}^t - B_{21}K_2^+ B_{24}^t)\lambda \\ &= B_{21}K_2^+ f_2 - B_{12}K_1^+ f_1 + B_{21}E_2\alpha_2 - B_{12}E_1\alpha_1 \end{aligned} \tag{A.3}$$

Après utilisation des trois équations du sous domaine 1 (entre 1 et 2, entre 1 et 3, entre 1 et 4), deux équations du sous domaine 2 (entre 2 et 3, entre 2 et 4), une équation du sous domaine 3 (entre 3 et 4), F du système 1.16 est :

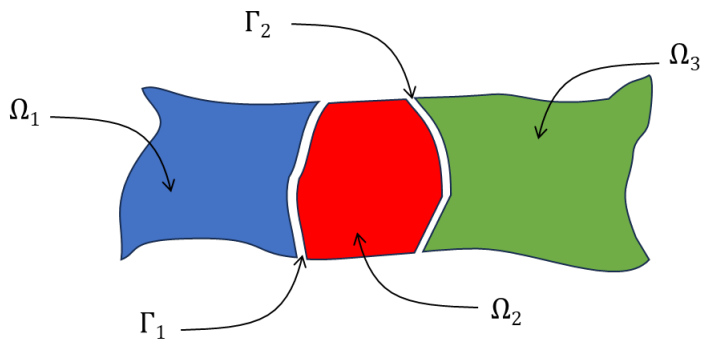

Figure A.1 – Découpage en 3 sous domaines

$$\begin{pmatrix}
B_{12}K_1^+B_{12}^t + B_{12}K_1^+B_{13}^t + B_{12}K_1^+B_{14}^t + B_{21}K_2^+B_{21}^t - B_{21}K_2^+B_{23}^t - B_{21}K_2^+B_{24}^t \\
B_{13}K_1^+B_{12}^t + B_{13}K_1^+B_{13}^t + B_{13}K_1^+B_{14}^t + B_{31}K_3^+B_{31}^t + B_{31}K_3^+B_{32}^t - B_{31}K_3^+B_{34}^t \\
B_{14}K_1^+B_{12}^t + B_{14}K_1^+B_{13}^t + B_{14}K_1^+B_{14}^t + B_{41}K_4^{-1}B_{41}^t + B_{41}K_4^{-1}B_{42}^t + B_{41}K_4^{-1}B_{43}^t \\
-B_{23}K_1^+B_{21}^t + B_{23}K_1^+B_{23}^t + B_{23}K_1^+B_{24}^t + B_{32}K_3^+B_{31}^t + B_{32}K_3^+B_{32}^t - B_{32}K_3^+B_{34}^t \\
-B_{24}K_2^+B_{21}^t + B_{24}K_2^+B_{23}^t + B_{24}K_2^+B_{24}^t + B_{42}K_4^{-1}B_{41}^t + B_{42}K_4^{-1}B_{42}^t + B_{42}K_4^{-1}B_{43}^t \\
-B_{34}K_3^+B_{31}^t - B_{34}K_3^+B_{32}^t + B_{34}K_3^+B_{34}^t + B_{43}K_4^{-1}B_{41}^t + B_{43}K_4^{-1}B_{42}^t + B_{43}K_4^{-1}B_{43}^t
\end{pmatrix}$$

La matrice E pour notre cas s'écrit sous la forme :

$$\begin{pmatrix}
B_{12}E_1 & -B_{21}E_2 & 0 \\
B_{13}E_1 & 0 & -B_{31}E_3 \\
B_{14}E_1 & 0 & 0 \\
0 & B_{23}E_2 & -B_{32}E_3 \\
0 & B_{24}E_2 & 0 \\
0 & 0 & B_{34}E_3
\end{pmatrix}$$

A.2 Appendix A.2

Nous allons construire le système à résoudre pour cette méthode. Nous allons utiliser l'exemple illustré dans la figure A.1. Cet exemple avec un découpage simple permet l'absence de cross-points ce qui facilite l'écriture de la méthode.

Le système global du type $Ku = f$ peut être écrit sous la forme :

$$\begin{bmatrix} K_{ii}^{(1)} & 0 & 0 & K_{ib}^{(1)} \\ 0 & K_{ii}^{(2)} & 0 & K_{ib}^{(2)} \\ 0 & 0 & K_{ii}^{(3)} & K_{ib}^{(3)} \\ K_{bi}^{(1)} & K_{bi}^{(2)} & K_{bi}^{(3)} & K_{bb} \end{bmatrix} \begin{bmatrix} u_i^{(1)} \\ u_i^{(2)} \\ u_i^{(3)} \\ u_b \end{bmatrix} = \begin{bmatrix} f_i^{(1)} \\ f_i^{(2)} \\ f_i^{(3)} \\ f_b \end{bmatrix}.$$

Nous imposons des condition de Neumann sur l'interface Γ_1 et nous utilisons l'indice b_1 pour designer ces variables. Nous imposons des conditions de Robin sur Γ_2 avec l'indice b_1 cette fois. Nous définissons aussi les différentes sous matrices et sous vecteurs :

$$K_{bi}^{(1)} = \begin{bmatrix} K_{b_1 i}^{(1)} \\ 0 \end{bmatrix}, \quad K_{bi}^{(2)} = \begin{bmatrix} K_{b_1 i}^{(2)} \\ K_{b_2 i}^{(2)} \end{bmatrix}, \quad K_{bi}^{(3)} = \begin{bmatrix} 0 \\ K_{b_2 i}^{(3)} \end{bmatrix} \tag{A.4}$$

$$K_{ib}^{(1)} = \begin{bmatrix} K_{ib_1}^{(1)} & 0 \end{bmatrix}, \quad K_{ib}^{(2)} = \begin{bmatrix} K_{ib_1}^{(2)} & K_{ib_2}^{(2)} \end{bmatrix}, \quad K_{ib}^{(3)} = \begin{bmatrix} 0 & K_{ib_2}^{(3)} \end{bmatrix} \tag{A.5}$$

$$K_{bb} = K_{bb}^{(1)} + K_{bb}^{(2)} + K_{bb}^{(3)} = \begin{bmatrix} K_{b_1 b_1}^{(1)} & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} K_{b_1 b_1}^{(2)} & 0 \\ 0 & K_{b_2 b_2}^{(2)} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & K_{b_2 b_2}^{(3)} \end{bmatrix} \tag{A.6}$$

$$f_b = f_b^{(1)} + f_b^{(2)} + f_b^{(3)} = \begin{bmatrix} f_{b_1}^{(1)} \\ 0 \end{bmatrix} + \begin{bmatrix} f_{b_1}^{(2)} \\ f_{b_2}^{(2)} \end{bmatrix} + \begin{bmatrix} 0 \\ f_{b_2}^{(3)} \end{bmatrix} \tag{A.7}$$

$$u_b = \begin{bmatrix} u_{b_1} \\ u_{b_2} \end{bmatrix} \tag{A.8}$$

Pour le sous domaine 1, le problème, après introduction de multiplicateurs de Lagrange, devient :

$$\begin{bmatrix} K_{ii}^{(1)} & K_{ib_1}^{(1)} \\ K_{b_1 i}^{(1)} & K_{b_1 b_1}^{(1)} \end{bmatrix} \begin{bmatrix} u_i^{(1)} \\ u_{b_1}^{(1)} \end{bmatrix} = \begin{bmatrix} f_i^{(1)} \\ f_{b_1}^{(1)} + \lambda_{b_1}^{(1)} \end{bmatrix} \tag{A.9}$$

Le problème du sous-domaine 2 quant à lui s'écrit :

$$\begin{bmatrix} K_{ii}^{(2)} & K_{ib_1}^{(2)} & K_{ib_2}^{(2)} \\ K_{b_1 i}^{(2)} & K_{b_1 b_1}^{(2)} & 0 \\ K_{b_2 i}^{(2)} & 0 & K_{b_2 b_2}^{(2)} + A_{b_2}^{(2)} \end{bmatrix} \begin{bmatrix} u_i^{(2)} \\ u_{b_i}^{(2)} \\ u_{b_2}^{(2)} \end{bmatrix} = \begin{bmatrix} f_i^{(2)} \\ f_{b_1}^{(2)} + \lambda_{b_1}^{(2)} \\ f_{b_2}^{(2)} + \lambda_{b_2}^{(2)} \end{bmatrix} \tag{A.10}$$

A coïncidant avec le complément de Schur (voir 1.4.4) dans le cas optimal et pour un découpage comme pour cette exemple. D'autres choix sont généralement considérés qui permettent d'avoir un problème bien posé, améliorer la convergence et sans coûter trop cher pour les construire.

Pour le troisième sous domaine :

$$\begin{bmatrix} K_{ii}^{(3)} & K_{ib_2}^{(3)} \\ K_{b_2i}^{(3)} & K_{b_2b_2}^{(3)} + A_{b_2}^{(3)} \end{bmatrix} \begin{bmatrix} u_i^{(3)} \\ u_{b_2}^{(3)} \end{bmatrix} = \begin{bmatrix} f_i^{(3)} \\ f_{b_2}^{(3)} + \lambda_{b_2}^{(3)} \end{bmatrix} \tag{A.11}$$

Nous posons aussi :

$$\lambda_{b1} = \lambda_{b1}^{(1)} = -\lambda_{b1}^{(2)} \tag{A.12}$$

Nous allons maintenant éliminer les variables intérieurs. D'après le système du sous-domaine 1 nous avons :

$$\begin{aligned} \lambda_{b_1} &= K_{b_1i}^{(1)} u_i^{(1)} + K_{b_1b_1}^{(1)} u_{b_1}^{(1)} - b_{f_1}^{(1)} \\ &= -K_{b_1i}^{(1)} K_{ii}^{(1)^{-1}} K_{ib_1}^{(1)} u_{b_1}^{(1)} + K_{b_1b_1}^{(1)} u_{b_1}^{(1)} - f_{b_1}^{(1)} + K_{b_1i}^{(1)} K_{ii}^{(1)^{-1}} f_i^{(1)} \\ &= S_{b_1b_1}^{(1)} u_{b_1}^{(1)} - c_{b_1}^{(1)} \end{aligned} \tag{A.13}$$

Avec $c_{b_1}^{(1)} := f_{b_1}^{(1)} - K_{b_1i}^{(1)} K_{ii}^{(1)^{-1}} f_i^{(1)}$. Nous pouvons écrire $u_{b_1}^{(1)}$ en fonction de λ_{b1} tel que pour $F^{(1)} = S_{b_1b_1}^{(1)^{-1}}$:

$$u_{b_1}^{(1)} = F^{(1)} \lambda_{b_1} + F^{(1)} c_{b_1}^{(1)} \tag{A.14}$$

Pour le sous domaine 2, nous avons :

$$\begin{aligned} \begin{bmatrix} \lambda_{b_1} \\ \lambda_{b_2}^{(2)} \end{bmatrix} &= \begin{bmatrix} K_{b_1i}^{(2)} u_i^{(2)} + K_{b_1b_1}^{(2)} u_{b_1}^{(2)} - b_{b_1}^{(2)} \\ K_{b_2i}^{(2)} u_i^{(2)} + \left(K_{b_2b_2}^{(2)} + A_{b_2}^{(2)} \right) u_{b_2}^{(2)} - b_{b_2}^{(2)} \end{bmatrix} \\ &= \begin{bmatrix} K_{b_1b_1}^{(2)} - K_{b_1i}^{(2)} K_{ii}^{(2)^{-1}} K_{ib_1}^{(2)} & -K_{b_1i}^{(2)} K_{ii}^{(2)^{-1}} K_{ib_2}^{(2)} \\ -K_{b_2i}^{(2)} K_{ii}^{(2)^{-1}} K_{ib_1}^{(2)} & K_{b_2b_2}^{(2)} - K_{b_2i}^{(2)} K_{ii}^{(2)^{-1}} K_{ib_2}^{(2)} + A_{b_2}^{(2)} \end{bmatrix} \begin{bmatrix} u_{b_1}^{(2)} \\ u_{b_2}^{(2)} \end{bmatrix} \\ &+ \begin{bmatrix} -f_{b_1}^{(2)} + K_{b_1i}^{(2)} K_{ii}^{(2)^{-1}} f_i^{(2)} \\ -f_{b_2}^{(2)} + K_{b_1i}^{(2)} K_{ii}^{(2)^{-1}} f_i^{(2)} \end{bmatrix} \\ &= \begin{bmatrix} S_{b_1b_1}^{(2)} & S_{b_1b_2}^{(2)} \\ S_{b_2b_1}^{(2)} & S_{b_2b_2}^{(2)} + A_{b_2}^{(2)} \end{bmatrix} \begin{bmatrix} u_{b_1}^{(2)} \\ u_{b_2}^{(2)} \end{bmatrix} - \begin{bmatrix} c_{b_1}^{(2)} \\ c_{b_2}^{(2)} \end{bmatrix} \end{aligned} \tag{A.15}$$

Écrire $u_{b_1}^{(2)}$ et $u_{b_2}^{(2)}$ en fonction des multiplicateurs de Lagrange donne :

$$\begin{bmatrix} u_{b_1}^{(2)} \\ u_{b_2}^{(2)} \end{bmatrix} = F^{(2)} \begin{bmatrix} \lambda_{b_1} \\ \lambda_{b_2}^{(2)} \end{bmatrix} + F^{(2)} \begin{bmatrix} c_{b_1}^{(2)} \\ c_{b_2}^{(2)} \end{bmatrix} \tag{A.16}$$

Avec

$$\begin{aligned} F^{(2)} &= \begin{bmatrix} S_{b_1b_1}^{(2)} & S_{b_1b_2}^{(2)} \\ S_{b_2b_1}^{(2)} & S_{b_2b_2}^{(2)} + A_{b_2}^{(2)} \end{bmatrix}^{-1} \\ &= \begin{bmatrix} F_{b_1b_1}^{(2)} & F_{b_1b_2}^{(2)} \\ F_{b_2b_1}^{(2)} & F_{b_2b_2}^{(2)} \end{bmatrix} \end{aligned} \tag{A.17}$$

(A.16) peut alors s'écrire comme :

$$\begin{bmatrix} u_{b_1}^{(2)} \\ u_{b_2}^{(2)} \end{bmatrix} = \begin{bmatrix} F_{b_1b_1}^{(2)}\lambda_{b_1} + F_{b_1b_2}^{(2)}\lambda_{b_2}^{(2)} + F_{b_1b_1}^{(2)}c_{b_1}^{(2)} + F_{b_1b_2}^{(2)}c_{b_2}^{(2)} \\ F_{b_2b_1}^{(2)}\lambda_{b_1} + F_{b_2b_2}^{(2)}\lambda_{b_2}^{(2)} + F_{b_2b_1}^{(2)}c_{b_1}^{(2)} + F_{b_2b_2}^{(2)}c_{b_2}^{(2)} \end{bmatrix} \tag{A.18}$$

Nous continuons avec le troisième sous domaine :

$$\begin{aligned} \lambda_{b_2}^{(3)} &= K_{b_2i}^{(3)}u_i^{(3)} + \left(K_{b_2b_2}^{(3)} + A_{b_2}^{(3)}\right)u_{b_2}^{(3)} - f_{b_2}^{(3)} \\ &= -K_{b_2i}^{(3)}K_{ii}^{(3)^{-1}}K_{ib_2}^{(3)}u_{b_2}^{(3)} + \left(K_{b_2b_2}^{(3)} + A_{b_2}^{(3)}\right)u_{b_2}^{(3)} - f_{b_2}^{(3)} + K_{b_2i}^{(3)}K_{ii}^{(3)^{-1}}f_i^{(3)} \\ &= \left(S_{b_2b_2}^{(3)} + A_{b_2}^{(3)}\right)u_{b_2}^{(3)} - c_{b_2}^{(3)} \end{aligned} \tag{A.19}$$

$u_{b_2}^{(3)}$ est alors écrite :

$$x_{b_2}^{(3)} = F^{(3)}\lambda_{b_2}^{(3)} + F^{(3)}c_{b_2}^{(3)} \tag{A.20}$$

Pour $F^{(3)} = \left(S_{b_2b_2}^{(3)} + A_{b_2}^{(3)}\right)^{-1}$:

$$u_{b_2}^{(3)} = F^{(3)}\lambda_{b_2}^{(3)} + F^{(3)}c_{b_2}^{(3)} \tag{A.21}$$

Les conditions de continuité sur l'interface $b1$ et sur l'interface $b2$ vues du sous-domaine 2 puis du sous-domaine 3 sont respectivement données par :

$$\begin{aligned} u_{b_1}^{(1)} - u_{b_1}^{(2)} &= 0 \\ \lambda_{b_2}^{(2)} + \lambda_{b_2}^{(3)} - \left(A_{b_2}^{(2)} + A_{b_2}^{(3)}\right)u_{b_2}^{(2)} &= 0 \\ \lambda_{b_2}^{(2)} + \lambda_{b_2}^{(3)} - \left(A_{b_2}^{(2)} + A_{b_2}^{(3)}\right)u_{b_2}^{(3)} &= 0 \end{aligned} \tag{A.22}$$

Nous pouvons alors écrire le système réduit sur l'interface tel que

$$\begin{bmatrix} F^{(1)} + F^{(2)}_{b_1 b_1} & -F^{(2)}_{b_1 b_2} & 0 \\ \left(A^{(2)}_{b_2} + A^{(3)}_{b_2}\right) F^{(2)}_{b_2 b_1} & I - \left(A^{(2)}_{b_2} + A^{(3)}_{b_2}\right) F^{(2)}_{b_2 b_2} & I \\ 0 & I & I - \left(A^{(2)}_{b_2} + A^{(3)}_{b_2}\right) F^{(3)} \end{bmatrix} \begin{bmatrix} \lambda_{b_1} \\ \lambda^{(2)}_{b_2} \\ \lambda^{(3)}_{b_2} \end{bmatrix}$$
$$= \begin{bmatrix} -F^{(1)} c^{(1)}_{b_1} + F^{(2)}_{b_1 b_1} c^{(2)}_{b_1} + F^{(2)}_{b_1 b_2} c^{(2)}_{b_2} \\ \left(A^{(2)}_{b_2} + A^{(3)}_{b_2}\right) \left(F^{(2)}_{b_2 b_1} c^{(2)}_{b_1} + F^{(2)}_{b_2 b_2} c^{(2)}_{b_1}\right) \\ \left(A^{(2)}_{b_2} + A^{(3)}_{b_2}\right) F^{(3)} c^{(3)}_{b_2} \end{bmatrix} \tag{A.23}$$

C'est avec des multiplicateurs de Lagrange différents des deux côtés de l'interface que nous arrivons à exprimer des conditions mixtes.

Table des figures

Liste des tableaux