Une méthode de prolongement régulier pour la simulation d'écoulements fluide/particules

Benoit Fabrèges

N° d'ordre :

Université Paris-Sud
Faculté des Sciences d'Orsay

THÈSE

présentée pour obtenir

LE GRADE DE DOCTEUR EN SCIENCES
DE L'UNIVERSITÉ PARIS XI

Spécialité : Mathématiques

par

Benoît Fabrèges

Sujet :

UNE MÉTHODE DE PROLONGEMENT RÉGULIER POUR LA SIMULATION D'ÉCOULEMENTS FLUIDE/PARTICULES

Soutenue le 6 Décembre 2012 devant la Commission d'examen :

M.	Roland Becker	(Rapporteur)
M.	Silvia Bertoluzza	(Examinatrice)
M.	Miguel A. Fernández	(Examinateur)
M.	Patrick Joly	(Rapporteur)
M.	Yvon Maday	(Examinateur)
M.	Bertrand Maury	(Directeur de thèse)

Table des matières

Perspectives

Introduction

Le sujet de cette thèse est la simulation du mouvement de particules rigide immergées. Nous proposons une méthode numérique de type variationnel pour prendre en compte la contrainte de mouvement rigide dans les équations régissant la partie fluide.

Simuler des sytèmes fluide/particules

On retrouve dans la nature de nombreux exemples de système fluide/particules solide. On peut penser par exemple au cas des globules rouges dans le sang, qui dans une première approximation peuvent être considérés comme des particules rigides. La concentration en globules rouges dans le sang peut atteindre une concentration allant jusqu'à 40%, ce qui constitue une suspension très dense. On trouve aussi de nombreux travaux en ce qui concerne l'étude rhéologique de suspension active ou passive. L'étude dans le cas de suspension passive diluée (avec une faible concentration volumique en particules) et à bas nombre de Reynolds à commencé avec les travaux d'Einstein qui prédit une évolution de la viscosité effective comme une fonction linéaire de la concentration volumique solide. Dans ce cas là, les particules sont suffisament éloignées pour pouvoir considérer qu'elles n'intéragissent pas les unes avec les autres. Dans le cas d'une suspension dense, chaque particule rigide agit sur les particules proches qui, à leur tour, agissent sur les autres particules proches. Les particules se voient donc à travers le fluide et les méthodes utilisées dans le cas dilué ne s'applique plus. On utilise alors des simulations numériques pour prédire le comportement d'un tel fluide complexe. Pour des suspensions actives, comme par exemple le cas de la nage de bactéries, il est possible d'obtenir des résultats très différents du cas des suspensions passives. En effet, dans le cas de bactéries auto-propulsées comme *Escherichia coli* ou bien *Bacillus subtilis*, on trouve des modèles qui prédisent une diminution de la viscosité effective en fonction de la concentration là où on observe une augmentation de la viscosité effective dans le cas de particules rigides passives. Une autre application plus physique est le cas de la fluidisation ou bien de la sédimentation. Dans le cas de la fluidisation on regarde un fluide passant à travers une collection dense de particules rigides, dans le cas de la sédimentation on s'intéresse par exemple à la vitesse limite de sédimentation d'une seule particule ou d'une suspension dense de particules rigides.

Si l'on s'intéresse au nombre de Reynolds dans ces applications, on s'aperçoit qu'il est généralement très faible ce qui permet de nous placer dans un régime de Stokes. En effet, dans le cas des bactéries, la taille caractéristique est de l'ordre du micromètre et les vitesses caractéristiques sont de l'ordre de la dizaine de micromètres par seconde. Ainsi,

même si la viscosité du fluide est de l'ordre de 10^{-3} (celle de l'eau par exemple), le nombre de Reynolds dans cette application est donc de l'ordre de 10^{-6}, ce qui justifie parfaitement l'utilisation d'un modèle de Stokes (sans inertie) en ce qui concerne le fluide.

Dans les applications on considère souvent des suspensions denses de particules rigides, le domaine fluide étudié comporte donc énormément de bords, proches les uns des autres. Afin de simuler de façon précise le mouvement des particules, il est donc nécessaire de résoudre la partie fluide suffisamment précisément jusqu'au bord du domaine. Le choix de la méthode de résolution est donc très important.

Présentation de la méthode développée

Une première approche consiste à utiliser un maillage conforme à la géométrie du problème mais dans nos applications, le fait que l'on considère des suspensions denses, et donc des domaines très perforés, est un problème pour cette méthode. En effet, il devient difficile et long de générer de tel maillage, notamment en dimension 3, surtout lorsqu'il faut le générer à chaque pas de temps. De plus, avec un maillage conforme, les conditions péridioques sont difficilement implémentables. Pour calculer la viscosité effective d'un fluide complexe, on a besoin de considérer un domaine périodique ce qui nous fait préférer l'utilisation d'une méthode sur maillage non conforme.

On a choisi d'utiliser une méthode variationnelle de type éléments finis. Les objectifs de la méthode que l'on a développée sont les suivants :

- On ne veut pas avoir à gérer de données sur le maillage et on veut pouvoir utiliser des solveurs rapides comme la transformée de Fourier rapide par exemple. On fait donc le choix d'une méthode sur maillage cartésien qui rempli ces deux conditions. On n'a besoin de stocker que le pas du maillage pour avoir toutes les informations sur le maillage qui restera le même tout au long de la simulation. On ne maille donc plus uniquement le domaine fluide mais tout le domaine englobant le fluide et les particules. Ceci nous permet de considérer des boites maillées avec un maillage cartésien qui ne voit plus la géométrie du problème.
- Comme expliqué plus haut, étant donnés les domaines considérés, on a besoin d'une méthode qui soit précise en espace jusqu'au bords. Le problème des méthodes sur maillage non conforme est justement la précision de la solution aux bords qui est dégradée par rapport à un maillage conforme car ces bords ne sont plus bien approchés. En effet, comme on dispose d'un maillage du domaine fluide/particules, le champ de vitesse solution est un prolongement à tout le domaine du champ de vitesse recherché, celui ci étant défini sur le domaine fluide uniquement. Or ce prolongement n'est en général pas aussi régulier que la solution exacte dans le domaine fluide, ce qui implique une perte d'ordre en espace au niveau discret si on utilise des éléments finis n'autorisant pas de discontinuité dans une maille. C'est par exemple le cas avec une méthode de pénalisation classique ou bien une méthode duale avec des multiplicateurs de Lagrange classiques et des éléments finis d'ordre 1. La solution sur le domaine tout entier comporte dans ce cas un saut des dérivées normales à travers le bord des particules qui n'est pas bien capté par ce type d'éléments finis. Une méthode pour contourner ce problème consiste à trouver une base d'éléments

finis autorisant les discontinuités des dérivées normales dans une maille. Une autre revient à chercher un prolongement régulier de la solution exacte définie sur le domaine fluide uniquement. C'est cette seconde approche que l'on a choisie afin de ne pas avoir de traitement particulier sur certaines mailles.

– Un dernier point est la façon dont les opérateurs du système de Stokes sont modifiés pour prendre en compte la géométrie du problème qui n'est plus vue par le maillage non conforme. On cherche à ne pas perturber ces opérateurs afin de toujours avoir la possibilité d'utiliser des solveurs rapides mais aussi pour qu'ils ne dépendent pas de la position des particules. Ceci permet de construire les matrices du système de Stokes une seule fois au début de la simulation et de les réutiliser lorsqu'on en a besoin. On peut donc effectuer des factorisations de manière hors ligne par exemple et les charger au début de la simulation pour gagner du temps. De plus on ne dégrade pas le conditionnement des matrices obtenu, on ne ralentit pas de ce fait la convergence d'un solveur itératif comme un *minres* ou un *gmres* par exemple. Un autre avantage est que l'on peut utiliser des préconditionneurs classique pour le problème de Stokes. La géométrie du problème sera vue dans le second membre de l'équation de Stokes.

La méthode présentée ici est une méthode éléments finis qui respecte les trois points énoncés ci-dessus. Ceci en fait donc une méthode d'ordre optimal et potentiellement rapide de part l'utilisation de solveurs comme la transformée de Fourier rapide ou bien des multigrilles géométriques. De plus, les opérateurs du problème de Stokes ne sont pas modifiés, seul le second membre change, ce qui en fait une méthode facile d'implémentation et robuste, pouvant résoudre non seulement des problèmes fluide/particules rigides mais qui peut s'adapter pour résoudre des problèmes avec des conditions différentes.

Travail effectué

La première partie de cette thèse est consacrée à la présentation du type de problème fluide/particules rigides qui nous intéresse et de la méthode que l'on a développée. Le chapitre 1 présente donc tout d'abord le système d'équations ainsi qu'un état de l'art des méthodes existantes permettant de résoudre des problèmes du même type. On s'attache à présenter les avantages et inconvénients de ces méthodes afin de montrer en quoi notre méthode diffère et ce qu'elle apporte. On présente notamment dans cet état de l'art des courbes d'erreurs en espace sur les méthodes rapidement implémentable dans un logiciel de résolution d'EDP. Le chapitre 2 concerne la méthode que l'on a développée. Tout d'abord sur un problème scalaire pour comprendre les différentes étapes de la méthode, puis sur le problème fluide/particules qui nous intéresse. La difficulté étant de prendre en compte la contrainte de mouvement rigide, on choisit de résoudre un problème non contraint défini sur tout le domaine et de contrôler la valeur du champ de vitesse sur le bord des particules par le prolongement du second membre dans les particules. On montre alors qu'il existe bien un prolongement permettant de contrôler la valeur du champ de vitesse sur le bord et on présente la technique utilisée pour trouver un tel prolongement. Le chapitre 3 fait l'objet d'un article soumis qui consiste en l'analyse numérique de l'approximation d'une distribution simple couche par une combinaison de masse de Dirac. En

effet, une des étapes de notre méthode consiste en la résolution d'un problème de Stokes non contraint dont le terme source est une distribution simple couche. On démontre tout d'abord des lemmes d'approximations qui nous sont utiles dans la preuve de la proposition énonçant le résultat sur l'erreur d'une telle approximation. On ilustre le résultat obtenu avec deux exemples d'applications sur des problèmes scalaires. Le premier est un problème de Poisson faisant intervenir une distribution simple couche dans le second membre et le second est un problème de Poisson sur un maillage perforé. La prise en compte des conditions de Dirichlet sur le bord de l'inclusion est réalisée à l'aide de multiplicateurs de Lagrange définis sur le bord. On montre un résultat sur l'erreur lorsque l'on approche ces multiplicateurs de Lagrange par une combinaison de masses de Dirac. Enfin, on termine ce chapitre par des cas tests visant à valider le résultat théorique énoncé. Dans le chapitre 4, on s'intéresse à la validation de la méthode au moyen de petits cas tests. On vérifie tout d'abord sur un problème scalaire que l'on obtient l'ordre optimal en espace. On réalise ensuite un cas test avec le problème fluide/particules qui nous intéresse. On place une particule rigide dans un champ de vitesse cisaillé en dimension 2. La particule est placée au centre du domaine carré si bien qu'elle tourne sur elle même sans se déplacer. La vitesse de rotation est connue expérimentalement, on peut donc la comparer avec celle trouvée par la simulation.

La deuxième partie se concentre sur le solveur de Stokes et la description du code *C* écrit permettant de résoudre numériquement le problème qui nous intéresse. Le chapitre 5 concerne donc le système obtenu après discrétisation des équations de Stokes. On présente des résultats sur la convergence de la méthode *minres* utilisée pour résoudre le système linéaire en fonction du préconditionneur utilisé. On montre en particulier que les préconditionneurs utilisés sont de bons préconditionneurs au sens où ils sont relativement rapide à inverser tout en gardant une convergence de la méthode en un nombre d'itération indépendant du pas du maillage. C'est une propriété essentielle pour ne pas voir le nombre d'itérations, permettant d'atteindre une tolérance fixée, exploser lorsque le pas du maillage devient petit. Le chapitre 6 consiste en une description de certaines parties du code *C* que l'on a écrit pour résoudre le problème avec la méthode développée. C'est un code qui utilise la librairie *Petsc*, parallèle et fonctionnant en deux et trois dimensions pour des particules rigides passives. On explique le code permettant de créer le maillage ainsi que celui de la création des différentes matrices et des solveurs. Cette partie est parallélisée grâce à Petsc mais toute la partie sur la gestion des particules rigides doit être adaptée en fonction. Les particules rigides pouvant se trouver sur plusieurs domaines appartenant à des processus différents, une attention particulière doit être prise pour éviter de réaliser trop de communications entre ces processus qui ralentiraient l'execution du code. Pour gérer les contacts rigides, nous avons couplé le code avec un logiciel existant, *SCoPI*, développé dans [48]. On décrit rapidement cet algorithme de contact et on explique la manière dont le couplage a été réalisé entre les deux codes.

La troisième partie concerne une application à la nage de micro-organisme et à la présentation de résultats en deux et trois dimensions. Le chapitre 7 décrit quelques simulations numériques en deux et trois dimensions. En deux dimensions, on présente un cas de sédimentation de 1008 particules régulièrement disposées dans le carré unité ainsi que l'évolution d'un tas de sable immergé dans le cas où le fluide est soumis à un cisaillement. Pour la simulation en trois dimensions, c'est un cas de sédimentation de 2000

particules dans le cube unité. Dans le chapitre 8, on regarde tout d'abord deux modèles de bactéries auto-propulsées qui consiste en une sphère rigide représentant la bactérie et en une ou deux forces, suivant le modèle utilisé, permettant à la bactérie de se déplacer. Ces deux modèles ajoutent donc des forces dans les équations de Stokes que l'on résout qui interviennent dans le second membre. On donne une description de la classe ajouté dans le code afin de gérer ces nouvelles forces. On réalise alors quelques simulations numériques sur le comportement de ces bactéries lorsqu'elles sont isolées ou en interaction avec d'autres micro-organismes. On présente ensuite un modèle de *squirmer* permettant de simuler, entre autres, le déplacement d'une goutte, immergée dans une solution contenant un surfactant, et se déplaçant sous l'effet Marangoni. C'est aussi un modèle pour certains micro-organismes qui se déplacent à l'aide de petits cils situés sur leur surface. Le modèle consiste cette fois-ci en la prescription de la vitesse sur le bord de la particule dans son référentiel. La condition de bord sur les particules dans les équations de Stokes que l'on résout n'est donc plus une condition de mouvement rigide mais on cherche à imposer la vitesse du fluide comme la somme de la vitesse dans le référentiel de la particule (qui est connue) et un mouvement rigide. On décrit alors les changements à effectuer dans le code pour pouvoir appliquer notre méthode et prendre en compte cette nouvelle contrainte.

Enfin le dernier chapitre concerne quelques perspectives sur la méthode. Un des inconvénient est qu'elle demande de résoudre beaucoup de problèmes de Stokes classiques. On suggère une approche par base réduite qui permettrait, pour un problème donné, de précalculer des contrôles de bases sur lesquels on exprimerait le contrôle recherché. Nous parlons aussi du passage aux équations de Navier-Stokes. La densité des particules pouvant être différentes de celle du fluide, on perd la propriété des matrices indépendantes de la position des particules. Une analyse numérique plus poussée, dans le cas des équations de Stokes, reste aussi à réaliser.

Première partie

Simulation directe d'un écoulement fluide/particules rigides par une méthode de type domaine fictif d'ordre optimal

Chapitre 1

Présentation du problème

Sommaire

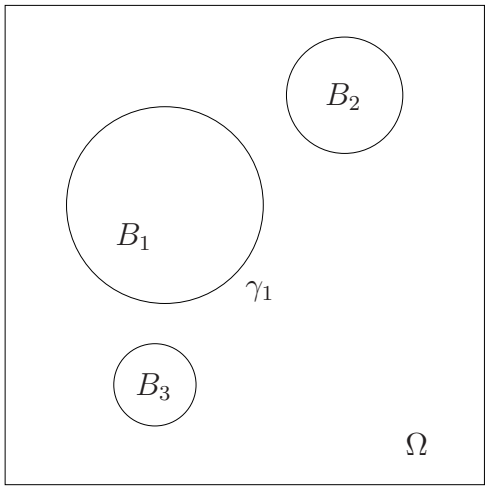

FIGURE 1.1 – Exemple de domaine considéré

1.1 Modélisation

Dans cette partie nous présentons le problème type que l'on souhaite résoudre. On considère un domaine rectangulaire que l'on note Ω et une collection de N particules sphériques inclues dans le domaine Ω. La i^{eme} particule est notée B_i avec $i = 1, \dots, N$ et la réunion des particules est notée B. Pour une configuration des particules rigides donnée, on souhaite trouver le champ de vitesse du fluide dans tout le domaine fluide $\Omega \setminus \bar{B}$. Le problème initial que l'on se pose est le suivant,

$$(1.1)\quad \begin{cases} -2\eta \operatorname{div}(D(\mathbf{u})) + \nabla p &= \rho_f \mathbf{f}\,, & \text{dans } \Omega \setminus \bar{B}\,, \\ \nabla \cdot \mathbf{u} &= 0\,, & \text{dans } \Omega \setminus \bar{B}\,, \\ \mathbf{u} &= 0\,, & \text{sur } \partial\Omega\,, \\ \mathbf{u} &= \mathbf{V}_i + \boldsymbol{\omega}_i \wedge \mathbf{r}_i\,, & \text{sur } \gamma_i = \partial B_i\,, \\ \displaystyle\int_{\gamma_i} \sigma(\mathbf{u})\mathbf{n}_i &= \mathbf{F}_i\,, \\ \displaystyle\int_{\gamma_i} \mathbf{r}_i \wedge \sigma(\mathbf{u})\mathbf{n}_i &= \mathbf{T}_i\,. \end{cases}$$

On note $\mathbf{u}$ la vitesse du fluide et p sa pression. On note $D(\mathbf{u})$ le tenseur des déformations,

$$D(\mathbf{u}) = \frac{\nabla \mathbf{u} + {}^t\nabla \mathbf{u}}{2}. \tag{1.2}$$

Le paramètre η est la viscosité dynamique du fluide et ρ_f est sa masse volumique. Le vecteur $\mathbf{f}$ est l'ensemble des forces massiques appliquées au fluide, comme la gravité si on veut prendre en compte la pression hydrostatique. La première équation est l'équation de la quantité de mouvement et la seconde est l'équation de conservation de la masse qui traduit l'hypothèse que le fluide est incompressible.

Les inconnues $\mathbf{V}_i$ et $\boldsymbol{\omega}_i$ sont respectivement la vitesse de translation et de rotation de la i^{eme} particule et le vecteur $\mathbf{r}_i$ est le vecteur radial. On considère que le fluide adhère aux

parois, sa vitesse est donc égale à la vitesse des particules sur leurs bords. L'expression de cette condition de bord traduit le fait que l'on considère des particules rigides (qui ne se déforment pas). Une particule qui ne se déforme pas ne peut que se translater et tourner sur elle même, d'où l'expression de cette condition de bord.

Dans les deux dernières équations la matrice $\sigma(\mathbf{u})$ est le tenseur des contraintes de Cauchy et s'écrit,

$$\sigma(\mathbf{u}) = \eta\left(\nabla\mathbf{u} + {}^t\nabla\mathbf{u}\right) - p\, I_d = 2\eta D(\mathbf{u}) - pI_d\,, \tag{1.3}$$

où I_d est la matrice identité en dimension d ($d = 2$ ou 3). Le vecteur $\mathbf{n_i}$ est la normale sortante du domaine fluide sur γ_i. Ainsi, la résultante des forces exercées par le fluide sur la particule s'écrit,

$$\mathbf{F}_{fluide\to particule} = -\int_{\gamma_i} \sigma(\mathbf{u})\mathbf{n}_i\,. \tag{1.4}$$

et le moment dû au fluide appliqué à la particule est :

$$\mathbf{T}_{fluide\to particule} = -\int_{\gamma_i} \mathbf{r}\wedge\sigma(\mathbf{u})\mathbf{n}_i\,. \tag{1.5}$$

Le Principe Foncamental de la Dynamique appliqué aux particules donne alors les équations suivantes sur chaque particule :

$$m_i\frac{d\mathbf{V}_i}{dt} = \mathbf{F}_i - \int_{\gamma_i}\sigma(\mathbf{u})\mathbf{n}_i\,, \tag{1.6}$$

$$\mathbf{I}_i\frac{d\boldsymbol{\omega}_i}{dt} + \boldsymbol{\omega}_i\wedge\mathbf{I}_i\boldsymbol{\omega}_i = \mathbf{T}_i - \int_{\gamma_i}\mathbf{r}\wedge\sigma(\mathbf{u})\mathbf{n}_i\,, \tag{1.7}$$

où $\mathbf{I}_i$ est le tenseur d'inertie et est donné par :

$$(\mathbf{I}_i)_{j,k} = \rho_s\int_{B_i} x_j x_k \quad \text{pour } j\neq k, 1\leq j,k\leq d\,,$$
$$(\mathbf{I}_i)_{j,j} = \rho_s\int_{B_i}\sum_{1\leq k\neq j\leq d} x_k^2 \quad \text{pour } 1\leq j\leq d\,.$$

Dans notre cas, on ne considère que des particules sphériques, on a donc les relations suivantes pour les termes extra diagonaux de $\mathbf{I}_i$:

$$(\mathbf{I}_i)_{j,k} = 0 \quad \text{pour } j\neq k\,,$$

et pour les termes diagonaux :

$$(\mathbf{I}_i)_{j,j} = \rho_s\frac{d-1}{d}\int_{B_i}|\mathbf{r}|^2\,.$$

Les tenseurs $\mathbf{I}_i$ sont donc des matrices identité à une constante multiplicative près :

$$\mathbf{I}_i = \rho_s\frac{d-1}{d}\left(\int_{B_i}|\mathbf{r}|^2\right) I_d\,.$$

Le terme non linéaire $\boldsymbol{\omega}_i \wedge \mathbf{I}_i\boldsymbol{\omega}_i$ de l'équation 1.7 est donc nul. De plus, comme on se place dans le contexte des équations de Stokes stationnaires, on considère que l'équilibre est atteint instantanément. Les dérivées en temps dans les équations 1.6 et 1.7 sont donc nulles. Le Principe Fondamental de la Dynamique donne finalement les relations suivantes :

$$\mathbf{F}_i - \int_{\gamma_i} \sigma(\mathbf{u})\mathbf{n}_i = 0\,,$$
$$\mathbf{T}_i - \int_{\gamma_i} \mathbf{r}_i \wedge \sigma(\mathbf{u})\mathbf{n}_i = 0\,,$$

qui sont exactement les deux dernières équations du problème 1.1.

Les inconnues du problème sont donc la vitesse $\mathbf{u}$ et la pression p du fluide ainsi que les N vitesses de translation $\mathbf{V}_i$ et les N vitesses de rotation $\boldsymbol{\omega}_i$.

1.2 État de l'art

On présente différentes méthodes utilisées pour résoudre ce genre de problème. La difficulté dans la résolution du système 1.1 est la prise en compte de la condition de bord sur les particules du fait de la géométrie complexe du domaine lorsqu'il contient un grand nombre d'inclusions rigides. En effet, dans ce cas, mailler le domaine fluide de façon conforme pour résoudre le problème peut s'avérer long et difficile lorsque les particules (représentées par des trous dans le maillage) sont proches les unes des autres. C'est encore plus le cas pour des simulations en trois dimensions avec un grand nombre de particules. De plus une attention particulière doit être porté à la génération du maillage dans le cas de conditions périodiques. Afin d'éviter les difficultés rencontrées pour la génération de maillage, de nombreuse méthodes ont été développées sur maillage non conforme. La géométrie du domaine est alors prise en compte d'une manière différente en modifiant, par exemple, les équations à résoudre. Pour ces méthodes, on va se concentrer sur deux difficultés. La première est l'ordre de l'erreur en espace qui peut être dégradé car la frontière est mal approchée avec un maillage non conforme. La seconde concerne la façon dont les opérateurs sont modifiés. Le conditionnement des matrices peut en effet être dégradé, ce qui ralenti la convergence des solveurs ou bien empêche l'utilisation de solveurs rapides.

1.2.1 Méthode sur maillage conforme

Dans le cas d'un maillage conforme, une formulation ALE (*Arbitrary Lagrangian Eulerian*) permet de réduire le nombre de remaillages car elle permet dans le cas de petites déformations d'adapter le maillage au lieu de le reconstruire entièrement. Dans [41] une formulation ALE est utilisée pour simuler le mouvement de particule rigide immergées en deux dimensions. On retrouve ce type de méthode dans [55] dans le cas d'un domaine bipériodique. Les auteurs de [49] ont utilisé une formulation ALE pour calculer la viscosité effective d'une suspension de particules passives en dimension deux. Takahashi et ses collaborateurs ont étudié la convergence du problème de particules rigides immergées

dans le contexte éléments finis avec une formulation ALE dans [54] pour l'équation de Stokes et [50] pour Navier-Stokes.

Codina et ses collaborateurs ont développé dans [25] une méthode type ALE sur maillage fixe qui consiste à utiliser une formulation ALE et à remailler à chaque pas de temps sur un maillage fixe en découpant les cellules intersectées par la frontière mobile. Dans le même esprit, Ilinca et Hétu dans [43] développent une méthode, non basée sur une formulation ALE, sur maillage fixe où cette fois encore les cellules sont découpées. Dans leurs travaux, Ilinca et Hétu ne considèrent cependant que des conditions de Dirichlet homogène et les degrés de liberté ajoutés sont enlevés des matrices. Dans le contexte volumes finis, la méthode *Cartesian Cut Cell* (voir [44]) suit le même principe et adapte un maillage cartésien en découpant les cellules du maillage intersectées par l'interface pour qu'il devienne conforme. Le problème de ces méthodes est l'apparition possible de petites cellules qui peut contraindre le pas de temps utilisé et dégrader le conditionnement des matrices. Des méthodes pour recoller plusieurs cellules sont alors utilisées (voir les références dans [44]) afin d'empêcher la formation de ces petites cellules. De plus, l'ajout de noeuds à l'interface pour découper les cellules engendre des modifications sur les matrices à chaque pas de temps lorsque l'interface est mobile.

1.2.2 Méthode de domaine fictif et multiplicateurs de Lagrange

De nombreuses méthodes sur maillage fixe, non conforme ont été développées pour ne pas avoir à gérer le maillage. C'est le cas des méthodes de domaine fictif où le domaine d'intérêt est plongé dans un domaine plus grand avec une géométrie plus simple. Dans un contexte éléments finis, Glowinski et ses collaborateurs ont beaucoup travaillé sur ces méthodes pour la simulation du mouvement de particules rigides immergées en imposant la contrainte de mouvement rigide avec des multiplicateurs de Lagrange. Pour illustrer cette méthode sur un cas simple, on prend le problème jouet suivant :

$$(1.8)\quad \begin{cases} -\Delta u &= f \quad \text{dans } \Omega \setminus \bar{B}\,, \\ u &= 0 \quad \text{sur } \partial\left(\Omega \setminus \bar{B}\right)\,, \end{cases}$$

où f est une fonction L^2, Ω le carré unité et B une boule inclue dans Ω. La formulation variationnelle de ce problème est :

Trouver $u \in H_0^1(\Omega \setminus \bar{B})$ telle que :

$$\int_{\Omega\setminus\bar{B}} \nabla u \cdot \nabla v = \int_{\Omega\setminus\bar{B}} fv \quad \forall v \in H_0^1(\Omega \setminus \bar{B})\,. \tag{1.9}$$

Le domaine fictif est le domaine Ω tout entier et le problème est alors d'imposer la condition de Dirichlet sur l'interface ∂B. L'idée ici est d'étendre le terme source f par une fonction valant zéro dans B (que l'on note toujours f) et d'étendre la solution u en une fonction valant 0 dans B. Ces prolongements permettent alors de travailler avec des intégrales définies sur tout le domaine fictif Ω. La formulation 1.9 devient alors,

Trouver $u \in V$ telle que :

$$\int_{\Omega} \nabla u \cdot \nabla v = \int_{\Omega} fv \quad \forall v \in V\,, \tag{1.10}$$

où V est l'espace,

$$V = \left\{ v \in H_0^1(\Omega) : v_{|\partial B} = 0 \right\} \ .$$

Si un multiplicateur de Lagrange défini sur le bord ∂B est utilisé pour relaxer la contrainte de l'espace V, une formulation possible est :

Trouver $(u, \lambda) \in H_0^1(\Omega) \times L^2(\partial B)$ telles que :

$$\begin{aligned} \int_\Omega \nabla u \cdot \nabla v - \int_{\partial B} \lambda v &= \int_\Omega f v \quad \forall v \in H_0^1(\Omega) \, , \\ \int_{\partial B} \mu u &= 0 \quad \forall \mu \in L^2(\partial B) \ . \end{aligned} \tag{1.11}$$

Dans le cas d'un multiplicateur de Lagrange distribué (défini en volume), une formulation possible est :

Trouver $(u, \lambda) \in H_0^1(\Omega) \times H^1(B)$ telles que :

$$\begin{aligned} \int_\Omega \nabla u \cdot \nabla v - \langle \lambda, v \rangle &= \int_\Omega f v \quad \forall v \in H_0^1(\Omega) \, , \\ \langle \mu, u \rangle &= 0 \quad \forall \mu \in H^1(B) \, , \end{aligned} \tag{1.12}$$

où $< \, . \, , \, . \, >$ dénote un produit scalaire dans $H^1(B)$. On ne parlera pas de l'existence et de l'unicité de ces solutions, on peut trouver ces résultats dans les références qui suivent. Dans [36], les auteurs simulent, entre autre, l'écoulement de l'air autour d'une aile d'avion fixe en résolvant les équations de Navier-Stokes incompressibles avec condition de Dirichlet. Les conditions de Dirichlet sur le bord de l'aile sont imposées à l'aide d'un multiplicateur de Lagrange défini sur le bord. Dans [37] c'est le mouvement de particules rigides immergées qui est simulé où le mouvement rigide est supposé connu *a priori*. Ici encore un multiplicateur de Lagrange de bord est utilisé pour imposer le mouvement rigide. Le mouvement de particules rigides immergées dans un fluide incompressible est simulé dans [35] où la contrainte de mouvement rigide est imposée cette fois-ci par un multiplicateur de Lagrange distribué (défini en volume dans les particules). Dans [31] Girault et Glowinski ont fait l'analyse numérique de la méthode de domaine fictif avec multiplicateur de Lagrange de bord dans le cas d'un problème avec condition de Dirichlet. La même analyse dans le cas des équations de Navier-Stokes incompressibles stationnaires est faite dans [32]. Dans le cas d'un multiplicateur de Lagrange distribué, on trouvera une analyse de Tomas dans [71]. Dans tous les cas, l'erreur en espace de ces méthodes de domaine fictif est d'ordre $1/2 - \epsilon$ pour tout ϵ positif en norme H^1 pour des éléments finis d'ordre 1 (voir [65] pour des éléments finis Q_1). On perd donc l'ordre optimal de l'erreur qui est d'ordre 1 dans le cas d'un maillage conforme. Cette perte d'ordre est due au fait que la solution obtenue n'est pas un prolongement régulier de la solution sur le domaine initial. Avec ces méthodes, la solution du problème initial est en effet prolongée par le mouvement rigide dans les particules donc un saut des dérivées normales aux bords apparaît.

Nous avons implémenté une méthode de domaine fictif avec multiplicateur de Lagrange de bord dans le logiciel *FreeFem++* afin de calculer l'ordre de l'erreur en espace sur le cas

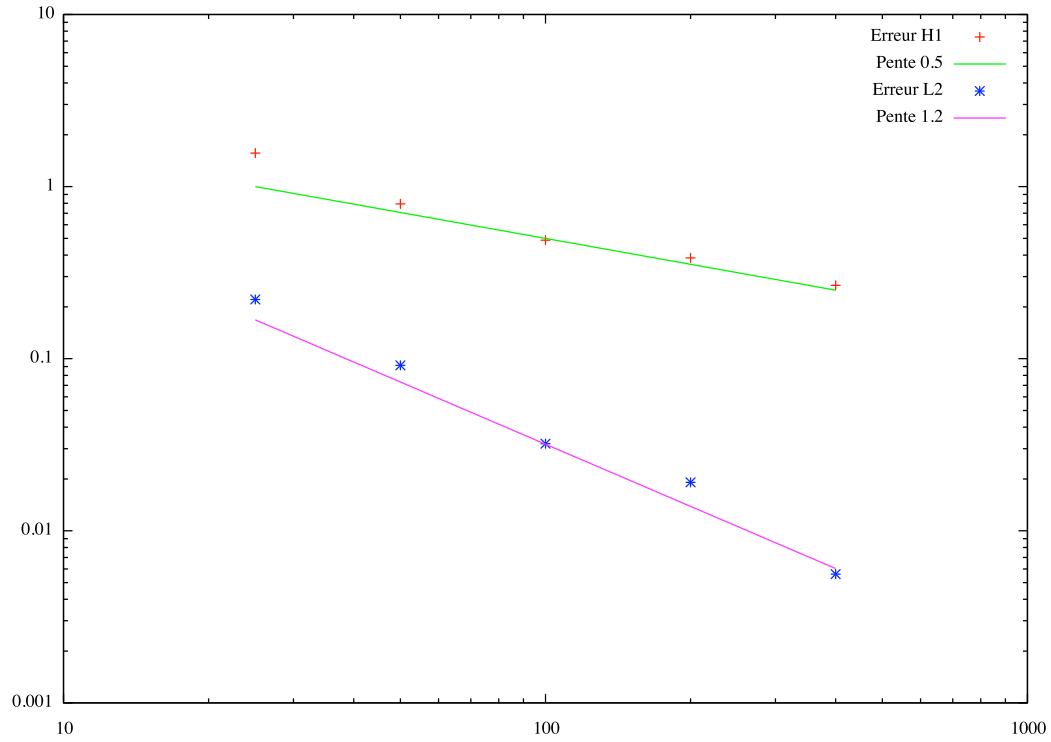

FIGURE 1.2 – Erreur en norme H^1 et L^2 avec des multiplicateur de Lagrange

test suivant dont on connaît la solution exacte :

$$(1.13)\quad \begin{cases} -\Delta u &= 0\,, & \text{dans } \Omega \setminus \bar{\mathcal{O}}\,, \\ u &= \log\left(\frac{(x-x_c)^2+(y-y_c)^2}{R^2}\right)\,, & \text{sur } \partial\Omega\,, \\ u &= 0\,, & \text{sur } \partial\mathcal{O}\,, \end{cases}$$

où (x_c, y_c) sont les coordonnées du centre de la boule $\mathcal{O}$ et R son rayon. La solution exacte de ce problème est :

$$u(x,y) = \log\left(\frac{(x-x_c)^2+(y-y_c)^2}{R^2}\right)\,.$$

On calculera l'erreur en espace en interpolant la solution calculée et la solution exacte sur un maillage très fin et conforme à la géométrie de $\Omega \setminus \bar{\mathcal{O}}$. La figure 1.2 représente les courbes d'erreurs pour les normes L^2 et H^1. L'ordre obtenu numériquement est légèrement meilleur que l'ordre théorique ce qui n'est pas surprenant. Les figures 1.3 et 1.4 représentent respectivement une coupe horizontale de la solution passant par le centre de l'inclusion et les lignes de niveaux de la solution. Dans les deux figures on voit bien le saut des dérivées sur le bord de l'inclusion.

1.2.3 Méthode de domaine fictif et pénalisation

Une autre façon d'imposer une contrainte avec une méthode de domaine fictif est d'utiliser une méthode de pénalisation. Dans ce cas, la formulation 1.10 du problème

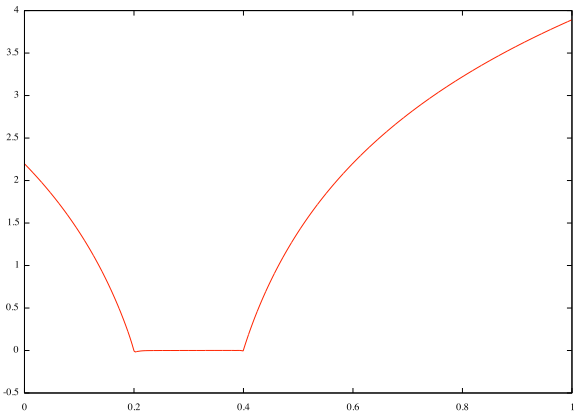

FIGURE 1.3 – Coupe de la solution de 1.13 avec des multiplicateurs de Lagrange

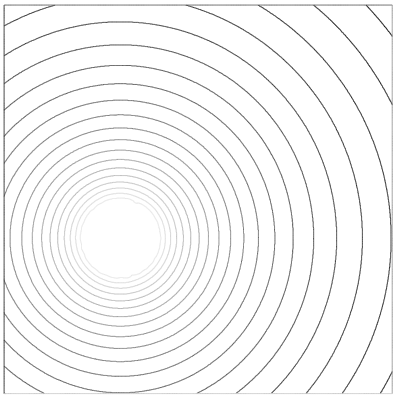

FIGURE 1.4 – Lignes de niveaux de la solution de 1.13 avec des multiplicateurs de Lagrange

jouet 1.8 devient par exemple,

Trouver $u_\epsilon \in H^1_0(\Omega)$ telle que

$$\int_\Omega \nabla u_\epsilon \cdot \nabla v + \frac{1}{\epsilon}\int_B u_\epsilon v = \int_\Omega fv \quad \forall v \in H^1_0(\Omega)\,. \tag{1.14}$$

Ce problème variationnel est équivalent à résoudre le problème de minimisation suivant :

$$\text{Trouver} \qquad u_\epsilon = \underset{v\in H^1_0(\Omega)}{\operatorname{argmin}} \quad \frac{1}{2}\int_\Omega |\nabla v|^2 + \frac{1}{2\epsilon}\int_B |v|^2 - \int_\Omega fv\,. \tag{1.15}$$

L'idée de la méthode est donc de prendre un ϵ suffisament petit de sorte que u_ϵ soit forcée à être petite pour minimiser l'intégrale sur la boule B. Les résultats de convergence sont énoncés et prouvés dans les références qui suivent. Dans [46], les auteurs imposent par pénalisation la contrainte de mouvement rigide en imposant au tenseur des déformations d'être nul dans les particules. Une analyse rigoureuse de la méthode de pénalisation est réalisée par dans [57]. Avec cette méthode, tout comme avec un multiplicateur de Lagrange, il est montré que l'erreur en espace est d'ordre 1/2 en norme H^1. On retrouve en effet une discontinuité dans les dérivées normales au niveau du bord des particules qui n'est pas bien approchée par le maillage non conforme. De plus, les matrices sont modifiées avec l'ajout du terme de pénalisation qui dégrade le conditionnement. En revanche, la méthode est relativement simple à implémenter si l'on dispose d'un solveur classique. Nous avons implémenté cette méthode avec le logiciel *FreeFem++* et nous avons résolu le problème 1.13. La figure 1.5 représente les courbes d'erreur obtenues, on retrouve bien un ordre 1/2 en norme H^1 et un ordre 1 en norme L^2. La figure 1.6 représente les lignes de niveau de la solution obtenues dans le cas d'un maillage assez fin et la figure 1.7 représente une coupe horizontale de la solution passant par le centre de la boule. On peut observer très clairement le saut des dérivées normales à travers le bord de la boule $\mathcal{O}$ et la solution prolongée par 0 dedans.

Dans [23], les auteurs utilisent une méthode de pénalisation pour imposer une condition de Dirichlet homogène. Au lieu de contraindre la solution à être nulle, ils lui imposent d'être égale à un prolongement régulier de la solution dans la particule (qui est lui nul sur le bord) qui est construit en même temps que la solution par un processus itératif. Ceci permet d'avoir la continuité des dérivées normales sur le bord des particules et de retrouver l'ordre optimal. Cette méthode appliquée au problème jouet 1.8 consiste à trouver u_ϵ solution du problème pénalisé suivant :

$$-\Delta u_\epsilon + \frac{1}{\epsilon}(u_\epsilon - v_\epsilon)\chi_B = f\,,$$

où v_ϵ est un prolongement régulier de la solution exacte qu'il reste à construire. Le processus itératif pour construire ce prolongement est constitué de trois étapes. Tout d'abord, on repère l'interface ∂B par une fonction level-set ϕ qui est la fonction distance à ∂B, négative dans B et positive en dehors. On calcule alors la solution pénalisée classique solution de :

$$-\Delta u_\epsilon^{(0)} + \frac{1}{\epsilon}u_\epsilon^{(0)}H(\phi) = f\,,$$

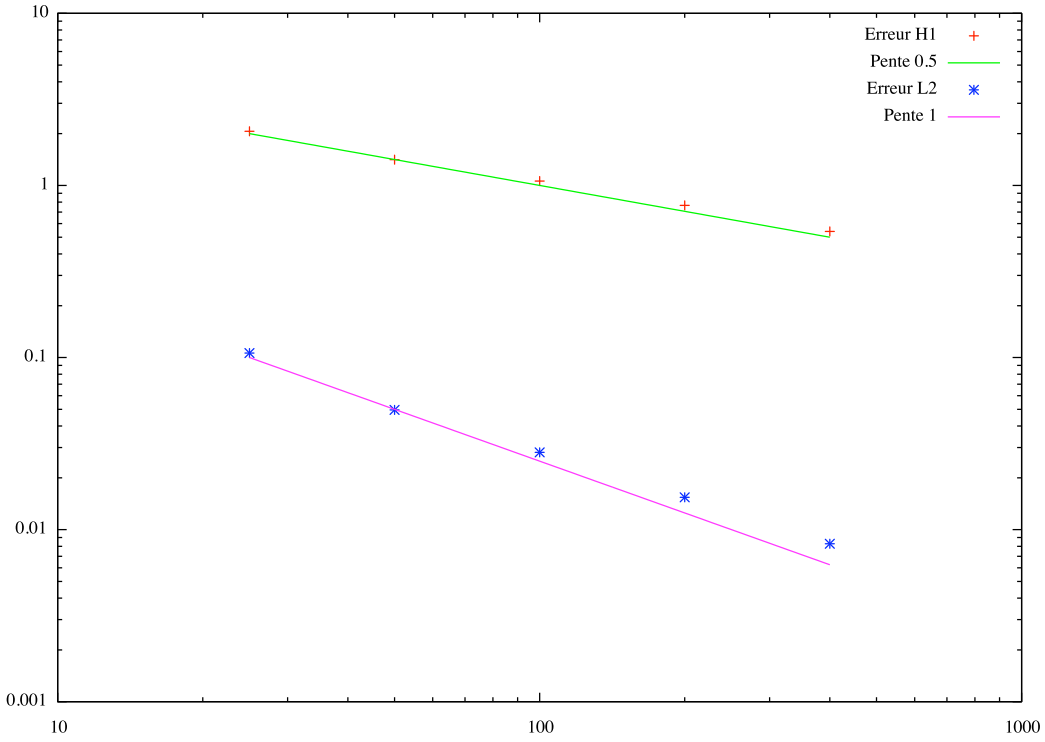

FIGURE 1.5 – Erreur en norme H^1 et L^2 pour la méthode de pénalisation

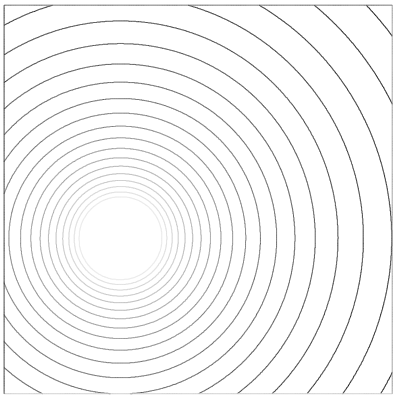

FIGURE 1.6 – Ligne de niveau de la solution de 1.13 avec une méthode de pénalisation

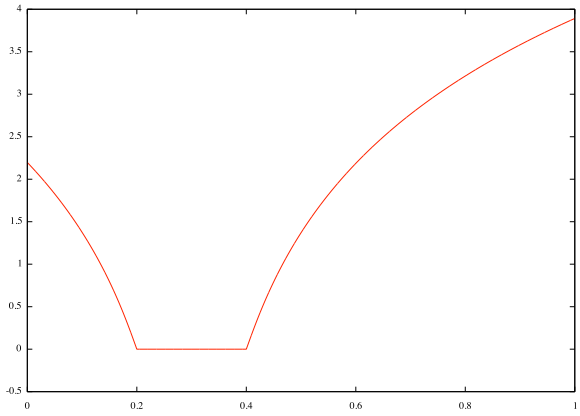

FIGURE 1.7 – Coupe de la solution de 1.13 avec une méthode de pénalisation

où H est la fonction de Heaviside, puis on cherche v_ϵ sous la forme $\alpha\phi$, avec α une fonction scalaire qui approche les dérivées normales sur ∂B de la solution exacte prolongée, en suivant les étapes suivantes :

– étape 1 : On calcule α_k de la façon suivante :

$$\alpha_k = \nabla v_\epsilon^{(k)} \cdot \frac{\nabla\phi}{|\nabla\phi|}.$$

– étape 2 : Les valeurs de α_k sont prolongées dans B en réalisant quelques itérations de l'équation d'advection :

$$\partial_t \alpha_k + H(\phi+\delta)\frac{\nabla\phi}{|\nabla\phi|} \cdot \nabla\alpha_k = 0,$$

où δ est un paramètre petit qui vaut la moitié du pas du maillage par exemple.

– étape 3 : On pose $v_\epsilon^{(k)} = \alpha_k\phi$ et on résout le problème pénalisé suivant :

$$-\Delta u_\epsilon^{(k+1)} + \frac{1}{\epsilon}H(\phi)\left(u_\epsilon^{(k+1)} - v_\epsilon^{(k+1)}\right) = f.$$

On répète alors les étapes jusqu'à convergence. C'est une méthode qui permet d'obtenir une solution sans saut des dérivées normales à l'interface mais qui reste une méthode de pénalisation, les matrices sont donc modifiées par l'ajout des termes en $1/\epsilon$.

Dans un contexte de volume finis, Angot et ses collaborateurs imposent, entre autres, des conditions de Dirichlet par pénalisation en raffinant le maillage au niveau des interfaces. Ainsi, même si l'ordre n'est pas optimal, le but est de raffiner suffisamment le maillage sur les bords pour obtenir une erreur équivalente (voir [66]). Dans cette optique de raffinement de maillage, on trouve un algorithme dans [14] dans le cas des éléments finis. Le maillage n'est alors plus conforme si on utilise des mailles quadrangulaires et il est prouvé pour cet algorithme un ordre optimal dans le cas des équations de Stokes formulées sous la forme d'un problème point-selle.

1.2.4 Les méthodes dites de *frontières immergées*

Dans un contexte différence finies, Peskin a développé la méthode de frontière immergée (voir [62,63]). On considère une frontière immergée dans un fluide dont la vitesse est régie par les équations de Navier-Stokes. La méthode consiste à représenter la frontière par une collection de points Lagrangiens et d'approcher les forces singulières s'exerçant sur cette frontière comme une combinaison de masses de Dirac en ces points. La fontière se déplace donc en advectant la collection de points par la vitesse du fluide. Pour résoudre ce problème il est nécéssaire de pouvoir interpoler la vitesse en les points Lagrangiens représentant la frontière et de pouvoir transmettre les forces singulières sur la grille cartésienne du fluide. Comme les points de la frontière ne sont en général pas des points de la grille, Peskin utilise des fonctions régularisantes pour approcher les Diracs afin de connecter le fluide et la représentation de la frontière. Ces fonctions régularisantes étalent les Diracs sur quelques cellules de la grille du fluide et sont définies dans [62] par :

$$d(r) = \begin{cases} \dfrac{1}{4h}\left(1+\cos\left(\dfrac{\pi r}{2h}\right)\right) & \text{si } |r| < 2h \\ 0 & \text{sinon,} \end{cases}$$

où h représente le pas de la grille fluide. C'est une méthode d'ordre 1 en norme L^2 qui a donné naissance à beaucoup d'extensions et de variantes. Elle a notamment été adaptée au cas éléments finis dans [18] en une méthode permettant de gérer les Dirac de façon faible.

La méthode IIM (*Immersed Interface Method*) développée par Leveque et Li dans [51,52] permet de résoudre des problèmes avec des coefficients discontinus et des solutions comprenant des discontinuités. C'est une méthode en différences finies où le schéma est modifié au niveau des noeuds proches de l'interface de façon à prendre en compte le saut de la quantité discontinue. Il est démontré dans [42] que c'est une méthode d'ordre 2 en norme L^2. Il y a plusieurs applications au cas d'une particule rigide immergée : dans [73], Xu simule le champ de vitesse en 2D d'un fluide contenant un corps rigide dont le mouvement est imposé et dans [69] pour un fluide contenant des interfaces fixe ou mobile et des bords rigides.

1.2.5 Les formulations *à la Nitsche*

Revenons au contexte éléments finis avec les formulations dites *à la Nitsche* (voir [59]) qui ressemblent aux formulations par pénalisation. Pour illustrer ce type de formulation avec notre problème jouet, on défini l'espace suivant,

$$W = \left\{ v \in H^1(\Omega \setminus \bar{B}) : v_{|\partial\Omega} = 0 \right\}.$$

En multipliant l'équation aux dérivées partielles du problème jouet par une fonction test dans l'espace W, on obtient la formulation suivante,

$$\int_{\Omega\setminus\bar{B}} \nabla u \cdot \nabla v - \int_{\partial B} \nabla u \cdot n v = \int_{\Omega\setminus\bar{B}} f v \quad \forall v \in W\,. \tag{1.16}$$

L'idée est d'imposer la condition de Dirichlet sur ∂B avec l'équation suivante :

$$\int_{\partial B} uv = 0 \quad \forall v \in W\,. \tag{1.17}$$

On peut voir cette équation comme une formulation faible des conditions de Dirichlet. Ensuite pour symétriser la formulation obtenue en ajoutant (1.16) et (1.17), on considère l'équation suivante :

$$\int_{\partial B} u\nabla v \cdot n = 0 \quad \forall v \in W\,. \tag{1.18}$$

La formulation *à la Nitsche* est alors obtenue avec la somme de (1.16) $-$ (1.18) $+\gamma/h$ (1.17), où γ est un paramètre à fixer (pris suffisament grand pour imposer la condition de Dirichlet) et où h est le pas de maillage :

$$\int_{\Omega\setminus\bar{B}} \nabla u \cdot \nabla v - \int_{\partial B} \nabla u \cdot nv - \int_{\partial B} u\nabla v \cdot n + \frac{\gamma}{h}\int_{\partial B} uv = \int_{\Omega\setminus\bar{B}} fv \quad \forall v \in W. \tag{1.19}$$

Comme la formulation est symétrique, ce problème revient à chercher la solution du problème de minimisation suivant

$$u = \underset{v\in W}{\operatorname{argmin}} \quad J_1(v) + \frac{\gamma}{h}J_2(v)\,. \tag{1.20}$$

où les fonctionnelles J_1 et J_2 sont définies ainsi,

$$J_1(v) = \frac{1}{2}\int_{\Omega\setminus\bar{B}} |\nabla v|^2 - \int_{\partial B} \nabla v \cdot nv - \int_{\Omega\setminus\bar{B}} fv\,, \tag{1.21}$$

$$J_2(v) = \int_{\partial B} |v|^2\,. \tag{1.22}$$

On retrouve une formulation similaire à la formulation par pénalisation. Au niveau discret, il reste les intégrales à calculer sur des domaines complexes avec un maillage non conforme. Dans [39], les auteurs utilisent une formulation *à la Nitsche* et cherchent la solution sous la forme de la somme de deux fonctions définies dans chaque sous domaine. Pour faire le parallèle avec notre problème jouet 1.8, cela revient à chercher notre solution sous la forme,

$$u = u_{\Omega\setminus\bar{B}} + u_B \tag{1.23}$$

où $u_{\Omega\setminus\bar{B}}$ est dans l'espace W et u_B est dans l'espace $H^1(B)$. Dans notre cas, on peut prendre directement u_B nulle contrairement à ce qui est fait dans [39] où les auteurs résolvent un problème de la chaleur dans tout le domaine. Deux sous maillage du maillage global sont alors considérés : celui qui recouvre $\Omega \setminus \bar{B}$ et celui qui recouvre B. Les mailles qui sont intersectées par l'interface ∂B sont donc "doublées". Ceci permet de considérer des solutions discontinues à l'interface ∂B et de retrouver l'ordre optimal. Dans [39], les intégrales sont calculées à l'aide de formule de quadrature. Dans [8], une méthode similaire est utilisée mais avec une formulation *à la Nitsche* différente qui conduit à une formulation non symétrique mais qui permet de minimiser la différence entre la trace de la solution calculée par éléments finis et la condition de Dirichlet dans une certaine

norme. Deux cas sont traités : celui où la condition de Dirichlet est imposée en utilisant les noeuds du maillage à l'extérieur du domaine (dans B pour notre problème jouet) et celui où les noeuds utilisés sont ceux à l'intérieur du maillage appartenant à une maille intersectée par l'interface. Il est observé que cette méthode est d'ordre 2 en norme L^2 avec des éléments finis de degrés 1. Une formulation proche de la précédente est utilisée par les mêmes auteurs dans [9] qui est cette fois ci symétrique lorsque le problème est symétrique. Elle est illustrée par des applications au cas des équations de Stokes et de Navier-Stokes. Burman présente une méthode type pénalisation dans [21] permettant de controler la sensibilité du conditionnement des matrices par rapport à la position du maillage par rapport au domaine. Elle est utilisée dans [22] avec la méthode de [39] pour imposer de façon faible des conditions de type Dirichlet. L'ajout de ces termes de pénalisation ne dégrade pas l'ordre optimal obtenu précédemment.

1.2.6 La *Fat Boundary Method*

Toujours dans le contexte éléments finis, la *Fat Boundary Method* (FBM) développée dans [56] est une méthode sur maillage fixe permettant de conserver l'ordre optimal. La présentation de la méthode dans [56] est faite sur le même problème jouet 1.8 que le notre. Elle consiste à considérer un petit domaine local ω autour de l'inclusion (un anneau autour de l'inclusion sphérique par exemple) et à résoudre deux problèmes couplés au lieu du problème initial 1.8 :

$$\left\{\begin{array}{ll} (a): & \left\{\begin{array}{rcll} -\Delta v &=& f\,, & \text{dans } \omega\,,\\ v &=& \hat{u}\,, & \text{sur } \gamma'\,,\\ v &=& 0\,, & \text{sur } \gamma\,, \end{array}\right.\\[2ex] (b): & \left\{\begin{array}{rcll} -\Delta \hat{u} &=& f + \left.\dfrac{\partial v}{\partial n}\right|_{\gamma} \delta_\gamma\,, & \text{dans } \Omega\,,\\ \hat{u} &=& 0\,, & \text{sur } \partial\Omega\,, \end{array}\right. \end{array}\right.$$

où f a été prolongée par zéro dans B comme pour la méthode de domaine fictif et où γ et γ' sont les deux bord de ω avec $\gamma = \partial B$. L'expression $\left.\frac{\partial v}{\partial n}\right|_\gamma \delta_\gamma$ représente la forme linéaire dans $H^{-1}(\Omega)$ définie par :

$$\varphi \in H_0^1(\Omega) \longrightarrow \int_\gamma \frac{\partial v}{\partial n}\varphi\,.$$

Le problème (a) est résolu sur un maillage conforme et est localisé sur une petite couronne autour de ω. Si $(\hat{u}, v)$ est solution du système, alors la restriction de $\hat{u}$ à ω est égale à la fonction v. Le problème (a) permet de transmettre au problème (b) le bon saut des dérivées normales de la solution à travers le bord γ, la fonction $\hat{u}$ étant en effet nulle dans le domaine B. Par une méthode de point fixe on peut alors résoudre ce système d'équations et une solution du problème initial 1.8 est donnée par la restriction de $\hat{u}$ au domaine $\Omega \setminus \bar{B}$. Une analyse complète de la FBM à été faite dans [15]. Il est montré que l'ordre de l'erreur en norme H^1 de $\hat{u}$ sur le domaine $\Omega\setminus(\bar{B}\cup\bar{\omega})$ est optimal et de même pour l'ordre de l'erreur de v sur le domaine ω. Les figures 1.8 et 1.9 représentent respectivement les courbes d'erreur de $\hat{u}$ et de v réalisées sur le cas test 1.13 dont on connaît la solution

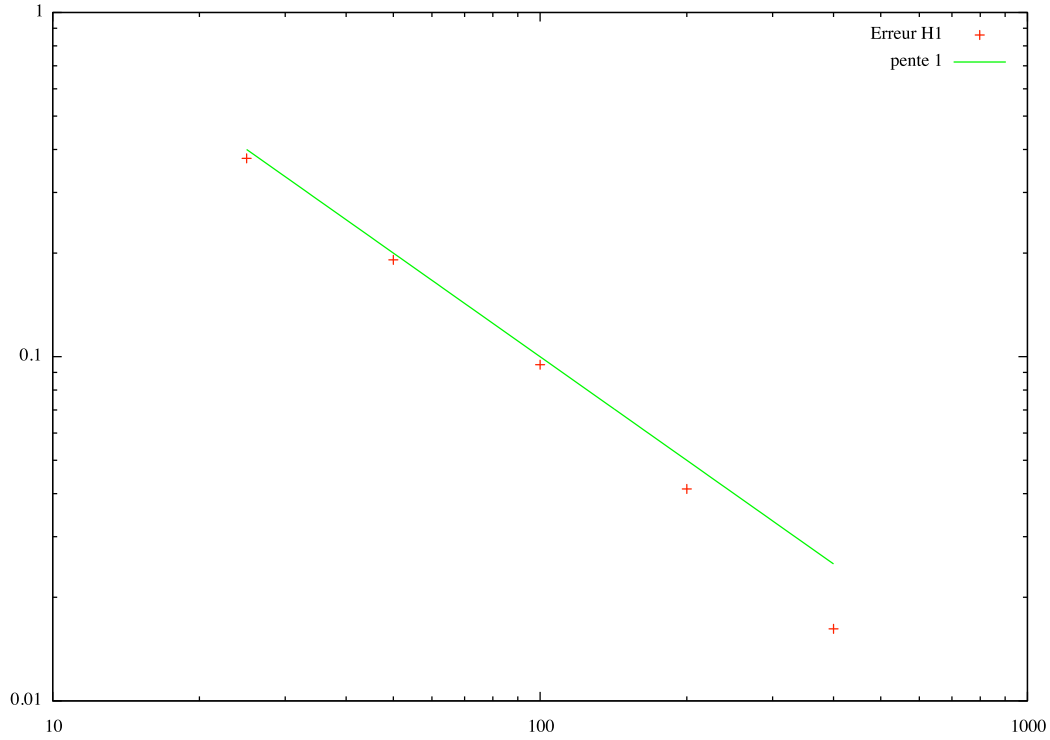

FIGURE 1.8 – Erreur en norme H^1 sur $\Omega \setminus (\bar{B} \cup \bar{\omega})$ de la FBM

exacte. On retrouve bien l'ordre 1 pour les deux approximations, comme démontré dans l'analyse numérique de la méthode.

1.2.7 Avantages et inconvénients

En conclusion, les méthodes sur maillage conforme ne sont pas facile à implémenter, un soin particulier doit notamment être pris dans la gestion de conditions de bord périodique et dans la gestion du remaillage à chaque pas de temps. Elles ont néanmoins l'avantage de pouvoir approcher la solution de notre problème initial (1.1) avec un ordre optimal puisqu'il n'y a pas de discontinuité des dérivées à gérer. À l'inverse, les méthodes de domaine fictif avec multiplicateurs de Lagrange ou avec de la pénalisation sont facilement implémentables dans un solveur éléments finis de Stokes et le maillage peut être cartésien autorisant l'usage de solveurs rapides tels que la FFT ou bien des solveurs multigrille géométrique. Le désavantage de ces deux méthodes est que la solution obtenue n'est pas suffisamment régulière et comporte un saut des dérivées normales sur le bord des particules. Ce saut n'est pas bien capté par le maillage non conforme, il en résulte une erreur d'ordre 1/2 pour des éléments finis d'ordre 1 en norme H^1, là où l'ordre optimal est de 1. Plusieurs méthodes ont été développées pour récupérer cet ordre optimal tout en utilisant un maillage non conforme, elles consistent soit à capter les discontinuités de la solution à travers le bord des particules soit à chercher une solution régulière du problème initial (1.1) de sorte que le saut des dérivées normales disparaisse et que la solution éléments finis puisse approcher correctement la solution exacte. Cependant certaines de

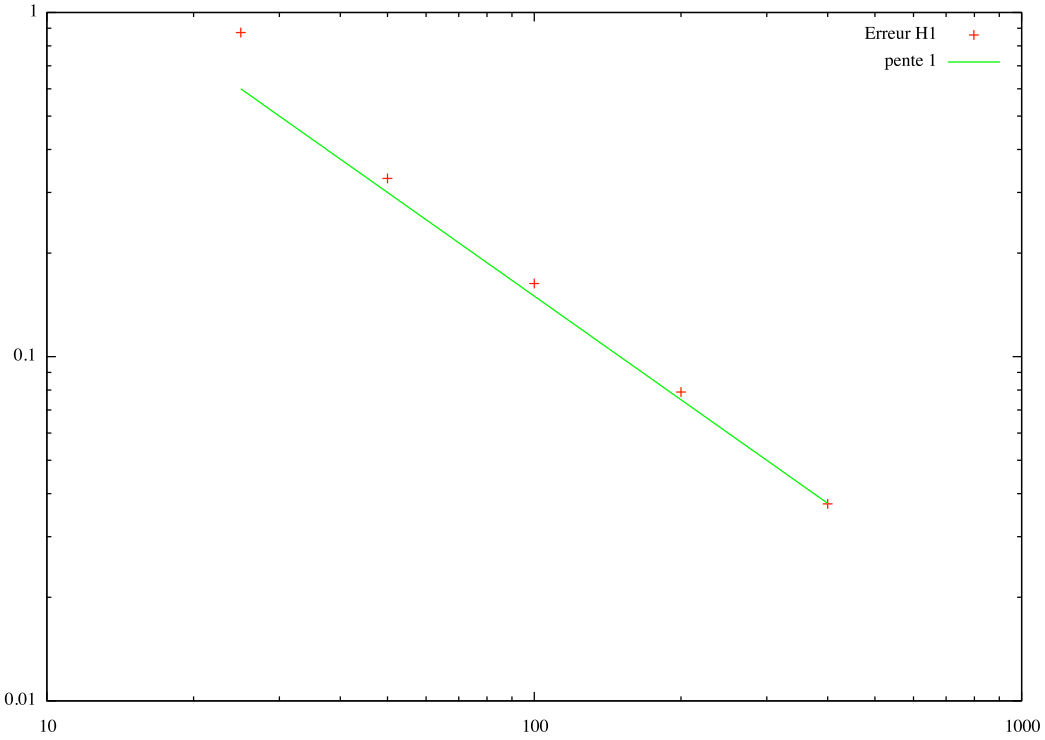

FIGURE 1.9 – Erreur en norme H^1 sur ω de la FBM

ces méthodes dégradent le conditionnement des matrices du système ce qui peut poser des problèmes de convergence des algorithmes de résolution. De plus, la plupart des méthodes font intervenir des matrices qui dépendent de la géométrie du problème et qui doivent donc être modifiées à chaque fois que les particules se déplacent. La méthode développée consiste à chercher un prolongement régulier de la solution tout en conservant des matrices indépendantes de la position des particules.

1.2.8 Les approches par contrôle

On présente dans cette section des méthodes par contrôle, proches de celle développée ici. Ce sont des méthodes qui consistent à chercher un prolongement régulier de la solution exacte à tout le domaine en utilisant un contrôle. On a déjà vu une méthode similaire en pénalisation dans [23]. Dans [4], Atamian, Dinh, Glowinski, He et Périaux présentent une méthode de type domaine fictif basée sur une formulation par contrôle. Ils utilisent le prolongement du second membre comme un contrôle afin d'imposer différentes conditions de bord. Le contrôle est cherché comme le minimum d'une fonctionnelle. Ils appliquent cette méthode tout d'abord à des problèmes de Poisson pour imposer une condition de Dirichlet ou Neumann sur le bord de l'inclusion, puis ils l'appliquent à un problème de type Helmholtz. C'est le sujet de la thèse de Bruno Perret (voir [61]) qui applique aussi cette méthode à un problème de Helmholtz. Dans [5], Atamian et Joly utilisent aussi cette

approche par contrôle pour résoudre le problème d'Helmholtz suivant :

$$\begin{cases} -\Delta u - \omega^2 u &= 0\,, & \text{dans } \mathbb{R}^n \setminus \Omega\,, \\ u &= g\,, & \text{sur } \Gamma\,, \\ \dfrac{\partial u}{\partial r} - i\omega u &= O\left(r^{\frac{1-n}{2}}\right) & \text{pour } r \to +\infty\,, \end{cases}$$

où Ω est un obstacle et Γ son bord. C'est la condition de Dirichlet sur le bord de l'obstacle qui est imposée en utilisant un contrôle. Le problème est prolongé à $\mathbb{R}^n$ tout entier et le second membre est le contrôle en question qui est cherché de telle sorte que la restriction de la solution du problème prolongé soit la solution du problème départ :

$$\begin{cases} -\Delta \tilde{u} - \omega^2 \tilde{u} &= v\,, & \text{dans } \mathbb{R}^n \setminus \Omega\,, \\ \dfrac{\partial \tilde{u}}{\partial r} - i\omega \tilde{u} &= O\left(r^{\frac{1-n}{2}}\right) & \text{pour } r \to +\infty \end{cases}$$

Tout le problème revient à trouver un contrôle qui nous donne la solution exacte. C'est fait en minimisant la fonctionnelle suivante :

$$\int_\Gamma (\tilde{u} - g)^2$$

qui est nulle lorsque $\tilde{u}$ vérifie la condition de Dirichlet recherchée. Enfin dans [7], Badea et Daripa utilise une approche par contrôle pour résoudre des problèmes de type Poisson. Le domaine est inclu dans un disque plus grand ce qui permet l'utilisation d'un algorithme rapide basé sur une décomposition de Fourier. La recherche du contrôle pour imposer une condition de Dirichlet sur le domaine d'origine devient alors un système sur les coefficients de Fourier, permettant une résolution rapide du problème.

Comme on le verra dans le chapître suivant, notre approche est aussi une approche par contrôle mais qu'on applique ici à la résolution d'un problème de Stokes, dans le but d'imposer la contrainte de mouvement rigide.

Chapitre 2

Présentation de la méthode

Sommaire

2.1 Une méthode de prolongement régulier

La méthode développée dans cette thèse est une méthode éléments finis sur des maillages fixes, cartésiens. L'avantage d'un maillage cartésien est la possibilité de pouvoir utiliser des solveurs rapides comme les solveurs multigrilles géométriques. De plus les maillages cartésiens n'ont pas besoin d'être construits et ne prennent donc pas de place en mémoire, ce qui est un plus dans le cas de grosses simulations où la mémoire est restreinte. Les objectifs sont alors :

- Conserver l'ordre optimal bien que le maillage ne soit pas conforme.
- Ne pas modifier les opérateurs du problème initial, comme c'est le cas avec une méthode de pénalisation classique.
- Conserver les mêmes matrices tout au long d'une simulation en temps. Toutes les opérations sur les matrices (création, factorisations éventuelles, ...) peuvent donc être effectuées une seule fois au tout début de la simulation.

Dans l'esprit de [4, 5, 61, 6, 7, 23], la méthode utilise le prolongement du second membre dans les particules pour contrôler la valeur de la solution sur les bords. On présente la méthode dans le cas du problème jouet (1.8) de la section précédente. L'idée est de trouver une fonction g dans l'espace $L^2(\Omega)$ telle que la solution du problème suivant,

$$(2.1)\quad \left\{ \begin{array}{rcll} -\Delta u_g & = & f\chi_{\Omega\setminus\bar{B}} + g\chi_B\,, & \text{dans } \Omega\,, \\ u_g & = & 0\,, & \text{sur } \partial\Omega\,, \end{array} \right.$$

vérifie

$$u_{g|_{\Omega\setminus\bar{B}}} = u,$$

où u est la solution du problème jouet 1.8 et $\chi_{\Omega\setminus\bar{B}}$ et χ_B sont respectivement la fonction caractéristique de $\Omega\setminus\bar{B}$ et de B. Comme f et g sont des fonctions de L^2, la solution de (2.1) est dans l'espace de Sobolev $H^2(\Omega)$ et on peut donc s'attendre à retrouver l'ordre optimal puisqu'il n'y a plus de saut des dérivées normales sur le bord ∂B. La solution u_g n'est donc pas nulle dans B, mais la partie qui nous intéresse est la restriction à $\Omega\setminus\bar{B}$. Tout le problème revient donc à trouver une telle fonction g qui nous permet d'obtenir par restriction la solution du problème sur le domaine perforé. Pour ce faire, on minimise la fonction coût suivante :

$$J(g) = \frac{1}{2}\int_{\partial B} u_g^2\,.$$

On peut voir avec cette définition que si une fonction existe, telle que la restriction de u_g au domaine perforé soit la solution du problème jouet u, alors la fonction coût est nulle. De plus, le minimum de la fonction coût est bien zéro, l'idée est donc de minimiser cette fonctionnelle pour trouver un candidat g possible. En ce qui concerne l'existence d'une telle fonction g, on peut la choisir comme le Laplacien d'un prolongement régulier de la solution exacte du problème jouet à tout le domaine Ω. Un tel prolongement existe et il en existe même une infinité, ce qui nous donne une infinité de fonctions g possibles. L'idée est d'en trouver une qui convient en minimisant la fonction coût à l'aide d'un algorithme de gradient conjugué par exemple. Le choix de la fonction g dépend donc de l'initial guess que l'on choisira. Dans le cas du problème de Stokes, il faut montrer non pas l'existence d'un

prolongement régulier de la solution exacte, mais l'existence d'un prolongement régulier à divergence nulle pour satisfaire la contrainte d'incompressibilité sur le domaine entier.

Dans le but de minimiser la fonctionnelle J, on s'intéresse à son gradient. On peut montrer, et nous mèneront une démarche analogue dans le cas du problème de Stokes qui nous intéresse (proposition 2.19), que le gradient de cette fonctionnelle est donné par la solution du problème de Poisson suivant :

$$(2.2)\quad \left\{ \begin{array}{rcll} -\Delta w_g & = & u_{g|\partial B}\delta_{\partial B}\,, & \text{dans } \Omega\,, \\ w_g & = & 0\,, & \text{sur } \partial\Omega\,, \end{array} \right.$$

où le terme source est défini de façon faible contre des fonctions tests par :

$$\left\langle u_{g|\partial B}\delta_{\partial B}\,,\, v \right\rangle = \int_{\partial B} u_g v\,.$$

En utilisant, par exemple, un algorithme de gradient à pas simple pour minimiser la fonctionnelle J, on a besoin à chaque itération du gradient de la fonctionnelle qui est calculé en résolvant numériquement le problème 2.2. Pour construire le second membre de ce problème on a besoin de calculer la solution u_g du problème 2.1, ce qui fait donc au total deux problèmes de Poisson à résoudre pour calculer le gradient de la fonctionnelle et mettre à jour la fonction g.

L'algorithme est donc relativement simple, on minimise la fonctionnelle J pour trouver une fonction g qui convient et on a besoin de résoudre deux problèmes de Poisson par itération pour calculer le gradient de la fonctionnelle. A chaque itération, la solution du problème 2.1 est régulière dans tout le domaine Ω, et on prend la restriction de la solution de ce problème lorsque notre algorithme de minimisation à convergé. On peut remarquer qu'on satisfait déjà certains points de l'introduction. On peut utiliser un maillage cartésien dans le but d'utiliser des solveurs rapides pour résoudre les problèmes de Poisson (FFT, Multigrille géométrique). Les opérateurs ne sont pas modifiés, seul les seconds membres changent et contiennent l'information sur la géométrie du problème. On parlera de l'ordre de l'erreur en espace dans la section suivante consacrée à l'application de cette méthode dans le cas du problème de Stokes avec condition de mouvement rigide sur le bord des inclusions (au lieu d'une condition homogène). La différence réside dans le choix de la fonctionnelle à minimiser. Elle est plus compliquée mais reste similaire au cas scalaire que l'on vient de voir.

2.2 Détail de la méthode sur le problème de Stokes

En ce qui concerne le problème de Stokes initial, on suppose que le second membre $\mathbf{f}$ est une fonction de $L^2(\Omega)^d$ nulle sur B. Le domaine Ω est un rectangle et les inclusions sont sphériques, donc avec des bords réguliers, la solution $(\mathbf{u}, p)$ du problème initial 1.1 est donc dans l'espace produit $H^2(\Omega \setminus \bar{B})^d \times H^1(\Omega \setminus \bar{B})$. On a alors la proposition suivante sur le lien entre la solution du problème de Stokes non contraint (équivalent au problème 2.1 du cas scalaire) et la solution du problème de Stokes initial 1.1 :

Proposition 2.1. *On prend les mêmes notations que dans le problème 1.1. Pour tout i entier compris entre 1 et N, on note $C_i(d)$ la constante définie par :*

$$C_i(d) = \begin{cases} \dfrac{{R_i}^2}{2}, & si\ d = 2\,, \\ \dfrac{3{R_i}^2}{5}, & si\ d = 3\,. \end{cases} \tag{2.3}$$

Supposons qu'il existe une fonction $\mathbf{g}$ appartenant à $L^2(B)^d$, prolongée par zéro sur $\Omega \setminus \bar{B}$, telle que la restriction à $\Omega \setminus \bar{B}$ de la solution $(\mathbf{u}_g, p_g)$ du problème de Stokes incompressible suivant :

$$(2.4)\quad \begin{cases} -2\eta\,\mathrm{div}(D(\mathbf{u}_g)) + \nabla p_g &= \rho_f \mathbf{f}\,\chi_{\Omega\setminus\bar{B}} + \mathbf{g}\,\chi_B \\ &\quad + \displaystyle\sum_{i=1}^{N} \frac{\mathbf{F}_i}{|B_i|}\,\chi_{B_i} \\ &\quad + \displaystyle\sum_{i=1}^{N} \frac{d}{C_i(d)|B_i|(d-1)}\,(\mathbf{T}_i \wedge \mathbf{r}_i)\,\chi_{B_i}\,, & \text{dans } \Omega\,, \\ \nabla \cdot \mathbf{u}_g &= 0\,, & \text{dans } \Omega\,, \\ \mathbf{u}_g &= 0\,, & \text{sur } \partial\Omega\,, \end{cases}$$

soit solution du problème 1.1, alors $\mathbf{g}$ vérifie les relations suivantes :

$$\int_{B_i} \mathbf{g} = 0 \quad \forall i = 1, \ldots, N\,, \tag{2.5}$$

$$\int_{B_i} \mathbf{r}_i \wedge \mathbf{g} = 0 \quad \forall i = 1, \ldots, N\,. \tag{2.6}$$

Réciproquement, si une fonction $\mathbf{g}$ dans $L^2(B)^d$ vérifie les conditions 2.5 et 2.6 est telle que la solution $(\mathbf{u}_g, p_g)$ du problème 2.4 soit un mouvement rigide sur chaque particule, alors la restriction de $(\mathbf{u}_g, p_g)$ à $\Omega \setminus \bar{B}$ est solution du problème 1.1.

Remarque 2.2. *Dans le cas 2D la vitesse de rotation $\boldsymbol{\omega}$ est en fait un réel mais nous continuerons à l'écrire en gras $\boldsymbol{\omega}$. En 2D on peut d'ailleurs considérer que l'on se trouve dans le plan $z = 0$ et écrire la vitesse de rotation comme un vecteur dirigé selon l'axe z. On retrouve alors les formules du produit vectoriel en utilisant celles en 3D. Pour le produit vectoriel d'un réel par un vecteur on obtient donc un vecteur :*

$$\omega \wedge \mathbf{r} = \begin{pmatrix} 0 \\ 0 \\ \omega \end{pmatrix} \wedge \begin{pmatrix} r_1 \\ r_2 \\ 0 \end{pmatrix} = \omega \begin{pmatrix} -r_2 \\ r_1 \\ 0 \end{pmatrix}$$

et le produit vectoriel de deux vecteurs donne un réel :

$$\tilde{\mathbf{r}} \wedge \mathbf{r} = \begin{pmatrix} \tilde{r}_1 \\ \tilde{r}_2 \\ 0 \end{pmatrix} \wedge \begin{pmatrix} r_1 \\ r_2 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \tilde{r}_1 r_2 - \tilde{r}_2 r_1 \end{pmatrix}$$

Preuve de la proposition 2.1 : Soit $\mathbf{g} \in L^2(B)^d$ telle que la restriction de la solution du problème 2.4 soit solution de 1.1. En particulier, les relations suivantes sont vraies :

$$\int_{\gamma_i} \sigma(\mathbf{u}_g)\mathbf{n}_i = \mathbf{F}_i \,, \tag{2.7}$$

$$\int_{\gamma_i} \mathbf{r}_i \wedge \sigma(\mathbf{u}_g)\mathbf{n}_i = \mathbf{T}_i \,. \tag{2.8}$$

Soit i un entier compris entre 1 et N, on intègre la première équation de 2.4 sur le domaine B_i et après utilisation de la formule de Green sur le membre de gauche, on obtient :

$$\int_{\gamma_i} \sigma(\mathbf{u}_g)\mathbf{n}_i = \int_{B_i} \mathbf{g} + \frac{1}{|B_i|}\int_{B_i} \mathbf{F}_i + \frac{d}{C_i(d)|B_i|(d-1)}\int_{B_i} \mathbf{T}_i \wedge \mathbf{r}_i \,,$$

car la fonction test est constante. Or $\mathbf{T}_i$ est constant sur B_i et la dernière intégrale est nulle par symétrie puisqu'on ne considère que des sphères. De même, la force $\mathbf{F}_i$ est constante sur B_i et la deuxième intégrale du membre de droite est donc égale à $\mathbf{F}_i$. La relation 2.7 nous permet d'obtenir la relation 2.5.

Pour la relation 2.6, on refait la même chose en multipliant l'équation par la fonction test égale à $\boldsymbol{\omega} \wedge \mathbf{r}_i$. Une telle fonction test décrit un mouvement rigide, les termes qui apparaissent après la formule de Green sont donc tous nuls sauf les termes de bords. On a en effet :

$$D(\boldsymbol{\omega} \wedge \mathbf{r}_i) = 0 \,,$$
$$\nabla \cdot (\boldsymbol{\omega} \wedge \mathbf{r}_i) = 0 \,.$$

On a donc cette fois-ci la relation :

$$\begin{aligned}\int_{\gamma_i} \sigma(\mathbf{u}_g)\mathbf{n}_i \cdot (\boldsymbol{\omega} \wedge \mathbf{r}_i) = &\int_{B_i} \mathbf{g} \cdot (\boldsymbol{\omega} \wedge \mathbf{r}_i) \\ &+ \frac{1}{|B_i|}\int_{B_i} \mathbf{F}_i \cdot (\boldsymbol{\omega} \wedge \mathbf{r}_i) \\ &+ \frac{d}{C_i(d)|B_i|(d-1)}\int_{B_i} (\mathbf{T}_i \wedge \mathbf{r}_i) \cdot (\boldsymbol{\omega} \wedge \mathbf{r}_i).\end{aligned} \tag{2.9}$$

Comme précédemment, la force $\mathbf{F}_i$ est constante sur B_i, la deuxième intégrale est donc nulle puisque $\boldsymbol{\omega}$ est aussi une constante. Les propriétés du produit mixte permettent d'écrire les intégrales restantes sous la forme,

$$\int_{\gamma_i} \sigma(\mathbf{u}_g)\mathbf{n}_i \cdot (\boldsymbol{\omega} \wedge \mathbf{r}_i) = \boldsymbol{\omega} \cdot \int_{\gamma_i} \mathbf{r}_i \wedge \sigma(\mathbf{u}_g)\mathbf{n}_i \,,$$
$$\int_{B_i} \mathbf{g} \cdot (\boldsymbol{\omega} \wedge \mathbf{r}_i) = \boldsymbol{\omega} \cdot \int_{B_i} \mathbf{r}_i \wedge \mathbf{g} \,,$$
$$\int_{B_i} (\mathbf{T}_i \wedge \mathbf{r}_i) \cdot (\boldsymbol{\omega} \wedge \mathbf{r}_i) = \boldsymbol{\omega} \cdot \int_{B_i} \mathbf{r}_i \wedge (\mathbf{T}_i \wedge \mathbf{r}_i) \,.$$

Le triple produit vectoriel dans le membre de droite de la troisième relation s'écrit :

$$\mathbf{r}_i \wedge (\mathbf{T}_i \wedge \mathbf{r}_i) = |\mathbf{r}_i|^2 \mathbf{T}_i - (\mathbf{r}_i \cdot \mathbf{T}_i)\, \mathbf{r}_i \,.$$

On écrit composante par composante cette relation et on l'intègre sur B_i, ce qui donne :

$$\int_{B_i} \mathbf{r}_i \wedge (\mathbf{T}_i \wedge \mathbf{r}_i) = \int_{B_i} |\mathbf{r}_i|^2 \mathbf{T}_i - \int_{B_i} \left(\left(\sum_{j=1}^{d} r_{i,j} T_{i,j} \right) r_{i,k} \right)_{k=1,\dots,d} .$$

Comme on ne considère que des particules sphériques, on obtient :

$$\int_{B_i} r_{i,j} r_{i,k} = 0 \quad j \neq k \,,$$

$$\int_{B_i} r_{i,k}^2 = \frac{1}{d} \int_{B_i} |\mathbf{r}_i|^2 = \frac{C_i(d)|B_i|}{d} \,.$$

Ce qui donne :

$$\int_{B_i} \mathbf{r}_i \wedge (\mathbf{T}_i \wedge \mathbf{r}_i) = C_i(d)|B_i|\mathbf{T}_i - \frac{C_i(d)|B_i|}{d}\mathbf{T}_i = \frac{C_i(d)|B_i|(d-1)}{d}\mathbf{T}_i \,.$$

En injectant toutes ces relations dans l'équation 2.9 et en utilisant l'équation 2.8, on obtient le résultat 2.6 de l'énoncé.

L'autre sens est clair : si $\mathbf{g}$ vérifie les relations 2.5 et 2.6 alors les équations traduisant l'équilibre des forces au niveau des particules sont vérifiées (comme on vient de le voir) et les conditions de bords sont aussi toutes vérifiées. Enfin la restriction de la solution à $\Omega \setminus \bar{B}$ vérifie bien les équations de Stokes incompressible. □

On cherche donc une fonction $\mathbf{g}$ dans l'espace noté K_B et définie par la

Définition 2.3.

$$K_B = \left\{ \mathbf{g} \in L^2(B)^d \,:\, \int_{B_i} \mathbf{g} = 0, \quad \int_{B_i} \mathbf{r}_i \wedge \mathbf{g} = 0 \quad \forall i \in \{1, \dots, N\} \right\} \tag{2.10}$$

telle que la restriction de la solution de 2.4 à $\Omega \setminus \bar{B}$ soit solution de 1.1. On peut donc voir cette fonction $\mathbf{g}$ comme un contrôle qui nous permet d'ajuster la solution sur le bord des particules. On peut noter que la solution n'aura pas de sens physique dans les particules mais on ne s'intéresse qu'à sa restriction au domaine $\Omega \setminus \bar{B}$.

2.3 Existence et non unicité d'un contrôle

Dans cette section, nous démontrons l'existence d'un contrôle et une façon de construire d'autres contrôles possibles qui nous permettent d'obtenir la solution du problème initial (il n'y a donc pas unicité). On commence par un exemple simple en une dimension pour illustrer l'idée des preuves dans les cas $2D$ et $3D$. On démontre ensuite l'existence dans le cas $2D$ puis dans le cas $3D$ en insistant sur les différences induites par la dimension.

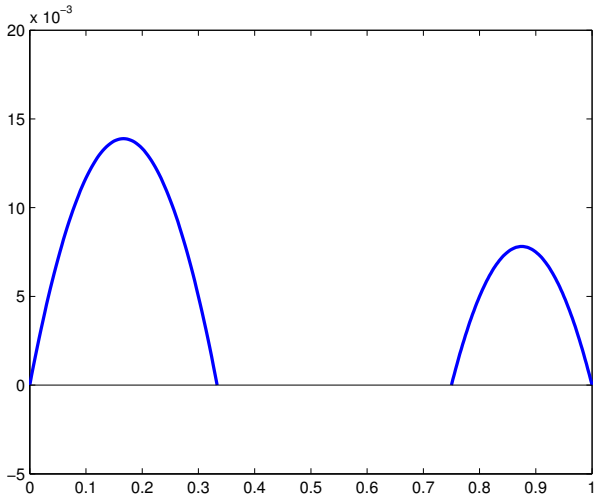

FIGURE 2.1 – Solution du problème $1D$ 2.12

2.3.1 Un exemple en une dimension

On se donne deux réels a et b tels que $0 < a < b < 1$. Le problème que l'on se pose est une version $1D$ du problème jouet 1.8 :

$$(2.12)\quad \begin{cases} -u'' = 1\,, & \text{dans } [0,a]\cup[b,1]\,,\\ u(0) = u(1) = 0\,,\\ u(a) = u(b) = 0\,. \end{cases}$$

Le domaine $[a,b]$ représente l'inclusion B du problème jouet. La solution de ce problème peut être calculée à la main et est constituée de deux paraboles (figure 2.1) :

$$\begin{cases} u(x) = -\dfrac{1}{2}x(x-a) & \text{dans } [0,a]\,,\\ u(x) = -\dfrac{1}{2}(x-b)(x-1) & \text{dans } [b,1]\,. \end{cases}$$

Pour montrer l'existence d'un contrôle, l'idée est d'en construire un en prolongeant la solution exacte de façon régulière. Dans ce cas $1D$ on peut le faire facilement puisqu'on connaît la solution exacte et un prolongement régulier possible est le suivant (figure 2.2) :

$$u(x) = (x-a)(x-b)\frac{(1-b-a)x + a(2b-1)}{2(b-a)^2} \quad \text{dans } [a,b]\,.$$

Pour avoir le contrôle il ne reste plus qu'à dériver deux fois la solution sur $[a,b]$. On voit bien qu'il est possible d'avoir autant de prolongements différents que l'on veut puisque, tant que le prolongement est régulier, il peut être égal à n'importe quelle fonction dans $[a,b]$. Un autre exemple de prolongement possible est le suivant (figure 2.3) :

$$u(x) = \frac{1}{2}\sin\left(\frac{5\pi x}{a}\right)\sin\left(\frac{\pi x}{b}\right)(x-a)(x-b) \quad \text{dans } [a,b]\,.$$

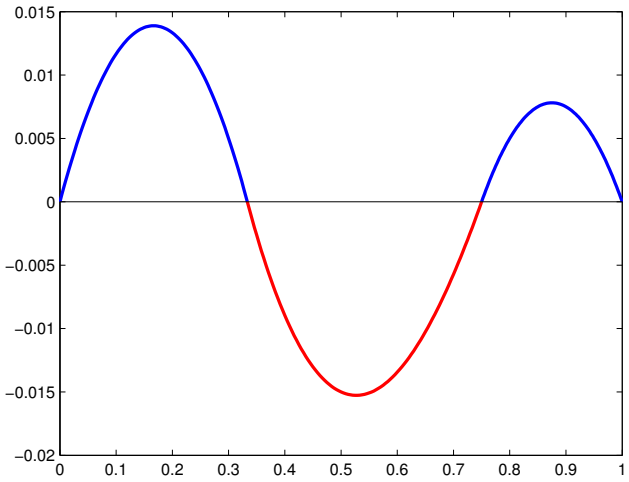

FIGURE 2.2 – Un prolongement possible de la solution du problème 2.12

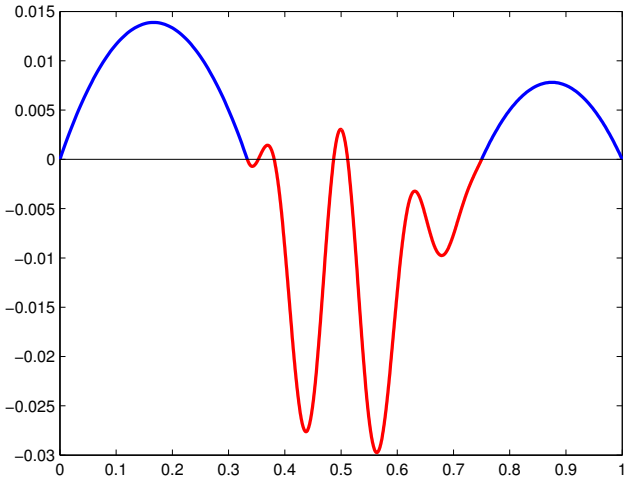

FIGURE 2.3 – Un autre prolongement de la solution du problème 2.12

Dans le cas $2D$ et $3D$ on cherche à faire la même chose, excepté qu'on ne cherche pas seulement un prolongement régulier de la solution mais un prolongement régulier à divergence nulle et que la solution exacte n'est bien sûr pas connue.

2.3.2 Existence dans le cas 2D

On montre l'existence d'un contrôle adapté sous l'hypothèse supplémentaire qu'aucune particule ne touche le bord $\partial\Omega$ de telle sorte que le domaine $\Omega \setminus \bar{B}$ est alors Lipschitz. On cherche à faire la même chose que dans le cas $1D$, avec la difficulté qu'on ne connaît pas la solution exacte. Soit $(\mathbf{u}, p)$ la solution du problème de départ 1.1 qui est dans l'espace $H^2(\Omega \setminus \bar{B})^2 \times H^1(\Omega \setminus \bar{B})$. On cherche donc à montrer l'existence d'un prolongement H^2 à divergence nulle de la vitesse à Ω tout entier et un prolongement H^1 pour la pression.

Pour la vitesse on utilise un résultat d'existence d'une fonction courant d'un champ à divergence nulle que l'on peut trouver dans [33] (théorème 3.1) :

Théorème 2.4. *Soit $\mathcal{O}$ un ouvert borné Lipschitzien de $\mathbb{R}^2$. On note Γ_0 le bord extérieur de $\mathcal{O}$ et Γ_i pour $i = 1, \ldots, p$, les autres composantes de $\partial\mathcal{O}$. Le produit de dualité entre $H^{-1/2}(\Gamma_i)$ et $H^{1/2}(\Gamma_i)$ est noté $\langle ., . \rangle_{\Gamma_i}$. Soit $\mathbf{v}$ une fonction de $L^2(\mathcal{O})^2$, elle satisfait les conditions suivantes :*

$$\operatorname{div} \mathbf{v} = 0 \quad \langle \mathbf{v} \cdot \mathbf{n}\,,\, \mathbf{1} \rangle_{\Gamma_i} = 0\,,$$

si et seulement s'il existe une fonction $\psi \in H^1(\mathcal{O})$ telle que,

$$\mathbf{v} = \operatorname{rot} \psi\,.$$

Dans notre cas, ce résultat dit que la solution $\mathbf{u}$ satisfait les relations :

$$\nabla \cdot \mathbf{u} = 0, \quad \int_{\partial\Omega} \mathbf{u} \cdot \mathbf{n} = 0, \quad \int_{\partial B_i} \mathbf{u} \cdot \mathbf{n}_i = 0,$$

si et seulement s'il existe une fonction scalaire ψ dans $H^1(\Omega \setminus \bar{B})$ telle que,

$$\mathbf{u} = \operatorname{rot} \psi\,.$$

On a bien une fonction à divergence nulle et les intégrales sur les bords sont bien nulles puisque sur les bords des particules la vitesse est un mouvement rigide $\mathbf{V}_i + \boldsymbol{\omega}_i \wedge \mathbf{r}_i$. Or $\boldsymbol{\omega}_i \wedge \mathbf{r}_i$ est orthogonal au vecteur normal qui est colinéaire à $\mathbf{r}_i$ et $\mathbf{V}_i$ est une constante donc son intégrale contre le vecteur normal est nulle sur un contour fermé. Il reste le bord $\partial\Omega$ qui ne pose pas de problème puisque l'on impose des conditions de Dirichlet homogène. Il faut noter que même si ce n'était pas le cas, on aurait le même résultat puisque le champ est à divergence nulle :

$$0 = \int_{\Omega \setminus \bar{B}} \nabla \cdot \mathbf{u} = \int_{\partial\Omega} \mathbf{u} \cdot \mathbf{n} + \int_{\partial B} \mathbf{u} \cdot \mathbf{n}\,.$$

L'intégrale sur le bord ∂B étant nulle comme on vient de le voir, l'intégrale sur le bord $\partial\Omega$ est nulle aussi. Il existe donc une fonction ψ dans $H^1(\Omega \setminus \bar{B})$ telle que $\mathbf{u} = \operatorname{rot} \psi$, ce qui s'écrit :

$$\mathbf{u} = \begin{pmatrix} \partial_y \psi \\ -\partial_x \psi \end{pmatrix}.$$

Comme la solution $\mathbf{u}$ est dans $H^2(\Omega \setminus \bar{B})^2$, on en déduit que la fonction courant ψ est dans $H^3(\Omega \setminus \bar{B})$. On cherche maintenant à prolonger ψ de façon régulière à tout le domaine Ω, on utilise pour cela le théorème de Stein (voir [2]). Afin de pouvoir énoncer ce théorème, ous introduisont les définitions suivantes :

Définition 2.5. *Soit Ω un domaine de $\mathbb{R}^n$. Pour des entiers m et p donnés, un opérateur de l'espace de Sobolev $W^{m,p}(\Omega)$ dans l'espace de Sobolev $W^{m,p}(\mathbb{R}^n)$ est appelé un opérateur de (m,p)-prolongement simple pour Ω s'il existe une constante $K = K(m,p)$ telle que pour tout u dans $W^{m,p}(\Omega)$ les conditions suivantes sont vraies :*

1. $Eu(x) = u(x) \quad$ *p.p. dans* $\Omega \quad$,
2. $\|Eu\|_{m,p,\mathbb{R}^n} \leq K\|u\|_{m,p,\Omega} \quad$.

L'opérateur E est appelé un opérateur de m-prolongement fort pour Ω si E est un opérateur linéaire envoyant les fonctions définies presque partout sur Ω sur des fonctions définies presque partout sur $\mathbb{R}^n$ et si, pour tout p, $1 \leq p < \infty$ et pour tout entier k, $0 \leq k \leq m$, la restriction de E à $W^{k,p}(\Omega)$ est un opérateur de (k,p)-prolongement simple pour Ω.

L'opérateur E est enfin appelé un opérateur de prolongement total pour Ω si E est un opérateur de m-prolongement fort pour Ω pour tout m.

Définition 2.6 (Condition de Lipschitz locale forte)**.** *Un domaine Ω satisfait la condition de Lipschitz locale forte s'il existe des réels positifs δ et M, un recouvrement localement fini $\{U_j\}$ de $\partial\Omega$, et, pour tout j, une fonction rélle f_j de $n-1$ variables, telles que les conditions suivantes soient vraies :*

1. *Pour un nombre fini R, toute collection de $R+1$ ouverts U_j à une intersection vide.*
2. *Pour toute paire de points x, y dans $\Omega_\delta = \{x \in \Omega : d(x, \partial\Omega) < \delta\}$ telle que $|x-y| < \delta$, il existe j tel que :*

$$x, y \in V_j = \{x \in U_j : d(x, \partial U_j) > \delta\} \ .$$

3. *Chaque fonction f_j satisfait la condition de Lipschitz avec la constante M, c'est-à-dire que si $\xi, \rho \in \mathbb{R}^{n-1}$ alors on a*

$$|f(\xi) - f(\rho)| \leq M|\xi - \rho| \, .$$

4. *Pour un système de coordonnées cartésiennes $(\xi_{j,1}, \ldots, \xi_{j,n})$ dans U_j, $\Omega \cap U_j$ est représenté par l'inégalité :*

$$\xi_{j,n} < f_j(\xi_{j,1}, \ldots, \xi_{j,n-1}) \, .$$

Remarque 2.7. *Comme mentionné dans [2], si le domaine Ω considéré est borné, alors ces conditions traduisent le fait que le bord de Ω est localement Liptschitz, c'est-à-dire que chaque point x de $\partial\Omega$ doit avoir un voisinage U_x tel que $U_x \cap \partial\Omega$ soit le graphe d'une fonction Lipschitz.*

Théorème 2.8 (Stein)**.** *Si Ω est un domaine de $\mathbb{R}^n$ satisfaisant la condition forte de Liptschitz locale, alors il existe un opérateur de prolongement total de Ω.*

On suppose que les particules ne touchent pas le bord $\partial\Omega$ et ne se touchent pas entre elles, ce qui implique que le domaine $\Omega \setminus \bar{B}$ que l'on considère est Lipschitz et on peut appliquer le théorème de prolongement de Stein qui donne l'existence d'un prolongement de ψ à $H^3(\mathbb{R}^2)$. On note $\tilde{\psi}$ ce prolongement et on prolonge $\mathbf{u}$ en un champ $\tilde{\mathbf{u}}$ à Ω en posant $\tilde{\mathbf{u}} = \operatorname{rot}\tilde{\psi}$. Ainsi, $\tilde{\mathbf{u}}$ est une fonction de $H^2(\Omega)^2$ à divergence nulle. Pour la pression on peut utiliser directement le théorème de prolongement pour obtenir un prolongement $\tilde{p}$ dans $H^1(\Omega)$. On a donc maintenant un prolongement régulier de la solution du problème 1.1 à Ω, on trouve un contrôle qui convient en le définissant dans B par :

$$\mathbf{g} = -2\eta \operatorname{div}(\tilde{\mathbf{u}}) + \nabla\tilde{p} - \sum_{i=1}^{N} \frac{\mathbf{F}_i}{|B_i|}\chi_{B_i} - \sum_{i=1}^{N} \frac{d}{C_i(d)|B_i|(d-1)} (\mathbf{T}_i \wedge \mathbf{r}_i)\ \chi_{B_i}\,.$$

Le couple $(\tilde{\mathbf{u}}, \tilde{p})$ vérifie bien les équations du problème 2.4 et sa restriction à $\Omega \setminus \bar{B}$ est bien la solution du problème 1.1.

2.3.3 Existence dans le cas 3D

En trois dimensions on fait la même chose qu'en deux dimensions sauf pour montrer que la fonction courant est dans $H^3(\Omega \setminus \bar{B})^d$ qui ne découle pas directement de la définition. On supposera ici aussi qu'aucune particule ne touche le bord $\partial\Omega$. On utilise certains résultats de [3] :

Théorème 2.9. *Soit $\mathcal{O}$ un ouvert borné Lipschitzien de $\mathbb{R}^3$. On note Γ_0 le bord extérieur de $\mathcal{O}$ et Γ_i pour $i = 1, \dots, p$ les autres composantes de $\partial\mathcal{O}$. Le produit de dualité entre $H^{-1/2}(\Gamma_i)$ et $H^{1/2}(\Gamma_i)$ est noté $\langle .\,, .\rangle_{\Gamma_i}$. Soit $\mathbf{v}$ dans $L^2(\mathcal{O})^3$ telle que la divergence de $\mathbf{v}$ soit dans $L^2(\mathcal{O})$. Alors $\mathbf{v}$ satisfait les relations :*

$$\operatorname{div}\mathbf{v} = 0, \quad \langle \mathbf{v}\cdot\mathbf{n}\,,\, \mathbf{1}\rangle_{\Gamma_i} = 0,$$

si et seulement s'il existe un potentiel vecteur $\boldsymbol{\psi}$ dans $L^2(\mathcal{O})^3$ tel que sa divergence et son rotationnel soient respectivement dans $L^2(\mathcal{O})$ et $L^2(\mathcal{O})^3$ et qui satisfait de plus les relations suivantes :

$$\mathbf{u} = \operatorname{rot}\boldsymbol{\psi}, \quad \nabla\cdot\boldsymbol{\psi} = 0, \quad \boldsymbol{\psi}\cdot\mathbf{n} = 0 \text{ sur } \partial\mathcal{O}\,.$$

Ce champ $\boldsymbol{\psi}$ est unique.

Ce théorème est analogue à celui qu'on peut trouver dans [33] avec les conditions supplémentaires sur le potentiel qui donnent l'unicité. On ne peut pas conclure directement que le potentiel est dans H^3 contrairement au cas $2D$, on utilise pour cela le résultat suivant qu'on trouvera aussi dans [3] :

Théorème 2.10. *Soit $\mathcal{O}$ un domaine de classe $\mathcal{C}^{m,1}$ pour un entier $m \geq 1$. Alors les espaces de fonctions :*

$$\left\{\mathbf{v} \in L^2(\mathcal{O})^3 \,:\, \operatorname{rot}\mathbf{v} \in H^{m-1}(\mathcal{O})^3,\, \operatorname{div}\mathbf{v} \in H^{m-1}(\mathcal{O}) \text{ et } \mathbf{v}\wedge\mathbf{n} \in H^{m-1/2}(\partial\mathcal{O})^3\right\}$$

et

$$\left\{\mathbf{v}\in L^2(\mathcal{O})^3\ :\ \mathrm{rot}\,\mathbf{v}\in H^{m-1}(\mathcal{O})^3,\ \mathrm{div}\,\mathbf{v}\in H^{m-1}(\mathcal{O})\ et\,\mathbf{v}\cdot\mathbf{n}\in H^{m-1/2}(\partial\mathcal{O})^3\right\}\ ,$$

s'injectent continûment dans $H^m(\mathcal{O})^3$.

On suppose qu'aucune particule ne touche le bord $\partial\Omega$ et n'est en contact avec une autre particule et on considère un sous domaine $\mathcal{O}\setminus\bar{B}$ de $\Omega\setminus\bar{B}$ avec des bords réguliers et contenant toutes les particules. L'existence du potentiel $\boldsymbol{\psi}$ dans $\mathcal{O}\setminus\bar{B}$ est donnée par le théorème 2.9 et on peut montrer qu'il est dans $H^3(\mathcal{O}\setminus\bar{B})$ avec le théorème 2.10 pour $m=3$. En utilisant alors comme dans le cas $2D$ le théorème de prolongement de Stein, on étend $\boldsymbol{\psi}$ à tout le domaine $\mathcal{O}$ en un potentiel $\tilde{\boldsymbol{\psi}}$ dans $H^3(\mathcal{O})^3$. On conclut alors de la même façon qu'en deux dimensions.

2.3.4 Non unicité

Nous précisons dans cette section pourquoi le contrôle n'est pas défini de façon unique en dimension $d=2$ et 3. Comme dans l'exemple $1D$, l'idée est que le prolongement de la solution du problème 1.1 peut être quelconque dans les particules à condition qu'il reste régulier.

Proposition 2.11. *Soit* $\mathbf{g}$ *un contrôle tel que la restriction de* $\mathbf{u}_g$*, solution de 2.4, soit solution de 1.1. Soit* $\boldsymbol{\psi}$ *une fonction de* $H^2(B)^d$ *vérifiant les propriétés suivantes :*

$$\nabla\cdot\boldsymbol{\psi}=0,\quad \boldsymbol{\psi}_{|\partial B}=0,\quad D(\boldsymbol{\psi})\mathbf{n}=0\ \textit{sur}\ \partial B\,.$$

Alors la fonction de $L^2(B)^d$ *suivante :*

$$\boldsymbol{\phi}=\mathbf{g}-2\eta\,\mathrm{div}(D(\boldsymbol{\psi}))$$

est aussi un contrôle qui convient.

Preuve : Le champ de vitesse $\tilde{\mathbf{u}}_{g,\psi}$ défini sur Ω par :

$$\begin{array}{rcll}\tilde{\mathbf{u}}_{g,\psi} &=& \mathbf{u}_g\,, & \text{dans } \Omega\setminus\bar{B}\,,\\ \tilde{\mathbf{u}}_{g,\psi} &=& \mathbf{u}_g+\boldsymbol{\psi}\,, & \text{dans } B\,,\end{array}$$

permet aussi d'obtenir la solution de 1.1 ($\boldsymbol{\psi}$ est nulle sur le bord des particules) et est solution du problème :

$$\left\{\begin{array}{rcll}-2\eta\,\mathrm{div}(D(\tilde{\mathbf{u}}_{g,\psi}))+\nabla p_{g,\psi} &=& \rho_f\mathbf{f}\,\chi_{\Omega\setminus\bar{B}}+\boldsymbol{\phi}\,\chi_B & \\ && +\displaystyle\sum_{i=1}^N\frac{\mathbf{F}_i}{|B_i|}\,\chi_{B_i} & \\ && +\displaystyle\sum_{i=1}^N\frac{d}{C_i(d)|B_i|(d-1)}\,(\mathbf{T}_i\wedge\mathbf{r}_i)\,\chi_{B_i}\,, & \text{dans } \Omega\,,\\ \nabla\cdot\tilde{\mathbf{u}}_{g,\psi} &=& 0\,, & \text{dans } \Omega\,,\\ \tilde{\mathbf{u}}_{g,\psi} &=& 0\,, & \text{sur } \partial\Omega\,,\end{array}\right.$$

où $\boldsymbol{\phi}$ est la fonction de l'énoncé que l'on suppose prolongée par zéro dans $\Omega \setminus \bar{B}$ dans le problème de Stokes. On montre alors que $\boldsymbol{\phi}$ est bien dans l'espace K_B. Par formule de Green et grâce aux propriétés de $\boldsymbol{\psi}$ on a, pour tout $\boldsymbol{\omega} \in \mathbb{R}^d$:

$$\int_{B_i} \operatorname{div}(D(\boldsymbol{\psi})) = \int_{\partial B_i} D(\boldsymbol{\psi})\mathbf{n} = 0\,,$$

$$\boldsymbol{\omega} \cdot \int_{B_i} \mathbf{r} \wedge \operatorname{div}(D(\boldsymbol{\psi})) = \int_{B_i} \operatorname{div}(D(\boldsymbol{\psi})) \cdot (\boldsymbol{\omega} \wedge \mathbf{r}) = \int_{\partial B_i} D(\boldsymbol{\psi})\mathbf{n} \cdot (\boldsymbol{\omega} \wedge \mathbf{r}) = 0\,.$$

Il n'y a donc pas unicité et nous ne chercherons pas à l'obtenir en sélectionnant un prolongement en particulier. □

2.4 Minimisation d'une fonction coût

La difficulté consiste à trouver un prolongement $\mathbf{g}$ qui donne la solution de 1.1 à partir de celle de 2.4. Pour cela, nous proposons dans cette section une méthode qui consiste à minimiser une fonctionnelle à l'aide d'un algorithme de type gradient conjugué. Nous commençons par présenter cette fonctionnelle puis nous calculons son gradient en vue de la minimiser.

2.4.1 Définition de la fonctionnelle

L'idée est de minimiser en norme L^2 sur le bord des particules, la différence entre la fonction $\mathbf{u}_g$ pour un $\mathbf{g}$ donné et son mouvement rigide. Supposons que $\mathbf{u}_g$ soit un mouvement rigide $\mathbf{V}_i + \boldsymbol{\omega}_i \wedge \mathbf{r}_i$ sur le bord de la particule i, alors on a :

$$\mathbf{V}_i = \frac{1}{|\partial B_i|} \int_{\partial B_i} \mathbf{u}_g\,, \tag{2.12}$$

$$\boldsymbol{\omega}_i = \frac{C_d}{{R_i}^2 |\partial B_i|} \int_{\partial B_i} \mathbf{r}_i \wedge \mathbf{u}_g\,, \tag{2.13}$$

où la constante C_d vaut :

$$C_d = 1 \quad \text{si } d = 2\,,$$
$$C_d = \frac{3}{2} \quad \text{si } d = 3\,.$$

Le calcul est très similaire à celui de la preuve de la proposition 2.1 et on ne le détaillera donc pas. On considère alors la fonctionnelle suivante, définie sur K_B et à valeurs dans $\mathbb{R}$:

$$J(\mathbf{g}) = \sum_{i=1}^{N} \frac{1}{2} \int_{\partial B_i} \left| \mathbf{u}_g - \frac{1}{|\partial B_i|} \int_{\partial B_i} \mathbf{u}_g - \frac{C_d}{{R_i}^2 |\partial B_i|} \left(\int_{\partial B_i} \mathbf{r}_i \wedge \mathbf{u}_g \right) \wedge \mathbf{r}_i \right|^2 . \tag{2.14}$$

Étant données les relations 2.12 et 2.13, si $\mathbf{u}_g$ vérifie la contrainte de mouvement rigide sur le bord des particules, la fonctionnelle est nulle au point $\mathbf{g}$ et réciproquement. On cherche

donc une fonction $\mathbf{g}$ de K_B telle que $J(\mathbf{g}) = 0$. C'est une fonctionnelle quadratique qu'on écrit sous une forme plus simple. Pour cela on a besoin de définir le problème suivant :

Définition 2.12. *Pour un* $\mathbf{g}$ *donné dans* K_B*, on note* $(\tilde{\mathbf{u}}_g, \tilde{p}_g)$ *la solution du problème suivant :*

$$(2.15)\quad \begin{cases} -2\eta \operatorname{div}(D(\tilde{\mathbf{u}}_g)) + \nabla \tilde{p}_g &= \mathbf{g}\,\chi_B\,, & \text{dans } \Omega\,, \\ \nabla \cdot \tilde{\mathbf{u}}_g &= 0\,, & \text{dans } \Omega\,, \\ \tilde{\mathbf{u}}_g &= 0\,, & \text{sur } \partial\Omega\,. \end{cases}$$

Remarque 2.13. *Une solution de 2.4 peut alors s'exprimer comme la somme de deux solutions :*

$$\mathbf{u}_g = \mathbf{u}_0 + \tilde{\mathbf{u}}_g\,, \tag{2.16}$$

où $\mathbf{u}_0$ *est la solution de 2.4 avec* $\mathbf{g} = 0$*.*

On définit aussi les applications qui à un champ de vitesse associent son mouvement rigide sur les particules :

Définition 2.14. *Soit* i *un entier compris entre* 1 *et* N*, on note* $\mathcal{R}_i$ *l'application définie sur* $H^2(\Omega)^d$ *telle que :*

$$\mathcal{R}_i(\mathbf{v}) = \frac{1}{|\partial B_i|}\int_{\partial B_i} \mathbf{v} + \frac{C_d}{{R_i}^2|\partial B_i|}\left(\int_{\partial B_i} \mathbf{r}_i \wedge \mathbf{v}\right) \wedge \mathbf{r}_i\,.$$

Ceci nous permet de réécrire la fonctionnelle sous la forme :

$$J(\mathbf{g}) = \sum_{i=1}^{N} \frac{1}{2}\int_{\partial B_i} |\mathbf{u}_g - \mathcal{R}_i(\mathbf{u}_g)|^2\,.$$

On définie une autre fonctionnnelle similaire pour simplifier les calculs qui est aussi définie sur K_B :

$$\tilde{J}(\mathbf{g}) = \sum_{i=1}^{N} \frac{1}{2}\int_{\partial B_i} |\tilde{\mathbf{u}}_g - \mathcal{R}_i(\tilde{\mathbf{u}}_g)|^2\,.$$

On utilise maintenant 2.16 et on développe le module pour obtenir l'expression :

$$J(\mathbf{g}) = \tilde{J}(\mathbf{g}) + \sum_{i=1}^{N}\int_{\partial B_i} (\tilde{\mathbf{u}}_g - \mathcal{R}_i(\tilde{\mathbf{u}}_g)) \cdot (\mathbf{u}_0 - \mathcal{R}_i(\mathbf{u}_0)) + J(0)\,.$$

Afin de simplifier un peu plus cette expression, on énonce une proposition qui nous servira aussi dans la section suivante :

Proposition 2.15. *Soit* $\mathbf{v}$ *une fonction de* $H^2(\Omega)^d$*,* $\mathbf{V}$ *un élément de* $\mathbb{R}^d$ *et* $\boldsymbol{\omega}$ *un élément de* $\mathbb{R}$ *si* $d = 2$ *et* $\mathbb{R}^3$ *si* $d = 3$*. On a la relation d'orthogonalité :*

$$\int_{\partial B_i} (\mathbf{v} - \mathcal{R}_i(\mathbf{v})) \cdot (\mathbf{V} + \boldsymbol{\omega} \wedge \mathbf{r}_i) = 0\,.$$

Preuve : Comme on ne considère que des sphères, les intégrales ne faisant intervenir que le produit vectoriel entre une constante et le vecteur radial sont nulles. On a donc :

$$\int_{\partial B_i} \mathbf{v} - \mathcal{R}_i(\mathbf{v}) = 0\,,$$

et il ne reste plus que la partie avec la rotation $\boldsymbol{\omega} \wedge \mathbf{r}_i$. En développant $\mathcal{R}_i(\mathbf{v})$ on obtient alors :

$$\int_{\partial B_i} \mathbf{v} \cdot (\boldsymbol{\omega} \wedge \mathbf{r}_i) - \frac{C_d}{{R_i}^2 |\partial B_i|} \left(\int_{\partial B_i} \mathbf{r}_i \wedge \mathbf{v} \right) \wedge \mathbf{r}_i \cdot (\boldsymbol{\omega} \wedge \mathbf{r}_i)\ .$$

Après un calcul qu'on ne détaillera pas ici, on obtient la relation :

$$\frac{C_d}{{R_i}^2 |\partial B_i|} \int_{\partial B_i} \left(\int_{\partial B_i} \mathbf{r}_i \wedge \mathbf{v} \right) \wedge \mathbf{r}_i \cdot (\boldsymbol{\omega} \wedge \mathbf{r}_i) = \int_{\partial B_i} \mathbf{v} \cdot (\boldsymbol{\omega} \wedge \mathbf{r}_i)\,,$$

ce qui nous donne le résultat énoncé. □

On peut donc écrire la fonctionnelle sous la forme suivante :

$$J(\mathbf{g}) = \tilde{J}(\mathbf{g}) + \sum_{i=1}^{N} \int_{\partial B_i} \tilde{\mathbf{u}}_g \cdot (\mathbf{u}_0 - \mathcal{R}_i(\mathbf{u}_0)) + J(0)\,. \tag{2.17}$$

À partir de cette expression, on va maintenant identifier l'opérateur sous-jacent.

2.4.2 Opérateur de la fonctionnelle

On établit tout d'abord la proposition suivante qui nous permet d'écrire l'expression 2.17 sous la forme de produit scalaire dans K_B, ce qui nous permettra d'identifier les opérateurs de cette fonctionnelle quadratique.

Proposition 2.16. *Soit* $\mathbf{h} \in K_B$ *et* $\tilde{\mathbf{u}}_h$ *la solution de 2.15 pour* $\mathbf{g} = \mathbf{h}$. *Soit* $\mathbf{v}$ *une fonction de* $H^2(\Omega)^d$, *on note* $(\mathbf{w}, \boldsymbol{\pi})$ *la solution du problème :*

$$(2.18)\quad \begin{cases} -2\eta\,\mathrm{div}(D(\mathbf{w})) + \nabla\pi &= \displaystyle\sum_{i=1}^{N} (\mathbf{v} - \mathcal{R}_i(\mathbf{v}))\,\delta_{\partial B_i}\,, & \text{dans } \Omega\,, \\ \nabla \cdot \mathbf{w} &= 0\,, & \text{dans } \Omega\,, \\ \mathbf{w} &= 0\,, & \text{sur } \partial\Omega\,. \end{cases}$$

Le terme source de 2.18 est défini dans $H^{-1}(\Omega)$ *de façon faible contre les fonctions de* $H_0^1(\Omega)^d$ *par :*

$$\langle (\mathbf{v} - \mathcal{R}_i(\mathbf{v}))\,\delta_{\partial B_i}\,,\, \boldsymbol{\varphi} \rangle = \int_{\partial B_i} (\mathbf{v} - \mathcal{R}_i(\mathbf{v})) \cdot \boldsymbol{\varphi}\,.$$

Alors on a :

$$\sum_{i=1}^{N} \int_{\partial B_i} \tilde{\mathbf{u}}_h \cdot (\mathbf{v} - \mathcal{R}_i(\mathbf{v})) = (\mathcal{P}_{K_B}(\mathbf{w})\,,\, \mathbf{h})\ ,$$

où $\mathcal{P}_{K_B}$ est la projection L^2 sur K_B donnée sur chaque B_i par :

$$\mathcal{P}_{K_B}(\mathbf{w})_{|_{B_i}} = \mathbf{w}_{|_{B_i}} - \frac{1}{|B_i|}\int_{B_i}\mathbf{w} - \frac{d}{C_i(d)|B_i|(d-1)}\left(\int_{B_i}\mathbf{r}_i\wedge\mathbf{w}\right)\wedge\mathbf{r}_i\,,$$

où la constante $C_i(d)$ à été définie dans la proposition 2.1.

Remarque 2.17. *On remarque que contrairement au problème 2.4, le problème 2.18 admet des solutions moins régulières à cause de la distribution simple couche dans le second membre. Les solutions* $\mathbf{w}$ *auront en effet un saut des dérivées au niveau des bords ∂B_i si le second membre n'est pas nul, donc si* $\mathbf{v}$ *n'est pas un mouvement rigide sur le bord des particules. La solution* $\mathbf{w}$ *est en fait dans l'espace de Sobolev $H^{3/2}(\Omega)^d$.*

Preuve de la proposition 2.16 : On note S la somme qui nous intéresse :

$$S = \sum_{i=1}^{N}\int_{\partial B_i}\tilde{\mathbf{u}}_h\cdot(\mathbf{v}-\mathcal{R}_i(\mathbf{v}))\ .$$

On utilise la formulation variationnelle du problème 2.18 pour passer d'une intégrale sur une surface à une intégrale en volume :

$$\frac{\eta}{2}\int_\Omega D(\mathbf{w}):D(\boldsymbol{\varphi}) - \int_\Omega \boldsymbol{\pi}\nabla\cdot\boldsymbol{\varphi} = \sum_{i=1}^{N}\int_{\partial B_i}(\mathbf{v}-\mathcal{R}_i(\mathbf{v}))\cdot\boldsymbol{\varphi}\quad\forall\boldsymbol{\varphi}\in H^1_0(\Omega)\,.$$

On peut prendre $\boldsymbol{\varphi}=\tilde{\mathbf{u}}_h$ comme fonction test et en utilisant le fait que la divergence de $\tilde{\mathbf{u}}_h$ est nulle on obtient :

$$S = \frac{\eta}{2}\int_\Omega D(\mathbf{w}):D(\tilde{\mathbf{u}}_h)\,.$$

On fait la même chose avec la formulation variationnelle du problème 2.15 dont $\tilde{\mathbf{u}}_h$ est solution :

$$\frac{\eta}{2}\int_\Omega D(\tilde{\mathbf{u}}_h):D(\boldsymbol{\varphi}) - \int_\Omega \tilde{p}_h\nabla\cdot\boldsymbol{\varphi} = \int_B \mathbf{h}\cdot\boldsymbol{\varphi}\quad\forall\boldsymbol{\varphi}\in H^1_0(\Omega)\,.$$

On prend alors cette fois ci $\boldsymbol{\varphi}=\mathbf{w}$ et on obtient donc :

$$S = \int_B \mathbf{h}\cdot\mathbf{w}\,.$$

On a presque l'égalité souhaité, il faut juste vérifier que l'on peut projeter $\mathbf{w}$ sur K_B sans que cela ne change l'égalité. Il faut donc vérifier que les fonctions de K_B sont orthogonales aux mouvements rigides. On a en effet les relations suivantes :

$$\int_{B_i}\mathbf{h}\cdot\mathbf{V}=0\quad\forall\mathbf{V}\in\mathbb{R}^d \text{ car } \int_B\mathbf{h}=0\,,$$
$$\int_{B_i}\mathbf{h}\cdot(\boldsymbol{\omega}\wedge\mathbf{r}_i) = \boldsymbol{\omega}\cdot\int_{B_i}\mathbf{r}_i\wedge\mathbf{h}=0\quad\forall\boldsymbol{\omega} \text{ car } \int_{B_i}\mathbf{r}_i\wedge\mathbf{h}=0\,.$$

On en déduit donc :

$$S = \sum_{i=1}^{N} \int_{B_i} \mathbf{h} \cdot \left(\mathbf{w} - \frac{1}{|B_i|} \int_{B_i} \mathbf{w} - \frac{d}{C_i(d)|B_i|(d-1)} \left(\int_{B_i} \mathbf{r}_i \wedge \mathbf{w} \right) \wedge \mathbf{r}_i \right)$$
$$= \int_B \mathbf{h} \cdot \mathcal{P}_{K_B}(\mathbf{w}) \, ,$$

où la constante $C_i(d)$ à été définie dans la proposition 2.1. On a donc écrit S sous la forme d'un produit scalaire dans K_B :

$$S = (\mathcal{P}_{K_B}(\mathbf{w}) \, , \, \mathbf{h}) \ ,$$

ce qu'il fallait démontrer. □

Notations 2.18. *Dans le cas particulier où la fonction* $\mathbf{v}$ *de la proposition 2.16 est la solution* $\mathbf{u}_g$ *du problème 2.4 pour un* $\mathbf{g}$ *donné, on notera* $\mathbf{w}_g$ *la solution* $\mathbf{w}$ *du problème 2.18. De même si la fonction* $\mathbf{v}$ *est la solution* $\tilde{\mathbf{u}}_g$ *du problème 2.15, on notera* $\tilde{\mathbf{w}}_g$ *la solution* $\mathbf{w}$ *du problème 2.18.*

Proposition 2.19. *Soit* A *l'opérateur défini par :*

$$\begin{array}{rccc} A : & K_B & \longrightarrow & K_B \\ & \mathbf{g} & \longrightarrow & \mathcal{P}_{K_B}(\tilde{\mathbf{w}}_g) \end{array}$$

La fonctionnelle J *se met sous la forme :*

$$J(\mathbf{g}) = \frac{1}{2} \left(A\mathbf{g} \, , \, \mathbf{g} \right) + \left(\mathcal{P}_{K_B}(\mathbf{w}_0) \, , \, \mathbf{g} \right) + J(0) \, .$$

Preuve : On rappelle l'expression 2.17 de la fonctionnelle :

$$J(\mathbf{g}) = \tilde{J}(\mathbf{g}) + \sum_{i=1}^{N} \int_{\partial B_i} \tilde{\mathbf{u}}_g \cdot (\mathbf{u}_0 - \mathcal{R}_i(\mathbf{u}_0)) + J(0) \, .$$

On applique la proposition 2.16 avec $\mathbf{h} = \mathbf{g}$ et $\mathbf{v} = \mathbf{u}_0$. On peut alors écrire le terme central sous la forme d'un produit scalaire dans K_B :

$$\sum_{i=1}^{N} \int_{\partial B_i} \tilde{\mathbf{u}}_g \cdot (\mathbf{u}_0 - \mathcal{R}_i(\mathbf{u}_0)) = (\mathcal{P}_{K_B}(\mathbf{w}_0) \, , \, \mathbf{g}) \ .$$

En utilisant la proposition 2.15, on peut écrire $\tilde{J}(\mathbf{g})$ sous la forme :

$$\tilde{J}(\mathbf{g}) = \frac{1}{2} \sum_{i=1}^{N} \int_{\partial B_i} \tilde{\mathbf{u}}_g \cdot (\tilde{\mathbf{u}}_g - \mathcal{R}_i(\tilde{\mathbf{u}}_g)) \ .$$

On applique alors à nouveau la proposition précédente avec $\mathbf{h} = \mathbf{g}$ et $\mathbf{v} = \tilde{\mathbf{u}}_g$, ce qui nous donne :

$$\tilde{J}(\mathbf{g}) = \frac{1}{2} (\mathcal{P}_{K_B}(\tilde{\mathbf{w}}_g) \, , \, \mathbf{g}) \ .$$

On peut alors écrire la fonctionnelle J sous la forme de l'énoncé. □

Remarque 2.20. *On obtient directement le gradient de J en tout point $\mathbf{g}$ de K_B :*

$$\nabla J(\mathbf{g}) = A\mathbf{g} + \mathcal{P}_{K_B}(\mathbf{w}_0)\,.$$

En utilisant l'identité 2.16, on a la relation suivante :

$$\tilde{\mathbf{w}}_g + \mathbf{w}_0 = \mathbf{w}_g\,,$$

ce qui permet d'écrire le gradient sous une forme plus condensée :

$$\nabla J(\mathbf{g}) = \mathcal{P}_{K_B}(\mathbf{w}_g)\,.$$

On remarque que ce gradient est nul si la solution $\mathbf{u}_g$ de 2.4 vérifie la contrainte de mouvement rigide sur le bord des particules. Dans ce cas, la fonction $\mathbf{w}_g$ est en effet nulle puisque le terme source dans 2.18 est nul.

Le calcul du gradient (et de $A\mathbf{g}$) passe donc par la résolution du problème de Stokes incompressible 2.18. On présente dans la suite quelques propriétés de l'opérateur A.

2.4.3 Propriétés de l'opérateur de la fonctionnelle

Dans cette section, on montre que l'opérateur A défini à la section précédente est un opérateur auto-adjoint compact et qu'il admet donc une suite de valeurs propres décroissantes vers 0. On présente alors les premiers vecteurs propres dans plusieurs cas différents : avec une seule particule dans le domaine Ω et avec deux particules proches et éloignées. On commence donc avec la proposition suivante :

Proposition 2.21. *L'opérateur A défini par :*

$$\begin{array}{rccc} A: & K_B & \longrightarrow & K_B \\ & \mathbf{g} & \longrightarrow & \mathcal{P}_{K_B}(\tilde{\mathbf{w}}_g) \end{array}$$

est un opérateur auto-adjoint compact.

Preuve : On commence par montrer que A est auto-adjoint. On se donne donc deux fonctions $\mathbf{g}$ et $\mathbf{h}$ de K_B et on regarde le produit scalaire correspondant :

$$(A\mathbf{g}\,,\,\mathbf{h}) = (\mathcal{P}_{K_B}(\tilde{\mathbf{w}}_g)\,,\,\mathbf{h})\ .$$

On utilise la proposition 2.16 pour repasser à une intégrale sur le bord des particules et faisant intervenir la fonction $\tilde{\mathbf{u}}_h$:

$$(\mathcal{P}_{K_B}(\tilde{\mathbf{w}}_g)\,,\,\mathbf{h}) = \sum_{i=1}^{N} \int_{\partial B_i} \tilde{\mathbf{u}}_h \cdot (\tilde{\mathbf{u}}_g - \mathcal{R}_i(\tilde{\mathbf{u}}_g))\,.$$

La proposition 2.15 nous donne une propriété d'orthogonalité ce qui nous permet d'écrire :

$$\begin{aligned} \sum_{i=1}^{N} \int_{\partial B_i} \tilde{\mathbf{u}}_h \cdot (\tilde{\mathbf{u}}_g - \mathcal{R}_i(\tilde{\mathbf{u}}_g)) &= \sum_{i=1}^{N} \int_{\partial B_i} (\tilde{\mathbf{u}}_h - \mathcal{R}_i(\tilde{\mathbf{u}}_h)) \cdot (\tilde{\mathbf{u}}_g - \mathcal{R}_i(\tilde{\mathbf{u}}_g)) \\ &= \sum_{i=1}^{N} \int_{\partial B_i} (\tilde{\mathbf{u}}_h - \mathcal{R}_i(\tilde{\mathbf{u}}_h)) \cdot \tilde{\mathbf{u}}_g\,. \end{aligned}$$

On termine alors en utilisant à nouveau la proposition 2.16 pour faire apparaître l'opérateur :

$$\sum_{i=1}^{N} \int_{\partial B_i} (\tilde{\mathbf{u}}_h - \mathcal{R}_i(\tilde{\mathbf{u}}_h)) \cdot \tilde{\mathbf{u}}_g = (\mathbf{g}\,,\, A\mathbf{h})\ .$$

L'opérateur A est donc bien auto-adjoint.

Pour la compacité, on note $\mathcal{S}^{-1}$ l'inverse de l'opérateur de Stokes qui à un second membre dans K_B associe la solution du problème 2.15 et $\mathcal{W}^{-1}$ l'opérateur qui à un second membre de type distribution simple couche sur ∂B associe la solution du problème 2.18. On note de plus $\mathcal{T}_{\partial B}$ l'opérateur qui à un champ associe la distribution simple couche sur le bord des inclusions. On peut donc écrire l'opérateur A comme la composée de ces opérateurs :

$$A = \mathcal{P}_{K_B} \mathcal{W}^{-1} \mathcal{T}_{\partial B} \mathcal{S}^{-1}.$$

On remarque que l'inverse de l'opérateur de Stokes est compact (voir [70] par exemple). Comme A est la composée de l'inverse de l'opérateur de Stokes avec des opérateurs continus, on en déduit que A lui même est compact (voir [20]). □

Ce résultat nous permet de dire que K_B possède une base hilbertienne formée de vecteurs propres de A que l'on peut ordonner de telle sorte que la suite des valeurs propres de A soit décroissante et tende vers 0 (voir [20]). On peut donc décomposer les contrôles sur cette base de vecteurs propres. Pour appréhender la manière dont l'algorithme est susceptible de converger, nous nous intéressons maintenant à la suite des vecteurs propres de cet opérateur. À l'aide d'une méthode de la puissance, on peut chercher numériquement le vecteur propre $\mathbf{g}_0$ associés à la plus grande valeur propre λ_0. Pour trouver numériquement les vecteurs propres associés aux valeurs propres suivantes, nous utilisons une méthode de déflation. On note $\mathbf{g}_k$ les éléments de la base hilbertienne formée de vecteurs propres de A et λ_k les valeurs propres associées. On considère alors B_k l'opérateur défini par :

$$B_k = A - \sum_{i=0}^{k} \lambda_i(\mathbf{g}_i\,,\,.)\mathbf{g}_i\,.$$

C'est un opérateur dont la plus grande valeur propre est la k+1 ième valeur propre de A. La méthode de déflation consiste donc à utiliser une méthode de la puissance sur l'opérateur B_k au lieu de A ce qui permet de trouver les premiers vecteurs propres de A associés aux plus grandes valeurs propres de A. On présente tout d'abord le cas d'une particule de rayon 0.1 centrée au milieu du carré unité. Les figures 2.4, 2.5, 2.6 et 2.7 représentent les lignes de courant des premiers vecteurs propres trouvés numériquement. On constate qu'ils oscillent de plus en plus. Dans le cas où le contrôle est lisse, ceci suggère d'obtenir avec précision les coefficients correspondant aux grandes valeurs propres dans la décomposition du contrôle sur la base hilbertienne formée des vecteurs propres de l'opérateur pour avoir une bonne approximation du contrôle. Dans ce cas là, quelques itérations suffisent donc pour avoir une solution comme on le verra dans la partie suivante.

Les figures 2.8 et 2.9 représentent cette fois les premiers vecteurs propres dans le cas de deux particules éloignées. L'une se trouve au point $(0.5, 0.5)$ et l'autre au point $(0.5, 1.5)$ dans le rectangle $[0, 1] \times [0, 2]$. Les vecteurs propres sont tout à fait semblables au cas

FIGURE 2.4 – Vecteurs propres dans le cas d'une seule particule

FIGURE 2.5 – Vecteurs propres dans le cas d'une seule particule

FIGURE 2.6 – Vecteurs propres dans le cas d'une seule particule

FIGURE 2.7 – Vecteurs propres dans le cas d'une seule particule

FIGURE 2.8 – Vecteurs propres dans le cas de deux particules éloignées

d'une seule particule, elles ne se voient donc pas. On peut ainsi espérer qu'en régime dilué (peu de particules) les particules soient suffisamment éloignées les unes des autres et que la méthode utilisée pour minimiser la fonctionnelle converge à la même vitesse que dans le cas d'une seule particule.

En comparaison, les figures 2.10 et 2.11 représentent le cas de deux particules proches. La première est située au point $(0.5, 1.125)$ et l'autre au point $(0.5, 0.875)$. Elles se trouvent dans le même rectangle que dans le cas précédent et ont un rayon de 0.1. La distance les séparant est donc d'un demi rayon. On peut voir cette fois ci que l'allure des vecteurs propres est différente du cas d'une seule particule ou de deux particules éloignées. Ici, on peut s'attendre à ce que l'algorithme de minimisation de la fonctionnelle converge plus lentement.

2.4.4 Algorithme

Pour minimiser la fonctionnelle on va utiliser un algorithme de gradient conjugué. On présente tout d'abord les étapes nécessaires pour calculer l'image d'une fonction $\mathbf{g}$ de K_B par A :

FIGURE 2.9 – Vecteurs propres dans le cas de deux particules éloignées

FIGURE 2.10 – Vecteurs propres dans le cas de deux particules proches

FIGURE 2.11 – Vecteurs propres dans le cas de deux particules proches

Calcul de $A\mathbf{g}$:

1. Étant donnée une fonction $\mathbf{g}$ dans K_B, résoudre le problème 2.4

$$\begin{cases} -2\eta\,\mathrm{div}(D(\tilde{\mathbf{u}}_g)) + \nabla\tilde{p}_g &= \mathbf{g}\,\chi_B\,, & \text{dans } \Omega\,, \\ \nabla\cdot\tilde{\mathbf{u}}_g &= 0\,, & \text{dans } \Omega\,, \\ \tilde{\mathbf{u}}_g &= 0\,, & \text{sur } \partial\Omega\,. \end{cases}$$

2. Construire la distribution simple couche à partir de la solution $\tilde{\mathbf{u}}_g$

$$\sum_{i=1}^{N} (\tilde{\mathbf{u}}_g - \mathcal{R}_i(\tilde{\mathbf{u}}_g))\,\delta_{\partial B_i}\,.$$

3. Résoudre le problème 2.18

$$\begin{cases} -2\eta\,\mathrm{div}(D(\tilde{\mathbf{w}}_g)) + \nabla\tilde{\boldsymbol{\pi}}_g &= \displaystyle\sum_{i=1}^{N} (\tilde{\mathbf{u}}_g - \mathcal{R}_i(\tilde{\mathbf{u}}_g))\,\delta_{\partial B_i}\,, & \text{dans } \Omega\,, \\ \nabla\cdot\tilde{\mathbf{w}}_g &= 0\,, & \text{dans } \Omega\,, \\ \tilde{\mathbf{w}}_g &= 0\,, & \text{sur } \partial\Omega\,. \end{cases}$$

4. Projeter la solution $\tilde{\mathbf{w}}_g$ sur l'espace K_B en calculant $\mathcal{P}_{K_B}(\tilde{\mathbf{w}}_g)$:

$$\mathcal{P}_{K_B}(\tilde{\mathbf{w}}_g) = \sum_{i=1}^{N} \left(\tilde{\mathbf{w}}_g - \frac{1}{|B_i|}\int_{B_i} \tilde{\mathbf{w}}_g - \frac{d}{C_i(d)|B_i|(d-1)} \left(\int_{B_i} \mathbf{r}_i \wedge \tilde{\mathbf{w}}_g \right) \wedge \mathbf{r}_i \right) \chi_{B_i}\,.$$

Remarque 2.22. *Pour calculer l'image d'une fonction* $\mathbf{g}$ *par* A *il est donc nécessaire de résoudre deux problèmes de Stokes. L'avantage est que ce sont des problèmes de Stokes classiques, l'opérateur de Stokes n'est pas modifié pour imposer une contrainte par exemple. Les matrices sont donc les mêmes pour les deux problèmes et elles ne dépendent pas de la position des particules qui est prise en compte uniquement dans les second membres.*

Avant d'écrire l'algorithme de gradient conjugué, on rappelle que la fonctionnelle s'écrit :

$$J(\mathbf{g}) = (A\mathbf{g}\,,\,\mathbf{g}) + (\mathcal{P}_{K_B}(\mathbf{w}_0)\,,\,\mathbf{g}) + J(0)$$

et qu'on cherche donc à résoudre le problème :

$$A\mathbf{g} = -\mathcal{P}_{K_B}(\mathbf{w}_0)\,.$$

Algorithme de gradient conjugué :

1. Soit $\mathbf{g}_0$ un contrôle choisi initialement.

2. Calculer

$$\mathbf{r}_0 = -(\mathcal{P}_{K_B}(\mathbf{w}_0) + A\mathbf{g}_0) = -\mathcal{P}_{K_B}(\mathbf{w}_{g_0})$$

et poser $\mathbf{p}_0 = \mathbf{r}_0$

3. On pose $k = 0$. Tant que le critère d'arrêt n'est pas satisfait :
   (a) En utilisant l'algorithme précédent, calculer

$$\alpha_k = \frac{(\mathbf{r}_k\,,\,\mathbf{r}_k)}{(\mathbf{p}_k\,,\,A\mathbf{p}_k)}\,.$$

   (b) Mettre à jour le contrôle :

$$\mathbf{g}_{k+1} = \mathbf{g}_k + \alpha_k\mathbf{p}_k\,.$$

   (c) Calculer le nouveau résidu :

$$\mathbf{r}_{k+1} = \mathbf{r}_k - \alpha_k A\mathbf{p}_k\,.$$

   (d) Calculer le coefficient β_k :

$$\beta_k = \frac{(\mathbf{r}_{k+1}\,,\,\mathbf{r}_{k+1})}{\mathbf{r}_k\,,\,\mathbf{r}_k}\,.$$

   (e) Calculer la nouvelle direction de descente :

$$\mathbf{p}_{k+1} = \mathbf{r}_{k+1} + \beta_k\mathbf{p}_k\,.$$

   (f) Incrémenter $k = k + 1$.
4. Fin tant que.

Remarque 2.23. *Pour calculer le premier résidu* $\mathbf{r}_0$*, il faut calculer le gradient de* J *en* $\mathbf{g}_0$*. Pour cela, on utilise le même algorithme que celui pour calculer le produit* $A\mathbf{g}$ *mais en utilisant le problème 2.4 à la première étape au lieu de 2.15.*

Remarque 2.24. *Pour pouvoir calculer* $A\mathbf{g}$ *il faut que* $\mathbf{g}_k$ *soit dans l'espace* K_B *à chaque itération. Pour cela on choisi toujours* $\mathbf{g}_0$ *nulle, qui est bien dans* K_B*. On a ainsi* $\mathbf{r}_0$ *et* $\mathbf{p}_0$ *dans l'espace* K_B *puisqu'ils sont égaux à l'opposé de la projection de* $\mathbf{w}_0$ *sur* K_B*. Pour un* k *donné, on suppose que* $\mathbf{r}_k, \mathbf{p}_k$ *et* $\mathbf{g}_k$ *sont dans* K_B*. Alors on a bien* $\mathbf{g}_{k+1}$ *dans* K_B *car par définition, l'image de* A *est dans* K_B*. De même pour* $\mathbf{r}_{k+1}$ *et donc pour* $\mathbf{p}_{k+1}$*. Pour avoir une solution qui est dans* K_B*, il suffit donc de partir d'une fonction* $\mathbf{g}_0$ *dans* K_B *; on choisira de prendre la fonction nulle.*

Remarque 2.25. *On minimise une fonctionnelle quadratique en résolvant à l'aide d'un gradient conjugué un système linéaire du type* $A\mathbf{g} = \mathbf{b}$*. On attire alors l'attention sur le fait que même l'évaluation de* $A\mathbf{g}$ *pour un* $\mathbf{g}$ *donné n'est pas exacte puisque c'est la*

projection de la solution numérique d'un problème de Stokes lui même résolu par un algorithme itératif. L'algorithme global est donc l'imbrication de deux méthodes itératives (par exemple un solveur minres dans un solveur gradient conjugué).

Chapitre 3

Approximation d'une distribution simple couche par une combinaison de masses de Dirac

Sommaire

3.1 Introduction

Dans cette partie on s'intéresse à l'approximation numérique de la distribution simple couche qui apparaît dans le problème 2.18. On s'intéresse donc à des problèmes de la forme :

$$\begin{aligned} -\Delta u &= \varphi\delta_\gamma \quad \text{dans } \Omega\,, \\ u &= 0\,, \quad \text{sur } \partial\Omega\,, \end{aligned}$$

où Ω est un domaine borné de $\mathbb{R}^d$ et $\varphi\delta_\gamma$, dans $H^{-1}(\Omega)$, peut être écrit formellement comme :

$$\langle \varphi\delta_\gamma, v\rangle_{H^{-1},\,H^1_0} = \int_\gamma \varphi v\,,$$

où γ est une variété régulière de dimension $d-1$ (typiquement le bord d'un sous domaine connexe $\omega \subset\subset \Omega$) et φ est dans un espace de Sobolev fractionnaire $H^{-1/2+s}(\gamma)$ avec $s>0$.

Ce genre de problèmes sont régulièrement rencontrés en modélisation numérique. Peskin et Mc Queen (voir [64]) ont modélisé le mouvement des parois du coeur par une collection de fibres élastiques immergées dans un fluide. Les forces appliquées au fluide par ces fibres sont des distributions simple couche. Dans un contexte similaire, Gerbeau et ses collaborateurs [67] ont modélisé plus récemment une valve par une interface élastique immergée dans un fluide. De même, Cottet et Maitre dans [26] suivent l'interface d'une membrane élastique immergée dans un fluide par une méthode level-set et les forces élastiques appliquées sur la membrane apparaissent dans le terme source des équations de Navier-Stokes. Dans [40] Herrmann suit l'interface entre deux fluides par une méthode level-set et la tension de surface est une distribution simple couche.

On retrouve aussi ces problèmes de distribution simple couche dans le contexte des méthodes de domaine fictif. Dans [35] et [16] par exemple, les auteurs utilisent des multiplicateurs de Lagrange pour imposer des conditions de Dirichlet. Ces multiplicateurs de Lagrange agissent comme des distributions singulières de forces supportées par le bord. Dans [60], cette même approche est proposée pour simuler la sédimentation de particules rigides dans un fluide. La contrainte de mouvement rigide est imposée par des multiplicateurs de Lagrange supportés sur le bord des particules rigides. Ces distributions simple couche se retrouvent aussi dans la FBM (voir [56]) lors de la résolution des problèmes locaux autour des inclusions.

Il existe beaucoup d'approches pour résoudre ces problèmes numériquement, le choix dépendant du cadre général du problème discret. Dans [64], les auteurs ont choisi de résoudre le problème dans un cadre différence finie. Ils ont une grille sur le domaine tout entier pour résoudre la partie fluide et une collection uniforme de points sur les fibres, qui ne sont pas nécessairement les mêmes points que ceux de la grille. Pour prendre en compte les forces exercées par les fibres sur le fluide, ils régularisent les Dirac en chaque point des fibres pour étaler la distribution simple couche sur la grille du fluide. C'est la méthode de frontière immergée qu'on a déjà mentionnée dans une partie précédente. On peut trouver des applications de cette méthode dans la section 9 de [63] et, par exemple, dans [34,58,72,47] pour des plus récentes. Une autre façon de traiter les Dirac est d'injecter

directement le saut des dérivées normales dans le schéma différence finies, comme c'est le cas avec la méthode IIM de Leveque et Li (voir [51]).

Dans un cadre variationnel, il est possible de calculer les intégrales impliquant une distribution simple couche. Dans un contexte éléments finis, Glowinski et ses collaborateurs utilisent dans [35] des éléments finis constants dans chaque maille pour approcher les multiplicateurs de Lagrange. Les intégrales au niveau discret peuvent donc être calculées exactement. La méthode de Peskin à été adaptée à un contexte éléments finis dans [18,43]. Dans le même esprit, il est possible d'approcher la distribution simple couche par une combinaison de masses de Dirac sur des points décrivant l'interface. C'est la méthode utilisée dans [67,60,16] pour approcher les multiplicateurs de Lagrange définis sur les interfaces du domaine de calcul. Cette méthode ne requiert aucun maillage supplémentaire, seulement une collection de points décrivant l'interface, et peut s'implémenter dans un code relativement facilement. De plus les intégrales qui apparaissent dans la formulation variationnelle et faisant intervenir la distribution simple couche ne sont plus que des evaluations de fonctions en les points décrivant l'interface. On choisit d'approcher la distribution simple couche du problème 2.18 avec cette méthode.

À notre connaissance, cette approche n'a pas été justifié d'un point de vue théorique. Une des difficultés est qu'une fonctionnelle linéaire parfaitement définie, le terme source dans notre problème que l'on considérera dans H^{-1}, est remplacée par une combinaison de masses de Dirac, ce qui n'a *pas* de sens au niveau continu (dès que la dimension d est plus grande que 1) puisque les fonctions de H^1 ne sont pas continues.

L'objectif de cette partie est de donner une analyse numérique de cette méthode dans le cas d'un problème de Poisson en dimension 2 avec une distribution simple couche pour terme source. On établit tout d'abord une estimation de l'erreur dans le cas où φ est dans $H^{-1/2+s}$ avec $0 < s \leq 1/2$ et on montre que l'ordre de l'erreur sature dès que s est plus grand que 0.5 (lorsque φ est au moins dans L^2). Ensuite nous présentons quelques exemples d'applications sur lesquels cette estimation de l'erreur peut être utilisée pour obtenir l'erreur globale. Enfin, nous présenterons des résultats numériques pour valider l'analyse numérique.

3.2 Résultats théoriques

3.2.1 Lemmes d'approximations en espace

On démontre dans cette section les lemmes qui serviront dans la démonstration du théorème sur l'estimation de l'erreur en espace. Soit Ω un domaine borné de $\mathbb{R}^2$ et ω un sous domaine régulier de bord γ. On considère une distribution simple couche supportée sur γ :

$$\varphi\delta_\gamma \in H^{-1}(\Omega) \ : \ v \in V = H_0^1(\Omega) \longmapsto \langle \varphi,\, v \rangle_{H^{-1/2+s}(\gamma),\, H^{1/2-s}(\gamma)} \,,$$

avec $\varphi \in H^{-1/2+s}(\gamma)$, $0 \leq s < 1$. Cette expression a un sens car v est dans $H^1(\Omega)$ de telle sorte que sa trace est dans $H^{1/2}(\gamma) \subset H^{1/2-s}$.

Étant donnée une triangulation conforme T_h de Ω, on note V_h l'espace des éléments finis P^1 associés :

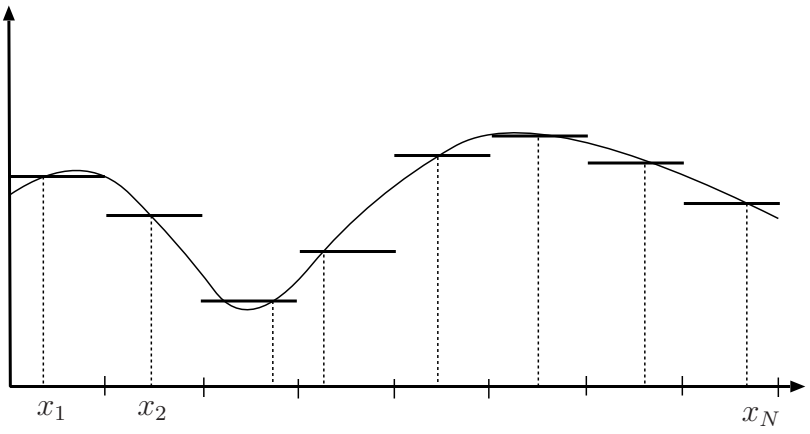

FIGURE 3.1 – interpolation P^0

$$V_h = \left\{ v_h \in C^0(\overline{\Omega}) \,,\ v_{h|K} \text{ est affine } ,\ \forall K \in T_h \right\}.$$

On s'intéresse à l'approximation de $\varphi\delta_\gamma$ dans V_h par une combinaison de masses de Dirac. Encore une fois cette formulation n'a pas de sens au niveau continu puisque $V' = H^{-1}(\Omega)$ ne contient aucune masse de Dirac. Le premier lemme énoncé concerne l'approximation d'une fonction H^1 en dimension 1 par son interpolée P^0.

Lemme 3.1. *Soit I l'intervalle unité $(0,1)$, $h > 0$, et (T_h) une famille quasi uniforme de triangulation de I. On représente T_h par ses sous intervalles γ_1, ..., γ_N (on n'écrit pas volontairement la dépendance de γ_i en h pour alléger les notations). La quasi uniformité exprime :*

$$ch \leq |\gamma_i| \leq Ch\,, \quad \textit{avec } 0 < c < C.$$

On considère alors x_1, ..., x_N, avec $x_i \in \gamma_i$. Pour tout $v \in H^1(I)$, on note v_h l'interpolation constante par morceaux correspondante (voir Fig. 3.1)

$$v_h = \sum_{i=1}^{N} v(x_i)\chi_{\gamma_i}.$$

Pour tout r, $0 \leq r < 1/2$, on a

$$\|v - v_h\|_{H^r} \leq Ch^{1-r} \left|v\right|_1$$

où $\left|v\right|_1$ est la semi norme H^1, et la norme de Sobolev fractionnaire est définie par :

$$\|w\|_{H^r} = \left(\int_I |w|^2 \right)^{1/2} + \left(\int_I \int_I \frac{|w(y) - w(x)|^2}{|y-x|^{1+2r}} \right)^{1/2}, \quad 0 < r < 1/2.$$

Preuve : On commence par l'intégrale sur $I \times I$. On note $w_h = v - v_h$ et en remarquant que pour tout x sur γ_i, $v_h(x) = v(x_i)$, on a :

$$\begin{aligned}\int_I\int_I \frac{|w_h(y)-w_h(x)|^2}{|y-x|^{1+2r}} &= \sum_i\sum_j \int_{\gamma_i}\int_{\gamma_j} \frac{|v(y)-v(x_j)-v(x)+v(x_i)|^2}{|y-x|^{1+2r}} \\ &= \sum_i \int_{\gamma_i}\int_{\gamma_i} \frac{|v(y)-v(x)|^2}{|y-x|^{1+2r}} + \sum_i\sum_{j\neq i} \int_{\gamma_i}\int_{\gamma_j} \frac{|w_h(y)-w_h(x)|^2}{|y-x|^{1+2r}} \\ &= A+B.\end{aligned}$$

Le premier terme (terme diagonal) s'écrit :

$$\begin{aligned}A = \sum_i \int_{\gamma_i}\int_{\gamma_i} \frac{|v(y)-v(x)|^2}{|y-x|^{1+2r}} &\leq \sum_i \int_{\gamma_i}\int_{\gamma_i} \frac{\int_{\gamma_i}|v'(t)|^2\,dt}{|y-x|^{2r}} \\ &\leq \frac{1}{(1-2r)} h^{2-2r} \int_I |v'(t)|^2 .\end{aligned}$$

Pour les termes extra diagonaux, on remarque tout d'abord que sur chaque γ_i,

$$|w_h|^2 = \left|\int_{x_i}^x v'(t)\,dt\right|^2 \leq h\int_{\gamma_i}|v'(t)|^2 .$$

Le second terme B peut être estimé :

$$\begin{aligned}B &\leq 4\sum_i\sum_{j\neq i}\int_{\gamma_i}\int_{\gamma_j}\frac{|w_h(x)|^2}{|y-x|^{1+2r}} \\ &\leq 4h\sum_i\int_{\gamma_i}|v'(t)|^2\sum_{j\neq i}\int_{\gamma_i}\int_{\gamma_j}\frac{1}{|y-x|^{1+2r}}.\end{aligned}$$

Dans le cas $r>0$, la dernière somme (sur $j\neq i$) est plus petite que deux fois

$$\begin{aligned}\int_0^h dx\int_h^1 \frac{dy}{|y-x|^{1+2r}} = -\frac{1}{2r}\int_0^h dx\left[(y-x)^{-2r}\right]_h^1 &\leq \frac{1}{2r}\int_0^h (h-x)^{-2r}\,dx \\ &\leq \frac{1}{2r(1-2r)}h^{1-2r},\end{aligned}$$

si bien que

$$B \leq \frac{4}{r(1-2r)}h^{2-2r}\int_I |v'(t)|^2 .$$

Ainsi, on obtient :

$$\int_I\int_I \frac{|w_h(y)-w_h(x)|^2}{|y-x|^{1+2r}} \leq \frac{C}{r(1-2r)}h^{2-2r}\,|v|_{1,I}^2 .$$

Le terme d'ordre 0 est simplement :

$$\int_I |w_h|^2 = \sum_i \int_{\gamma_i} |w_h|^2 \leq h^2 \, |v|^2_{1,I} \, .$$

D'où l'ordre $1 - r$ du lemme. □

Remarque 3.2. *Si $r = 0$, la norme H^r est simplement la norme L^2 et il n'y a alors qu'un terme d'ordre 0.*

Remarque 3.3. *Ce résultat peut être mis en défaut dans le cas $-1/2 < r < 0$. En effet, même si la définition de la semi norme dans la définition de la norme H^r peut être étendue pour $r < 0$, ce n'est pas le cas pour la norme L^2. On ne peut pas simplement l'enlever non plus car il faut contrôler les fonctions constantes d'une certaine façon. De plus l'intégrale sur $I \times I$ est d'ordre h^{2-2r} même si $r < 0$, ce doit donc aussi être vrai pour l'expression qui remplace la norme L^2 et qui contrôle les fonctions constantes. Par exemple, prendre la moyenne de la fonction n'est pas suffisant parce que c'est toujours un terme d'ordre 1. C'est pour cette raison que l'ordre ne peut, peut être, pas aller plus loin que 1, même dans le cas $r < 0$.*

Le second lemme est un lemme de trace pour les fonctions discrète. On peut majorer la trace d'une fonction P^1 par sa norme en volume mais on perd un $h^{-1/2}$:

Lemme 3.4. *Soit Ω le carré unité dans $\mathbb{R}^2$, et $\omega \subset\subset \Omega$ un sous domaine régulier dont on note γ le bord. Soit $(T_h)_h$ une famille régulière de triangulations, et V_h l'espace éléments finis P^1 associé, on suppose que le rayon de courbure de γ est plus grand que h. L'opérateur de trace γ_0 est surjectif de V_h (équippé de la norme H^1) dans $H^1(\gamma)$, avec*

$$|\gamma_0(v_h)|_{1,\gamma} \leq \frac{C}{h^{1/2}} \, |v_h|_{1,\Omega} \, .$$

Preuve : Parce que v_h est une fonction P^1, son gradient est constant sur chaque maille de T_h. Soit Q_γ l'ensemble des mailles qui ont une intersection non vide avec γ, on a

$$\begin{aligned} |\gamma_0(v_h)|^2_{1,\gamma} = \int_\gamma |\nabla v_h|^2 &= \sum_{Q \in Q_\gamma} |\nabla v_h|^2_{\infty,Q} \, |Q \cap \gamma| \\ &= \frac{1}{h^2} \sum_{Q \in Q_\gamma} |v_h|^2_{1,Q} \, |Q \cap \gamma| \, . \end{aligned}$$

Comme le rayon de courbure de γ est plus grand que h, l'inégalité suivante est vraie,

$$|Q \cap \gamma| \leq Ch \, ,$$

où C est une constante indépendante de h. Ainsi, l'inégalité précédente s'écrit :

$$\begin{aligned} |\gamma_0(v_h)|^2_{1,\gamma} &\leq \frac{C}{h} \sum_{Q \in Q_\gamma} |v_h|^2_{1,Q} \\ &\leq \frac{C}{h} |v_h|^2_{1,\Omega} , \end{aligned}$$

qui est l'estimation du lemme. □

Remarque 3.5. *La restriction sur le rayon de courbure de γ n'est pas forte car pour h assez petit elle est vraie si γ est suffisament régulière.*

3.2.2 Estimation de l'erreur d'approximation

On dispose des lemmes que l'on va utiliser pour démontrer le résultat sur l'estimation de l'erreur de l'approximation d'une distribution simple couche par une combinaison de Dirac. Le théorème est le suivant :

Théorème 3.6. *Soit Ω le carré unité de $\mathbb{R}^2$, et $\omega \subset\subset \Omega$ un sous domaine régulier de bord γ. Soit $(T_h)_h$ une famille régulière de triangulation et V_h l'espace éléments finis P^1 associé. Soit $0 < s \leq 1/2$ et $\varphi \in H^{-1/2+s}(\gamma)$. Soit $(S_{\tilde{h}})$ une famille quasi uniforme de triangulations de γ. On représente $S_{\tilde{h}}$ par ses sous intervalles $\gamma_1, \ldots, \gamma_{N_{\tilde{h}}}$. On considère $x_1, \ldots, x_{N_{\tilde{h}}}$, avec $x_i \in \gamma_i$ et l'approximation suivante de φ,*

$$\varphi_h^{\tilde{h}} = \sum_{i=1}^{N_{\tilde{h}}} \lambda_i \delta_{x_i} \tag{3.1}$$

où les réels λ_i sont choisis comme suit,

$$\lambda_i = \langle \varphi, \chi_{\gamma_i} \rangle . \tag{3.2}$$

Il existe une constante $C > 0$ telle que :

$$\left| \langle \varphi, v_h \rangle - \left\langle \varphi_h^{\tilde{h}}, v_h \right\rangle \right| \leq C \sqrt{\frac{\tilde{h}}{h}} \, \tilde{h}^s \, \|\varphi\|_{-1/2+s,\gamma} |v_h|_{1,\Omega} \tag{3.3}$$

pour toute fonction v_h dans V_h et où $\langle \ . \ , \ . \ \rangle$ dénote le crochet de dualité entre $H^{-1/2+s}(\gamma)$ et $H^{1/2-s}(\gamma)$.

Preuve : D'après la définition de $\varphi_h^{\tilde{h}}$, on a,

$$\begin{aligned} \left| \langle \varphi, v_h \rangle - \left\langle \varphi_h^{\tilde{h}}, v_h \right\rangle \right| &= \left| \langle \varphi, v_h \rangle - \left\langle \varphi, \sum_{i=1}^{N_{\tilde{h}}} v_h(x_i) \chi_{\gamma_i} \right\rangle \right| \\ &\leq \|\varphi\|_{-1/2+s,\gamma} \| v_h - \sum_{i=1}^{N_{\tilde{h}}} v_h(x_i) \chi_{\gamma_i} \|_{1/2-s,\gamma} . \end{aligned}$$

La fonction $\tilde{v}_{\tilde{h}}$ définie par

$$\tilde{v}_{\tilde{h}} = \sum_{i=1}^{N_{\tilde{h}}} v_h(x_i)\chi_{\gamma_i}$$

l'interpolation constante par morceaux de v_h sur γ. Afin de pouvoir appliquer le lemme 3.1 on effectue un changement de variable pour envoyer la courbe γ sur $[0,1]$,

$$\begin{array}{rccc} X: & [0,1] & \longrightarrow & \gamma \\ & t & \longrightarrow & X(t)\,. \end{array}$$

La variable t désigne l'abscisse curviligne donc $X(t)$ est le point sur γ tel que la longueur de l'arc $X(0)X(t)$ est égale à $|\gamma|\,t$. Ainsi, en notant $w_h^{\tilde{h}}$ la fonction $v_h - \tilde{v}_{\tilde{h}}$, on a,

$$\begin{aligned} \left|w_h^{\tilde{h}}\right|^2_{1/2-s,\gamma} &= \int_\gamma \int_\gamma \frac{\left|w_h^{\tilde{h}}(y) - w_h^{\tilde{h}}(x)\right|^2}{|y-x|^{2-2s}} \\ &= \int_0^1 \int_0^1 \frac{\left|w_h^{\tilde{h}} \circ X(t) - w_h^{\tilde{h}} \circ X(\tau)\right|^2}{|X(t) - X(\tau)|^{2-2s}} X'(t)X'(\tau)dtd\tau\,. \end{aligned}$$

Pour tout t dans $[0,1]$, $X'(t)$ est égal à $|\gamma|$ donc le changement de variable s'écrit :

$$\left|w_h^{\tilde{h}}\right|_{1/2-s,\gamma} = |\gamma| \left|w_h^{\tilde{h}} \circ X\right|_{1/2-s,[0,1]}\,.$$

La même chose est valable pour la norme L^2. La longueur des sous intervalles γ_i de γ est $\tilde{h}$, donc la longueur des sous intervalles correspondant dans $[0,1]$ est $\tilde{h}/|\gamma|$. On applique le lemme 3.1 avec $r = 1/2 - s$ et $h = \tilde{h}/|\gamma|$,

$$\begin{aligned} \|v_h - \tilde{v}_{\tilde{h}}\|_{1/2-s,\gamma} &= |\gamma|\, \|v_h \circ X - \tilde{v}_{\tilde{h}} \circ X\|_{1/2-s,[0,1]} \\ &\leq \frac{|\gamma|\, C}{|\gamma|^{1/2+s}}\, \tilde{h}^{1/2+s}\, |v_h \circ X|_{1,[0,1]}\,. \end{aligned}$$

Encore une fois, avec le changement de variable, on a,

$$\|v_h - \tilde{v}_{\tilde{h}}\|_{1/2-s,\gamma} \leq \frac{C}{|\gamma|^{1/2+s}}\, \tilde{h}^{1/2+s}\, |v_h|_{1,\gamma}\,.$$

Maintenant le lemme de trace 3.4 nous permet d'étendre la semi norme H^1 sur γ de v_h à tout le domaine Ω,

$$\|v_h - \tilde{v}_{\tilde{h}}\|_{1/2-s,\gamma} \leq \frac{C}{|\gamma|^{1/2+s}}\, \frac{\tilde{h}^{1/2+s}}{h^{1/2}}\, |v_h|_{1,\Omega}\,,$$

ce qui termine la preuve. □

Remarque 3.7. *Si $1/2 < s \leq 1$, on ne peut pas espérer avoir un meilleur ordre que celui pour $s = 1/2$. En effet, voici un exemple en dimension 1 où φ est une fonction régulière et où l'erreur est toujours d'ordre 1.*

Le domaine est $(0,\, 1)$ et φ est la fonction constante égale à 1. On prend $h = \tilde{h}/2$ et la fonction v_h est définie par :

$$v_h(x) = \begin{cases} -\dfrac{\left|x - \tilde{h}/2\right|}{h} + 1 & \text{dans } [0,\, \tilde{h}] \\ 0 & \text{sinon.} \end{cases}$$

Si on prend le point $\tilde{h}/2$ pour approcher φ dans $[0,\, \tilde{h}]$, l'erreur s'écrit,

$$\left|\langle \varphi,\, v_h\rangle - \left\langle \varphi_h^{\tilde{h}},\, v_h\right\rangle\right| = \left|\frac{\tilde{h}}{2} - \left(\int_0^{\tilde{h}} \varphi\right) v_h\left(\frac{\tilde{h}}{2}\right)\right| = \frac{\tilde{h}}{2}\,.$$

Ainsi l'erreur est d'ordre 1 dès que s est plus grand que 0.5.

Remarque 3.8. *Dans le cas $s = 0$ la définition de l'approximation de la distribution simple couche est différente car les fonctions caractéristiques ne sont pas dans $H^{1/2}$. Néanmoins, ces fonctions caractéristiques peuvent être remplacées par une partition de l'unité et le résultat du théorème 3.6 reste vrai. Ainsi, il y a toujours convergence si $\tilde{h}$ est tel que $\tilde{h}/h$ tend vers 0 quand h tend 0.*

3.3 Exemples d'applications

3.3.1 Un problème de Poisson

On présente dans cette partie un problème de Poisson avec condition de Dirichlet homogène et une distribution simple couche comme terme source. On s'intéresse à l'estimation d'erreur de ce problème lorsque la distribution simple couche est approchée par une combinaison de masses de Dirac.

Proposition 3.9. *Soit Ω le carré unité et $\omega \subset\subset \Omega$ un sous domaine avec un bord γ régulier. On considère le problème de Poisson suivant :*

$$\left|\begin{array}{rcll} -\Delta u & = & \varphi\delta_\gamma & \text{dans } \Omega \\ u & = & 0 & \text{sur } \partial\Omega \end{array}\right.$$

où φ est dans $H^{-1/2+s}(\gamma)$, $0 < s \leq 1/2$.

Soit $(T_h)_h$ une famille régulière de triangulation et V_h l'espace éléments finis P^1 associés. on approche φ par une fonction $\varphi_{\tilde{h}}^h$ selon les définitions 3.1 et 3.2. On note u_h l'approximation éléments finis de u et on suppose que $\tilde{h}$ est de l'ordre de h, alors l'estimation d'erreur est :

$$\|u - u_h\|_1 \leq C\left(\sqrt{h}|u|_2 + h^s\|\varphi\|_{-1/2+s,\gamma}\right)\,,$$

où $|.|_2$ dénote la semi norme H^2.

Preuve : Le lemme de Strang (voir [24]) nous donne l'estimation d'erreur suivante :

$$\|u-u_h\| \leq C\left(\inf_{v_h\in V_h}\|u-v_h\| + \sup_{w_h\in V_h}\frac{|\langle\varphi,w_h\rangle - \left\langle\varphi_h^{\tilde{h}},w_h\right\rangle|}{\|w_h\|}\right).$$

La première partie nous donne l'erreur habituelle et est d'ordre 1/2 dans notre cas car T_h est un maillage non conforme et V_h est l'espace éléments finis des fonctions P^1. La seconde partie du membre de droite de l'inégalité est donnée par le théorème 3.6. L'estimation d'erreur peut donc être précisée et devient :

$$\|u-u_h\|_1 \leq C\left(\sqrt{h}|u|_2 + \sqrt{\frac{\tilde{h}}{h}}\,\tilde{h}^s\|\varphi\|_{-1/2+s,\gamma}\right).$$

En particulier, si $\tilde{h}$ est égal à h, l'estimation d'erreur est :

$$\|u-u_h\|_1 \leq C\left(\sqrt{h}|u|_2 + h^s\|\varphi\|_{-1/2+s,\gamma}\right),$$

ce qui termine la preuve. □

3.3.2 Approximation de la solution d'un problème de point-selle

On considère dans cet exemple un problème de Poisson dans un domaine contenant une inclusion rigide avec des conditions de Dirichlet homogène aux bords. La condition de Dirichlet sur le bord de l'inclusion est imposée avec des multiplicateurs de Lagrange de bord. Ces multiplicateurs de Lagrange sont des distributions simples couches que l'on approche par une combinaison de masses de Dirac. On montre tout d'abord une proposition sur l'erreur d'approximation lorsque l'espace des multiplicateurs de Lagrange discret n'est pas inclus dans l'espace des multiplicateurs de Lagrange continu. Ensuite on utilise cette proposition et le théorème 3.6 pour obtenir une estimation de l'erreur du problème de Poisson.

Proposition 3.10. *Soit V un espace de Hilbert, a une forme bilinéaire continue coercive dans V et f une application de V'. Soit Λ un autre espace de Hilbert, $B \in \mathcal{L}(V,\Lambda)$ et K le noyau de B. On considère le problème consistant à trouver u dans K tel que $a(u,v) = \langle f, v\rangle$ pour tout v dans K, et sa formulation point-selle :*

$$\begin{array}{lcll} a(u,v) + \langle B^*\lambda, v\rangle & = & \langle f, v\rangle & \forall v \in V \\ (\mu, Bu) & = & 0 & \forall \mu \in \Lambda\,. \end{array}$$

Soit V_h un sous espace de dimension finie de V et $\Lambda_{\tilde{h}}$ un espace de dimension finie, non nécessairement inclus dans Λ. Soit $B_{\tilde{h}}^h$ une application linéaire continue de V dans $\Lambda_{\tilde{h}}$, on note $K_{\tilde{h}}^h$ l'approximation de K :

$$K_{\tilde{h}}^h = \left\{v_h \in V_h,\ \left((B_{\tilde{h}}^h)^* v_h,\, \mu_{\tilde{h}}\right) = 0 \quad \forall \mu_{\tilde{h}} \in \Lambda_{\tilde{h}}\right\}.$$

Le problème point-selle approché est :

$$\begin{aligned} a(u_h, v_h) + \left\langle (B_{\tilde{h}}^h)^* \lambda_{\tilde{h}},\, v_h \right\rangle &= \langle f,\, v_h \rangle \quad \forall v_h \in V_h \\ \left(\mu_{\tilde{h}},\, B_{\tilde{h}}^h\, u_h \right) &= 0 \qquad\quad \forall \mu_{\tilde{h}} \in \Lambda_{\tilde{h}}\,. \end{aligned}$$

On a alors l'estimation d'erreur suivante :

$$\|u - u_h\|_V \leq \left(1 + \frac{\|a\|}{\alpha}\right) \inf_{w_h \in K_{\tilde{h}}^h} \|w_h - u\| + \frac{1}{\alpha} \inf_{\mu_{\tilde{h}} \in \Lambda_{\tilde{h}}} \|\xi - \left(B_{\tilde{h}}^h\right)^* \mu_{\tilde{h}}\|_{V'}$$

où α est la constante de coercivité de a et ξ est la forme linéaire dans V définie par :

$$a(u, v) + \langle \xi, v \rangle = \langle f, v \rangle \quad \forall v \in V\,.$$

Preuve : u_h est la solution du problème point-selle discret, on a donc :

$$a(u_h, v_h) = \langle f, v_h \rangle \quad \forall v_h \in K_{\tilde{h}}^h\,.$$

Pour tout $w_h \in K_{\tilde{h}}^h$, on écrit $v_h = u_h - w_h \in K_{\tilde{h}}^h$. Ainsi,

$$\begin{aligned} a(v_h, v_h) &= a(u_h - w_h, v_h) \\ &= \langle f, v_h \rangle - a(w_h, v_h)\,. \end{aligned}$$

Comme u est la solution de notre problème on a :

$$a(u, v_h) + \langle \xi, v_h \rangle = \langle f, v_h \rangle\ .$$

Ainsi, pour tout $\mu_{\tilde{h}}$ dans $\Lambda_{\tilde{h}}$, on a :

$$\begin{aligned} a(v_h, v_h) =& a(u, v_h) + \langle \xi, v_h \rangle - a(w_h, v_h) - (B_{\tilde{h}}^h v_h, \mu_{\tilde{h}}) \\ =& a(u - w_h, v_h) + \left\langle \xi - \left(B_{\tilde{h}}^h\right)^* \mu_{\tilde{h}}, v_h \right\rangle\ . \end{aligned}$$

On prend le module de l'égalité précédente et on obtient :

$$\alpha \|v_h\|^2 \leq \|a\| \|u - w_h\| \|v_h\| + \|\xi - \left(B_{\tilde{h}}^h\right)^* \mu_{\tilde{h}}\|_{V'} \|v_h\|\,.$$

Mais $v_h = u_h - w_h$ on a donc :

$$\alpha \|u_h - w_h\| \leq \|a\| \|u - w_h\| + \|\xi - \left(B_{\tilde{h}}^h\right)^* \mu_{\tilde{h}}\|_{V'}\,.$$

Ainsi,

$$\begin{aligned} \|u - u_h\| &\leq \|u - w_h\| + \|w_h - u_h\| \\ &\leq \left(1 + \frac{\|a\|}{\alpha}\right) \|w_h - u\| + \frac{1}{\alpha} \|\xi - \left(B_{\tilde{h}}^h\right)^* \mu_{\tilde{h}}\|_{V'} \quad \forall w_h \in K_{\tilde{h}}^h\ , \mu_{\tilde{h}} \in \Lambda_{\tilde{h}}\,, \end{aligned}$$

ce qui est le résultat de l'énoncé. □

On présente maintenant le problème de Poisson qui nous intéresse et les choix que l'on fait pour approcher les différents espaces qui entres en jeu. On s'intéresse ensuite à l'ordre de l'erreur obtenu avec une telle discrétisation.

Problème continu : Soit Ω un domaine borné de $\mathbb{R}^2$ et $\mathcal{O}$ un sous espace ouvert non vide de Ω. On regarde le problème suivant :

$$(3.4)\quad \left\{ \begin{array}{rcll} -\Delta u & = & f & \text{dans } \Omega \setminus \bar{\mathcal{O}} \\ u & = & 0 & \text{sur } \gamma \\ u & = & 0 & \text{sur } \partial\Omega\,, \end{array} \right.$$

où f est une fonction de $L^2(\Omega \setminus \mathcal{O})$ et γ le bord de $\mathcal{O}$. On utilise une méthode de domaine fictif et on prolonge la fonction f par 0 dans $\mathcal{O}$. On note toujours f ce prolongement et on note V l'espace $H_0^1(\Omega)$ et K l'espace contraint :

$$K = \left\{ v \in V;\ v_{|\gamma} = 0 \right\}\,.$$

On impose la condition de Dirichlet avec des multiplicateurs de Lagrange. L'espace des multiplicateurs de Lagrange choisi est $\Lambda = L_0^2(\gamma)$, qui est l'espace des fonctions L^2 sur γ à moyenne nulle. Soit B l'application suivante :

$$\begin{array}{rccc} B: & V & \longrightarrow & \Lambda \\ & v & \longrightarrow & v_{|\gamma}\,. \end{array}$$

On a $K = \ker B$ et le problème point-selle correspondant est :

Trouver $(u, \lambda) \in V \times \Lambda$, tel que

$$\begin{array}{rcll} \displaystyle\int_\Omega \nabla u \cdot \nabla v + \int_\gamma \lambda v & = & \displaystyle\int_\Omega f v & \forall v \in V \\ \displaystyle\int_\gamma \mu u & = & 0 & \forall \mu \in \Lambda\,. \end{array}$$

Problème discret : Soit $\tilde{h} > 0$, on discrétise le bord γ en choisissant $N_{\tilde{h}}$ points $x_{i,\tilde{h}} \in \gamma$ équirépartis tels que que la distance entre deux points consécutifs est $\tilde{h}$. Soit $V_h \subset V$ un sous espace de dimension finie de V. L'espace des multiplicateurs de Lagrange et l'espace contraint sont approchés de la façon suivante :

$$\Lambda_{\tilde{h}} = \mathbb{R}^{N_{\tilde{h}}}$$

$$K_{\tilde{h}}^h = \left\{ v_h \in V_h;\ v_h(x_{i,\tilde{h}}) = 0 \right\}\,.$$

On peut remarquer qu'avec ce choix, l'espace $\Lambda_{\tilde{h}}$ n'est pas inclus dans Λ. L'application B devient :

$$\begin{array}{rccc} B_{\tilde{h}}^h: & V_h & \longrightarrow & \Lambda_{\tilde{h}} \\ & v_h & \longrightarrow & \left(v_h(x_{i,\tilde{h}})\right)_{i=1\ldots N_{\tilde{h}}}\,. \end{array}$$

On a alors le problème discrétisé suivant :

Trouver $(u_h, \lambda_{\tilde{h}}) \in V_h \times \Lambda_{\tilde{h}}$, tel que

$$\begin{aligned} a(u_h, v_h) + \left\langle \lambda_H,\, B^h_{\tilde{h}} v_h \right\rangle &= \langle f,\, v_h \rangle \quad \forall v_h \in V_h \\ \left\langle \mu_{\tilde{h}},\, B^h_{\tilde{h}} u_h \right\rangle &= 0 \qquad\quad \forall \mu_{\tilde{h}} \in \Lambda_{\tilde{h}}\,. \end{aligned}$$

Où a est la forme bilinéaire définie sur $V \times V$ par

$$a(u, v) = \int_\Omega \nabla u \cdot \nabla v\,.$$

Estimation d'erreur : On applique la proposition 3.10 et on obtient l'estimation d'erreur suivante :

$$\|u - u_h\|_V \leq \left(1 + \frac{\|a\|}{\alpha}\right) \inf_{w_h \in K^h_{\tilde{h}}} \|w_h - u\| + \frac{1}{\alpha} \inf_{\mu_{\tilde{h}} \in \Lambda_{\tilde{h}}} \|\xi - \left(B^h_{\tilde{h}}\right)^* \mu_{\tilde{h}}\|_{V'}\,.$$

Comme dans l'exemple précédent, la première partie de l'estimation est l'erreur usuelle qui traduit la qualité d'approximation de l'espace V par V_h. La seconde partie est l'erreur due à l'approximation de la forme linéraire ξ par une combinaison de masses de Dirac. En effet, pour tout $\mu_{\tilde{h}} = (\mu_i)_{1,\ldots,N_{\tilde{h}}}$ dans $\Lambda_{\tilde{h}}$ et pour tout v_h in V_h, on a :

$$\left\langle \left(B^h_{\tilde{h}}\right)^* \mu_{\tilde{h}}\ ,\ v_h \right\rangle = \left(\mu_{\tilde{h}}\ ,\ B^h_{\tilde{h}} v_h\right) = \sum_{i=1}^{N_{\tilde{h}}} \mu_i v_h(x_i) = \left\langle \sum_{i=1}^{N_{\tilde{h}}} \mu_i \delta_{x_i}\ ,\ v_h \right\rangle\,.$$

L'estimation d'erreur de la proposition 3.10 devient alors,

$$\|u - u_h\|_1 \leq \left(1 + \frac{\|a\|}{\alpha}\right) \inf_{w_h \in K^h_H} \|w_h - u\| + \frac{1}{\alpha} \inf_{\mu_{\tilde{h}} \in \Lambda_{\tilde{h}}} \frac{\left\langle \xi - \sum_{i=1}^{N_{\tilde{h}}} \mu_i \delta_{x_i}\ ,\ v_h \right\rangle}{\|v_h\|}\,.$$

Ainsi, en appliquant le résultat du théorème 3.6, on a :

$$\|u - u_h\|_1 \leq C \left(\sqrt{h}\|u\|_1 + \sqrt{\frac{\tilde{h}}{h}}\ \tilde{h}^{1/2} \|\xi\|_{0,\gamma}\right)\,.$$

Remarque 3.11. *Si on choisit $\tilde{h}$ égal à h, l'estimation devient :*

$$\|u - u_h\|_1 \leq C\sqrt{h}\,(\|u\|_1 + \|\xi\|_{0,\gamma})\,.$$

Dans ce cas, l'ordre de l'erreur est toujours de $1/2$ en norme H^1 et on ne perd donc rien en approchant les multiplicateurs de Lagrange de cette façon. Si par contre l'approximation de V par V_h était d'ordre supérieur à $1/2$, alors l'erreur serait restreinte par le choix que l'on a fait pour approcher les multiplicateurs de Lagrange.

3.4 Validation des résultats

Dans cette section nous présentons deux exemples pour valider l'estimation d'erreur 3.3 du théorème 3.6. Le premier est un exemple en dimension 1 qui montre que dans le cas $\tilde{h} = h$ l'ordre de l'erreur est celui prédit par le théorème 3.6. Le second est un exemple en dimension 2 où la solution d'un problème de Poisson avec une distribution simple couche en terme source est calculée numériquement. Ce terme source est le Laplacien d'une fonction avec un saut des dérivées normales à travers une interface. La solution exacte est donc connue et on calcule l'erreur numérique dans le cas $\tilde{h} = h$ pour plusieurs valeurs de s.

3.4.1 Un cas test 1D

Soit φ la fonction définie par :

$$\varphi = \frac{1}{x^{1-s}}\,,$$

où s est positif, et v_h est la fonction affine définie dans $[0,\, \tilde{h}]$ par :

$$v_h(x) = -\frac{\left|x - \frac{\tilde{h}}{2}\right|}{h} + 1\,.$$

L'estimation du théorème 3.6 peut être calculée explicitement. En effet, on a :

$$\langle \varphi,\, v_h \rangle = \int_0^{\tilde{h}} \varphi v_h = \frac{\tilde{h}^s}{s} + \left(\frac{1}{s(s+1)} - \frac{1}{s(s+1)2^s} - \frac{1}{2s}\right)\frac{\tilde{h}^{s+1}}{h}\,.$$

Soit x_i dans $[0,\, \tilde{h}]$, la seconde partie est :

$$\left\langle \varphi_h^{\tilde{h}},\, v_h \right\rangle = \left(\int_0^{\tilde{h}} \varphi\right) v_h(x_i) = \left(-\frac{\left|x_i - \frac{\tilde{h}}{2}\right|}{h} + 1\right)\frac{\tilde{h}^s}{s}\,.$$

Soit $0 \leq \alpha \leq 1$, on pose $x_i = \alpha\tilde{h}$. L'estimation de l'erreur s'écrit alors :

$$\left|\langle \varphi,\, v_h \rangle - \left\langle \varphi_h^{\tilde{h}},\, v_h \right\rangle\right| = C(s)\frac{\tilde{h}^{s+1}}{h}\,,$$

où $C(s)$ est une constante qui ne dépend que de s. Si $h = C\tilde{h}$, où C est une constante, l'erreur est d'ordre s qui est exactement l'erreur prédite par le théorème 3.6.

3.4.2 Validation numérique de l'exemple du problème de Poisson

Dans cette section Ω est le carré unité et γ est un cercle de rayon R. Pour étudier le comportement de l'approche pour un terme source $\varphi\delta_\gamma$, avec φ dans $H^{-1/2+s}(\gamma)$, on construit une fonction φ comme une série infinie de fonctions sinus :

$$\varphi = C\sum_{n=1}^{\infty} n^{-s}\sin(n\theta)$$

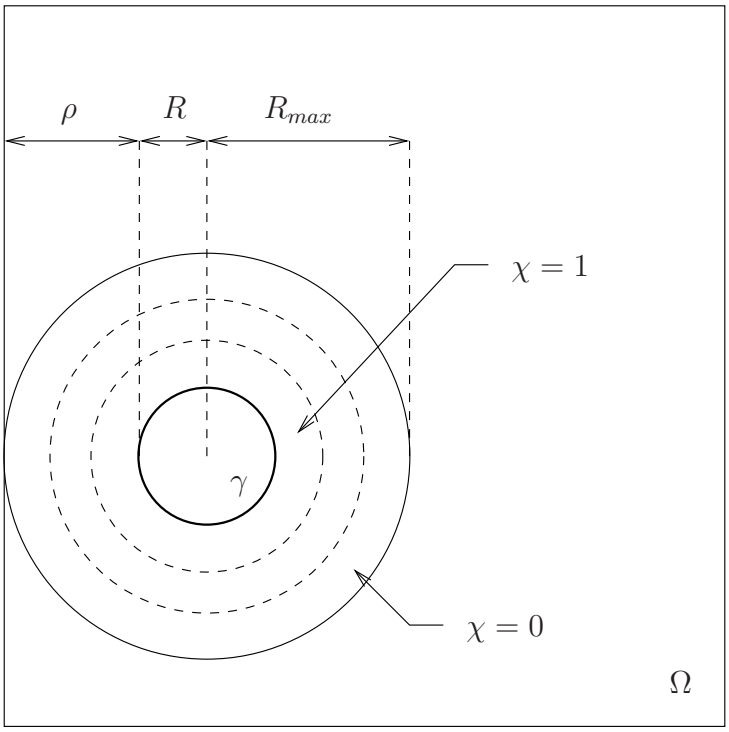

FIGURE 3.2 – Notations

On construit maintenant une fonction u telle que le saut de ses dérivées normales sur γ soit la fonction φ si bien que u est la solution d'un problème de Poisson de la forme :

$$\left\{ \begin{array}{rcll} -\Delta u & = & \varphi\delta_\gamma + f & \text{in } \Omega \\ u & = & 0 & \text{on } \partial\Omega \end{array} \right.$$

où f sera définie plus loin.

On a tout d'abord besoin d'une fonction chapeau régulière pour s'assurer que la solution u soit égale à zéro sur le bord de Ω. On note χ_ϵ la fonction C^2 définie dans $\mathbb{R}$ par

$$\chi_\epsilon(x) = \left\{ \begin{array}{ll} 1 & \text{si } x \leq 0 \\ \dfrac{-6}{\epsilon^5}x^5 + \dfrac{15}{\epsilon^4}x^4 - \dfrac{10}{\epsilon^3}x^3 + 1 & \text{si } 0 < x < \epsilon \\ 0 & \text{si } x \geq \epsilon\,. \end{array} \right.$$

On définit ρ comme la distance entre γ et $\partial\Omega$ (voir figure 3.2). On note R_{max} la somme de R et ρ. On peut maintenant définir la fonction suivante :

$$u(r,\theta) = \left\{ \begin{array}{ll} \chi_{\rho/3}\left(r - \left(R + \dfrac{\rho}{3}\right)\right) \displaystyle\sum_{n=1}^{\infty} \left(\frac{r}{R}\right)^{-n} n^{-s-1} \sin(n\theta) & \text{si } \ r \geq R \\ u\left(r + \left(1 - \dfrac{r}{R}\right) R_{max}\right) & \text{si } \ r < R. \end{array} \right.$$

C'est une fonction qui a un saut de ses dérivées normales sur γ et oscille très rapidement

proche de γ. Son Laplacien en dehors des particules est donné par :

$$\begin{aligned}\Delta u(r,\theta) = \chi''_{\rho/3}\left(r-\left(R+\frac{\rho}{3}\right)\right)\sum_{n=1}^{\infty}\left(\frac{r}{R}\right)^{-n} n^{-s-1}\sin(n\theta)\\ + \chi'_{\rho/3}\left(r-\left(R+\frac{\rho}{3}\right)\right)\sum_{n=1}^{\infty}\frac{1-2n}{R}\left(\frac{r}{R}\right)^{-n-1} n^{-s-1}\sin(n\theta)\,.\end{aligned}$$

Le saut φ de ses dérivées normales est :

$$\begin{aligned}\varphi &= \left(\left(1-\frac{R_{max}}{R}\right)-1\right)\frac{\partial u}{\partial r}(R,\theta)\\ &= \frac{R_{max}}{R^2}\sum_{n=1}^{\infty} n^{-s}\sin(n\theta)\end{aligned}$$

qui est dans $H^{-1/2+s-\epsilon}(\gamma)$ pour tout $\epsilon > 0$. Cela signifie qu'on a la solution exacte du problème suivant :

$$\left\{\begin{array}{rcll} -\Delta u &=& \varphi\delta_\gamma + f & \text{in } \Omega\\ u &=& 0 & \text{on } \partial\Omega\end{array}\right.$$

où f est donné par le Laplacien de u en dehors des particules.

Pour calculer la solution numérique de ce problème, on tronque la série à un ordre N plus grand que $2\pi/h$ de telle sorte qu'il y ait au moins une période de la fonction $\sin(N\theta)$ dans une maille du maillage. On calcule la solution numérique pour plusieurs s avec des éléments finis Q_1 et on la compare avec la solution exacte (voir figure 3.3). Dans tous les tests, l'ordre N est égal à 2^{12} et h prend des valeurs comprises entre 2^{-6} et 2^{-10}. Le pas $\tilde{h}$ est égal à h afin que la norme H^1 de l'erreur soit de la forme :

$$\|u-u_h\|_1 \leq C\left(\sqrt{h}|u|_2 + h^s\|\varphi\|_{-1/2+s,\gamma}\right).$$

On peut voir sur la figure 3.3 que l'ordre de l'erreur semble rester proche de 0.6 dès que s est plus grand que 0.5. Cela est dû à l'ordre d'approximation en espace de 1/2 car on utilise un maillage cartésien, donc non conforme, avec des éléments finis d'ordre 1, mais cela pourrait aussi être dû à la saturation de l'ordre lorsque s est plus grand que 0.5 (voir la remarque 3.7).

La figure 3.4 représente l'ordre de l'erreur en fonction de s. Lorsque s est plus petit que 0.5, la pente est égale à 1 ce qui est ce que l'on attend. L'ordre numérique est meilleur que celui trouvé théoriquement, mais ce phénomène de super-convergence arrive souvent lorsque l'on calcule des solutions numériques d'EDP.

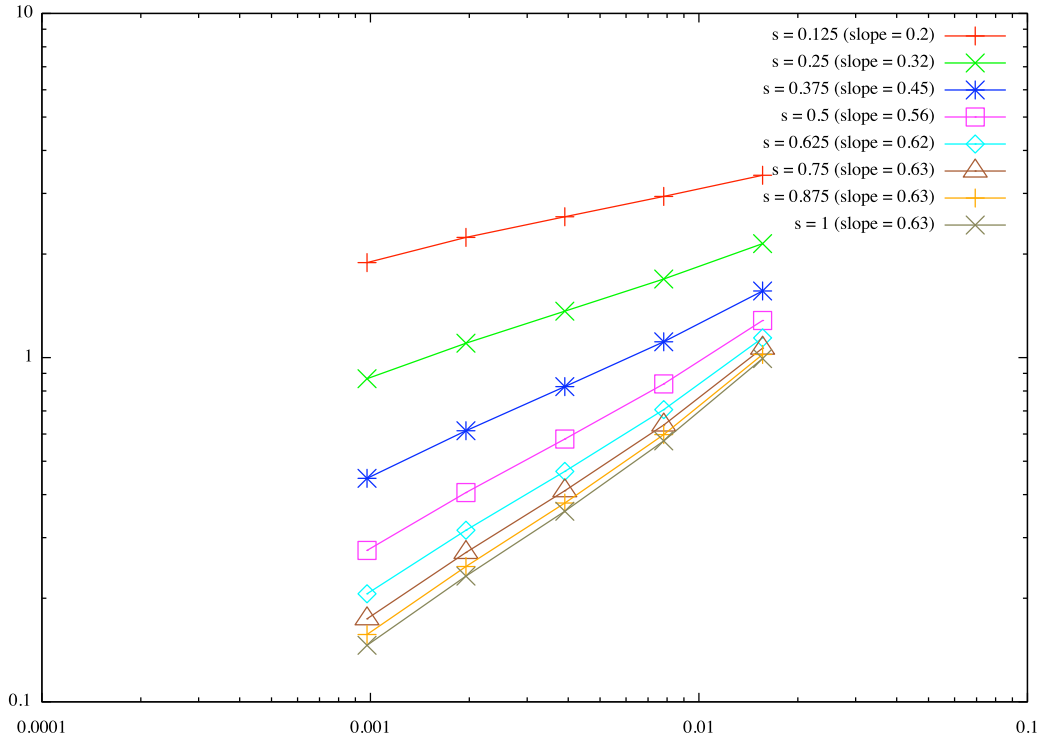

FIGURE 3.3 – Erreur H^1 pour quelques valeurs de s

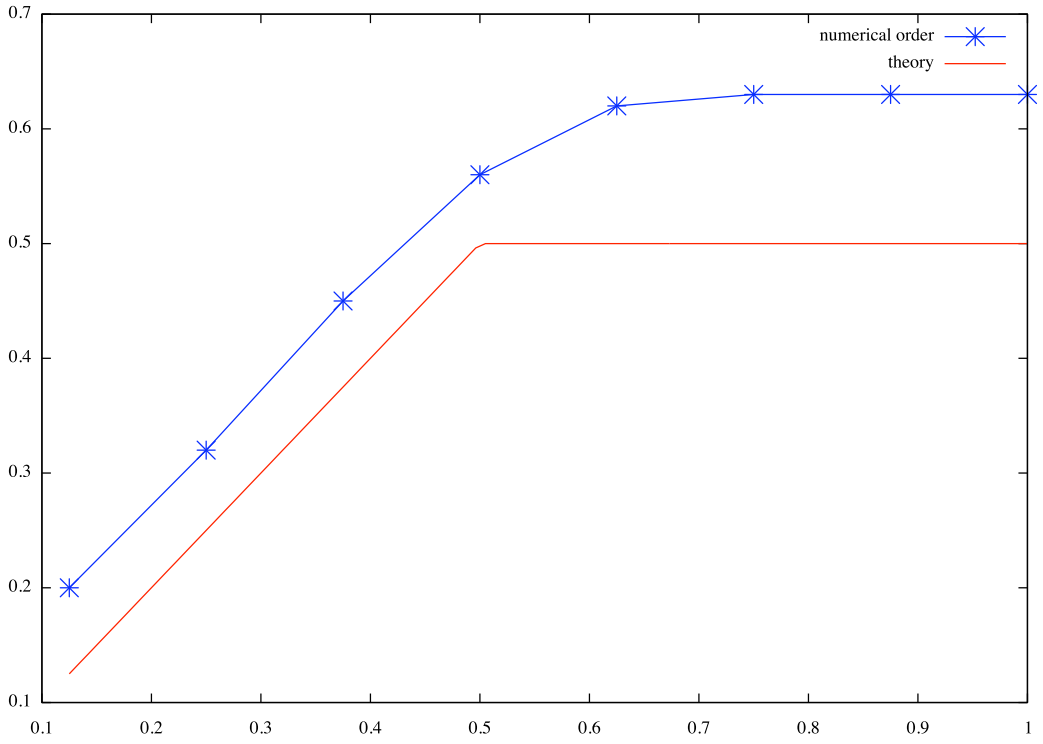

FIGURE 3.4 – Ordre de l'erreur H^1 en fonction de s

Chapitre 4

Analyse de l'erreur : théorie et validation numérique

Sommaire

4.1 Analyse de l'erreur dans le cas scalaire

Dans cette partie on donne des résultats sur l'estimation d'erreur de la méthode pour un problème de Poisson dans le but de montrer qu'elle est bien d'ordre 1 en norme H^1. Ensuite on utilise le même cas test que l'on a utilisé dans l'état de l'art pour obtenir les courbes de convergence et pour montrer que sur ce cas test, on retrouve bien l'ordre 1 en norme H^1.

4.1.1 Analyse numérique de l'erreur

Le cas que l'on étudie pour l'analyse de l'erreur est celui du problème jouet 1.8 de la partie état de l'art que l'on rappelle ici :

$$(4.1)\quad \begin{cases} -\Delta u &= f\,, & \text{dans } \Omega \setminus \bar{B}\,, \\ u &= 0\,, & \text{sur } \partial\left(\Omega \setminus \bar{B}\right)\,, \end{cases}$$

où Ω désigne le carré unité et B une inclusion sphérique dans Ω. La méthode de prolongement régulier appliquée à ce problème consiste à résoudre le problème 2.1 :

$$(4.2)\quad \begin{cases} -\Delta u_g &= f\chi_{\Omega\setminus\bar{B}} + g\chi_B\,, & \text{dans } \Omega\,, \\ u_g &= 0\,, & \text{sur } \partial\Omega\,. \end{cases}$$

Ici on impose des conditions de Dirichlet homogène sur le bord ∂B, on minimise donc la fonctionnelle suivante :

$$J(g) = \frac{1}{2}\int_B u_g^2\,,$$

qui est aussi une fonctionnelle quadratique dont le gradient est donné par la restriction à B de la solution du problème de Poisson suivant :

$$(4.3)\quad \begin{cases} -\Delta w_g &= u_g\delta_{\partial B}\,, & \text{dans } \Omega\,, \\ w_g &= 0\,, & \text{sur } \partial\Omega\,. \end{cases}$$

Remarque 4.1. *On ne fait pas les démonstrations de la forme du gradient puisqu'elles sont très similaires à celles du problème initial 1.1 en plus simple. On peut remarquer que dans ce cas là, il n'y a pas besoin de projeter la solution w_g sur le bon espace pour obtenir le gradient comme dans le cas des particules rigides puisqu'ici on impose une condition de Dirichlet connue, on n'a donc pas besoin d'équations supplémentaires.*

On reprend les notations de la partie précédente et on énonce maintenant un lemme sur l'erreur en norme L^2 du problème 4.3 lorsque la distribution simple couche $u_g\delta_{\partial B}$ est approchée par une combinaison de masses de Dirac. On montre qu'il est possible de récupérer l'ordre 1 en norme L^2 :

Lemme 4.2. *Soit g dans $L^2(B)$, prolongé par zéro dans le domaine $\Omega \setminus \bar{B}$. On note u_g la solution du problème 4.2 et w_g celle de 4.3.*

Soit $(T_h)_h$ une famille régulière de triangulations et V_h l'espace éléments finis P^1 associé. On note alors $w_h^{\tilde{h}}$ la solution approchée de 4.3 dans V_h. Alors, si $\tilde{h} = h^{3/2}$, on a l'estimation suivante :

$$\|w_g - w_h^{\tilde{h}}\|_{0,\Omega} \leq Ch\,.$$

Remarque 4.3. *Le problème 4.3 est un problème de Poisson avec une distribution simple couche dans le second membre et donc, tout comme dans l'exemple de la partie précédente, on ne s'attend pas à retrouver l'ordre optimal puiqu'on dispose d'un maillage fixe non conforme à la géométrie (et même cartésien en pratique). Par contre, ce lemme montre qu'on peut quand même obtenir l'ordre 1 en norme L^2 si on choisi $\tilde{h}$ convenablement en fonction de h.*

Preuve du lemme 4.2 : On adapte la preuve du lemme de Aubin-Nitsche pour montrer l'ordre 2 en norme L^2 dans le cas d'un second membre régulier. On considère le problème auxilliaire suivant :

$$(4.4)\quad \left\{ \begin{array}{rcll} -\Delta v & = & w_g - w_h^{\tilde{h}}\,, & \text{dans } \Omega\,, \\ v & = & 0\,, & \text{sur } \partial\Omega\,. \end{array} \right.$$

On note $\varphi = u_g \delta_{\partial B}$ la distribution simple couche du problème 4.3 et $\varphi_h^{\tilde{h}}$ son approximation par une combinaison de masses de Dirac. La formulation variationnelle du problème continu est :

$$\int_\Omega \nabla w_g \cdot \nabla \phi = \langle \varphi\,,\, \phi \rangle \quad \forall \phi \in H_0^1(\Omega)\,,$$

où $< .\,,\, . >$ est le crochet de dualité entre $H^{-1}(\Omega)$ et $H_0^1(\Omega)$. On note alors v_h l'approximation P^1 de la solution du problème 4.4 que l'on choisit comme fonction test dans la formulation variationnelle précédente :

$$\int_\Omega \nabla w_g \cdot \nabla v_h = \langle \varphi\,,\, v_h \rangle\,. \tag{4.5}$$

On refait exactement la même chose mais sur le problème discret approchant 4.3 et on obtient :

$$\int_\Omega \nabla w_h^{\tilde{h}} \cdot \nabla v_h = \left\langle \varphi_h^{\tilde{h}}\,,\, v_h \right\rangle\,. \tag{4.6}$$

On soustrait alors 4.5 et 4.6 ce qui nous donne :

$$\int_\Omega \nabla (w_g - w_h^{\tilde{h}}) \cdot \nabla v_h = \left\langle (\varphi - \varphi_h^{\tilde{h}})\,,\, v_h \right\rangle\,. \tag{4.7}$$

On considère maintenant la formulation variationnelle du problème auxilliaire 4.4 :

$$\int_\Omega \nabla v \cdot \nabla \phi = \int_\Omega (w_g - w_h^{\tilde{h}})\phi \quad \forall \phi \in H_0^1(\Omega)\,.$$

On choisit une fonction test particulière $\phi = w_g - w_h^{\tilde{h}}$ qui est bien dans $H_0^1(\Omega)$ et on obtient :

$$\int_\Omega \nabla v \cdot \nabla (w_g - w_h^{\tilde{h}}) = \int_\Omega |w_g - w_h^{\tilde{h}}|^2\,. \tag{4.8}$$

En soustrayant cette fois ci 4.7 et 4.8 on obtient la relation suivante :

$$\int_\Omega \nabla(v - v_h) \cdot \nabla(w_g - w_h^{\tilde{h}}) - \left\langle (\varphi - \varphi_h^{\tilde{h}}) \, , \, v_h \right\rangle = \int_\Omega |w_g - w_h^{\tilde{h}}|^2 \, .$$

Avec une inégalité de Cauchy-Schwarz on a alors :

$$\|w_g - w_h^{\tilde{h}}\|_{0,\Omega}^2 \leq |v - v_h|_{1,\Omega} \left| w_g - w_h^{\tilde{h}} \right|_{1,\Omega} + \left| \left\langle (\varphi - \varphi_h^{\tilde{h}}) \, , \, v_h \right\rangle \right| \, ,$$

où $|.|_{1,\Omega}$ désigne la semi-norme H^1 sur Ω. On estime chaque partie :

1. Le terme $|v - v_h|_{1,\Omega}$ est l'erreur classique due à l'approximation de la solution v par des éléments finis d'ordre 1. Comme le second membre est L^2 on obtient une erreur d'ordre 1 en norme H^1 :
$$|v - v_h|_{1,\Omega} \leq Ch|v|_{2,\Omega} \, ,$$
où $|.|_{2,\Omega}$ est la semi-norme H^2. On peut alors majorer cette semi-norme H^2 par la norme L^2 du second membre grâce à la continuité de la solution par rapport aux données, ce qui nous donne l'estimation :
$$|v - v_h|_{1,\Omega} \leq Ch \, \|w_g - w_h^{\tilde{h}}\|_{0,\Omega} \, .$$

2. Le terme $\left| w_g - w_h^{\tilde{h}} \right|_{1,\Omega}$ est d'ordre 1/2 puisque w_g est une fonction avec un saut des dérivées normales à travers le bord ∂B et que l'on considère des maillages non conformes à la géométrie. On a donc pour ce terme :
$$\left| w_g - w_h^{\tilde{h}} \right|_{1,\Omega} \leq C\sqrt{h} \, .$$

3. Enfin l'erreur sur la distribution simple couche est donnée par le résultat du théorème 3.6 :
$$\left| \left\langle (\varphi - \varphi_h^{\tilde{h}}) \, , \, v_h \right\rangle \right| \leq C \sqrt{\frac{\tilde{h}}{h}} \, \tilde{h}^s \, \|u_{g_{|\partial B}}\|_{-1/2+s,\partial B} |v_h|_{1,\Omega} \, .$$
Dans notre cas $s = 2$ car u_g est dans $H^2(\Omega)$ et donc sa trace sur ∂B est dans $H^{3/2}(\partial B)$. On a cependant vu dans la remarque 3.7 qu'il y avait une saturation de l'ordre de l'erreur lorsque s était plus grand que 1/2. De plus, d'après l'énoncé du lemme, on prend $\tilde{h} = h^{3/2}$ ce qui conduit à l'estimation suivante :
$$\left| \left\langle (\varphi - \varphi_h^{\tilde{h}}) \, , \, v_h \right\rangle \right| \leq C \, h \, |v_h|_{1,\Omega} \, .$$
En prenant dans la formulation variationnelle vérifiée par v_h la fonction test égale à v_h, on obtient l'inégalité $|v_h|_{1,\Omega} \leq C\|w_g - w_h^{\tilde{h}}\|_{0,\Omega}$ ce qui nous donne finalement l'estimation :
$$\left| \left\langle (\varphi - \varphi_h^{\tilde{h}}) \, , \, v_h \right\rangle \right| \leq Ch \, \|w_g - w_h^{\tilde{h}}\|_{0,\Omega} \, .$$

Pour finir, on remplace toutes ces estimations dans l'estimation globale et on obtient alors :

$$\|w_g - w_h^{\tilde{h}}\|_{0,\Omega} \leq Ch\,,$$

qui est bien le résultat de l'énoncé. □

Proposition 4.4. *Soit g une fonction de $L^2(B)$, u_g et w_g les solutions respectives de 4.2 et 4.3. Soit $(T_h)_h$ une famille régulière de triangulations et V_h l'espace éléments finis P^1 associés.*

Soit $\tilde{h} = h^{3/2}$, on note w_h la solution approchée de 4.3 dans V_h en approchant la distribution simple couche $u_g\delta_{\partial B}$ par une combinaison de masses de Dirac comme dans le théorème 3.6. On note enfin u_h la solution éléments finis du problème 4.2 dans V_h avec w_h comme contrôle. On a alors l'estimation d'erreur suivante :

$$|u_h - u_{w_g}|_{1,\Omega} \leq Ch\,.$$

Preuve : On reprend toutes les notations du théorème 3.6. En appliquant le lemme de Strang au problème 4.2 on a :

$$\|u_{w_g} - u_h\|_{1,\Omega} \leq C\left(\inf_{v_h\in V_h}\|u_{w_g} - v_h\|_{1,\Omega} + \sup_{v_h\in V_h}\frac{1}{\|v_h\|_{1,\Omega}}\int_\Omega (w_g - w_h)v_h\right)\,.$$

Après une inégalité de Cauchy-Schwartz, la deuxième partie se majore par l'erreur L^2 de w_h. Cette erreur est estimée dans le lemme 4.2 et est d'ordre 1 dans les conditions de l'énoncé de la proposition. La première partie de l'estimation quant à elle est d'ordre 1 puisque c'est l'estimation classique de l'erreur avec des éléments finis d'ordre 1. □

Remarque 4.5. *La résolution du problème contenant la distribution simple couche n'est pas faite de manière optimale parce qu'on ne dispose pas d'un maillage conforme. Néanmoins, la solution de ce problème apparaît comme un second membre dans le problème 4.2 et donc son erreur en norme L^2 intervient. Ainsi même si l'ordre de l'erreur n'est pas optimal pour le problème 4.3, l'ordre en norme L^2 est suffisamment élevé pour pouvoir l'obtenir dans le problème qui nous donnera la solution du problème initial.*

Remarque 4.6. *Ici on a approché la distribution simple couche par une combinaison de masses de Dirac et on a montré que sous certaines hypothèses on obtenait bien l'ordre 1 sur le problème 4.2. Bien entendu, toute autre approximation de la distribution simple couche d'ordre 1 en norme L^2 fonctionne pour obtenir cet ordre. On pourrait par exemple considérer un maillage $1D$ du bord des particules et approcher la distribution simple couche par des éléments finis P^0 sur ce maillage.*

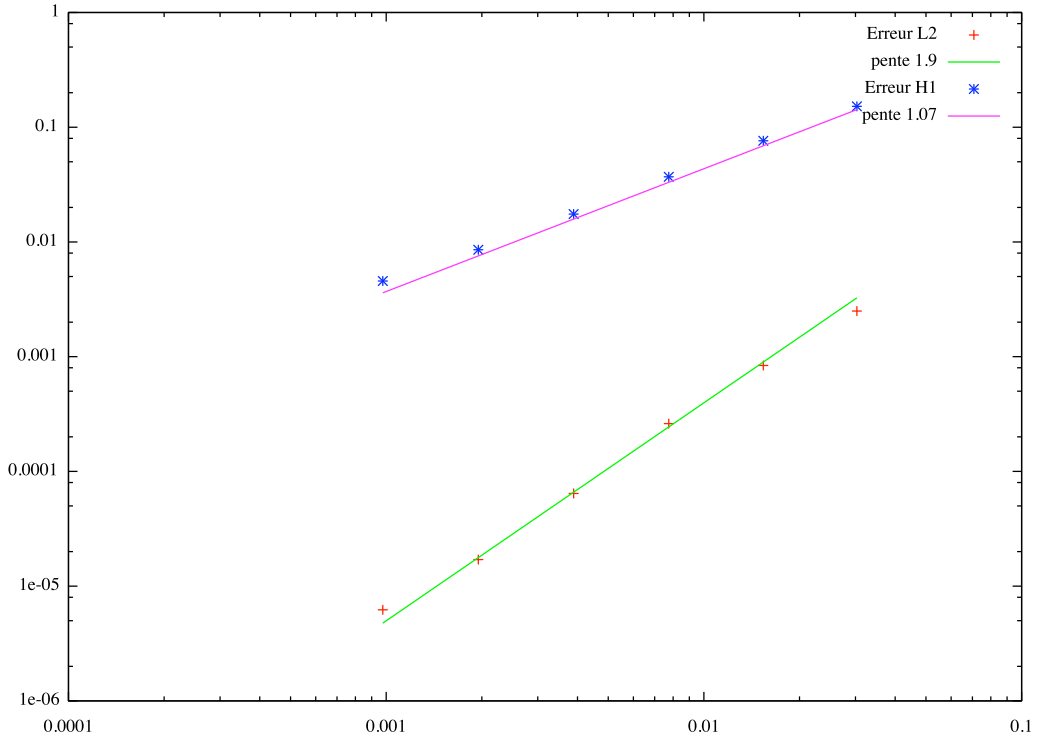

FIGURE 4.1 – Ordre de l'erreur en espace avec la *Smooth Extension Method*

4.1.2 Un cas test scalaire

On reprend le cas test qui nous a servi pour retrouver les ordres d'erreur en espace des autres méthodes :

$$(4.9)\quad \begin{cases} -\Delta u &= 0\,, & \text{dans } \Omega\setminus\bar{\mathcal{O}}\,,\\ u &= \log\left(\frac{(x-x_c)^2+(y-y_c)^2}{R^2}\right)\,, & \text{sur } \partial\Omega\,,\\ u &= 0\,, & \text{sur } \partial\mathcal{O}\,. \end{cases}$$

Le code que l'on a écrit pour notre méthode utilise dans le cas scalaire en $2D$ des éléments finis Q_1 sur maillage cartésien. Pour calculer l'erreur en espace, on a lancé le code pour plusieurs pas de maillage et interpolé la solution obtenue sur un maillage très fin, conforme à la géométrie, généré avec le logiciel *FreeFem++*. On a ensuite calculé l'erreur sur ce maillage très fin en comparant la solution obtenue avec la solution exacte. La figure 4.1 représente l'erreur en norme L^2 et H^1 et on voit que sur cet exemple on retrouve bien l'ordre optimal.

La figure 4.2 représente des lignes de niveaux de la solution obtenue avec la méthode de prolongement régulier. On voit bien qu'il n'y a pas de saut des dérivées normales parce qu'on a un prolongement bien régulier de la solution à l'intérieur de l'inclusion qui a un rayon de 0.1 et est centrée en $(0.3, 0.4)$ dans le carré unité. La figure 4.3 représente d'ailleurs une coupe horizontale de la solution en passant par le centre de l'inclusion sur laquelle on voit très bien que la solution est régulière contrairement aux coupes que l'on a obtenu pour les méthodes de pénalisation ou de multiplicateurs de Lagrange par exemple.

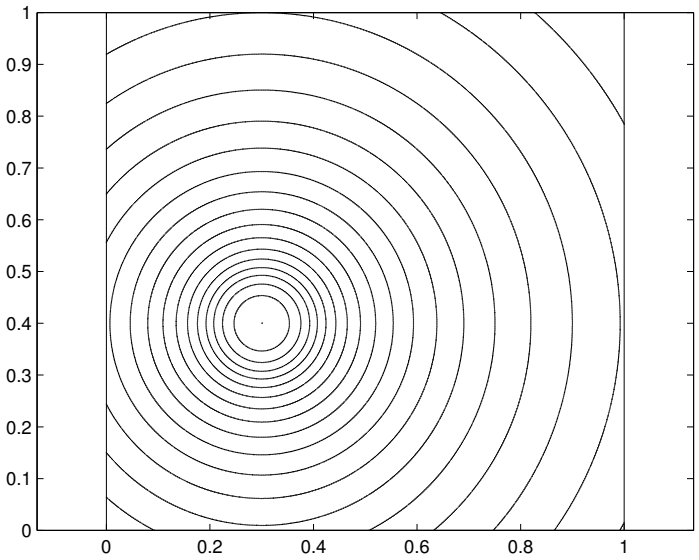

FIGURE 4.2 – Lignes de niveaux de la solution du cas test

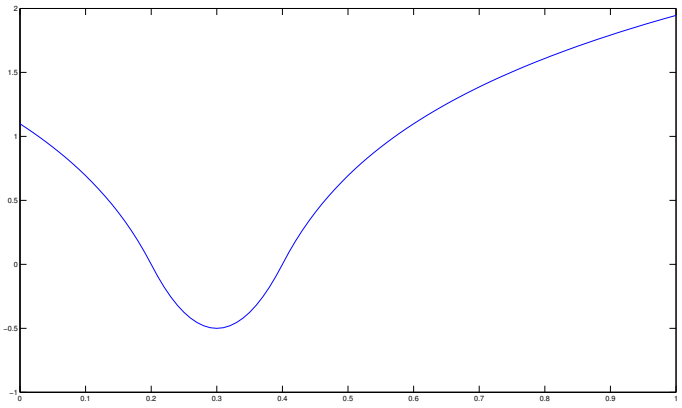

FIGURE 4.3 – Coupe horizontale de la solution du cas test

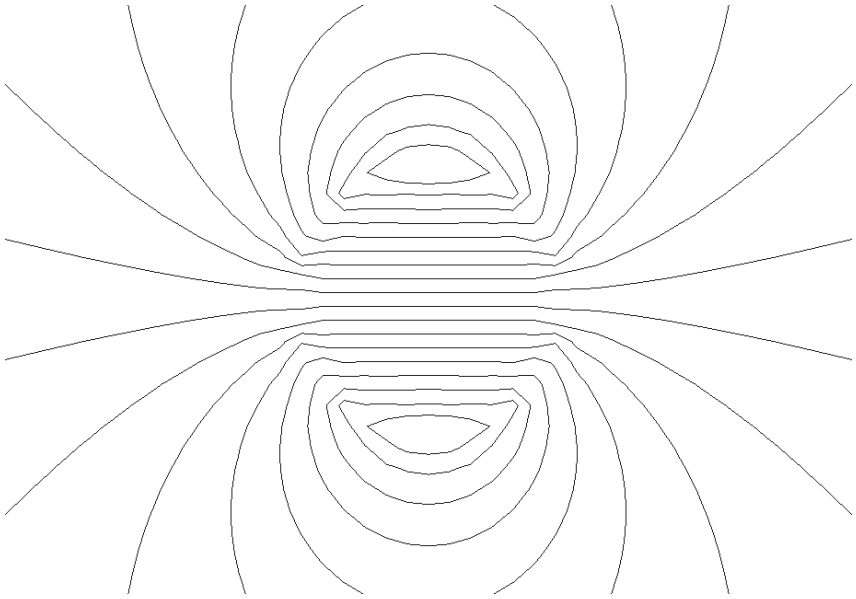

FIGURE 4.4 – Lignes de niveaux de la pression avec la méthode de prolongement régulier

A titre indicatif, on a réalisé la simulation d'une particule seule en sédimentation dans le cas Stokes avec la méthode du prolongement régulier et une méthode de pénalisation classique. Le but est ensuite de comparer les courbes de niveau de la pression. Elles sont représentées sur la figure 4.4 pour notre méthode et sur la figure 4.5 pour la méthode de pénalisation. On représente de plus sur la figure 4.6 la pression dans le cas de la pénalisation sur maillage conforme. Alors que les courbes de niveaux dans le cas de la méthode par prolongement régulier sont bien continues (la pression est H^1), celles obtenues avec la méthode de pénalisation ne le sont pas puisque la solution est moins régulière. Dans le cas de la pénalisation conforme, on retrouve une pression continue ce qui est attendu puisque le saut des dérivées normales est alors bien captée par le maillage. On représente sur la figure 4.7 les contours pour la pression avec la méthode de prolongement régulier dans le cas où l'on prend très peu de point sur chaque bord (32 en pression sur la figure). Les contours restent bien continus, ceci laisse donc penser que la méthode est bien d'ordre optimal dans le cas Stokes aussi. En comparaison, la figure 4.8 représente les isovaleurs de la pression dans le cas de le pénalisation avec la même grille (32 points en pression sur le bord).

4.2 Un cas test dans le cas Stokes

Dans cette partie on s'intéresse au cas d'une particule rigide sans masse dans un champ cisaillé en deux dimensions. L'idée de la simulation est de considérer une boîte, remplie de fluide, et une particule rigide au milieu de la boîte. Les bords haut et bas se translatent avec une vitesse opposée de telle sorte que le fluide est cisaillé et impose à la particule de tourner sur elle même. Cette vitesse de rotation peut être mesurée expérimentalement et numériquement ce qui fournit un bon cas test pour le code. Mathématiquement on

FIGURE 4.5 – Lignes de niveaux de la pression avec la méthode de pénalisation

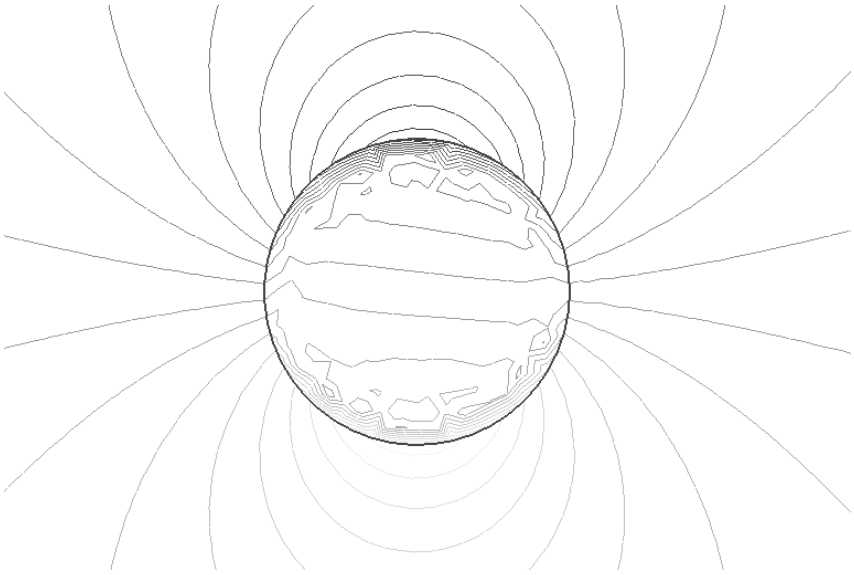

FIGURE 4.6 – Lignes de niveaux de la pression avec la méthode de pénalisation sur maillage conforme

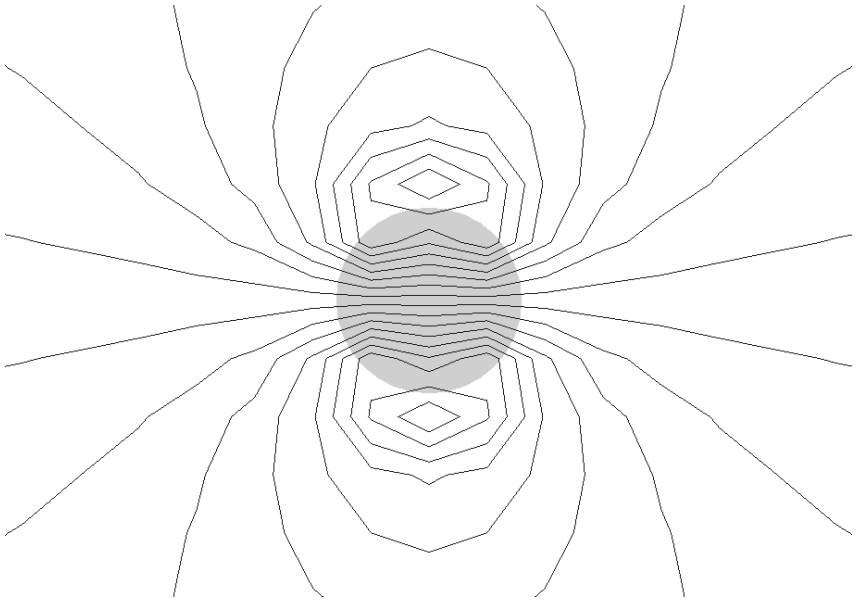

FIGURE 4.7 – Lignes de niveaux de la pression avec la méthode de prolongement régulier et un maillage grossier

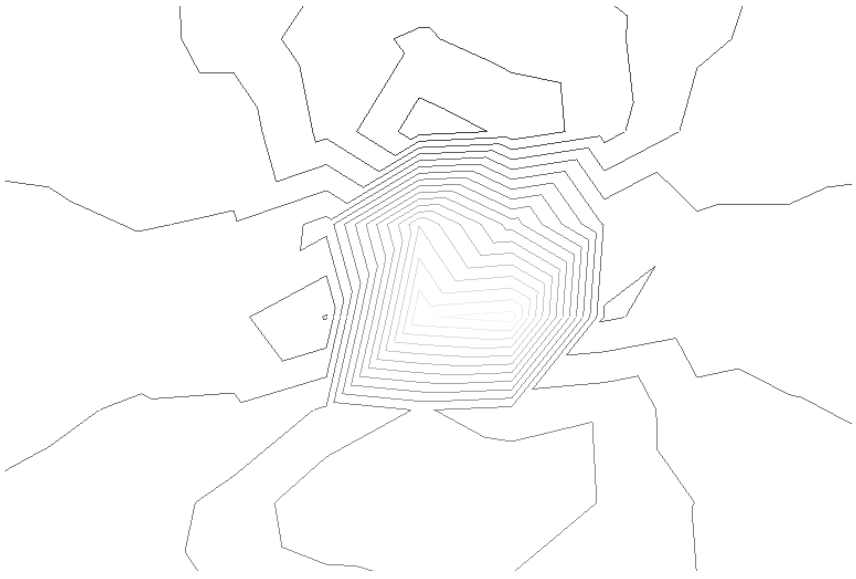

FIGURE 4.8 – Lignes de niveaux de la pression avec une méthode de pénalisation et un maillage grossier

considère une boîte rectangulaire que l'on note Ω et on note B la particule rigide qui se trouve au milieu du rectangle. On veut alors résoudre le problème initial 1.1 avec cette fois ci des conditions de Dirichlet non homogène sur les bords haut et bas et des conditions périodiques sur les bords latéraux. Pour les conditions de Dirichlet on imposera une vitesse horizontale $-\mathbf{U}$ sur le bord du haut et $\mathbf{U}$ sur le bord du bas. Le problème résolu est donc le suivant :

$$(4.10)\quad \begin{cases} -2\eta \operatorname{div}(D(\mathbf{u})) + \nabla p &= \rho_f \mathbf{f}\,, & \text{dans } \Omega \setminus \bar{B}\,,\\ \nabla \cdot \mathbf{u} &= 0\,, & \text{dans } \Omega \setminus \bar{B}\,,\\ \mathbf{u} &= \mathbf{U}\,, & \text{sur } \Gamma_1\,,\\ \mathbf{u} &= -\mathbf{U}\,, & \text{sur } \Gamma_3\,,\\ \mathbf{u}_{|\Gamma_2} &= \mathbf{u}_{|\Gamma_4}\,,\\ \mathbf{u} &= \mathbf{V} + \boldsymbol{\omega} \wedge \mathbf{r}\,, & \text{sur } \partial B\,,\\ \displaystyle\int_{\gamma_i} \sigma(\mathbf{u})\mathbf{n}_i &= 0\,,\\ \displaystyle\int_{\gamma_i} \mathbf{r}_i \wedge \sigma(\mathbf{u})\mathbf{n}_i &= 0\,. \end{cases}$$

Dans le cas où l'on dispose d'une boîte de hauteur H, il est connu que la vitesse de rotation de la particule converge vers la moitié du taux de cisaillement $\dot{\gamma}$ lorsque le rayon de la particule tend vers 0. En effet, il est possible de trouver une solution analytique dans le cas où l'on se place en milieu infini. Cette solution est donnée en $3D$ dans [13] et vaut en $2D$:

$$\begin{aligned} u_x &= A\dot{\gamma} r \cos^2(\theta)\sin(\theta) + \frac{\dot{\gamma}}{2}\left(\frac{R}{r}\right)^4 r\cos(\theta) - \dot{\gamma} r \sin(\theta)\\ u_y &= A\dot{\gamma} r \cos(\theta)\sin^2(\theta) + \frac{\dot{\gamma}}{2}\left(\frac{R}{r}\right)^4 r\sin(\theta)\\ A &= 2\left(\frac{R}{r}\right)^2 - 2\left(\frac{R}{r}\right)^4 \end{aligned}$$

où l'origine est placée au centre de la particule et r et θ sont les composantes en coordonnées polaires. La vitesse de la particule est donnée par la formule 2.13 :

$$\boldsymbol{\omega} = \frac{1}{R^2|\partial B|}\int_{\partial B} \mathbf{r} \wedge \mathbf{u} = \frac{1}{R^2|\partial B|}\int_{\partial B} R\cos(\theta)u_y - R\sin(\theta)u_x\,.$$

Après calcul de l'intégrale, on obtient la formule suivante pour la vitesse de rotation :

$$\boldsymbol{\omega} = \frac{\dot{\gamma}}{2}\,.$$

Remarque 4.7. *Dans les simulations numériques, on se place en domaine borné et la particule voit les bords du domaine, ce qui se traduit par une diminution de la vitesse de rotation. La vitesse de rotation ne sera donc pas exactement $\dot{\gamma}/2$ mais doit tendre vers cette valeur lorsque le rayon de la particule tend vers 0.*

$2R/H$	$\boldsymbol{\omega}$
0.5	8.860732e-01
0.25	9.711395e-01
0.125	9.928006e-01
0.0625	9.981915e-01
0.03125	9.995354e-01
0.015625	9.998712e-01
0.0078125	9.999678e-01

TABLE 4.1 – Évolution de la vitesse de rotation d'une particule dans un champ cisaillé

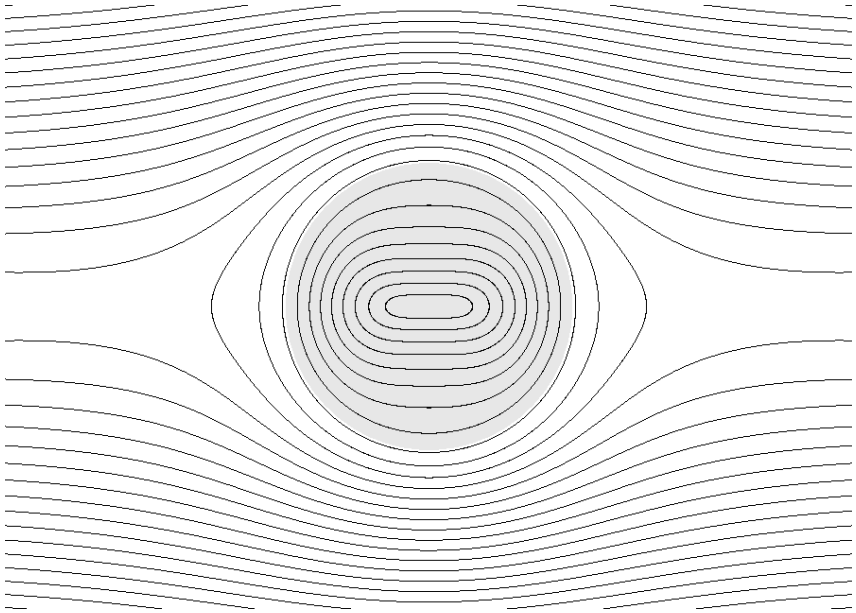

FIGURE 4.9 – Ligne de niveaux dans le cas d'une particule dans un champ cisaillé

Dans notre cas, on a choisi de prendre pour Ω le carré unité et de faire varier le rapport $2R/H$ où H est la hauteur de la boîte (ici on a donc $H = 1$). On prend $\mathbf{U} = (1, 0)$ ce qui donne :

$$\boldsymbol{\omega} = \frac{\dot{\gamma}}{2} = \frac{2\mathbf{U}_x}{2H} = 1$$

Les résultats obtenus sont résumés dans le tableau 4.1, on observe bien une convergence de la vitesse vers la valeur $\dot{\gamma}/2$ qui vaut 1 dans notre cas. On représente sur la figure 4.9 les lignes de niveaux de la solution obtenue. On peut observer que la solution dans la particule n'est pas un mouvement rigide conformément à ce qu'on attendait puisque la solution obtenue est régulière dans le domaine Ω tout entier.

Deuxième partie

Solveurs et code

Chapitre 5

Solveurs et préconditionneurs de Stokes

Sommaire

5.1 Solveur de Stokes

Dans cette partie on décrit le choix du type des éléments finis et on montre que ce sont des éléments stables pour le problème de Stokes. On a choisi de prendre des éléments stables pour ne pas avoir à utiliser des méthodes de stabilisation qui ajouteraient potentiellement de nouveaux paramètres à choisir. Une fois qu'on aura présenté le problème discrétisé, on se concentrera sur le solveur du problème de Stokes qui revient en permanence dans la méthode. Enfin, pour améliorer la rapidité de convergence du solveur de Stokes, on étudiera différentes façons de préconditionner la matrice du système linéaire à résoudre. On s'appuiera beaucoup sur le chapitre concernant l'équation de Stokes de [30] dans cette partie.

5.1.1 Discrétisation

On se concentre sur la résolution d'un problème de Stokes du type 2.4 où l'on considère un second membre général $\mathbf{f}$:

$$(5.1)\quad \begin{cases} -2\eta \operatorname{div}(D(\mathbf{u})) + \nabla p &= \mathbf{f}\,, \quad \text{dans } \Omega\,, \\ \nabla \cdot \mathbf{u} &= 0\,, \quad \text{dans } \Omega\,, \\ \mathbf{u} &= 0\,, \quad \text{sur } \partial\Omega\,. \end{cases}$$

La formulation variationnelle de ce problème est la suivante :

Trouver $(\mathbf{u}, p) \in H_0^1(\Omega) \times L^2(\Omega)$ tel que :

$$\frac{\eta}{2}\int_\Omega D(\mathbf{u}) : D(\mathbf{v}) - \int_\Omega p\nabla\cdot\mathbf{v} = \langle \mathbf{f}\,,\, \mathbf{v}\rangle \quad \forall \mathbf{v} \in H_0^1(\Omega)\,,$$
$$\int_\Omega q\nabla\cdot\mathbf{u} = 0 \quad \forall q \in L^2(\Omega)\,,$$

où $<,>$ est le crochet de dualité entre H^{-1} et H_0^1.

Lorsqu'on impose une condition de Dirichlet sur toute la frontière, la pression n'est alors définie qu'à une constante près. Pour fixer cette constante, on demande à ce que la pression soit dans l'espace des fonctions L^2 à moyenne nulle que l'on notera $L_0^2(\Omega)$. C'est un comportement que l'on cherche à garder au niveau discret et qui n'est pas vérifié pour tout les types d'éléments finis comme on le verra plus loin. Au niveau continu, on obtient l'unicité de la pression faible à une constante près grâce à la condition *inf-sup* :

$$\inf_{q,\|q\|_{L_0^2(\Omega)}\neq 0} \sup_{\mathbf{v}\neq 0} \frac{(q, \nabla\cdot\mathbf{v})}{\|\mathbf{v}\|_{H^1(\Omega)^d}\|q\|_{L_0^2(\Omega)}} \geq \gamma > 0\,,$$

où la norme $\|.\|_{L_0^2(\Omega)}$ est définie par :

$$\|q\|_{L_0^2(\Omega)} = \left\| q - \frac{1}{|\Omega|}\int_\Omega q \right\|_{L^2}\,.$$

La condition *inf-sup* est prouvée dans [19] par exemple. Pour montrer l'unicité de la solution faible, on considère le problème homogène ($\mathbf{f} = 0$) et on prend $\mathbf{v} = \mathbf{u}$ et $q = p$ dans la formulation variationnelle. On obtient alors :

$$\int_\Omega |D(\mathbf{u})|^2 = 0\,.$$

On a donc que $\mathbf{u}$ est un mouvement rigide dans Ω et comme on prend des conditions de Dirichlet homogène, on en déduit que la vitesse est nulle. En injectant cette information dans la formulation variationnelle, on obtient :

$$\int_\Omega p\nabla\cdot\mathbf{v} = 0 \quad \forall \mathbf{v} \in H_0^1(\Omega)\,.$$

La condition *inf-sup* nous permet alors de conclure puisqu'on a l'existence d'une constante non nulle γ telle que :

$$\sup_{v\neq 0} \frac{(q, \nabla\cdot\mathbf{v})}{\|\mathbf{v}\|_{H^1(\Omega)}} \geq \gamma \|q\|_{L_0^2(\Omega)} \quad \forall q \in L^2(\Omega), q \neq cste.$$

Il suffit alors de prendre $q = p$ pour obtenir que la norme L_0^2 de p est nulle et donc que p est une constante.

Au niveau discret, on choisit des espaces d'approximations de dimension finie de $H_0^1(\Omega)$ et de $L^2(\Omega)$ que l'on notera respectivement V_h et M_h et on considère la formulation variationnelle suivante :

Trouver $(\mathbf{u}_h, p_h) \in V_h \times M_h$ tel que :

$$\begin{aligned}\frac{\eta}{2}\int_\Omega D(\mathbf{u}_h) : D(\mathbf{v}_h) - \int_\Omega p_h \nabla\cdot\mathbf{v}_h = \langle \mathbf{f}_h\,,\, \mathbf{v}_h\rangle \quad \forall \mathbf{v}_h \in V_h\,,\\ \int_\Omega q_h \nabla\cdot\mathbf{u}_h = 0 \quad \forall q_h \in M_h\,,\end{aligned}$$

où $\mathbf{f}_h$ est une approximation de $\mathbf{f}$ comme par exemple une combinaison de masses de Dirac dans le cas de la distribution simple couche. On exprime alors les fonctions de V_h et de M_h comme une combinaison linéaire de fonctions de base. Dans la pratique, pour la vitesse on choisit des fonctions de base scalaires ϕ_k, pour k dans $\{1, \ldots, n\}$, pour une composante de la vitesse et on considère alors la base suivante pour V_h (ici en dimension 3) :

$$\{\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_{3n}\} = \left\{ \begin{pmatrix}\phi_1\\0\\0\end{pmatrix}, \ldots, \begin{pmatrix}\phi_n\\0\\0\end{pmatrix}, \begin{pmatrix}0\\\phi_1\\0\end{pmatrix}, \ldots, \begin{pmatrix}0\\\phi_n\\0\end{pmatrix}, \begin{pmatrix}0\\0\\\phi_1\end{pmatrix}, \ldots, \begin{pmatrix}0\\0\\\phi_n\end{pmatrix} \right\}.$$

On note ψ_j, pour j dans $\{1, \ldots, n_p\}$, les fonctions de base pour la pression. Avec un tel choix d'éléments finis, on est ramené à la résolution d'un système linéaire de la forme (toujours en 3 dimensions) :

$$\begin{pmatrix} A_x & {}^tC_{xy} & {}^tC_{xz} & {}^tB_x \\ C_{xy} & A_y & {}^tC_{yz} & {}^tB_y \\ C_{xz} & C_{yz} & A_z & {}^tB_z \\ B_x & B_y & B_z & 0 \end{pmatrix} \begin{pmatrix} u_x \\ u_y \\ u_z \\ p \end{pmatrix} = \begin{pmatrix} f_x \\ f_y \\ f_z \\ f_p \end{pmatrix}$$

que l'on peut écrire sous une forme plus condensée :

$$\begin{pmatrix} \mathbf{A} & {}^t\mathbf{B} \\ \mathbf{B} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ p \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ f_p \end{pmatrix}$$

où le bloc $\mathbf{A}$ correspond aux termes visqueux, ${}^t\mathbf{B}$ correspond à l'opérateur gradient et $\mathbf{B}$ à l'opérateur de l'opposé de la divergence.

Remarque 5.1. *Dans notre cas, le bloc* $\mathbf{A}$ *en vitesse est plus compliqué que dans le cas "classique" car on utilise la formulation* $D(\mathbf{u}_h) : D(\mathbf{v}_h)$ *au lieu de* $\nabla\mathbf{u}_h : \nabla\mathbf{v}_h$*. Dans le cas "classique" toutes les matrices* C *sont nulles et les matrices* A_x, A_y *et* A_z *sont toutes égales à la matrice du Laplacien scalaire* $\tilde{A}$ *:*

$$\begin{pmatrix} \tilde{A} & 0 & 0 & {}^tB_x \\ 0 & \tilde{A} & 0 & {}^tB_y \\ 0 & 0 & \tilde{A} & {}^tB_z \\ B_x & B_y & B_z & 0 \end{pmatrix} \begin{pmatrix} u_x \\ u_y \\ u_z \\ p \end{pmatrix} = \begin{pmatrix} f_x \\ f_y \\ f_z \\ f_p \end{pmatrix} .$$

Afin d'obtenir les mêmes propriétés que dans le cas continu, on veut que la pression discrète soit définie à une constante près. On reprend donc l'unicité des solutions du cas continu en considérant le problème homogène et en multipliant le système linéaire par la transposée de $\mathbf{u}$ pour la première ligne et par la transposée de p pour la deuxième ce qui donne :

$$\begin{aligned} {}^t\mathbf{u}\mathbf{A}\mathbf{u} + {}^t\mathbf{u}{}^t\mathbf{B}p = 0\,, \\ {}^tp\mathbf{B}\mathbf{u} = 0\,. \end{aligned}$$

On a donc que $\mathbf{A}\mathbf{u} = 0$ et comme $\mathbf{A}$ est symétrique, définie et positive, on en déduit que $\mathbf{u}$ est nulle. En injectant cette information dans la première ligne on obtient donc :

$${}^t\mathbf{B}p = 0\,.$$

De façon faible il faut que si :

$$\int_\Omega p_h \nabla\cdot\mathbf{v}_h = 0 \quad \forall \mathbf{v}_h \in V_h$$

alors la pression discrète p_h soit constante. Si on veut que la pression discrète soit définie à une constante près, le noyau de ${}^t\mathbf{B}$ doit être constitué des constantes uniquement et donc de dimension 1. L'idée est de choisir des éléments finis pour la vitesse et la pression de telle sorte qu'ils vérifient la condition *inf-sup discrète* :

$$\inf_{q_h \neq cste} \sup_{\mathbf{v}_h \neq 0} \frac{|(q_h, \nabla\cdot\mathbf{v}_h)|}{\|v_h\|_{H^1}\|q_h\|_{L^2_0}} \geq \gamma > 0\,,$$

ce qui garantie une pression discrète définie à une constante près quel que soit le maillage utilisé.

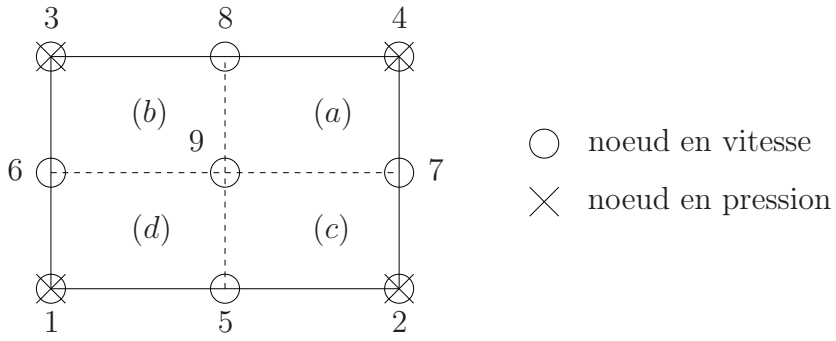

FIGURE 5.1 – Macro-élément constitué d'une cellule en pression

Remarque 5.2. *Toutes les combinaisons d'éléments finis ne vérifient pas cette condition. C'est le cas par exemple des éléments finis Q_1/Q_1 qui ne sont pas stables. Si un couple d'éléments finis ne vérifie pas la condition que le noyau de ${}^t\mathbf{B}$ est constitué des constantes, alors on peut ajouter des fonctions non constantes qui se trouve dans le noyau à la pression discrète et obtenir une nouvelle solution qui n'approche pas correctement celle au niveau continu. Ces fonctions non constantes dans le noyau s'appellent des modes de pression parasite (spurious pressure mode).*

Dans le code développé, on a choisi d'utiliser des éléments finis $4Q_1/Q_1$ en dimension 2 et $8Q_1/Q_1$ en dimension 3. On dispose donc de deux maillages cartésiens, un pour la vitesse et un pour la pression, celui pour la vitesse étant deux fois plus fin dans chaque direction d'espace. On prend alors des fonctions Q_1 sur les deux grilles pour approcher la vitesse et la pression. La terminologie vient du fait que dans une cellule pour la pression il y a 4 cellules pour la vitesse en dimension 2 et 8 en dimension 3 (voir figure 5.1 pour la dimension 2).

Remarque 5.3. *On prend des éléments finis Q_1 pour la vitesse et la pression pour la simplicité de l'implémentation dans le code et pour ne prendre que des éléments finis d'ordre* 1.

On démontre alors que ce choix conduit à des éléments finis mixtes stables :

Proposition 5.4. *Les élements $4Q_1/Q_1$ en dimension* 2 *et $8Q_1/Q_1$ en dimension* 3 *vérifient la condition inf-sup discrète.*

Preuve : On fait la preuve dans le cas de la dimension 2 en adaptant la méthode de [30] qui utilise les *macro-éléments*. L'idée est de considérer le même problème de Stokes mais sur de petits sous domaines, appelés *macro-éléments*, constitués typiquement d'une ou plusieurs cellules du maillage. Si sur ces macro-éléments la condition sur le noyau de ${}^t\mathbf{B}$ est respectée, on dit que ce sont des macro-éléments *stables*. Les résultats de Stenberg dans [68] montrent que si le domaine global est constitué de macro-éléments stables, alors la condition inf-sup discrète est vérifiée.

Considérons tout d'abord le cas d'un macro-élément $\mathcal{M}$ constitué d'une seule cellule $[0, 2h_x] \times [0, 2h_y]$ en pression et donc de 4 pour la vitesse (voir figure 5.1). On a des conditions de Dirichlet homogène et donc tous les degrés de liberté en vitesse sont nuls

sauf celui du milieu et on a 4 degrés de liberté en pression. On a donc deux fonctions de base en vitesse qui sont $\{(\phi,0),(0,\phi)\}$ et 4 en pression que l'on note ψ_1,ψ_2,ψ_3 et ψ_4. La fonction ϕ est une fonction Q_1 qui vaut 1 sur le noeud numéro 9 et zéro sur les bords du macro-élément. Les fonctions ψ_i sont des fonctions Q_1 qui valent 1 sur le noeud i et zéro sur les noeuds j avec $j \neq i$ et $j \in \{1,\ldots,4\}$. Il faut alors construire la matrice ${}^t\mathbf{B}$ et calculer son noyau. On peut donner les expressions dans $\mathcal{M}$ des fonctions de base introduites :

$$\begin{cases} \psi_1(x,y) = \dfrac{(2h_x-x)(2h_y-y)}{4h_xh_y} \\ \psi_2(x,y) = \dfrac{x(2h_y-y)}{4h_xh_y} \\ \psi_3(x,y) = \dfrac{(2h_x-x)y}{4h_xh_y} \\ \psi_4(x,y) = \dfrac{xy}{4h_xh_y} \end{cases} \qquad \phi(x,y) = \begin{cases} \dfrac{(2h_x-x)(2h_y-y)}{h_xh_y} & \text{dans } (a) \\ \dfrac{x(2h_y-y)}{h_xh_y} & \text{dans } (b) \\ \dfrac{(2h_x-x)y}{h_xh_y} & \text{dans } (c) \\ \dfrac{xy}{h_xh_y} & \text{dans } (d)\,. \end{cases}$$

On peut alors calculer les intégrales suivantes pour construire ${}^tB_x$ et ${}^tB_y$:

$$\begin{aligned} {}^tB_x &= \left(\left(-\int_{\mathcal{M}} \psi_j \partial_x \phi \right)_{1\leq j\leq 4} \right), \\ {}^tB_y &= \left(\left(-\int_{\mathcal{M}} \psi_j \partial_y \phi \right)_{1\leq j\leq 4} \right). \end{aligned}$$

Ce qui donne après calcul la matrice :

$${}^t\mathbf{B} = \frac{1}{4} \begin{pmatrix} -h_y & h_y & -h_y & h_y \\ -h_x & -h_x & h_x & h_x \end{pmatrix}.$$

Cette matrice à un noyau de dimension 2 qui contient les constantes mais aussi le vecteur $(1,-1,-1,1)$ par exemple. Avec une seule cellule on avait moins de degrés de liberté en vitesse qu'en pression on savait donc déjà que ça ne pouvait pas être stable. L'idée est donc de considérer un macro-élément $\mathcal{M}$ constitué de deux cellules en pression comme sur la figure 5.2 (on a représenté uniquement les noeuds apportant une contribution). On dispose ici de 3 noeuds en vitesse non nuls, donc de 6 degrés de liberté en vitesse en dimension 2, et de 6 degrés de liberté en pression aussi. Si on refait les mêmes calculs que précédemmment on obtient la matrice suivante :

$${}^t\mathbf{B} = \frac{1}{24} \begin{pmatrix} -6h_y & 6h_y & 0 & -6h_y & 6h_y & 0 \\ -6h_x & -6h_x & 0 & 6h_x & 6h_x & 0 \\ -3h_y & 0 & 3h_y & -3h_y & 0 & 3h_y \\ -h_y & -10h_y & -h_y & h_y & 10h_y & h_y \\ 0 & -6h_y & 6h_y & 0 & -6h_y & 6h_y \\ 0 & -6h_x & -6h_x & 0 & 6h_x & 6h_x \end{pmatrix}.$$

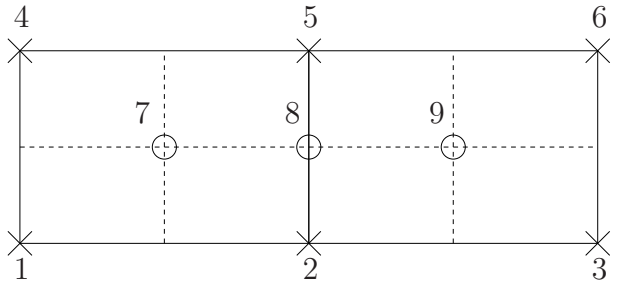

FIGURE 5.2 – Macro-élément constitué de deux cellules en pression

Si on note $(p_1, \ldots, p_6)$ un vecteur du noyau, on voit que les deux premières lignes donnent les relations :

$$p_5 = p_1$$
$$p_4 = p_2$$

De même les deux dernières lignes donnent :

$$p_6 = p_2$$
$$p_3 = p_5$$

On a donc finalement :

$$p_1 = p_3 = p_5 = \xi$$
$$p_2 = p_4 = p_6 = \eta$$

Il faut donc montrer que $\xi = \eta$, ce qu'on obtient en considérant la quatrième ligne. Le noyau est donc réduit aux constantes et ce macro-élément est stable. On peut faire la même chose avec deux cellules verticales et avec 3 cellules ce qui permet de considérer des maillages rectangulaires avec un nombre pair ou impair de cellules dans chaques directions de l'espace. Les éléments finis $4Q_1/Q_1$ vérifient donc la condition inf-sup discrète et sont donc stables. □

5.1.2 Méthode de résolution

On s'intéresse maintenant à la façon dont on va résoudre le système linéaire obtenu en utilisant les éléments finis $4Q_1/Q_1$. Une façon serait de considérer la formulation point selle :

$$\begin{cases} \mathbf{A}\mathbf{u} + {}^t\mathbf{B}p = \mathbf{f}\,, \\ \mathbf{B}u = 0\,, \end{cases}$$

et d'éliminer la vitesse dans la première équation en la multipliant par $\mathbf{B}\mathbf{A}^{-1}$. On obtient alors un système linéaire sur la pression uniquement :

$$\mathbf{B}\mathbf{A}^{-1}{}^t\mathbf{B}p = \mathbf{B}\mathbf{A}^{-1}\mathbf{f}\,. \tag{5.2}$$

On peut alors ensuite calculer la vitesse à partir de la pression en résolvant l'équation de Poisson associée. La difficulté est de résoudre l'équation en pression qui fait intervenir le complément de Schur $\mathbf{B}\mathbf{A}^{-1}\,{}^t\mathbf{B}$ qui est une matrice pleine. On peut par exemple utiliser un gradient conjugué et chercher un préconditionneur pour approcher le complément de Schur qui soit facile à inverser.

Remarque 5.5. *On peut remarquer qu'une itération de Richardson sur le système 5.2 :*

$$p_{k+1} = p_k + \rho(\mathbf{B}\mathbf{A}^{-1}\mathbf{f} - \mathbf{B}\mathbf{A}^{-1t}\mathbf{B}p_k)\,,$$

revient à faire une itération de l'algorithme d'Uzawa :

$$\begin{aligned}\mathbf{A}\mathbf{u}_k &= \mathbf{f} - {}^t\mathbf{B}p_k\,,\\ p_{k+1} &= p_k + \rho\mathbf{B}\mathbf{u}_k\,.\end{aligned}$$

Dans le code, on ne suit pas cette méthode de résolution, on s'est plutôt dirigé vers une formulation monolithique qui consiste à résoudre le gros système linéaire sans séparer les inconnues. La matrice est symétrique mais indéfinie ce qui empêche d'utiliser une méthode de gradient conjugué pour résoudre le système linéaire. On choisit d'utiliser une méthode de *minres*, qui est une méthode de Krylov, qui marche sur les matrices symétriques indéfinies. Elle consiste à chaque itération à chercher le nouveau résidu dans l'espace de Krylov de telle sorte qu'il soit de norme minimale. Pour une estimation de la convergence de la méthode, on suit à nouveau [30] :

Proposition 5.6. *Soit K une matrice symétrique non nécéssairement définie, on considère le système linéaire :*

$$K\mathbf{x} = \mathbf{b}$$

On se donne un vecteur $\mathbf{x}_0$ et on note $\mathbf{r}_0 = \mathbf{b} - K\mathbf{x}_0$ le résidu initial. la matrice K étant symétrique, on note $\mathbf{w}_j$ ses vecteurs propres qui forment une base orthogonale et λ_j les valeurs propres associées. On note enfin $\mathcal{P}_k$ l'espace des polynômes de degrés inférieur ou égales à k. On a alors la majoration suivante :

$$\frac{\|\mathbf{r}_k\|}{\|\mathbf{r}_0\|} \leq \min_{P_k\in\mathcal{P}_k, P_k(0)=1} \max_j |P_k(\lambda_j)|\,.$$

Remarque 5.7. *La convergence de la méthode dépend fortement de la répartition des valeurs propres de K. La méthode convergera d'autant plus vite que les valeurs propres de la matrice K sont regroupées. En effet si on se fixe une erreur finale ϵ, le degré k du polynôme P_k nécessaire pour que tous les $P_k(\lambda_j)$ soient plus petits que ϵ est plus petit dans le cas où toutes les valeurs propres λ_j sont regroupées par paquet.*

Preuve de la proposition 5.6 : Étant donné un résidu $\mathbf{r}_0$, la méthode minres consiste à chercher à l'étape k le nouveau résidu $\mathbf{r}_k$ dans l'espace de Krylov $\mathcal{K}_k$:

$$\mathbf{r}_k \in \mathcal{K}_k = \mathbf{r}_0 + \text{ vect}(K\mathbf{r}_0, K^2\mathbf{r}_0, \ldots, K^k\mathbf{r}_0)\,,$$

tel que la norme de $\mathbf{r}_k$ soit minimale. Le résidu $\mathbf{r}_k$ s'écrit donc sous la forme :

$$\mathbf{r}_k = P_k(K)\mathbf{r}_0\,,$$

où P_k est un polynôme de degré inférieur ou égal à k tel que $P_k(0) = 1$. Le polynôme P_k est optimal au sens où la norme de $\mathbf{r}_k$ est minimale. Comme la matrice K est symétrique, il existe une base orthogonale de vecteurs propres $\mathbf{w}_j$ sur laquelle on peut décomposer le résidu $\mathbf{r}_0$:

$$\mathbf{r}_0 = \sum_j \alpha_j \mathbf{w}_j \, .$$

On note $\mathcal{P}_k$ l'espace des polynôme de degré inférieur ou égal à k. On a donc la relation suivante :

$$\mathbf{r}_k = P_k(K)\mathbf{r}_0 = P_k(K) \sum_j \alpha_j \mathbf{w}_j = \sum_j \alpha_j P_k(\lambda_j)\mathbf{w}_j \, ,$$

où les λ_j sont les valeurs propres associées aux vecteurs propres $\mathbf{w}_j$. On exprime alors la norme de $\mathbf{r}_k$ que l'on majore :

$$\begin{aligned}
\|\mathbf{r}_k\| &= \min_{P_k \in \mathcal{P}_k, P_k(0)=1} \| \sum_j \alpha_j P_k(\lambda_j)\mathbf{w}_j \| \\
&= \min_{P_k \in \mathcal{P}_k, P_k(0)=1} \left(\alpha_j^2 P_k(\lambda_j)^2 < \mathbf{w}_j \, , \, \mathbf{w}_j > \right)^{1/2} \\
&\leq \min_{P_k \in \mathcal{P}_k, P_k(0)=1} \max_j |P_k(\lambda_j)| \left(\sum_j \alpha_j^2 < \mathbf{w}_j \, , \, \mathbf{w}_j > \right)^{1/2} .
\end{aligned}$$

On a donc en identifiant la norme de $\mathbf{r}_0$:

$$\frac{\|\mathbf{r}_k\|}{\|\mathbf{r}_0\|} \leq \min_{P_k \in \mathcal{P}_k, P_k(0)=1} \max_j |P_k(\lambda_j)| \, ,$$

ce qui est l'estimation de l'énoncé. □

Dans le cas des équations de Stokes, la matrice K est la matrice par bloc écrite dans la section précédente dont les valeurs propres tendent vers l'infini lorsque le pas de maillage tend vers 0. Elles ne sont donc pas regroupées en paquets et le nombre d'itérations pour atteindre une précision donnée augmente. On a regroupé dans le tableau 5.1 le nombre d'itérations nécessaires pour atteindre une erreur de 10^{-6} pour un problème de cavité entraînée dans le carré unité en deux dimensions. C'est un problème de Stokes incompressible homogène avec des conditions de Dirichlet homogène sur tous les bords sauf celui du haut où la vitesse est horizontale. Les résultats montrent que le nombre d'itérations nécessaires dépend du pas du maillage et augmente fortement. Ainsi, pour pouvoir prendre des maillages assez fins, il faut préconditionner le système. Un bon préconditionneur doit être facile à inverser et doit regrouper les valeurs propres de telle sorte que le nombre d'itérations ne dépende plus de la finesse du maillage comme c'est le cas ici.

5.1.3 Préconditionneur

On s'intéresse dans cette section au préconditionnement de la matrice de Stokes :

$$S = \begin{pmatrix} \mathbf{A} & {}^t\mathbf{B} \\ \mathbf{B} & \mathbf{0} \end{pmatrix} .$$

h	n = nombre d'itérations	temps (s)
2^{-4}	275	-
2^{-5}	633	0.1
2^{-6}	1217	0.74
2^{-7}	2218	5.66
2^{-8}	4044	67.8

TABLE 5.1 – Nombre d'itérations effectuées en fonction du pas d'espace h pour une erreur de 10^{-6} sur une cavité entraînée sans préconditionneur

On considère un préconditionneur $M = H^tH$ symétrique défini positif et le système préconditionné suivant :

$$H^{-1}S\,{}^t(H^{-1})\mathbf{y} = H^{-1}\mathbf{b}, \quad \mathbf{y} = {}^tH\mathbf{x}\,.$$

La matrice $H^{-1}S\,{}^t(H^{-1})$ est symétrique et possède autant de valeurs propres positives, négatives et nulles que S d'après la loi d'inertie de Sylvester. Étant donnée une approximation de la solution $\mathbf{x}_k$, le résidu de ce problème préconditionné est :

$$H^{-1}\mathbf{b} - H^{-1}S\,{}^t(H^{-1})\mathbf{y}_k = H^{-1}(\mathbf{b} - S\mathbf{x}_k) = H^{-1}\mathbf{r}_k\,.$$

Remarque 5.8. *Lorsque dans l'algorithme du minres on minimise la norme du résidu $\|H^{-1}\mathbf{r}_k\|$, on minimise donc la norme M^{-1} du résidu $\|\mathbf{r}_k\|_{M^{-1}} = (< M^{-1}\mathbf{r}_k\,,\,\mathbf{r}_k >)^{1/2}$. C'est pourquoi on utilise une matrice M symétrique définie positive, de façon à ce que $\|.\|_{M^{-1}}$ soit une norme et que le système préconditionné reste symétrique.*

On peut reprendre la preuve de la proposition 5.6 sur ce système préconditionné et on obtient alors l'estimation suivante :

$$\frac{\|\mathbf{r}_k\|_{M^{-1}}}{\|\mathbf{r}_0\|_{M^{-1}}} \leq \min_{P_k\in\mathcal{P}_k, P_k(0)=1} \max_j |P_k(\lambda_j)|\,, \tag{5.3}$$

où cette fois ci, les λ_j sont les valeurs propres de $H^{-1}S\,{}^t(H^{-1})$ qui sont les mêmes que celles de $M^{-1}S$.

Remarque 5.9. *L'estimation suggère bien de choisir un préconditionneur M de telle sorte que les valeurs propres de $M^{-1}S$ soient regroupées et ainsi obtenir une convergence, vers une erreur fixée, en un nombre fixe d'itérations indépendant de la finesse du maillage.*

On suit encore la référence [30] et on considère des préconditionneurs M diagonaux par bloc :

$$M = \begin{pmatrix} \mathbf{T} & \mathbf{0} \\ \mathbf{0} & \mathbf{P} \end{pmatrix}.$$

L'objectif est d'identifier des préconditionneurs faciles à inverser et tels que le nombre d'itérations pour résoudre un problème de Stokes soit indépendant de la finesse du maillage. On énonce tout d'abord un lemme qui exprime l'inégalité inf-sup discrète de façon algébrique. Ce lemme nous sera utile pour la proposition qui suit qui donne un exemple de matrices $\mathbf{T}$ et $\mathbf{P}$ faisant converger la méthode de minres en trois itérations.

Lemme 5.10. *On suppose que l'on dispose d'élément finis vérifiant la condition inf-sup discrète :*

$$0 < \gamma \leq \inf_{q_h \neq cste} \sup_{\mathbf{v}_h \neq 0} \frac{|(q_h\,,\,\nabla \cdot \mathbf{v}_h)|}{\|\mathbf{v}_h\|_1 \|q_h\|_0}\,.$$

Alors on a la relation :

$$\frac{\gamma^2}{C^2} \leq \frac{|<\mathbf{B}\mathbf{A}^{-1}\,{}^t\mathbf{B}q\,,\,q>|}{<Qq\,,\,q>} \leq C^2 \quad \forall q \neq cste\,,$$

où Q est la matrice de masse en pression, q est le vecteur des coefficients de q_h sur la base des éléments finis et C est une constante indépendante de h.

Preuve : Les produits scalaires des fonctions éléments finis peuvent s'écrire en fonction des matrices du problème. On a en effet les relations suivantes qui découlent de la définition des matrices :

$$\begin{aligned} |(q_h\,,\,\nabla \cdot \mathbf{v}_h)| = |{}^t q \mathbf{B}\mathbf{v}| = |<q\,,\,\mathbf{B}\mathbf{v}>|\,, \\ \|q_h\|_0 = {}^t q Q q =< Qq\,,\,q >^{1/2} \end{aligned}$$

.

Pour la norme de $\mathbf{v}_h$ dans $H_0^1(\Omega)$, on a équivalence entre la norme H^1 et la semi-norme H^1. Ensuite comme on est dans H_0^1, l'inégalité de Korn (voir [29]) nous donne l'équivalence entre la semi-norme H^1 et son analogue, que l'on notera $\epsilon(.)$, faisant intervenir le tenseur des déformations :

$$\epsilon(\mathbf{v}) = \left(\int_\Omega |D(\mathbf{v})|^2\right)^{1/2}\,.$$

On a donc les inégalités suivantes :

$$\frac{1}{C} < \mathbf{A}\mathbf{v}, \mathbf{v} >^{1/2} \leq \|\mathbf{v}_h\|_1 \leq C < \mathbf{A}\mathbf{v}, \mathbf{v} >^{1/2}\,,$$

où C est une constante positive. On obtient donc :

$$\begin{aligned} \gamma &\leq C \min_{q \neq cste} \max_{\mathbf{v} \neq 0} \frac{|<q\,,\,\mathbf{B}\mathbf{v}>|}{<\mathbf{A}\mathbf{v}, \mathbf{v}>^{1/2} < Qq\,,\,q>^{1/2}} \\ &= C \min_{q \neq cste} \frac{1}{<Qq\,,\,q>^{1/2}} \max_{\mathbf{w}=\mathbf{A}^{1/2}\mathbf{v} \neq 0} \frac{|<\mathbf{A}^{-1/2}\,{}^t\mathbf{B}q\,,\,\mathbf{w}>|}{<\mathbf{w}\,,\,\mathbf{w}>^{1/2}}\,. \end{aligned}$$

Or par inégalité de Cauchy-Schwartz on peut majorer le max par :

$$\max_{\mathbf{w} \neq 0} \frac{|<\mathbf{A}^{-1/2}\,{}^t\mathbf{B}q\,,\,\mathbf{w}>|}{<\mathbf{w}\,,\,\mathbf{w}>^{1/2}} \leq < \mathbf{A}^{-1/2}\,{}^t\mathbf{B}q,\ \mathbf{A}^{-1/2}\,{}^t\mathbf{B}q >^{1/2}\ .$$

Cette inégalité est une égalité si on prend $w = \mathbf{A}^{-1/2}\,{}^t\mathbf{B}q$, le max est donc atteint et on a :

$$\begin{aligned} \frac{\gamma}{C} &\leq \min_{q \neq cste} \frac{<\mathbf{A}^{-1/2}\,{}^t\mathbf{B}q,\ \mathbf{A}^{-1/2}\,{}^t\mathbf{B}q>^{1/2}}{<Qq\,,\,q>^{1/2}} \\ &= \min_{q \neq cste} \frac{<\mathbf{B}\mathbf{A}^{-1}\,{}^t\mathbf{B}q\,,\,q>^{1/2}}{<Qq\,,\,q>^{1/2}}\,. \end{aligned}$$

On obtient l'inégalité de gauche de l'énoncé en prenant le carré. Pour la partie de droite, on a tout d'abord :

$$\inf_{q_h \neq cste} \sup_{\mathbf{v}_h \neq 0} \frac{|(q_h\,,\,\nabla\cdot\mathbf{v}_h)|}{\|\mathbf{v}_h\|_1 \|q_h\|_0} \geq \frac{1}{C} \min_{q \neq cste} \frac{<\mathbf{B}\mathbf{A}^{-1}\,{}^t\mathbf{B}q\,,\,q>^{1/2}}{<Qq\,,\,q>^{1/2}}$$

on utilise Cauchy-Schwarz dans la condition inf-sup discrète :

$$\left|\int_\Omega q_h \nabla\cdot\mathbf{v}_h\right| \leq \|q_h\|_0 \left(\int_\Omega |\nabla\cdot\mathbf{v}_h|^2\right)^{1/2}.$$

Or la norme L^2 de la divergence se majore par la norme L^2 du gradient :

$$\int_\Omega |\nabla\cdot\mathbf{v}_h|^2 \leq \int_\Omega |\nabla\mathbf{v}_h|^2\,.$$

En remplaçant dans la condition inf-sup discrète, on peut la majorer par 1, d'où le résultat de l'énoncé. □

Proposition 5.11. *Soit S la matrice de Stokes que l'on écrit par bloc :*

$$S = \begin{pmatrix} \mathbf{A} & {}^t\mathbf{B} \\ \mathbf{B} & \mathbf{0} \end{pmatrix},$$

et M la matrice symétrique définie positive diagonale par bloc :

$$M = \begin{pmatrix} \mathbf{T} & \mathbf{0} \\ \mathbf{0} & \mathbf{P} \end{pmatrix}.$$

Alors, si les matrices $\mathbf{T}$ et $\mathbf{P}$ sont définies de la façon suivante :

$$\begin{aligned} \mathbf{T} &= \mathbf{A}\,, \\ \mathbf{P} &= \mathbf{B}\mathbf{A}^{-1}\,{}^t\mathbf{B}\,, \end{aligned}$$

le problème aux valeurs propres

$$S\begin{pmatrix} \mathbf{u} \\ p \end{pmatrix} = \lambda M \begin{pmatrix} \mathbf{u} \\ p \end{pmatrix},$$

possède exactement 3 valeurs propres non nulles.

Preuve : On regarde le problème aux valeurs propres :

$$\begin{pmatrix} \mathbf{A} & {}^t\mathbf{B} \\ \mathbf{B} & \mathbf{0} \end{pmatrix}\begin{pmatrix} \mathbf{u} \\ p \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}\mathbf{A}^{-1}\,{}^t\mathbf{B} \end{pmatrix}\begin{pmatrix} \mathbf{u} \\ p \end{pmatrix}.$$

Un vecteur $(\mathbf{u}, 0)$ qui vérifie $\mathbf{B}\mathbf{u} = 0$ est un vecteur propre de ce problème de valeur propre 1. La valeur propre 1 a donc une multiplicité égale à la dimension du noyau de $\mathbf{B}$ qui est au moins de $n_u - n_p$ si $\mathbf{B}$ est de taille $n_p \times n_u$. On fixe la pression en la prenant dans

l'espace des fonctions à moyenne nulle, elle ne peut donc pas être une constante non nulle. Ceci empêche d'obtenir la valeur propre zéro comme solution de ce problème. On suppose donc p non constante et les autres valeurs propres différentes de 1 vérifient alors :

$$\begin{aligned} \mathbf{Au} + {}^t\mathbf{B}p &= \lambda \mathbf{Au}\,, \\ \mathbf{Bu} &= \lambda \mathbf{B}\mathbf{A}^{-1}\,{}^t\mathbf{B}p\,. \end{aligned}$$

En éliminant la vitesse on obtient donc l'équation :

$$(\lambda^2 - \lambda - 1)\mathbf{B}\mathbf{A}^{-1}\,{}^t\mathbf{B}p = 0\,.$$

Comme on a choisi des éléments finis qui vérifient la condition inf-suf discrète, d'après le lemme 5.10, le complément de Schur $\mathbf{B}\mathbf{A}^{-1}\,{}^t\mathbf{B}$ est inversible et on obtient deux autres valeurs propres avec l'équation $\lambda^2 - \lambda - 1 = 0$:

$$\lambda_\pm = \frac{1 \pm \sqrt{5}}{2}\,.$$

On a donc trouvé toutes les valeurs propres de ce problème dont trois non nulles. □

Corollaire 5.12. *On prend les mêmes notations que dans la proposition précédente, la méthode minres appliquée au système linéaire dont S est la matrice et préconditionné par M converge exactement en 3 itérations.*

Preuve : la convergence du minres dépend des valeurs propres du système de la proposition précédente. Dans l'estimation 5.3 un polynôme de degrés 3 suffit à annuler toutes les valeurs propres du système, on obtient donc un résidu nul au bout de trois itérations. □

Remarque 5.13. *on a testé ce préconditionneur avec un code Matlab en calculant la matrice du complément de Schur. Sur des grilles de taille 64×64 en deux dimensions en vitesse, le minres converge en 2 itérations avec un résidu de norme inférieur à 10^{-13}. Ce choix de préconditionneur permet d'avoir une convergence en un minimum d'itérations du minres, par contre il n'est pas viable pour des grilles plus grosses car la matrice du complément de Schur est pleine et pour l'inverser il faut aussi inverser la matrice $\mathbf{A}$. Ce résultat suggère néanmoins d'utiliser une approximation du complément de Schur plus simple à inverser et à stocker tout en permettant de garder la convergence indépendante de la grille choisie.*

Remarque 5.14. *On retrouve, tout comme dans la résolution par élimination d'une des inconnues, qu'il faut approcher le complément de Schur d'une façon ou d'une autre pour obtenir un préconditionneur efficace.*

On montre maintenant que si on approche le complément de Schur par la matrice de masse en pression, alors on obtient un préconditionneur qui permet toujours d'obtenir une convergence en un nombre d'itération indépendant de la taille de la grille. De même si on approche la matrice $\mathbf{A}$ par la matrice diagonale par bloc du Laplacien $\tilde{\mathbf{A}}$.

Proposition 5.15. *Dans les conditions de la proposition 5.11, si on prend*

$$\mathbf{T} = \mathbf{A}\,,$$
$$P = Q\,,$$

alors le minres converge en un nombre d'itération indépendant du pas du maillage h.

Preuve : Comme dans le cas de la proposition précédente, on écrit le problème aux valeurs propres :

$$\mathbf{Au} + {}^t\mathbf{B}p = \lambda\mathbf{Au}\,,$$
$$\mathbf{Bu} = \lambda Qp\,.$$

Le cas $\lambda = 1$ est traîté de la même façon que dans la proposition précédente, et pour $\lambda \neq 1$ on a en multipliant la première équation par $\mathbf{BA}^{-1}$:

$$\lambda Qp + \mathbf{BA}^{-1}\,{}^t\mathbf{B}p = \lambda^2 Qp\,,$$
$$Q^{-1}\mathbf{BA}^{-1}\,{}^t\mathbf{B}p = (\lambda^2 - \lambda)p\,.$$

On note μ une valeur propre de $Q^{-1}\mathbf{BA}^{-1}\,{}^t\mathbf{B}$, on a alors la relation :

$$\lambda^2 - \lambda - \mu = 0\,.$$

Pour chaque valeur propre μ on obtient donc deux valeurs propres λ qui sont :

$$\lambda_\pm = \frac{1 \pm \sqrt{1+4\mu}}{2}\,.$$

Or, d'après le lemme 5.10, la matrice de masse en pression est spectralement équivalente au complément de Schur au sens où l'on a les inégalités suivantes :

$$\frac{\gamma^2}{C^2} \leq \frac{< \mathbf{BA}^{-1}\,{}^t\mathbf{B}p\,,\, p >}{< Qp\,,\, p >} \leq C^2\,,$$

où C est une constante indépendante de h. Les valeurs propres μ vérifient donc :

$$\frac{\gamma^2}{C^2} \leq \mu \leq C^2\,.$$

On obtient donc pour λ :

$$\lambda \in \left[\frac{1-\sqrt{1+4C^2}}{2}\,,\, \frac{1-\sqrt{1+4\frac{\gamma^2}{C^2}}}{2}\right] \cup \{1\} \cup \left[\frac{1+\sqrt{1+4\frac{\gamma^2}{C^2}}}{2}\,,\, \frac{1+\sqrt{1+4C^2}}{2}\right]\,.$$

Les valeurs propres λ sont donc bornées indépendemment de h, on en déduit donc que le minres converge en un nombre d'itérations indépendant de h. □

L'idée est de choisir des matrices spectralement équivalente au complément de Schur pour la partie pression et à la matrice $\mathbf{A}$ pour la partie vitesse. Le thérorème 6.6 de [30] montre que sous cette condition, le préconditionneur permet d'avoir une convergence indépendante de la finesse du maillage. On montre donc que l'on peut approcher la matrice $\mathbf{A}$ par la matrice diagonale par bloc du Laplacien $\tilde{\mathbf{A}}$.

Proposition 5.16. *Il existe deux constantes δ et Δ, indépendantes de h, telles que :*

$$\delta \leq \frac{< \mathbf{Au}, \mathbf{u} >}{< \tilde{\mathbf{A}}\mathbf{u}, \mathbf{u} >} \leq \Delta .$$

On dit que $\mathbf{A}$ *et* $\tilde{\mathbf{A}}$ *sont spectralement équivalentes.*

Preuve : La preuve de cette proposition a pratiquement été faite dans la preuve du lemme 5.10. En effet, la partie de gauche est obtenue avec l'inégalité de Korn dans le cas où l'on est dans H_0^1. L'autre partie vient de la majoration suivante :

$$\int_\Omega |D(\mathbf{u}_h)|^2 \leq \frac{1}{4} \int_\Omega |\nabla \mathbf{u}_h + {}^t\nabla \mathbf{u}_h|^2 \leq \int_\Omega |\nabla \mathbf{u}_h|^2 .$$

On a donc bien le résultat de l'énoncé. □

Le théorème 6.6 de [30] justifie alors le fait d'utiliser la matrice diagonale par bloc du Laplacien pour préconditionner notre problème de Stokes. L'avantage est que l'on peut alors utiliser des solveurs rapides des équations de Poisson pour inverser les systèmes linéaires du préconditionneur comme la transformée de Fourier rapide par exemple.

5.1.4 Résultat de convergence des différents préconditionneurs

Dans cette section on regarde le nombre d'itérations nécessaires au minres pour converger avec un résidu plus petit que 10^{-6} sur un problème de cavité entraînée et pour différents pas de maillage. Pour la vitesse on prendra soit la matrice $\mathbf{A}$ soit la matrice $\tilde{\mathbf{A}}$ et pour la pression on prendra soit la matrice de masse Q soit sa diagonale. On ne l'a pas fait mais on peut montrer (voir [30]) que la diagonale de la matrice de masse est spectralement équivalente à la matrice de masse ce qui justifie l'utilisation de ce préconditionneur.

Le tableau 5.2 résume les résultats obtenus en nombre d'itérations et en temps pour différents préconditionneurs inversés avec une décomposition LU. Les réels entre parenthèses représentent le temps en seconde pour résoudre le problème de Stokes uniquement. On ne compte pas la décomposition LU étant donnée qu'elle peut être calculée offline et récupérée ensuite. Les temps sont calculés comme la moyenne sur 100 simulations d'une cavité entraînée, ce qui permet d'avoir une bonne estimation du temps de calcul. On peut remarquer que pour tous les choix de préconditionneur, le nombre d'itérations nécessaires pour amener le résidu à 10^{-6} reste à peu près constant ce qui est le signe d'un bon préconditionneur (voir le tableau 5.1 dans le cas non préconditionné). Pour certaines tailles de grilles, même si utiliser la matrice du Laplacien par bloc augmente le nombre d'itérations, le temps de calcul est un peu plus faible qu'en utilisant la vraie matrice en vitesse. Les valeurs restent cependant très proches mais cela montre l'utilité d'utiliser des préconditionneurs moins bons en terme d'itérations mais meilleurs en ce qui concerne la facilité d'inversion. Par contre, en ce qui concerne l'utilisation de la diagonale de la matrice de masse au lieu de la matrice de masse elle même ce n'est pas le cas. On double le nombre d'itération et le gain en vitesse en utilisant la diagonale ne permet pas de compenser.

Le tableau 5.3 regroupe les résultats lorsqu'on utilise une transformée de Fourier rapide pour le préconditionnement en vitesse. Dans ce cas là, on utilise la matrice du Laplacien

h	$\mathbf{T} = \mathbf{A}$, $P = Q$	$\mathbf{T} = \tilde{\mathbf{A}}$, $P = Q$	$\mathbf{T} = \mathbf{A}$, $P = diag(Q)$	$\mathbf{T} = \tilde{\mathbf{A}}$, $P = diag(Q)$
2^{-4}	19 (0.00185)	33 (0.00234)	39 (0.0036)	54 (0.0034)
2^{-5}	19 (0.00852)	33 (0.01027)	43 (0.0178)	65 (0.0186)
2^{-6}	19 (0.05053)	32 (0.04901)	45 (0.1939)	66 (0.0903)
2^{-7}	19 (0.40709)	31 (0.49925)	43 (0.9191)	66 (1.2138)
2^{-8}	19 (3.0152)	30 (2.8548)	43 (5.1906)	64 (6.396)
2^{-9}	19 (14.22)	30 (12.2)	41 (20.72)	61 (30.9)
2^{-10}	19 (60.4)	30 (66.3)	41 (108.1)	?? (??)

TABLE 5.2 – Nombre d'itérations avec différents préconditionneurs et une méthode LU.

h	itérations	temps
2^{-4}	80	0.0072
2^{-5}	91	0.0275
2^{-6}	93	0.105
2^{-7}	90	0.43
2^{-8}	87	2.84
2^{-9}	86	13.49
2^{-10}	83	54.5
2^{-11}	79	213

TABLE 5.3 – Comportement du préconditionneur avec la FFT

par bloc pour la vitesse et on choisit de prendre la diagonale de la matrice de masse pour la pression pour que ce soit très facile à inverser. Le nombre d'itérations est beaucoup plus élevé mais cette méthode a l'avantage de pouvoir fonctionner sans construire de matrices ce qui nous permet de prendre des grilles assez fines sans avoir de problème de mémoire. De plus, la FFT étant un solveur rapide pour le Laplacien, les temps de calculs sont très faibles comparés à la méthode LU. Même dans le cas du premier précondionneur du tableau 5.2, la FFT reste plus rapide malgré le nombre d'itérations plus élevé. La taille mémoire étant beaucoup plus petite comparée à une méthode LU, c'est une très bonne façon de préconditionner le système de Stokes en deux dimensions en sequentiel.

On a lancé les mêmes calculs dans le cas $3D$. Les résultats sont récapiltulés dans les tableaux 5.4 et 5.5. En ce qui concerne les méthodes LU, la décomposition demande beaucoup de mémoire dés qu'on arrive à des pas d'espace supérieur à 2^{-6} lorsque l'on considère les termes croisés dans la matrice du préconditionneur. On s'est donc arrêté à un pas de 2^{-5} avec cette méthode. Avec une FFT, on n'a pas besoin de construire les matrices, seul le produit matrice/vecteur est codé, ce qui nous permet de prendre des pas d'espace beaucoup plus petit en trois dimensions. Dans ce cas là aussi on obtient de meilleurs résultats avec la FFT.

h	$\mathbf{T} = \mathbf{A}$, $P = Q$	$\mathbf{T} = \tilde{\mathbf{A}}$, $P = Q$	$\mathbf{T} = \mathbf{A}$, $P = diag(Q)$	$\mathbf{T} = \tilde{\mathbf{A}}$, $P = diag(Q)$
2^{-4}	29 (1.02216)	63 (1.14054)	67 (2.34465)	127 (2.22271)
2^{-5}	29 (27.5035)	64 (25.5409)	77 (65.2646)	150 (68.2274)

TABLE 5.4 – Nombre d'itérations avec différents préconditionneurs et une méthode LU en 3D.

h	itérations	temps
2^{-4}	217	0.8148
2^{-5}	254	9.1929
2^{-6}	248	98.456
2^{-7}	235	786.84

TABLE 5.5 – Comportement du préconditionneur avec la FFT en 3D

5.2 Prise en compte des particules

Dans cette section on explore le comportement du solveur permettant de minimiser la fonction coût pour imposer la contrainte de mouvement rigide sur le bord des particules. On regarde l'évolution du résidu dans le cas d'un régime dilué (ici avec une seule particule) et dans le cas d'un régime dense (quelques particules proches les unes des autres). On regarde ensuite les contrôles obtenus dans le cas d'une seule particule, de deux particules éloignées et de deux particules proches. On les compare alors avec les vecteurs propres de l'opérateur de la section 2.4.3.

5.2.1 Une particule seule

On considère une seule particule centrée dans le carré unité et de rayon 0.05. On ajoute la gravité pour faire sédimenter la particule mais on ne considère pas le terme de pression hydrostatique. Pour minimiser la fonctionnelle, on utilisera un gmres plutôt qu'un gradient conjugué au cas où, au niveau discret, l'opérateur ne soit pas exactement symétrique.

On représente alors sur la figure 5.3 l'évolution du résidu pour différentes valeurs de la précision demandée sur la résolution des problèmes de Stokes. On commence avec une précision de 10^{-2} pour finir à une précision de 10^{-10}. La grille cartésienne choisie dans ce cas est une grille de taille 256 par 256 en vitesse (et donc 128 par 128 en pression). On effectue 100 itérations du gmres pour minimiser la fonctionnelle et on trace le résidu à chaque itération. On peut observer que les résidus évoluent de la même façon au tout début puis commencent à stagner au fur et à mesure des itérations. Ceci suggère qu'il n'est pas nécessaire, dans ce cas, de prendre des précisions très élevées dans les solveurs de Stokes pour avoir un résultat comparable. Ici, prendre une précision de 10^{-3} permet d'avoir un code qui tourne deux fois plus vite qu'avec une précision de 10^{-6} pour un résidu pratiquement identique au bout d'une dizaine d'itérations. Pour 10^{-2}, on peut observer une saturation du résidu assez rapidement et sur un nombre d'itérations élevé. Une attention particulière doit donc être portée afin de choisir une précision optimale en

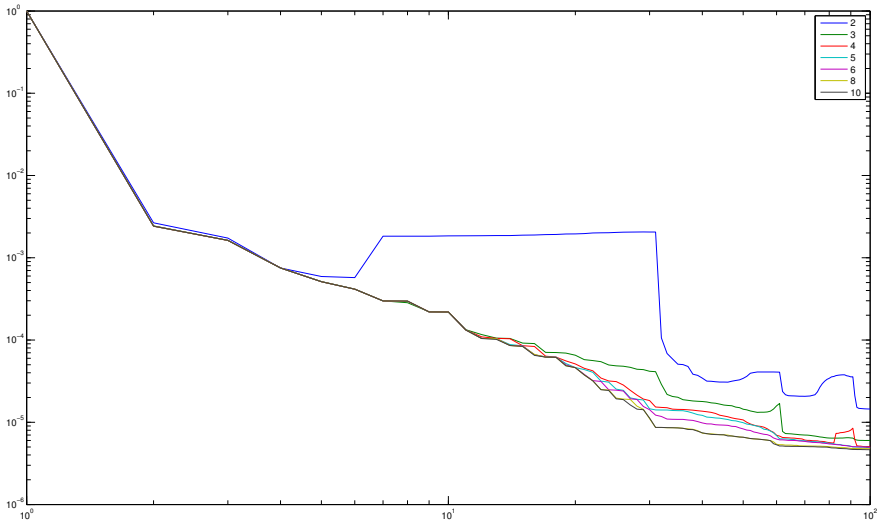

FIGURE 5.3 – Évolution du résidu pour différentes précisions sur le solveur Stokes

terme de temps de calcul et de précision du résultat. Une dernière propriété à obsever est le fait que, sur ce cas test de sédimentation, le résidu diminue très fortement sur les premières itérations, en particulier la première. On peut donc penser que quelques itérations suffisent pour obtenir un mouvement rigide ce qui est particulièrement intéressant puisqu'il faut résoudre deux problèmes de Stokes par itérations du gmres.

Afin de voir l'effet de la précision de la grille sur l'évolution du résidu, on trace sur les figures 5.4 et 5.5 l'évolution du résidu pour différents pas de maillage et une erreur de 10^{-3} et 10^{-6} sur le solveur de Stokes.

Sur la figure 5.4, on peut observer que prendre une précision de 10^{-3} ne suffit pas pour un nombre d'itérations élevé. Le solveur de Stokes n'est pas assez précis pour que le résidu continue de diminuer. L'évolution du résidu semble cependant indépendante du pas de maillage tout comme le solveur de Stokes préconditionné. On peut toutefois observer deux tendances différentes sur la première itération : pour des pas de 2^{-6} et 2^{-7}, le résidu ne chute pas aussi bas que pour les grilles de pas 2^{-8} à 2^{-10}. Sur la figure 5.5, la précision du solveur de Stokes est de 10^{-6}, c'est pourquoi on n'observe plus les oscillations du résidu dans les itérations élevées. Elles apparaissent plus tard si on augmente le nombre d'itération maximum. On retrouve les mêmes tendances que pour le cas d'une précision de 10^{-3} et on peut remarquer à nouveau que pour un nombre d'itérations inférieur à 10, le résidu est pratiquement le même sur les deux figures. La chute du résidu pour une grille grossière après 10 itérations peut être expliquée par le fait qu'il y a moins de points dans la particule, l'espace de Krylov est donc plus petit.

Afin de voir ce qu'il se passe lorsque le nombre de points dans la particule diminue, on a considéré l'évolution du résidu sur 10 itérations seulement et pour une particule de rayon de plus en plus petit. On peut alors voir les résultats sur la figure 5.6. Plus le nombre de points dans la particule est élevé, plus le résidu descend bas après la première itération. Il semble cependant qu'il y ait une valeur minimale qui pourrait dépendre du

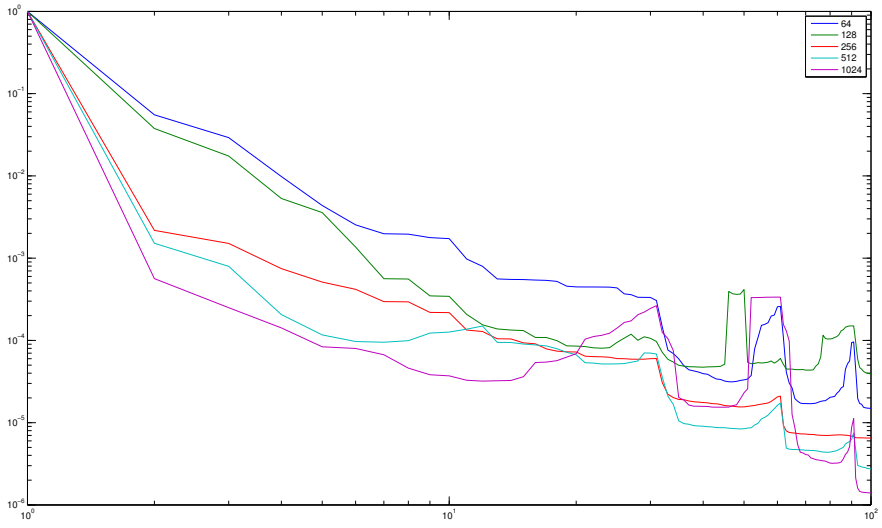

FIGURE 5.4 – Évolution du résidu pour différents pas de maillage et une erreur de 10^{-3} sur le solveur de Stokes

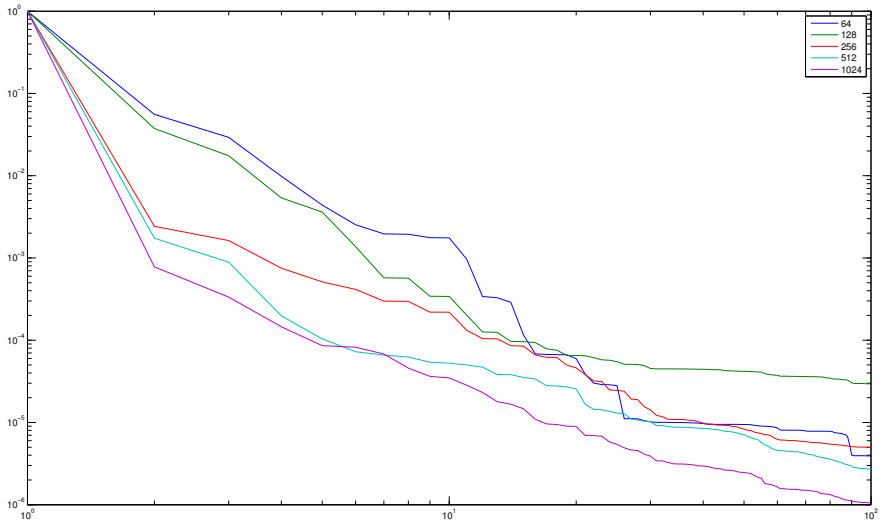

FIGURE 5.5 – Évolution du résidu pour différents pas de maillage et une erreur de 10^{-6} sur le solveur de Stokes

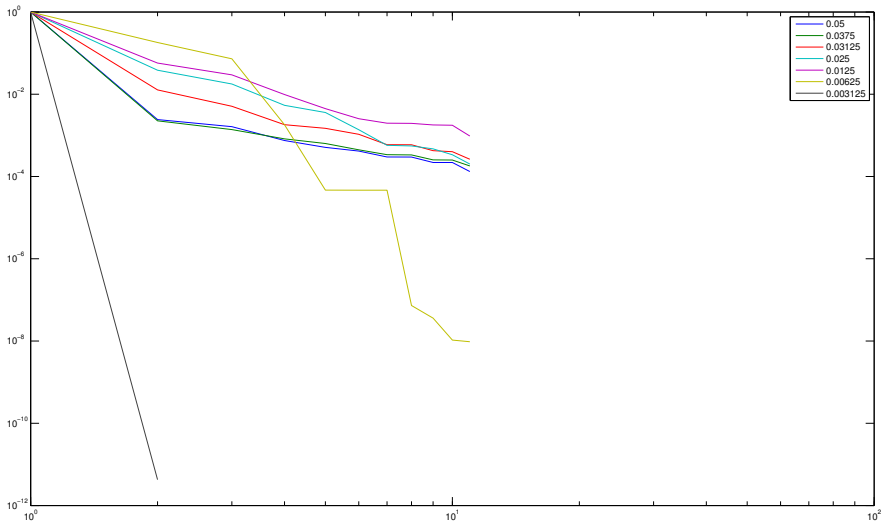

FIGURE 5.6 – Évolution du résidu pour différents rayons de la particule

pas du maillage. Deux courbes sortent du lot, ce sont celles pour de petits rayons donc lorsque le nombre de points dans la particule est très petit. On observe dans ce cas le même phénomène que sur la figure 5.5 pour un maillage grossier. Le nombre de points dans la particule étant petit, l'espace de minimisation est plus petit et le résidu peut donc atteindre la précision machine en un petit nombre d'itération. Pour le rayon le plus petit, il y a un seul point en vitesse dans la particule, d'où la convergence en une itération. Pour la seconde courbe il y a une dizaine de points dans la particule ce qui explique aussi la convergence rapide du résidu.

Après avoir regardé le résidu on regarde la solution obtenue avec quelques itérations du gmres pour minimiser la fonctionnelle d'une part, et sans aucune itération d'autre part. La figure 5.7 représente les lignes de courant de la solution obtenue dans le cas d'une précision de 10^{-6} sur le solveur de Stokes et sans aucune itération du gmres pour minimiser la fonctionnelle (avec un contrôle nul).

En comparaison, la figure 5.8 représente la solution obtenue après une itération du gmres. On ne voit plus la différence à l'oeil nu si on fait plus d'itérations et on peut voir que le champ de vitesse dans la particule n'est pas un mouvement rigide comme ce serait le cas avec une méthode de pénalisation ou de multiplicateur de Lagrange par exemple. La solution dans la particule n'est pas physique, c'est un prolongement régulier possible de la solution du problème initial.

On représente aussi sur la figure 5.9 la composante verticale du champ de vitesse sur la ligne horizontale passant par le centre du carré (et donc de la particule). La ligne pointillée représente la vitesse verticale sans que le mouvement rigide ne soit imposé sur le bord de la particule (aucune itération du gmres) et la ligne pleine représente la coupe de la solution après une itération. Sur cette coupe non plus on n'observe pas de différences lorsqu'on fait plus d'une itération pour imposer la contrainte de mouvement rigide, ce qui est totalement en accord avec le fait que le résidu chute rapidement après la première

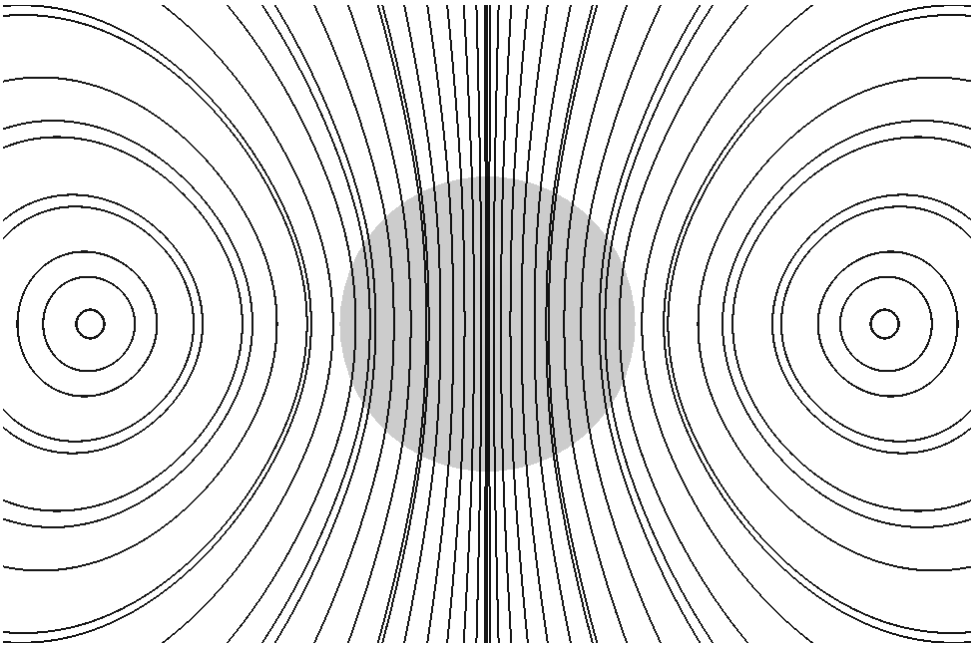

FIGURE 5.7 – Particule en sédimentation sans contrainte de mouvement rigide

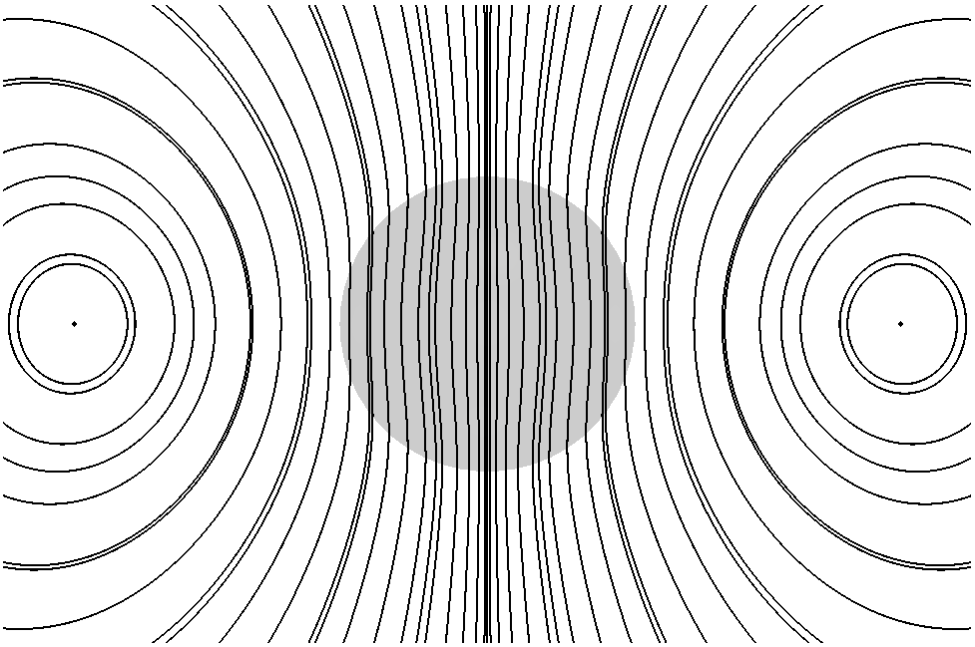

FIGURE 5.8 – Particule rigide en sédimentation

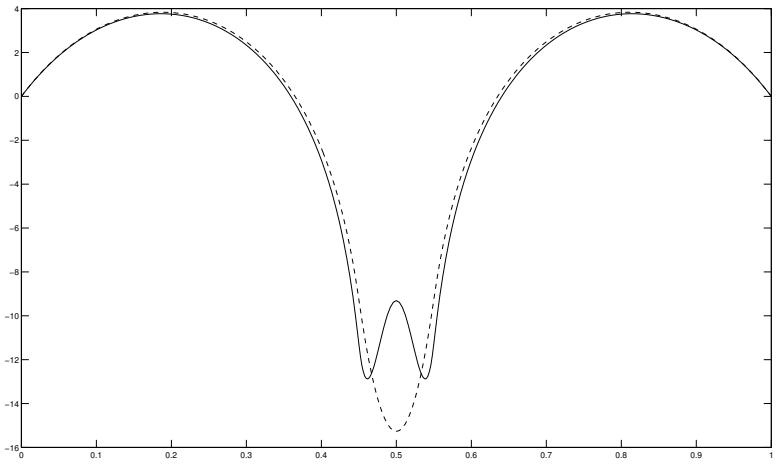

FIGURE 5.9 – Coupe de la composante verticale du champ de vitesse

FIGURE 5.10 – Lignes de courant et champ de vitesse du contrôle dans le cas d'une particule seule

itération. On observe en tout cas que la solution est bien régulière sur ces coupes.

Dans la section 2.4.3 on a regardé les vecteurs propres de l'opérateur de la fonctionnelle que l'on minimise. Comme c'est un opérateur compact auto-adjoint, on peut décomposer le contrôle sur une base de vecteurs propres dont la suite de valeurs propres associée tend vers 0. Les vecteurs propres associés aux grandes valeurs propres sont ceux qui oscillent le moins et qui représentent globalement le contrôle. On représente donc le contrôle sur la figure 5.10 avec les lignes de courant et le champ de vecteurs. On voit très clairement que les lignes de courant du contrôle sont très proches d'un des premiers vecteurs propres de l'opérateur de la fonctionnelle. Le gmres optimise donc le premier vecteur propre lors de la première itération ce qui donne l'aspect visuel à la solution puis corrige en optimisant sur des valeurs propres plus petites.

FIGURE 5.11 – Lignes de courant et champ de vitesse du contrôle dans le cas de deux particules éloignées

5.2.2 Deux particules proches et éloignées

On regarde maintenant le cas de deux particules dans le cas où elles sont éloignées puis dans le cas où elles sont proches l'une de l'autre. Le cas de deux particules éloignées est similaire à celui avec une seule particule. En effet, elles sont tellement éloignées qu'elles ne se voient pas à travers le fluide. La figure 5.11 représente le contrôle sur une particule dans le cas où l'on considère deux particules dans le rectangle $[0,1] \times [0,2]$ centrées en $(0.5, 1.5)$ et $(0.5, 0.5)$ et de rayon 0.05 toutes les deux. Le contrôle reste le même et les particules ne se voient pas.

Le cas de deux particules proches est plus intéressant. Les particules se voient à travers le fluide ce qui ralentit la convergence du gmres pour minimiser la fonctionnelle. On peut voir sur la figure 5.13 l'évolution du résidu sur 500 itérations dans le cas où l'on fait sédimenter 28 particules de rayon 0.05 chacune placée comme sur la figure 5.12. La distance minimale entre deux particules est supérieure à 10^{-3}, il se peut donc dans ce cas là qu'il n'y ait pas de point de fluide entre deux particules. On peut alors observer que le résidu diminue beaucoup plus lentement que dans le cas d'une seule particule, la vitesse est par contre constante. On n'a pas en particulier la chute après la première itération qui nous donne tout de suite une bonne solution. De plus on observe des saturations plus ou moins tôt suivant la taille de la grille (la légende représente le nombre de points en pression dans chaque direction de l'espace). Pour la première courbe qui correspond à un pas de maillage de 2^{-6}, la saturation apparaît aux alentours de 50 itérations ce qui reste beaucoup. Pour un maillage deux fois plus fin, elle apparaît encore plus tard et on ne la voit pas pour les autres tailles de grilles. Dans la pratique, on imposera aux particules de ne pas trop se rapprocher de façon à ce qu'il y ait toujours au moins une maille en pression entre deux particules, et donc deux en vitesse. Dans tous les cas, la vitesse de convergence est beaucoup plus lente et ce que suggère ces courbes c'est que le résidu relatif n'est pas dépendant de la taille de la grille.

Comme dans le cas d'une seule particule, on regarde maintenant les lignes de courant de la solution obtenue dans le cas de deux particules très proches. Elles sont centrées en $(0.445, 0.5)$ et en $(0.555, 0.5)$ et ont un rayon de 0.05 toutes les deux. On représente sur la figure 5.14 la solution dans le cas d'un contrôle nul, donc lorsque la contrainte de

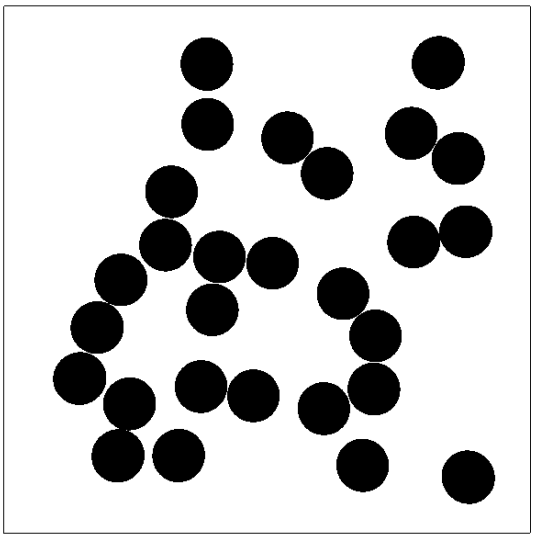

FIGURE 5.12 – Positions des particules dans le cas dense

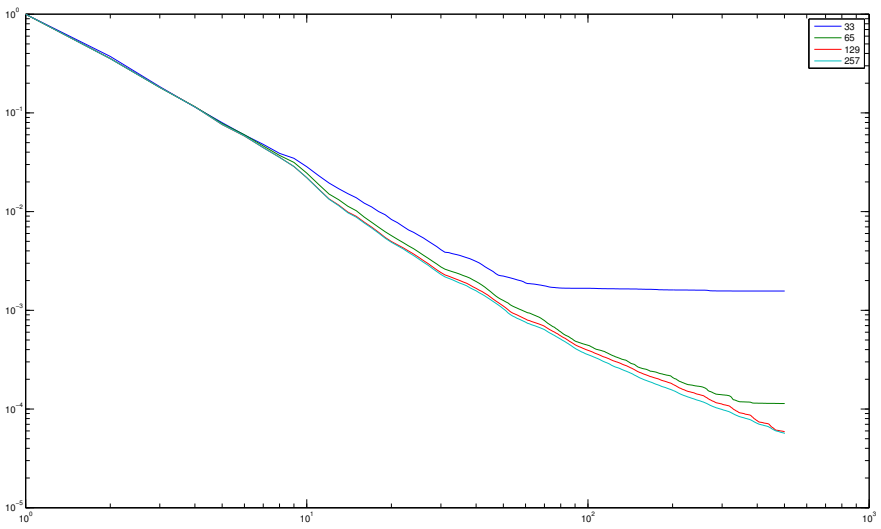

FIGURE 5.13 – Évolution du résidu dans le cas dense

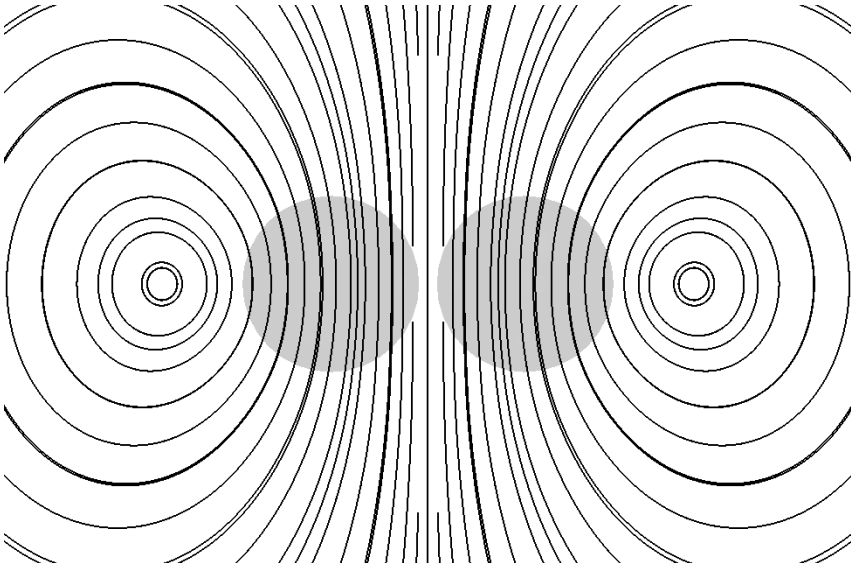

FIGURE 5.14 – Lignes de courant pour 2 particules proches

mouvement rigide n'est pas imposée, et sur la figure 5.15, on représente la solution après 10 itérations du gmres pour minimiser la fonctionnelle. Comme dans le cas d'une seule particule, la solution dans les inclusions n'est pas un mouvement rigide et n'a pas de signification physique. C'est un prolongement de la solution dans le domaine fluide.

La figure 5.16 représente, comme dans le cas d'une seule particule, la coupe de la composante verticale selon la droite horizontale passant par le centre des deux particules. La courbe pointillée représente la vitesse sans imposer la contrainte de mouvement rigide, celle en tiret représente la vitesse après une itération et la ligne pleine représente la vitesse après 40 itérations. Cette fois ci on n'a clairement plus une convergence très rapide en une itération comme dans le cas d'une seule particule car le contrôle n'est pas proche d'un vecteur propre, mais est bien la combinaison de plusieurs vecteurs propres dont certains sont associés à des valeurs propres petites.

On regarde alors le contrôle dans le cas de deux particules proches, situées l'une au dessus de l'autre, et on le compare avec les vecteurs propres de l'opérateur de la fonctionnelle obtenus dans la partie 2.4.3. La figure 5.17 représente le contrôle obtenu après une seule itération du gmres pour imposer la contrainte de mouvement rigide. Il est dans ce cas là aussi très proche d'un des premiers vecteurs propres. Par contre, on a vu que le résidu diminuait beaucoup plus lentement et que même après une dizaine d'itérations, on pouvait voir la solution changer. On trace donc dans la figure 5.18 le contrôle après 2 itérations et sur la figure 5.19 le contrôle après 10 itérations. Cette fois-ci, le contrôle n'a plus rien à voir avec un des premiers vecteurs propres et c'est ce qui explique la convergence plus lente. Le fait que les particules soient proches fait que le contrôle cherché est plus compliqué et contient des contributions de vecteurs propres associés à de petites valeurs propres qui ralentissent la convergence du gmres.

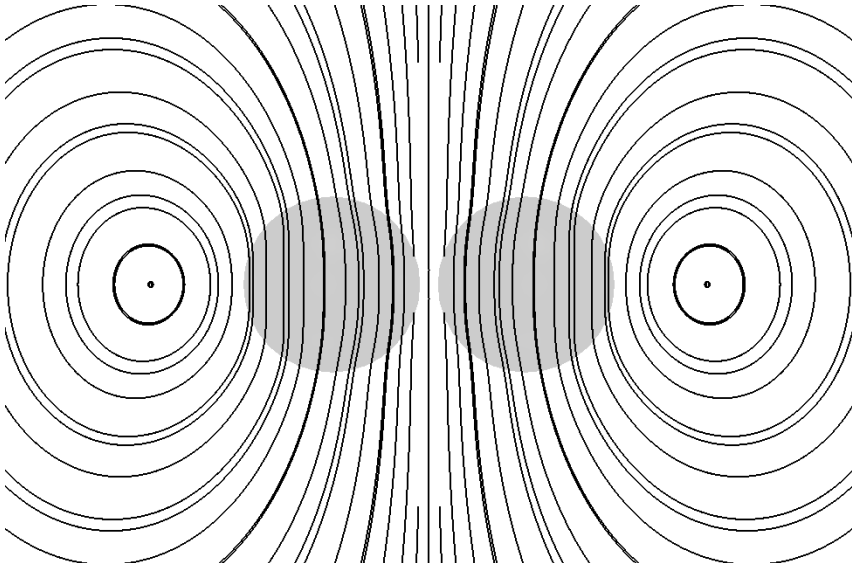

FIGURE 5.15 – Lignes de courant pour 2 particules rigides proches

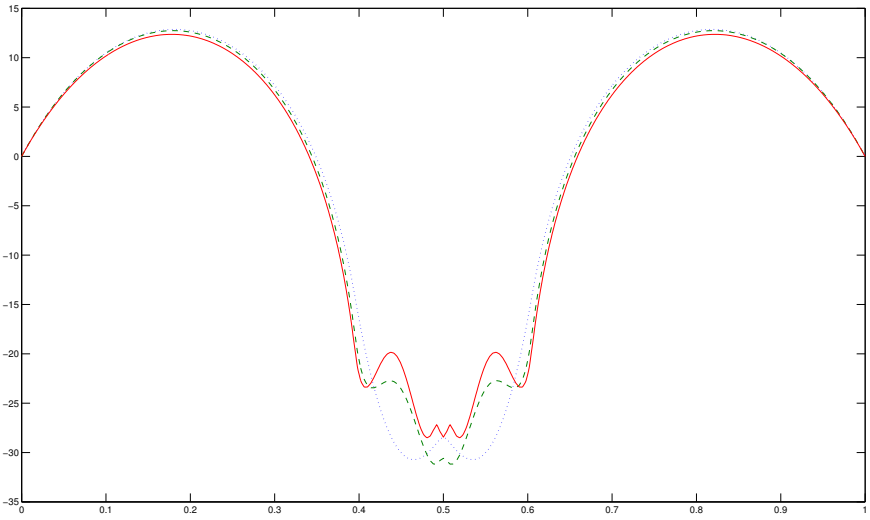

FIGURE 5.16 – Coupe de la composante verticale de la vitesse avec deux particules

FIGURE 5.17 – Lignes de courant et champ de vitesse du contrôle dans le cas de deux particules proches après une itération

FIGURE 5.18 – Lignes de courant et champ de vitesse du contrôle dans le cas de deux particules proches après deux itérations

FIGURE 5.19 – Lignes de courant et champ de vitesse du contrôle dans le cas de deux particules proches après dix itérations

Chapitre 6

Description du code

Sommaire

6.1 Structure du code

On détail maintenant le code développé dans la thèse et écrit en C++. Pour écrire ce code, on a choisi d'utiliser la librairie *PETSc* (voir [11, 10, 12]). C'est une librairie écrite en C, qui dispose de structures pour les vecteurs et matrices creuses ainsi que des solveurs de système linéaire. Les raisons pour lesquelles on utilise ce genre de librairie sont les suivantes :

- Elles sont testées par beaucoup d'utilisateur, on dispose donc rapidement d'opérations d'algèbre linéaire sans bug et bien plus optimisées que celles que l'on pourrait écrire soit même.
- Les méthodes sont déjà codées et utilisables directement. Il n'y pas besoin de passer du temps sur le développement de méthodes générales. Avec la librairie PETSc, il est par exemple possible de spécifier le type de solveur et de préconditionneur directement dans la ligne de commande utilisée pour lancer le code. C'est très agréable pour tester différentes méthodes de résolutions.
- Enfin, le code est déjà parallélisé. Toutes les opérations matrices/vecteurs, les solveurs, la décomposition du domaine sont déjà codés. C'est un avantage pour avoir un code parallèle rapidement. Il reste bien sûr à paralléliser toute la partie spécifique au problème que l'on souhaite résoudre, comme la gestion des particules dans notre cas, mais pour tout ce qui est opérations sur les matrices et les vecteurs, tout est fait.

Le code est constitué de deux classes principales : une classe pour le problème et une classe pour les particules. La classe `Problem` agit au niveau du domaine tout entier alors que la classe `Particle` concerne les opérations sur une particule, donc sur quelques points du maillage. On parlera de la gestion des particules dans la section suivante. On décrit maintenant le code permettant de minimiser la fonctionnelle quadratique. L'algorithme global est le suivant :

1. Création d'un objet de la classe `Config` contenant tous les paramètres du problème.
2. Création d'un objet de la classe `Domain` contenant toutes les données concernant le maillage cartésien.
3. Création des matrices pour les solveurs et les préconditionneurs une fois pour toute. Assemblage ou non des matrices (au choix). On calcule aussi les factorisations éventuelles.
4. Allocation des particules.
5. Boucle en temps :
   (a) Création d'un objet de la classe `Problem` dépendant de la position des particules.
   (b) Minimisation à l'aide d'un solveur type gradient conjugué ou gmres.
   (c) Calcul de la vitesse de chaque particule.

(d) Déplacement des particules en fonction de la vitesse calculée.

6. Fin de la boucle en temps.

Tout d'abord, on lit les paramètres du problème dans un fichier de configuration. Les paramètres sont les paramètres physiques qui interviennent dans le système d'équations mais aussi la taille du domaine rectangulaire, le pas de maillage dans chaque direction, le type de conditions aux bords que l'on veut mettre, etc. Pour l'instant seul les conditions de type Dirichlet ou périodiques sont prises en compte. Toutes ces données sont lues et sauvées dans un objet de la classe `Config` de notre code. C'est un objet qui est créé dés le départ et que l'on garde tout au long de la simulation. On a de plus choisi de mettre des paramètres pour le choix du solveur et du préconditionneur dans le fichier de configuration mais bien entendu, si un autre choix est fait en ligne de commande c'est celui-ci qui est gardé.

Une fois l'objet de la classe `Config` créé, on peut construire un objet de la classe `Domain` qui contient toutes les données sur le domaine. On ne considère que des maillages cartésiens et PETSc a justement un type de données pour les maillages uniformes. Dans cette classe `Domain` on retrouve donc le pas de maillage mais aussi la taille du domaine ainsi que le nombre de degrés de liberté du domaine de chaques processus en parallèle. C'est en effet à cet endroit que le domaine est découpé sur tous les processus de façon à ce qu'ils aient tous à peu près le même nombre de degrés de liberté. Comme on utilise des éléments finis $4Q_1Q_1$, on dispose en réalité de deux grilles cartésiennes : une pour la pression et une pour la vitesse. Celle pour la vitesse est deux fois plus fine que celle pour la pression. Avec PETSc, on crée un objet qui permet de stocker plusieurs grilles différentes. La fonction est la suivante :

```
DM pack;
DMCompositeCreate(PETSC_COMM_WORLD, &pack);
```

Il reste à créer les deux grilles et à les ajouter dedans. On pourra ensuite récupérer les grilles directement à partir de la variable `pack`. Dans notre cas, on veut créer deux grilles de tailles différentes mais on aimerait que le découpage selon les différents processus soit le même. On commence donc par créer la grille en pression en laissant PETSc décider du découpage suivant le nombre de processus demandé :

```
DM daPre;
DMDACreate2d(PETSC_COMM_WORLD, bnd[0], bnd[1], DMDA_STENCIL_BOX,
        np[0], np[1], PETSC_DECIDE, PETSC_DECIDE,
        1, stencil, 0, 0, &daPre);
DMDASetFieldName(daPre, 0, "p");
DMDASetUniformCoordinates(daPre, 0.0, length[0], 0.0, length[1], 0.0, 0.0);
```

Le premier argument de la fonction `DMDACreate2d` qui crée la grille en pression est le communicateur contenant tous les processus. Les deux arguments qui suivent sont le type de conditions aux bords utilisées. En parallèle, les domaines de chaque processus se chevauchent légèrement, le nombre de maille dans ce recouvrement étant choisi dans la variable `stencil` (que l'on prend à 1 dans le code). On crée donc une maille fantôme

en pression sur chaque bord du domaine local au processus. Le type de stencil utilisé est fixé par l'argument `DMDA_STENCIL_BOX`, qui veut dire que l'on prend aussi une maille fantôme dans les coins. Si les conditions de bords sont périodiques, on aura donc des mailles fantômes pour tout le monde, par contre si les conditions sont de type Dirichlet, les domaines locaux qui contiennent un bord du domaine global n'ont pas de mailles fantômes sur ce bord. Après le type de stencil utilisé, on donne le nombre de points dans chaque direction et on utilise les arguments `PETSC_DECIDE` pour laisser PETSc découper le domaine tout seul. L'argument 1 qui suit permet de dire qu'il y a un degré de liberté en chaque point du maillage pour la pression. On a déjà vu à quoi servait l'argument `stencil`, les deux zéros qui suivent servent si l'ont a pas mis `PETSC_DECIDE` dans le nombre de processus par direction. Ils permettent de spécifier le nombre de points pour chaque processus dans chaque direction. Comme on laisse PETSc se débrouiller tout seul, on met des zéros pour ne pas les prendre en compte. La troisième ligne permet de donner un nom à la grille et la quatrième ligne définie les coordonnées de chacun des points sur la grille.

Maintenant que la grille en pression est créée, on s'occupe de celle en vitesse qui est deux fois plus fine. On veut avoir le même découpage que la grille en pression, on va donc spécifier explicitement le découpage au lieu d'utiliser les arguments `PETSC_DECIDE`. On récupère tout d'abord le découpage réalisé par PETSc pour la grille en pression :

```
int cpux, cpuy;
DMDAGetInfo(daPre, 0, 0, 0, 0, &cpux, &cpuy, 0, 0, 0, 0, 0, 0, 0);

const int *lx, *ly;
DMDAGetOwnershipRanges(daPre, &lx, &ly, 0);

int lxu[cpux], lyu[cpuy];
for(int i=0; i<cpux; i++)
  lxu[i] = 2*lx[i]-1;

for(int i=0; i<cpuy; i++)
  lyu[i] = 2*ly[i]-1;
```

La fonction `DMDAGetInfo` permet de récupérer beaucoup d'informations sur une grille. On ne récupère que le nombre de processus dans chaque direction ici. On récupère le nombre de points par processus dans chaque direction avec la fonction `DMDAGetOwnershipRanges`. On construit ensuite deux tableaux où l'on fixe le nombre de points en vitesse pour chaque processus et dans chaque direction. Comme la grille en vitesse est deux fois plus fine, on a deux fois plus de points. La grille en vitesse est ensuite créée de la même façon que celle en pression :

```
DM daVel;
DMDACreate2d(PETSC_COMM_WORLD, bnd[0], bnd[1], DMDA_STENCIL_BOX,
        nv[0], nv[1], cpux, cpuy,
        2, 2*stencil, lxu, lyu, &daVel);
DMDASetFieldName(daVel, 0, "u");
DMDASetFieldName(daVel, 1, "v");
DMDASetUniformCoordinates(daVel, 0.0, length[0], 0.0, length[1], 0.0, 0.0);
```

C'est très similaire à la pression excepté qu'ici on spécifie explicitement le nombre de processus et de points dans chaque direction. On prend aussi un stencil deux fois plus grand que pour la pression et on dit à PETSc que la vitesse est un champ dans $\mathbb{R}^2$ (dans cet exemple, on est en deux dimensions).

De même que pour l'objet de la classe `Config`, cet objet de la classe `Domain` est créée dès le départ une fois pour toute et est utilisé tout au long de la simulation. En effet, puisqu'on utilise un maillage cartésien qui ne voit pas les particules, il ne dépend pas du temps et on peut donc utiliser le même à chaque fois.

Une fois le domaine découpé, on construit les matrices que l'on ne changera pas durant tout le problème. C'est un des avantages de cette méthode, les matrices ne dépendent pas de la position des particules. Les matrices étant creuses, on préalloue l'espace mémoire nécessaire en indiquant le nombre d'éléments non nuls de chaque matrice avant de les construire. Pour déclarer une matrice avec PETSc, on utilise les fonctions suivantes :

```
Mat A;
MatCreate(PETSC_COMM_WORLD, &A);
MatSetSizes(A, localsize, localsize, totalsize, totalsize);
MatSetFromOptions(A);
```

La fonction `MatSetSizes` permet de spécifier la taille de la matrice au niveau local et au niveau global. Beaucoup de fonctions sont disponibles pour ajuster les paramètres de la matrice, comme par exemple le type de format creux que l'on veut utiliser. Dans cet exemple, on laisse les paramètres par défaut et on ajoute l'appel à la fonction `MatSetFromOptions` qui permet de prendre en compte les informations ajoutées en ligne de commande lorsqu'on lance le programme. On précise ensuite où se trouve les éléments non nuls de la matrice afin de la préallouer correctement pour éviter de perdre du temps lors de son initialisation. En effet, PETSc dispose de fonctions qui réalisent cette préallocation directement. Malheureusement, cela ne marche pas avec le type d'éléments finis que l'on a choisi. Les exemples que l'on trouve dans la documentation concernent des préallocations et des assemblages de matrices dans le cas d'éléments finis Q_1Q_1 avec stabilisation. Il n'y a donc qu'une seule grille et PETSc est capable d'assembler la matrice tout seul dans ce cas là. Dans notre cas, on dispose de deux grilles différentes pour la pression et la vitesse et PETSc n'est pas capable de préallouer et d'assembler les matrices. Nous avons donc écrit nos propres fonctions de préallocations et d'assemblages. L'idée pour la préallocation est de trouver où se trouve les éléments non nuls de la matrice et de les transmettre à la fonction suivante via les tableaux `rowsu` et `colsu` :

```
MatPreallocateSetLocal(ltog, nrows, rowsu, ltog, ncols, colsu, dnz, onz);
```

Ici la variable `ltog` est un tableau faisant le lien entre indices locaux et indices globaux que l'on peut récupérer directement à partir du domaine via la variable `pack` qui contient les deux grilles :

```
ISLocalToGlobalMapping ltog;
DMGetLocalToGlobalMapping(pack, &ltog);
```

La variable `nrows` contient le nombre de lignes où l'on va ajouter des éléments non nuls et les indices de ces lignes sont donnés dans le tableau `rowsu`, de même pour les colonnes.

Les pointeurs `dnz` et `onz` sont nécessaire et sont gérés par PETSc, l'utilisateur n'a pas besoin de les modifier.

On initialise ensuite les matrices à l'aide de la fonction `MatSetValuesLocal` qui fonctionne un peu comme la fonction `MatPreallocateSetLocal` pour préallouer l'espace mémoire. On lui donne les indices des lignes et colonnes des entrées de la matrice que l'on veut modifier. Cette fonction ne rentre pas directement les valeurs dans la matrice mais les garde en caches. Pour construire réellement la matrice, il faut faire appel à deux fonctions qui vont entrer toutes les données en caches de façon optimisée. On peut donc ainsi ajouter petit à petit les valeurs sans perdre en temps. Les deux fonctions sont les suivantes :

```
MatAssemblyBegin(A, MAT_FINAL_ASSEMBLY);
MatAssemblyEnd(A, MAT_FINAL_ASSEMBLY);
```

où A est ici la matrice en question.

On initialise ensuite les tableaux contenant les informations sur les particules, comme la position des centres, leurs rayons, etc. Ne pas mettre de particules revient en fait à résoudre un problème de Stokes.

On entre maintenant dans la boucle en temps. À ce niveau là, on a toutes les informations pour créer le problème. Au début de la boucle, on crée un objet de la classe `Problem` qui va alors initialiser les particules. Le but est de créer un tableau d'objet de la classe `Particle`. Pour que cela fonctionne en parallèle, on a choisi de créer un tableau avec toutes les particules pour chaque processus. Dans la classe `Particle`, on met alors une variable à 1 si le processus a effectivement la particule en question et 0 sinon. Ceci permet ainsi à chaque processus de ne traiter que les particules qui sont dans son domaine. Il faut faire ici très attention dans le cas de conditions périodiques car une particule peut alors se trouver d'un coté et de l'autre du domaine global. L'avantage de créer un tableau avec toutes les particules pour chaque processus est que l'on va pouvoir utiliser des réductions pour calculer les intégrales permettant d'obtenir la vitesse de chaques particules afin que tous les processus aient toutes les vitesses. C'est essentiel car chaque processus déplace les particules à la fin du pas de temps, il est donc nécessaire que les particules se retrouvent au même endroit pour tout le monde. De plus, on ne considérera pas un nombre de l'ordre du millions de particules. La place mémoire occupée par ces tableaux est donc très petite comparée à celle des matrices du problème par exemple. Une fois qu'on a les matrices du problème, on a besoin de pouvoir résoudre des systèmes linéaires. Il faut pour cela construire un solveur à l'aide de la commande `KSPCreate` :

```
KSP solver;
KSPCreate(PETSC_COMM_WORLD, &solver);
KSPSetOptionsPrefix(solver, "stokes_");
KSPSetOperators(solver, A, B, SAME_PRECONDITIONER);
KSPSetFromOptions(solver);
```

La troisième ligne permet de donner un préfixe au solveur pour pouvoir spécifier ses paramètres directement en ligne de commande et uniquement pour lui. Par exemple pour dire que l'on veut utliser un minres, il suffira dans ce cas là d'ajouter l'option `-stokes_ksp_type minres` dans la ligne de commande qui appelle le programme. La fonction `KSPSetFromOptions` permet de prendre en compte ces options. Il est aussi tout à fait possible de rentrer ces paramètres directement dans le code mais on perd en sou-

plesse d'utilisation puisqu'il faut recompiler le code à chaque fois que l'on veut changer de solveur. La quatrième ligne permet d'attacher une matrice A et une matrice de préconditionneur B à un solveur. L'option `SAME_PRECONDITIONER` permet de dire que le préconditionneur reste le même lorsqu'on résout plusieurs fois le système linéaire. Pour résoudre un système linéaire il suffit alors d'utiliser la fonction `KSPSolve` :

```
KSPSolve(solver, rhs, sol);
```

L'argument `rhs` est le vecteur qui représente le second membre de l'équation et le vecteur `sol` est le vecteur qui va contenir la solution du système linéaire.

On minimise ensuite la fonctionnelle à l'aide de PETSc. Pour cela, on crée un produit matrice/vecteur chargé d'effectuer l'opération du premier algorithme de la section 2.4.4. Avec PETSc, on lui spécifie une fonction pour effectuer le produit matrice/vecteur. C'est fait en créant une matrice un peu spéciale avec les commandes suivantes :

```
Mat T;
MatCreate(PETSC_COMM_WORLD, &T);

MatSetSizes(T, size, size, PETSC_DECIDE, PETSC_DECIDE);
MatSetType(T, MATSHELL);
MatShellSetOperation(T, MATOP_MULT, (void(*)(void))gradJ);

MatContextJ *matctxJ = new MatContextJ;
/*
On initialise le contexte ici
*/
MatShellSetContext(T, matctxJ);
```

L'idée est donc de créer une matrice T et de spécifier qu'elle ne sera pas construire mais qu'elle réalisera le produit matrice/vecteur avec la commande `MatSetType(T, MATSHELL)`. On dit que c'est une opération qui fait le produit matrice/vecteur en utilisant notre propre fonction qui s'appelle ici `gradJ` avec la fonction `MatShellSetOperation`. la fonction `gradJ` doit avoir les mêmes arguments que la fonction qui réalise le produit matrice/vecteur dans PETSc. Elle est donc déclarée de la façon suivante :

```
PetscErrorCode gradJ(Mat A, Vec x, Vec y);
```

En particulier, on ne dispose d'aucunes autres informations que celle accessible par la matrice en question A et les vecteurs d'entrée et de sortie. C'est pour cette raison que l'on crée un contexte, qui est une structure dans laquelle on met toutes les variables dont on aura besoin pour effectuer le produit matrice/vecteur. On associe ensuite le contexte à la matrice avec l'appel `MatShellSetContext(T, matctxJ)`. Dans la fonction `gradJ`, il suffit alors d'appeler la fonction `MatShellGetContext` pour récupérer les données dont on a besoin. On est alors libre d'utiliser tout solveur de système linéaire qui n'a besoin que du produit matrice/vecteurs et non de la matrice tout entière. On utilise donc des méthodes de Krylov comme le gradient conjugué ou le gmres. La aussi on peut changer de solveur directement dans la ligne de commande.

Une fois le contrôle permettant d'imposer la condition de mouvement rigide trouvé, on peut résoudre un dernier problème de Stokes et intégrer la vitesse du fluide sur le bord des particules pour avoir leurs vitesses. Une communication collective permet alors de passer

cette information à tout le monde pour que chacuns déplacent les particules de la même façon. C'est fait de la façon suivante :

```
for (int i=0; i<_particles.size(); i++){
  if (_particles[i]->hasPart){
    _particles[i]->get_velocity(_dom, solLocal, mean, i);
  }
}
MPI_Allreduce(mean, mean_tot, dim*_particles.size(), MPI_DOUBLE,
      MPI_SUM, PETSC_COMM_WORLD);
```

On parcourt toutes les particules et si le processus en question contient effectivement la particule, il appelle la fonction `get_velocity` de la classe `Particles` qui permet d'intégrer le champ de vitesse fluide qui se trouve dans le vecteur `solLocal` sur le bord de la particule contenu dans le domaine du processus et de mettre la vitesse obtenue au bon endroit dans le tableau `mean` initialisé à zéro. Ainsi, lors de l'appel à la communication collective `MPI_Allreduce`, les processus qui n'ont pas la particule envoient une vitesse nulle et les autres envoient leur contribution. On fait une somme de toutes les contributions pour obtenir la vitesse réelle de la particule qui est alors stockée dans le tableau `mean_tot` et envoyée à tous les processus. Tout le monde a donc la bonne vitesse pour déplacer la particule. On utilise un simple schéma d'Euler explicite pour déplacer les centres des particules. Comme elles sont rigides, on sait à tout moment où se trouvent leurs bords.

6.2 Gestions des particules

La classe `Particle` est celle où l'on traîte toutes les opérations sur les particules. Ce sont les opérations locales que l'on trouve dans les différents algorithmes déjà vu. On décrit ici trois fonctions importantes : le constructeur de la classe qui initialise les différents tableaux liés à la particule, la construction de la distribution simple couche que l'on retrouve dans le problème permettant de calculer l'image d'un point par l'opérateur de la fonctionnelle et la projection sur l'ensemble des contrôles admissibles.

Le constructeur permet, si la particule ajoutée est bien une particule possédée par le processus en question, d'initialiser plusieurs tableaux utiles pour la suite. Tout d'abord on définit un petit carré, ou cube en 3D, qui contient la particule. Ceci nous permet de parcourir moins de points pour ensuite construire le tableau contenant les indices des points situés dans la particule. Le contrôle sur la particule peut alors être initialisé à zéro sur ces points. Le contrôle global est donc en réalité un ensemble de petit tableaux locaux, chacun défini dans la classe `Particle`. Dans le but de construire la distribution simple couche en l'approchant par une combinaison de masse de Dirac, on construit un ensemble de points régulièrement espacés sur la surface de la particule. En deux dimensions c'est très simple, il suffit de prendre des points régulièrement espacés sur le cercle qui définissent le bord de la particule. En trois dimensions, on prend des points régulièrement espacés sur un cube de référence que l'on translate et dilate ensuite pour en faire une sphère. On obtient de cette façon des points sur la surface de la sphère et en 2D comme en 3D on appelle `mesh_sphere` le tableau contenant les coordonnées de ces points.

Dans la classe `Particle`, on dispose d'une fonction interpolant un champ de vitesse

défini sur tout le domaine Ω en les points définis par le tableau `mesh_sphere`. Dans notre cas, on considère un maillage cartésien, les fonctions de bases sont donc connues et pour interpoler la valeur en un point d'un champ défini globalement, il suffit de repérer le carré (ou cube en 3D) dans lequel il se trouve et de calculer la participation des 4 (8 en 3D) fonctions de bases non nulles en ce point. On accède tout d'abord à la grille en vitesse à partir de la grille composite `pack` :

```
DM pack = dom->get_pack();
DM dau, dap;
DMCompositeGetEntries(pack, &dau, &dap);
```

la variable `dom` est un objet de la classe `Domain`, c'est de cette façon que l'on récupère la grille composite. On extrait ensuite les deux grilles en vitesse et en pression avec la fonction `DMCompositeGetEntries`. On récupère maintenant le champ de vitesse global dans un pointeur pour avoir accès aux valeurs.

```
Fieldv2d **psolu;
DMDAVecGetArray(dau, sol, &psolu);
```

la structure `Fieldv2d` est une structure que l'on a créée et qui contient deux réels en deux dimensions. Il existe une autre structure `Fieldv3d` qui contient trois réels en dimension trois pour faciliter l'accès aux différentes composantes de la vitesse comme on le verra juste après. On parcourt ensuite le tableau `mesh_sphere` et on repère le carré dans lequel se trouve le point :

```
for(int j=0; j<size; j++){
  // Coordonees du point
  p[0] = pmsphere_per[2*j];
  p[1] = pmsphere_per[2*j+1];

  // indice du coin bas/gauche du carre dans lequel il se trouve
  ind[0] = (int)floor(p[0]/_h[0]);
  ind[1] = (int)floor(p[1]/_h[1]);

  // distance entre le point et le coin
  dx[0] = p[0] - ind[0]*_h[0];
  dx[1] = p[1] - ind[1]*_h[1];
```

Ici `pmsphere_per` est un pointeur permettant d'accéder aux valeurs du tableau `mesh_sphere`. On calcule simplement l'indice du carré dans lequel se trouve le point auquel on va interpoler la vitesse et on calcule la distance du point au coin. Comme on connaît les fonctions de bases, on peut alors interpoler la vitesse en ce point :

```
pg[j] = (psolu[ind[1]][ind[0]].u*(dx[0]-_h[0])*(dx[1]-_h[1]) +
     psolu[ind[1]][ind[0]+1].u*dx[0]*(_h[1]-dx[1]) +
     psolu[ind[1]+1][ind[0]].u*(_h[0]-dx[0])*dx[1] +
     psolu[ind[1]+1][ind[0]+1].u*dx[0]*dx[1])/(_h[0]*_h[1]);

pg[j+size] = (psolu[ind[1]][ind[0]].v*(dx[0]-_h[0])*(dx[1]-_h[1]) +
     psolu[ind[1]][ind[0]+1].v*dx[0]*(_h[1]-dx[1]) +
     psolu[ind[1]+1][ind[0]].v*(_h[0]-dx[0])*dx[1] +
     psolu[ind[1]+1][ind[0]+1].v*dx[0]*dx[1])/(_h[0]*_h[1]);
```

On voit bien la contribution des 4 fonctions de bases non nulles en ce point pour les deux composantes de la vitesse. On accède aux différentes composantes de la vitesse avec le même indice en spécifiant avec un `.u` ou un `.v` si c'est la première ou la deuxième composante. C'est grâce à l'utilisation de la structure `Fieldv2d` que l'on peut utiliser le même indice de cette façon. Le tableau `g` contient à la fin l'ensemble des valeurs interpolées, que l'on pourra utiliser dans la suite pour construire la distribution simple couche ou bien calculer la vitesse de la particule. Bien entendu, ce tableau `g` est propre au processus et ne contient que les valeurs interpolées appartenant au domaine du processus en question. Il faudra ensuite faire des communications entre les différents processus pour prendre en compte toutes les contributions.

Pour construire la distribution simple couche en utilisant l'approximation par une combinaison de masses de Dirac présentée dans le chapître 3, on appelle tout d'abord, pour chaque particule, la fonction qui permet d'interpoler la vitesse sur le bord des particules présentée juste au dessus. En reprenant les notations du chapitre 3, on veut calculer les intégrales de la forme :

$$\int_\gamma \varphi\phi\,,$$

où φ est la distribution simple couche, γ le bord d'une particule et ϕ une fonction de base. Si on approche la distribution simple couche par une combinaison de masses de Dirac, l'intégrale devient une somme de la forme suivante :

$$\sum_{i=1}^{N} \lambda_i \phi(x_i)\,,$$

où les λ_i sont les coefficients de la combinaison de masse de Dirac qui sont égaux à l'intégrale de la distribution simple couche sur une petite portion du bord de la particule. On peut approcher cette intégrale par la valeur en un point, on choisit de prendre le point x_i :

$$\int_{\gamma_i} \varphi \simeq |\gamma_i| \varphi_h(x_i)\,,$$

où φ_h est définie avec le champ de vitesse discret. On obtient donc l'approximation suivante dans la formulation variationelle :

$$\int_\gamma \varphi\phi \simeq \sum_{i=1}^{N} |\gamma_i| \phi(x_i)\varphi_h(x_i)\,.$$

Dans la pratique, la taille $|\gamma_i|$ est constante, on peut donc la sortir de la somme et on obtient :

$$\int_\gamma \varphi\phi \simeq \frac{|\gamma|}{N} \sum_{i=1}^{N} \phi(x_i)\varphi_h(x_i)\,.$$

Au niveau du code, comme seul les fonctions de bases dont le support intersecte le bord d'une particule ajoutent une contribution à l'intégrale, on parcourt l'ensemble des points de `mesh_sphere` et on calcule les 4 (ou 8 en 3D) contributions des fonctions de bases

concernées. Les valeurs de $\varphi_h(x_i)$ sont obtenues avec la fonction qui interpole le champ de vitesse.

Pour faire la projection sur l'espace K, il faut calculer des moyennes en volume. Pour cela, on parcourt cette fois-ci le petit carré ou cube qui contient la particule avec un petit pas d'espace et on interpole la vitesse en chaque point situé dans la particule. On fait alors une moyenne pour approcher l'intégrale en volume. Bien sûr, ici aussi chaque processus calcule sa partie et tout est mis en commun à l'aide d'une réduction globale pour obtenir les vraies moyennes.

C'est dans cette classe aussi que l'on calcule la vitesse de translation et de rotation de la particule une fois le dernier problème de Stokes résolu. Cela consiste à interpoler le champ de vitesse sur les points de `mesh_sphere` contenu dans le domaine du processus, à les sommer et à diviser la valeur obtenue par le nombre total de point sur le bord de la particule. C'est la fonction `get_velocity` que l'on a déjà vu précédemment :

```
void StParticle::get_velocity(Domain *dom, Vec solLocal, double *mean,
            int ipart){
  int size;
  VecGetSize(_msphere, &size);
  size /= _dim;

  interpol(dom, solLocal);

  double *pg;
  VecGetArray(g, &pg);

  // Calcul de la moyenne de u_x et u_y
  for(int k=0; k<_dim; k++){
    mean[ipart*_dim + k] = 0.;
    for(int j = 0; j<size; j++){
      mean[ipart*_dim + k] += pg[j + k*size];
    }
    mean[ipart*_dim + k] /= _nbSurfPoints;
  }
  VecRestoreArray(g, &pg);
}
```

Les premières lignes servent à récupérer le nombre de point sur le bord de la particule contenu dans le domaine du processus. On appelle ensuite la fonction `interpol` qui interpole les valeurs du champ de vitesse fluide et les met dans le tableau `g`. On somme ensuite ces valeurs et on divise par le nombre de points total sur le bord de la particule donné par `_nbSurfPoints` qui est un membre privé de la classe `Particles`. Les fonctions `VecGetArray` et `VecRestoreArray` permettent d'accéder aux valeurs contenues dans le vecteur `g` comme si c'était un tableau classique. On effectue alors une réduction globale dans la classe `Problem`, comme on l'a vu précédemment, pour sommer toutes les contributions des processus et ainsi obtenir la vraie vitesse de translation et de rotation.

Au niveau du parallélisme, comme tous les processus ont toutes les particules, on a besoin de ne faire que des réductions globales lorsqu'on calcule des intégrales sur le bord ou dans les particules. On ne fait que des appels à la fonction `MPI_Allreduce` avec une somme (`MPI_SUM`). Ces réductions sont utilisées avant de calculer la distribution simple

couche pour avoir la moyenne de la vitesse et de la vitesse de rotation, avant de calculer la projection pour avoir de nouveau des moyennes mais en volume, et enfin avant de calculer la vitesse de translation et de rotation des particules à la fin du pas de temps.

6.3 Gestion des contacts

Lorsque l'on déplace les particules, il y a de forte chance qu'elles se chevauchent ce qui n'est pas une configuration admissible. Dans le code développé ici, il n'y a pas de gestion de contacts entre les particules, il faut donc prendre des pas de temps très petits pour éviter cela. Les particules ne doivent en effet jamais se toucher au niveau théorique.

Pour éviter de prendre de petits pas de temps, il est possible d'ajouter un algorithme de contact. Une version du code est couplée avec le logiciel *SCoPI* (voir [1,48]) qui est un algorithme de contact développé durant la thèse d'Aline Lefebvre en collaboration avec Sylvain Faure. C'est un code C++ qui déplace les particules à partir d'une vitesse a priori. Le principe est de projeter, à l'aide d'un algorithme d'Uzawa, les vitesses a priori sur un espace de vitesses admissibles, c'est à dire l'espace tel que si toutes les vitesses se trouvent dedans, les particules ne se chevaucheront pas une fois déplacées. En notant les centres des particules $\mathbf{x}_i^n$, on définit la distance signée entre deux particules :

$$D_{ij}(\mathbf{x}^n) = |\mathbf{x}_i^n - \mathbf{x}_j^n| - r_i - r_j \,,$$

où r_i et r_j sont les rayons des deux particules. On note aussi $\mathbf{G}_i j(\mathbf{x}^n)$ le gradient de cette distance qui est donné par :

$$\mathbf{G}_{ij}(\mathbf{x}^n) = (\ldots, 0, -\mathbf{e}_{ij}^n, 0, \ldots, 0, \mathbf{e}_{ij}^n, 0, \ldots) \,,$$

où le vecteur $\mathbf{e}_{ij}^n$ est défini par :

$$\mathbf{e}_{ij}^n = \frac{\mathbf{x}_j^n - \mathbf{x}_i^n}{|\mathbf{x}_j^n - \mathbf{x}_i^n|} \,.$$

Si on note dt le pas de temps, l'espace des vitesses admissibles est le suivant :

$$K(\mathbf{x}^n) = \left\{ \mathbf{V} \in \mathbb{R}^{2N} : D_{ij}(\mathbf{x}^n) + dt\mathbf{G}_{ij}(\mathbf{x}^n) \cdot \mathbf{V} \geq 0 \quad \forall i < j \right\} .$$

On peut montrer que cette contrainte est en fait le linéarisé de la contrainte suivante :

$$D_{ij}(\mathbf{x}^n + dt\mathbf{V}) \geq 0 \quad \forall i < j \,.$$

Ceci traduit qu'au pas de temps suivant, les particules ne doivent pas se chevaucher et implique que si la contrainte de l'espace $K(\mathbf{x}^n)$ est vérifiée, alors les particules ne se chevaucheront pas à l'instant suivant. Pour projeter les vitesses a priori, que l'on notera $\hat{\mathbf{V}}^n$, sur l'espace $K(\mathbf{x}^n)$, *SCoPI* utilise des multiplicateurs de Lagrange et un algorithme d'Uzawa. On note λ_{ij}^n le multiplicateur de Lagrange associé à la contrainte $D_{ij}(\mathbf{x}^n) + dt\mathbf{G}_{ij}(\mathbf{x}^n) \cdot \mathbf{V} \geq 0$ et $\boldsymbol{\lambda}^n$ le vecteur dans $\mathbb{R}_+^{N(N-1)/2}$ des multiplicateurs de Lagrange. Le problème point-selle qui est résolu à chaque itération en temps est le suivant (en dimension 2) :

$$\left\{ \begin{array}{l} \text{Trouver } (\mathbf{V}^n, \boldsymbol{\lambda}^n) \in \mathbb{R}^{2N} \times \mathbb{R}_+^{N(N-1)/2} \text{ tels que} \\ \mathcal{J}(\mathbf{V}^n, \boldsymbol{\Lambda}) \leq \mathcal{J}(\mathbf{V}^n, \boldsymbol{\lambda}^n) \leq \mathcal{J}(\mathbf{V}, \boldsymbol{\lambda}^n), \quad \forall (\mathbf{V}, \boldsymbol{\Lambda}) \in \mathbb{R}^{2N} \times \mathbb{R}_+^{N(N-1)/2} \,, \end{array} \right.$$

où la fonctionnelle $\mathcal{J}$ est définie de la façon suivante :

$$\mathcal{J}(\mathbf{V}, \mathbf{\Lambda}) = \frac{1}{2} \left| \mathbf{V} - \hat{\mathbf{V}}^n \right|^2 - \sum_{1 \leq i < j \leq N} \lambda_{ij} (D_{ij}(\mathbf{x}^n) + dt \mathbf{G}_{ij}(\mathbf{x}^n) \cdot \mathbf{V}).$$

C'est ce problème point-selle qui est résolu avec un algorithme d'Uzawa. S'il n'y a pas de contact entre deux particules i et j, alors $\lambda_{ij} = 0$ et le multiplicateur de Lagrange n'est pas activé.

Un des avantages de ce modèle de contact est qu'il est découplé du problème fluide. Il suffit d'une vitesse a priori pour qu'il fonctionne et peut donc être couplé avec n'importe quel solveur fluide qui retourne une vitesse a priori des particules. Pour réaliser le couplage, on a donc créé une nouvelle classe de vitesse a priori dans le code source de *SCoPI* dont le calcul passe par la résolution des équations de Stokes avec notre méthode. Le principe est donc le suivant :

1. SCoPI génère les particules sphériques.
2. Notre code récupère la position et le rayon des particules et calcule la vitesse du fluide correspondante.
3. Le code calcule la nouvelle vitesse des particules par intégration de la vitesse du fluide et l'envoie à SCoPI.
4. SCoPI déplace les particules en projetant les vitesses renvoyer par notre code sur l'ensemble des vitesses admissibles $K(\mathbf{x}^n)$ (qui permet aux particules de ne pas se chevaucher ou de ne pas entrer dans les obstacles).
5. Le code récupère les nouvelles positions et on recommence.

Avec SCoPI, il est possible de spécifier une distance minimale entre les particules à ne pas dépasser ce qui est très utile si on ne veut pas qu'il y ait moins d'un certain nombre de mailles entre chaques particules.

Troisième partie

Applications

Chapitre 7

Simulations numériques 2D et 3D

Sommaire

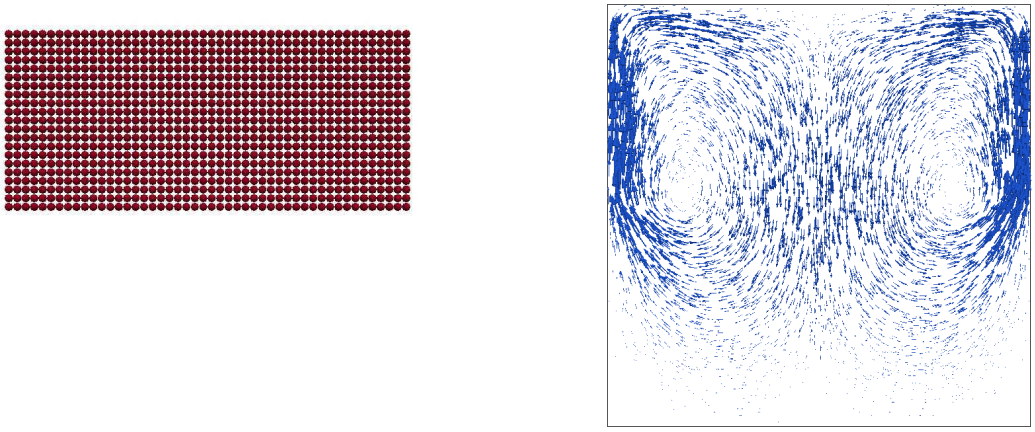

FIGURE 7.1 – Instant initial de la sédimentation de 1008 particules

Dans ce chapître on présente des simulations numériques en deux et trois dimensions réalisées avec le code que l'on a décrit au chapitre 6. En dimension 2 on présente deux simulations différentes. La première consiste en la sédimentation de 1008 particules, et la seconde est l'évolution d'un tas de particules rigides dans un champ cisaillé. Pour cette simulation, on a considéré deux types de modèle de contact différents donnés par *SCoPI* : un modèle de contact inélastique et un modèle de contact visqueux. Dans le second cas, le modèle considère qu'il y a une mince couche de fluide autour des particules, si bien qu'il faut une plus grande force que dans le cas inélastique pour que deux particules en contact se détachent l'une de l'autre. Les particules ont donc tendance à rester groupées. Pour la simulation $3D$, on présente le cas de la sédimentation de particules rigides dans un cube où la face supérieure est animée d'un mouvement de rotation qui entraîne le fluide.

7.1 Simulations 2D

7.1.1 Sédimentation de particules rigides

On considère un ensemble de 1008 particules réparties de façon régulière sur la partie supérieure du carré unité. Les particules ont toutes le même rayon et la même densité. La simulation a été réalisée sur une grille de taille 512×512 en pression et donc une grille de taille 1024×1024 en vitesse. La répartition initiale des particules est représentée sur la figure 7.1 ainsi que le champ de vitesse correspondant.

Les figures 7.2, 7.3 et 7.4 montrent l'évolution de la sédimentation. Les particules au centre chutent les premières ce qui fait remonter du fluide sur les cotés comme on peut le voir sur le champ de vitesse. Les particules sur les cotés commencent donc par remonter et à se diriger vers l'axe de symétrie vertical du carré pour sédimenter ensuite. Les champs de vitesse n'ont pas la même échelle d'une figure à l'autre pour des questions de visualisation car la vitesse augmente fortement entre le début de la simulation et ces trois instants.

Enfin les figures 7.5, 7.6, 7.7 et 7.8 représentent la fin de la sédimentation. Les particules sont toutes dans le bas du carré, exceptées quelques unes qui restent en haut. Ces particules sédimentent beaucoup plus lentement puisqu'elle sont isolées et ne sont donc pas entraînées

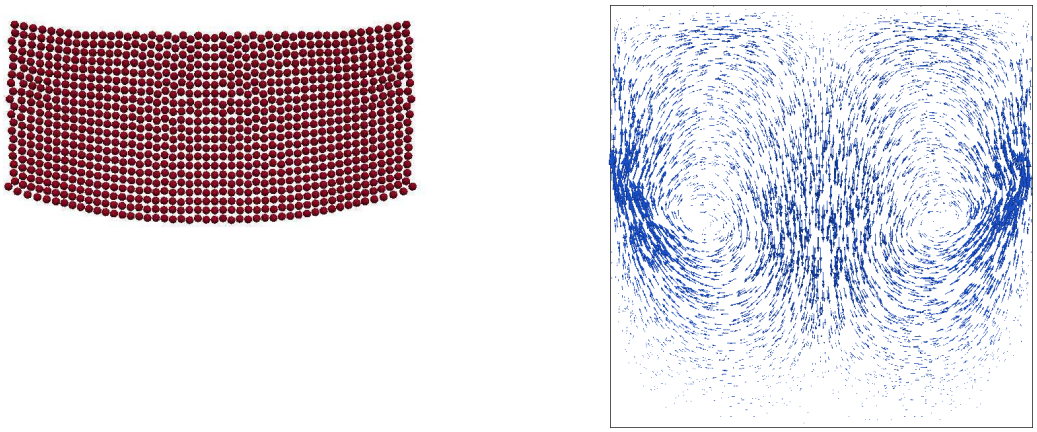

FIGURE 7.2 – Sédimentation après 20 itérations en temps

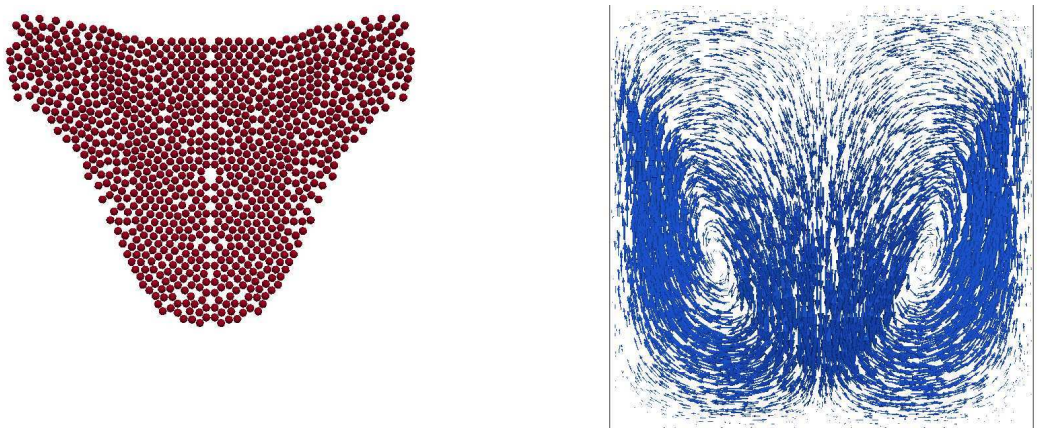

FIGURE 7.3 – Sédimentation après 40 itérations en temps

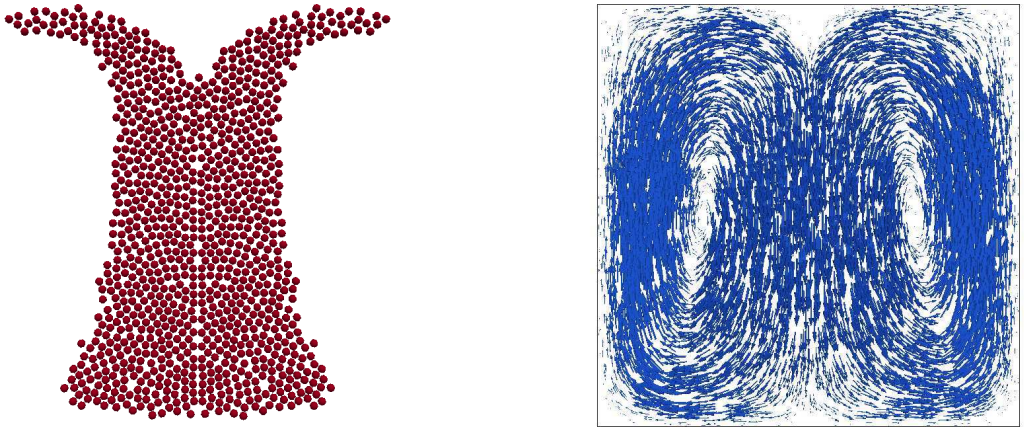

FIGURE 7.4 – Sédimentation après 50 itérations en temps

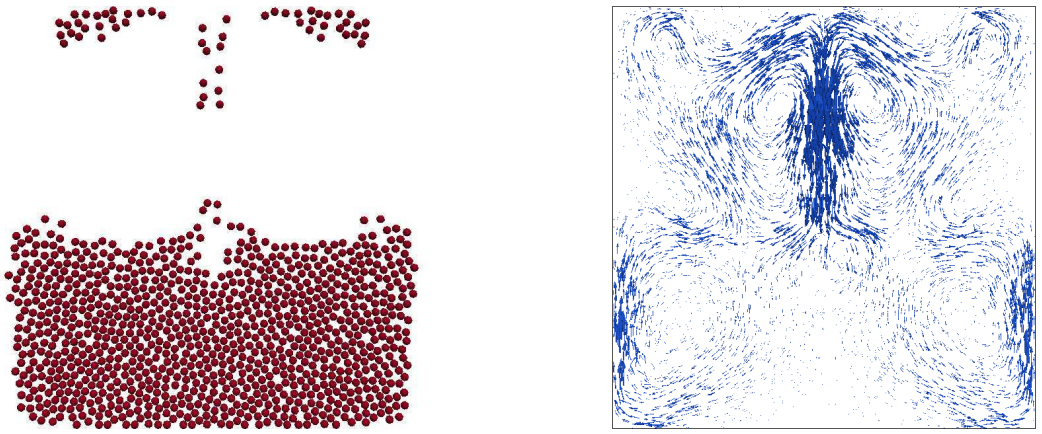

FIGURE 7.5 – Sédimentation après 100 itérations en temps

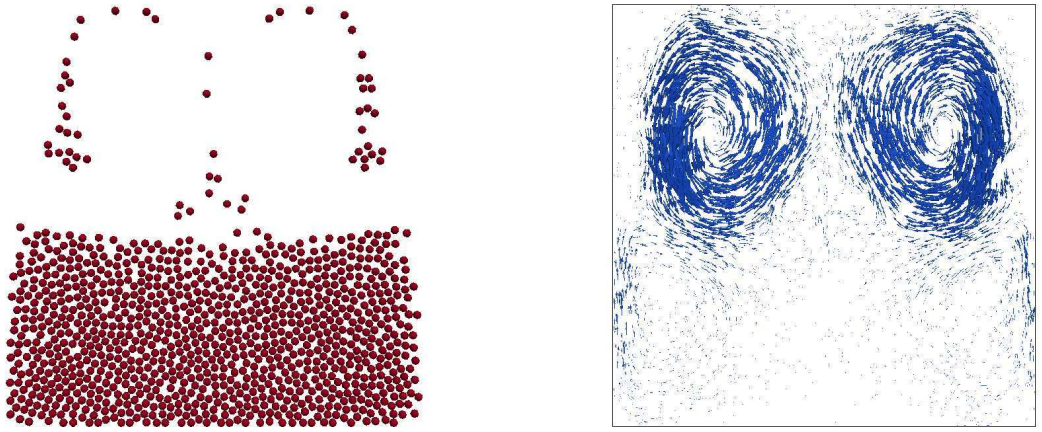

FIGURE 7.6 – Sédimentation après 175 itérations en temps

par la sédimentation des particules proches comme c'était le cas pour les autres. De même, les particules situées en bas du carré mettent beaucoup de temps à finir de sédimenter puisqu'il faut que le fluide remonte pour qu'elles puissent descendre. Le fluide doit donc passer dans un milieu très poreux ce qui prend du temps.

7.1.2 Un tas de sable

On présente maintenant une simulation d'un tas de particules rigides immergées dans un fluide soumis à un cisaillement. Le domaine est le rectangle $[0,1] \times [0,3]$ que l'on maille avec 64 points en pression par unité de longueur, soit 128 points en vitesse. On impose une vitesse horizontale non nulle sur le bord haut du rectangle et une vitesse nulle sur le bas. On prend alors des conditions périodiques sur les bords gauche et droit du rectangle. On obtient donc un cisaillement et on considère un tas de 300 particules. On peut voir ce problème comme la simulation de l'évolution d'un tas de sable immergé dans un fluide très visqueux pour justifier l'utilisation des équations de Stokes. On a besoin des équations de Navier-Stokes pour simuler des cas plus réalistes.

On considère tout d'abord un modèle de contact inélastique fourni par *SCoPI* qui est celui décrit dans le chapître 6. Les résultats sont représentés sur les figures 7.9, 7.10, 7.11,

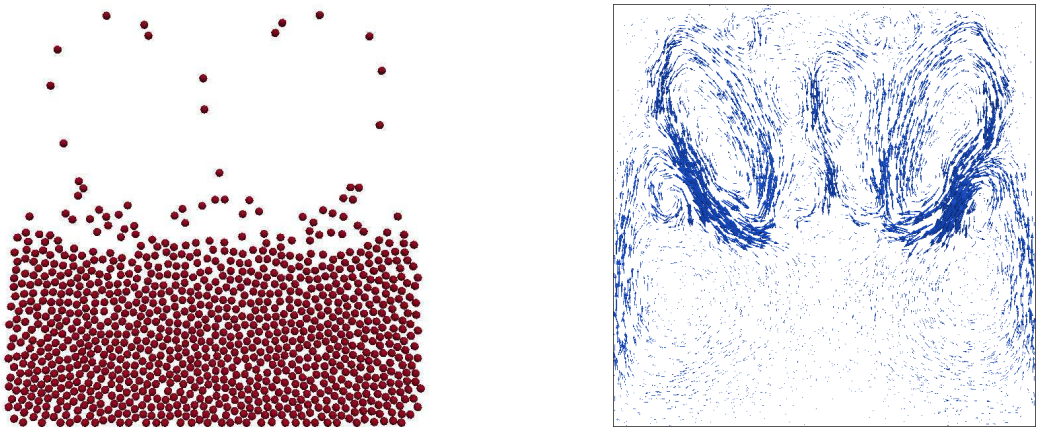

FIGURE 7.7 – Sédimentation après 225 itérations en temps

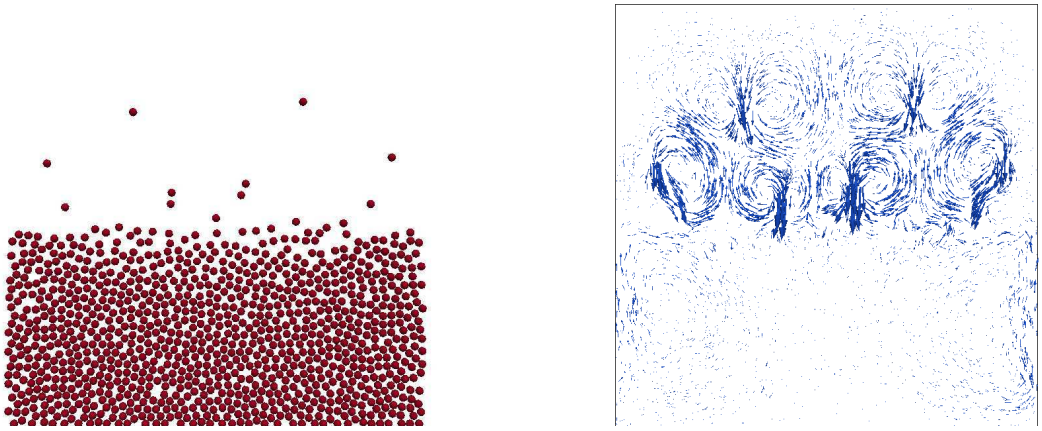

FIGURE 7.8 – Sédimentation après 450 itérations en temps

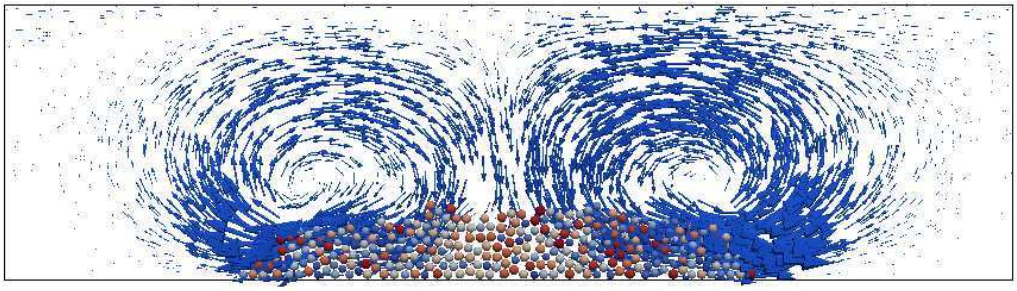

FIGURE 7.9 – Disposition du tas au temps 0 pour des contacts inélastiques

7.12 et 7.13. On peut observer que le tas s'étale en partant des deux cotés et finit par former une bande de particules rigides par périodicité. De même que dans le cas de la sédimentation, les champs de vitesse pour 20 et 50 itérations en temps n'ont pas la même échelle que les autres car les vitesses sont trop faibles pour pouvoir représenter les champs avec la même échelle. On peut néanmoins remarquer que c'est le cisaillement qui domine sur ces deux figures contrairement aux 3 premières.

On considère à présent la même simulation mais en remplaçant le modèle de contact inélastique par un modèle de contact visqueux. Une fine couche de fluide est modélisée autour des particules, il est donc plus difficile pour deux particules en contact de se décrocher. Les particules forment donc des paquets ce qui conduit à des résultats complètement différents. On part de la même configuration que dans le cas inélastique, le champ de vitesse est donc le même au départ. Les figures 7.14, 7.15 et 7.16 représentent la configuration après plusieurs itérations en temps.

Comme les particules se regroupent par paquet, le tas n'est pas aussi lisse que dans le cas inélastique et le champ de vitesse obtenu est donc complètement différent. On peut observer la formation de tourbillons qui font remonter certaines particules comme sur la figure 7.15 et 7.16.

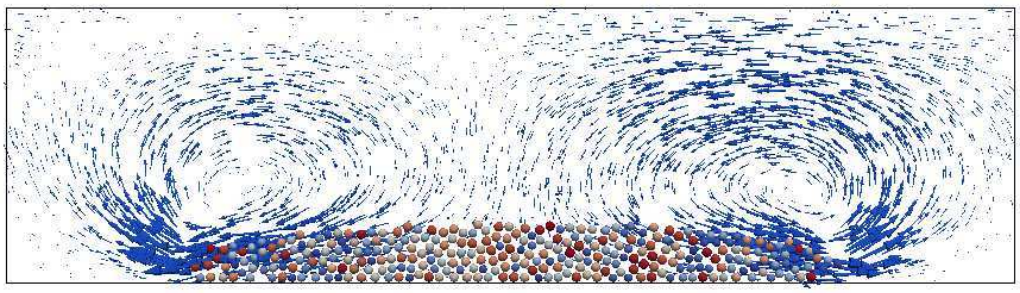

FIGURE 7.10 – Tas de sable après 5 itérations en temps

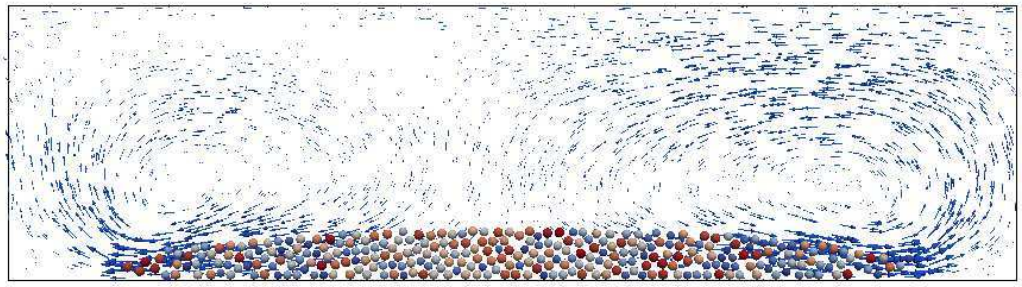

FIGURE 7.11 – Tas de sable après 10 itérations en temps

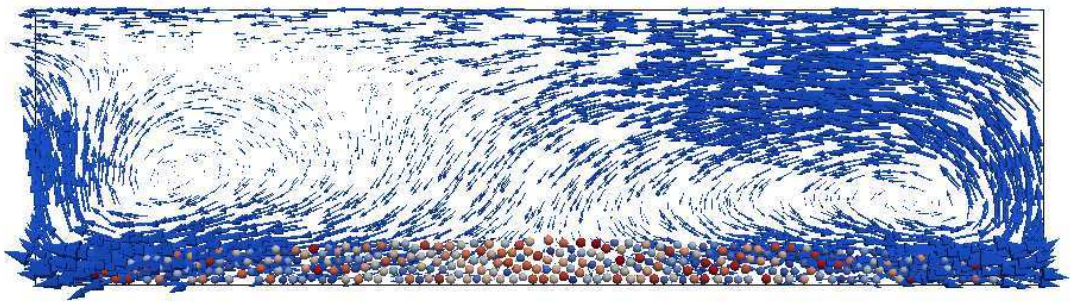

FIGURE 7.12 – Tas de sable après 20 itérations en temps

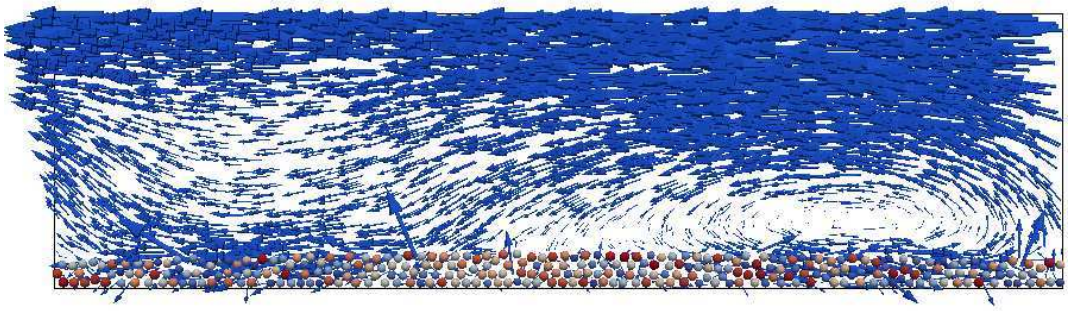

FIGURE 7.13 – Tas de sable après 50 itérations en temps

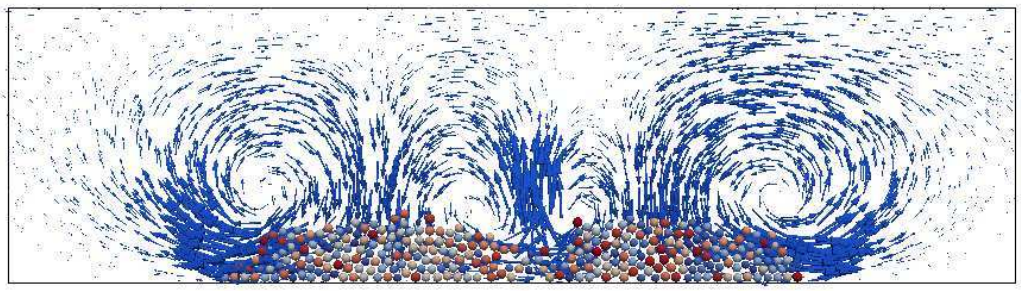

FIGURE 7.14 – Tas de sable avec contacts visqueux après 20 itérations en temps

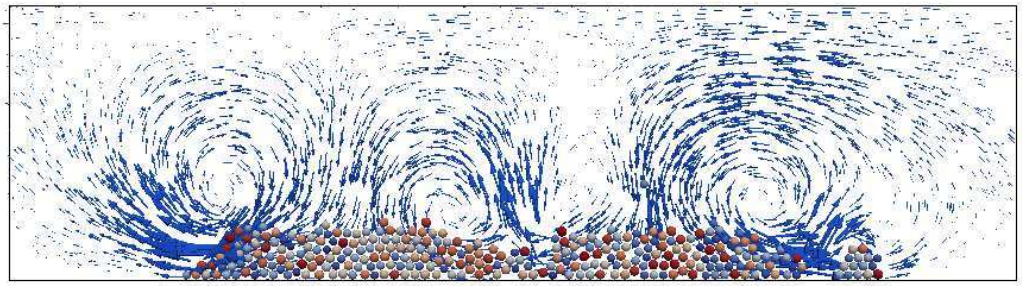

FIGURE 7.15 – Tas de sable avec contacts visqueux après 100 itérations en temps

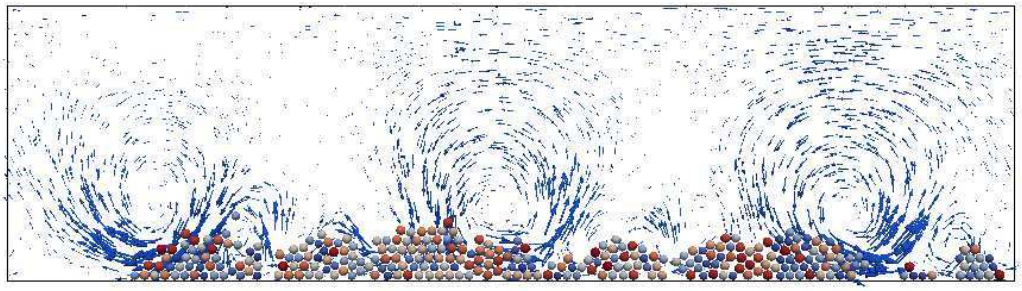

FIGURE 7.16 – Tas de sable avec contacts visqueux après 400 itérations en temps

7.2 Simulations 3D

7.2.1 Sédimentation de particules rigides en 3D

On regarde dans cette section le cas de la sédimentation de particules rigides en trois dimensions. La différence avec le cas en deux dimensions est qu'ici les particules n'ont pas toutes le même rayon et sont disposées aléatoirement dans la partie supérieure du cube unité. Les conditions de bord sont des conditions de Dirichlet homogène partout sauf sur la face supérieure qui est animée d'un mouvement de rotation. On laisse ainsi sédimenter 2000 particules dans le cube unité. La position initiale des particules et le champ de vitesse associé sont représentés sur les figures 7.17 et 7.18.

Au départ, la rotation due aux conditions de bords est dominante, puis les particules commencent à sédimenter et à s'entraîner entre elles. On peut le voir sur le champ de vitesse des figures 7.20 et 7.22. Le fluide remonte au centre et les particules chutent sur les cotés.

Enfin, tout comme dans le cas $2D$, les dernières particules sédimentent plus lentement. Une particule reste même accrochée dans un coin du fait des conditions de bords comme on peut le voir sur les figures 7.23 et 7.24.

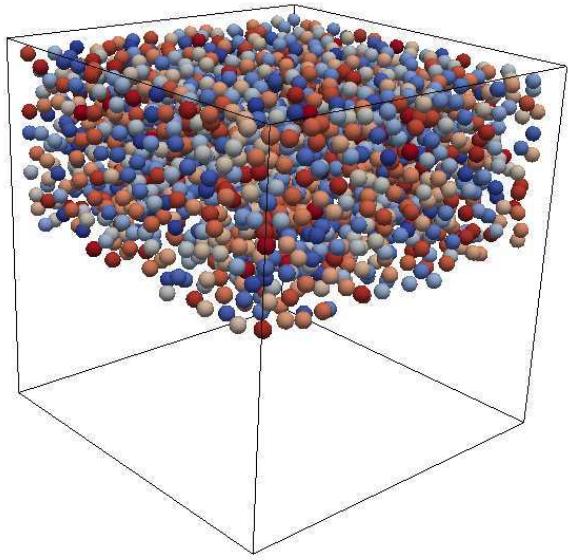

FIGURE 7.17 – Configuration initiale des particules

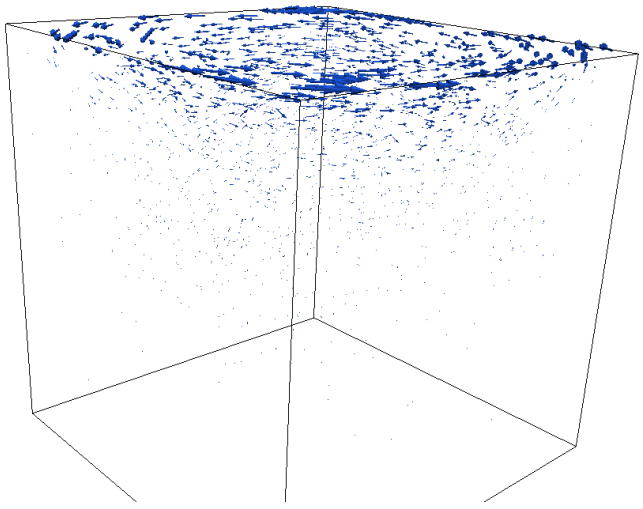

FIGURE 7.18 – Champ de vitesse initial

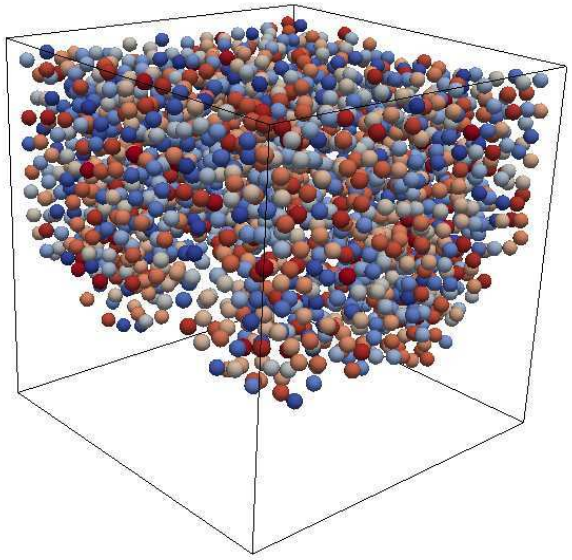

FIGURE 7.19 – Configuration des particules après 100 itérations en temps

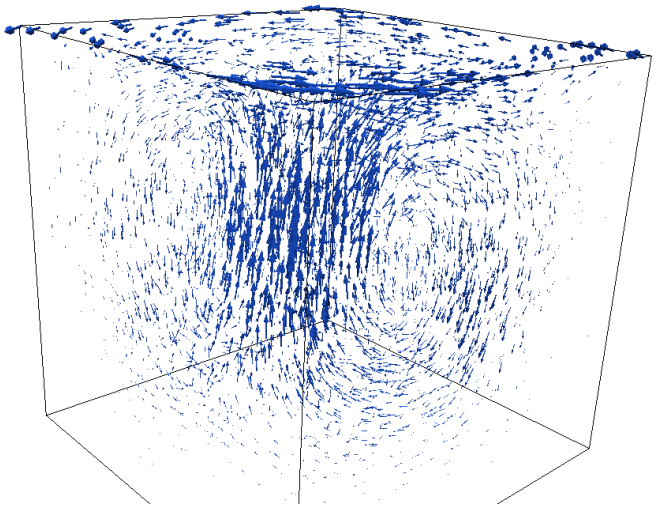

FIGURE 7.20 – Champ de vitesse après 100 itérations en temps

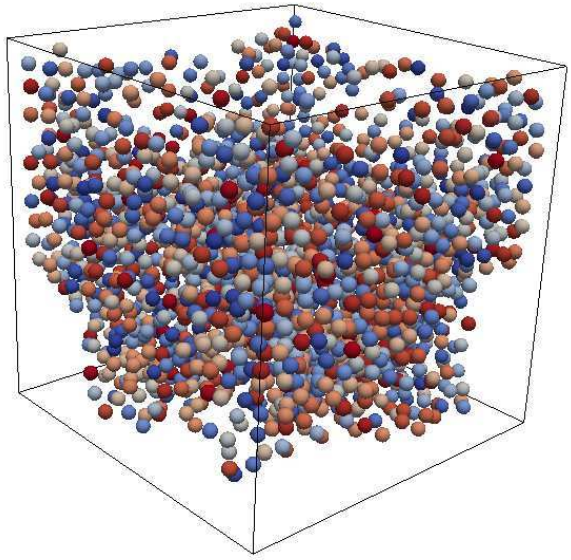

FIGURE 7.21 – Configuration des particules après 200 itérations en temps

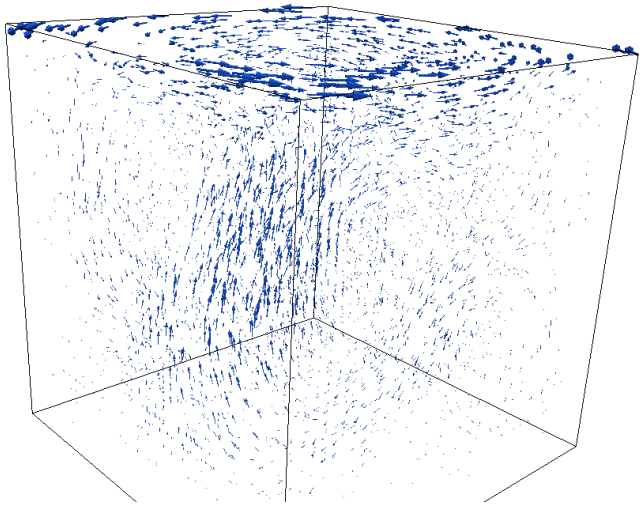

FIGURE 7.22 – Champ de vitesse après 200 itérations en temps

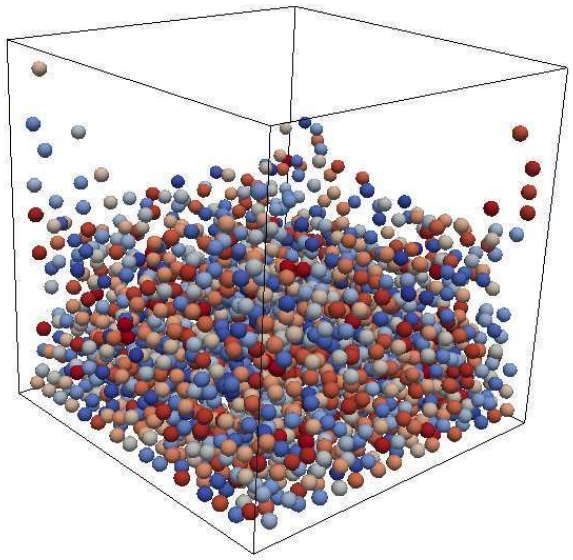

FIGURE 7.23 – Configuration des particules après 600 itérations en temps

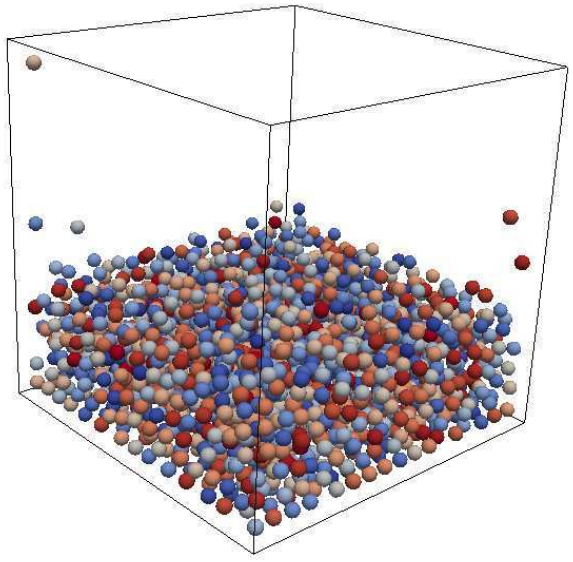

FIGURE 7.24 – Configuration des particules après 1000 itérations en temps

Chapitre 8

Suspensions actives

Sommaire

On présente dans ce chapitre une application à la nage de bactéries. Comme expliqué dans l'introduction, le but est ensuite de considérer des suspensions très denses de bactéries pour observer l'apparition de mouvement collectif dû au fait que les bactéries s'auto propulsent et sont donc actives contrairement aux suspensions de particules rigides simples qui sont passives. On présente dans la section suivante deux modèles de bactéries : les *pushers* et les *pullers*. On a ensuite mis à part le modèle des squirmers car il est légèrement différent de celui des deux autres qui font appel à des forces volumiques.

8.1 Modèles de *pusher* et de *puller*

8.1.1 Les *pushers*

On s'intéresse à des bactéries qui se propulsent à l'aide de flagelles comme par exemple *Escherichia Coli* (E. Coli) ou bien *Bacilius Subtilis* (B. Subtilis). Ce sont des bactéries à la forme plutôt cylindrique dont les dimensions sont de l'ordre du micro mètre. Elles sont donc très petites et à cette échelle le fluide dans lequel elles évoluent est essentiellement visqueux et les effets d'inertie peuvent être négligés. Ces bactéries s'auto propulsent à l'aide de flagelles qui tournent avec un mouvement d'hélices pour faire avancer la bactérie. Ce type de bactérie a deux principaux types de déplacements : un déplacement rectiligne dans une direction et une rotation sur elle même principalement pour changer de direction. Elles sont en effet trop petites pour percevoir un quelconque gradient de concentration d'une substance chemo-attractive. Elles se déplacent donc de façon rectiligne pour évaluer un tel gradient et changent de direction lorsque la concentration est trop faible. Elles passent donc beaucoup plus de temps à se déplacer en ligne droite qu'à changer de direction. Ici, on ne modélisera que le mouvement rectiligne, les bactéries ne changeront donc pas de direction suivant la concentration d'une certaine substance. Il faudrait pour cela ajouter des équations régissant l'évolution de la concentration de cette substance dans le fluide. On se concentre donc uniquement sur la partie fluide et la prise en compte des bactéries se propulsant à l'aide de flagelles.

Tout comme dans [38, 27], on modélise la bactérie par une particule rigide, en la prenant sphérique dans notre cas puisque notre code de calcul ne prend en compte que des particules sphériques. La prise en compte de formes différentes fait partie des perspectives. Le corps de la bactérie est donc modélisé par une sphère rigide et on prend en compte les flagelles en imposant un dipôle de force. On considère une ellipse à l'arrière de la particule sur laquelle on distribue une force et de même sur la particule (voir figure 8.1). De cette façon, la bactérie (la tête sphérique) avance et les flagelles sont pris en compte par la force distribuée dans l'ellipse.

On considère alors plusieurs bactéries, chacune notée B_i comme dans le cas des particules passives. On note B la réunion des bactéries. On note alors $\mathbf{F}_{b,i}$ la force résultante exercée sur la bactérie B_i et $\mathbf{F}_{p,i}$ la force résultante exercée au centre de l'ellipse E_i représentant les flagelles de la bactérie B_i. On a donc les relations $\mathbf{F}_{b,i} + \mathbf{F}_{p,i} = \mathbf{0}$. On obtient

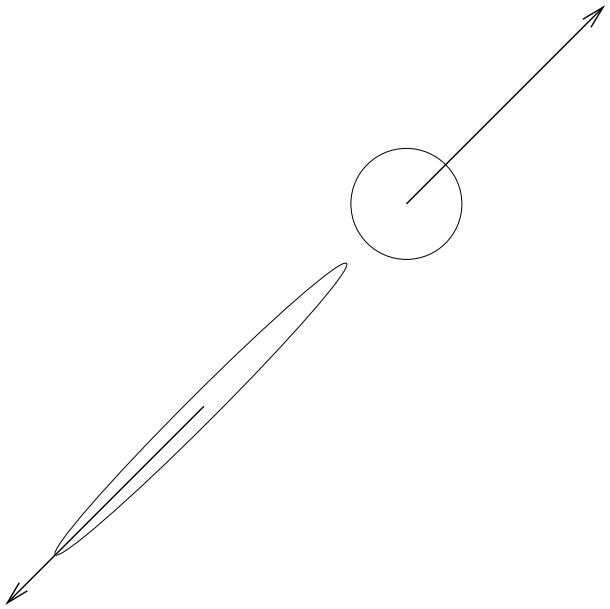

FIGURE 8.1 – Dipôle de force pour le pusher

les forces volumiques suivantes :

$$\mathbf{f}_{b,i} = \frac{1}{|B_i|}\mathbf{F}_{b,i}\,,$$
$$\mathbf{f}_{p,i} = \frac{1}{|E_i|}\mathbf{F}_{p,i}\,.$$

Comme expliqué dans l'introduction, on utilise les équations de Stokes pour modéliser le fluide, les équations à résoudre s'écrivent donc exactement sous la forme du problème initial 1.1 pour les particules passives avec un second membre particulier :

$$(8.1)\quad \begin{cases} -2\eta\,\mathrm{div}(D(\mathbf{u})) + \nabla p &= \rho_f\mathbf{f} + \displaystyle\sum_i \mathbf{f}_{p,i}\chi_{E_i}\,, & \text{dans } \Omega\setminus\bar{B}\,,\\ \nabla\cdot\mathbf{u} &= 0\,, & \text{dans } \Omega\setminus\bar{B}\,,\\ \mathbf{u} &= 0\,, & \text{sur } \partial\Omega\,,\\ \mathbf{u} &= \mathbf{V}_i + \boldsymbol{\omega}_i\wedge\mathbf{r}_i\,, & \text{sur } \gamma_i = \partial B_i\,,\\ \displaystyle\int_{\gamma_i}\sigma(\mathbf{u})\mathbf{n}_i &= \mathbf{F}_{b,i}\,,\\ \displaystyle\int_{\gamma_i}\mathbf{r}_i\wedge\sigma(\mathbf{u})\mathbf{n}_i &= 0\,. \end{cases}$$

Seul le second membre change pour prendre en compte les pushers, la méthode est ensuite la même que pour les particules passives. Il reste toutefois une différence : dans le cas des particules passives, comme on ne considère que des particules sphériques, nous n'avions pas besoin de prendre en compte la rotation de la particule sur elle même. Ici, pour que l'orientation de la bactérie évolue correctement au cours du temps, il faut prendre en

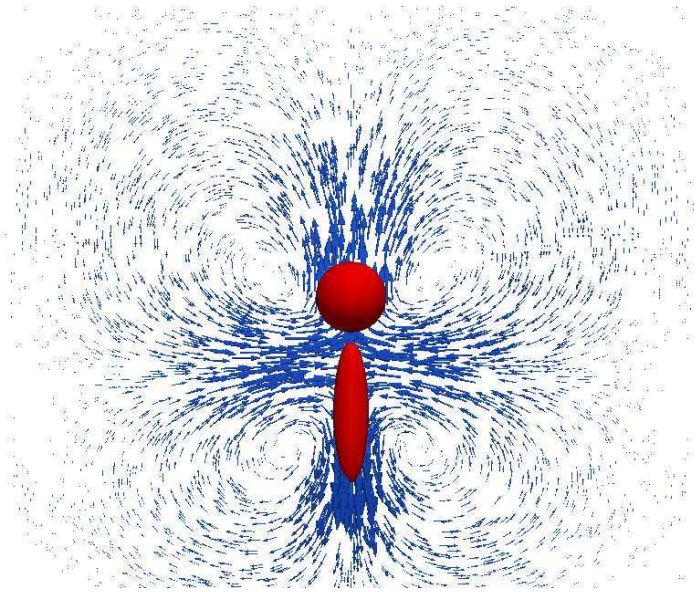

FIGURE 8.2 – Champ de vitesse d'un pusher

compte cette rotation de manière à ajuster la position de l'ellipse représentant les flagelles à chaque pas de temps. Il faut donc conserver l'angle entre la direction dans laquelle se déplace la bactérie et l'axe des abscisse par exemple. On modifie cet angle en calculant à chaque pas de temps la vitesse de rotation de la particule rigide représentant la bactérie.

Remarque 8.1. *Avec ce modèle, il est possible de trouver des ellipses qui intersectent les particules rigides ou bien qui sortent du domaine de calcul. Les ellipses ne sont pas représentées comme des particules, elles ne sont donc pas visibles puisqu'elles servent seulement de support pour la force qui apparaît dans le second membre. Pour notre méthode cela ne change rien, la force dans la particule rigide sera seulement différente mais le contrôle cherché fera toujours en sorte qu'il y ait bien un mouvement rigide sur le bord de la particule.*

On représente sur les figures 8.2 et 8.3 le champ de vitesse et les lignes de courant associées dans le cas d'un seul pusher dont la tête est centrée au centre du carré unité. Si on note θ l'angle de la bactérie par rapport à l'axe des abscisses, le centre de l'ellipse représentant les flagelles se trouve au point $(x_b - 3.25R_b cos(\theta), y_b - 3.25R_b sin(\theta))$, où x_b et y_b sont les coordonnées du centre de la particule rigide représentant la tête de la bactérie et R_b son rayon. Sur les figures, on a pris un angle de $\pi/2$ ce qui donne l'ellipse que l'on observe. Les rayons de l'ellipse sont de $2R_b$ pour le grand axe et $R_b/2$ pour le petit axe. Comme l'angle est de $\pi/2$ le grand axe est vertical. Le champ obtenu est en accord avec celui obtenu dans [27].

Sur les figures 8.4 et 8.5 on représente le champ de vitesse ainsi que les lignes de courant lorsque la bactérie se trouve près d'un mur. Cette fois encore, les résultats obtenus sont en accord avec ceux obtenus dans [27]. On teste aussi le cas de deux pushers proches, nageant dans la même direction. Le champ de vitesse a tendance à les ramener l'un vers l'autre tout en tournant sur eux mêmes dans un sens opposé. Lorsque l'angle est suffisamment

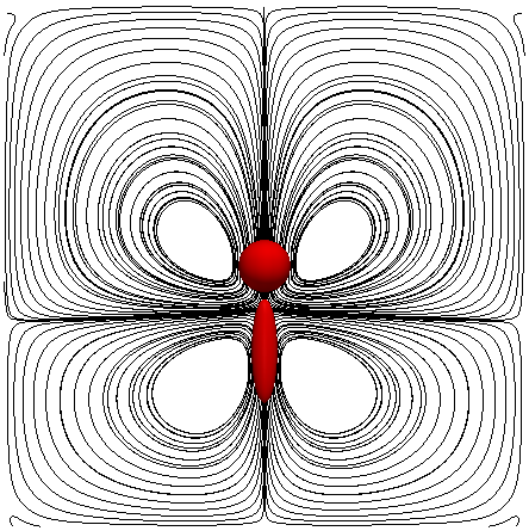

FIGURE 8.3 – Lignes de courant d'un pusher

grand, les deux pushers s'éloignent alors l'un de l'autre. C'est bien ce que l'on observe sur nos résultats aussi comme le montrent les figures 8.6 et 8.7 pour le champ de vitesse et les lignes de courants à l'instant de départ et 8.8 pour l'évolution de leurs centres en fonction du temps.

8.1.2 Les *pullers*

Le cas des pullers est un tout petit peu différent du cas des pushers. En effet, au lieu de se propulser à l'aide de flagelles, ces bactéries exercent un mouvement de traction au moyen de cils. On modélise les cils par deux ellipses situées devant la particule rigide qui représente la bactérie (voir figure 8.9). Les équations à résoudre sont donc les mêmes que pour le pusher excepté qu'il y a deux ellipses par sphère. La force s'exerçant dans les particules rigides est donc deux fois plus grande que celle s'exerçant dans chaque ellipse pour avoir un bilan des forces nul.

Comme pour les pushers, on a tracé le champ de vitesse obtenu dans le cas d'un seul puller centré au milieu du carré unité sur la figure 8.10. La particule rigide a la même taille que dans le cas d'un puller, de même pour les ellipses qui sont les mêmes. On trace aussi les lignes de courants associées sur la figure 8.11.

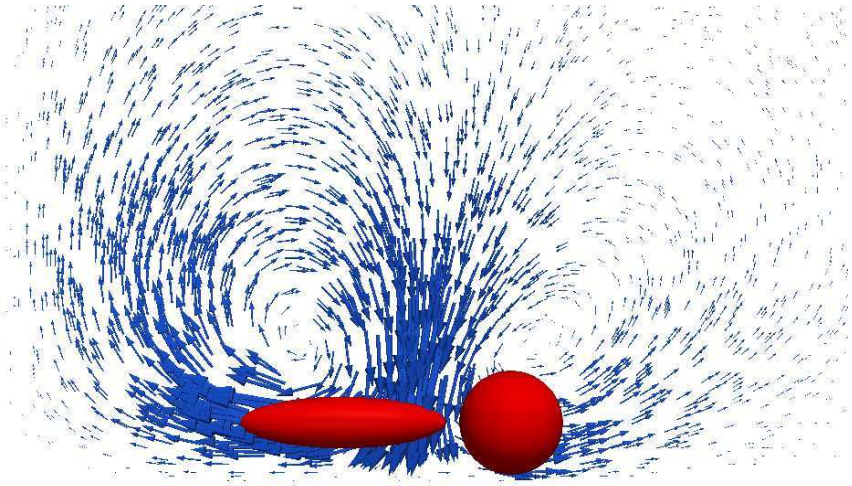

FIGURE 8.4 – Champ de vitesse d'un pusher près d'un bord.

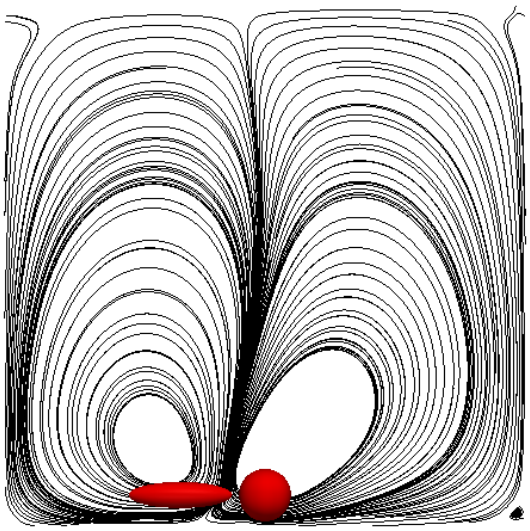

FIGURE 8.5 – Lignes de courant d'un pusher près d'un bord.

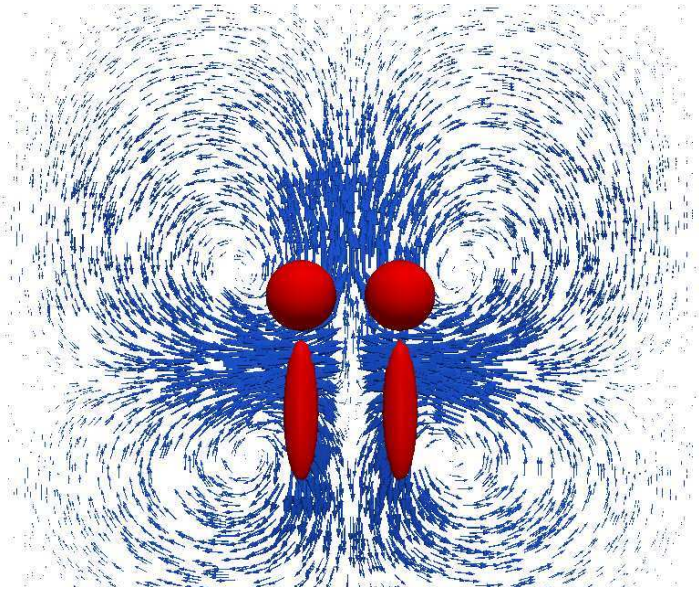

FIGURE 8.6 – Champ de vitesse de deux pushers proches

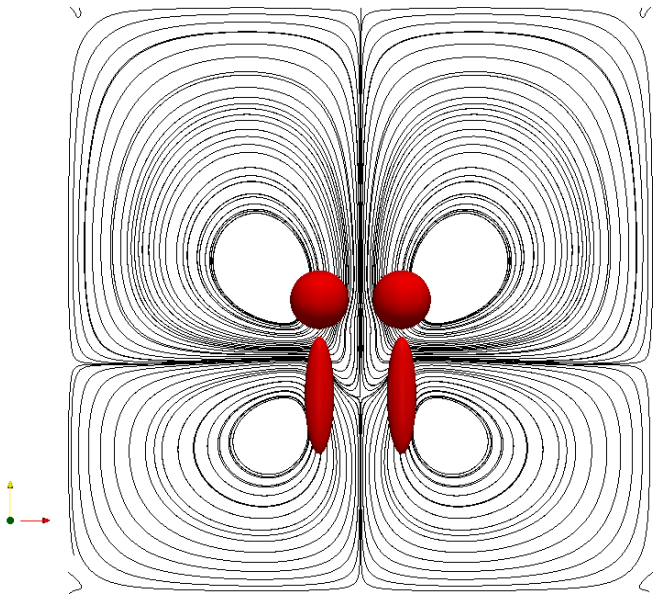

FIGURE 8.7 – Lignes de courant de deux pushers proches

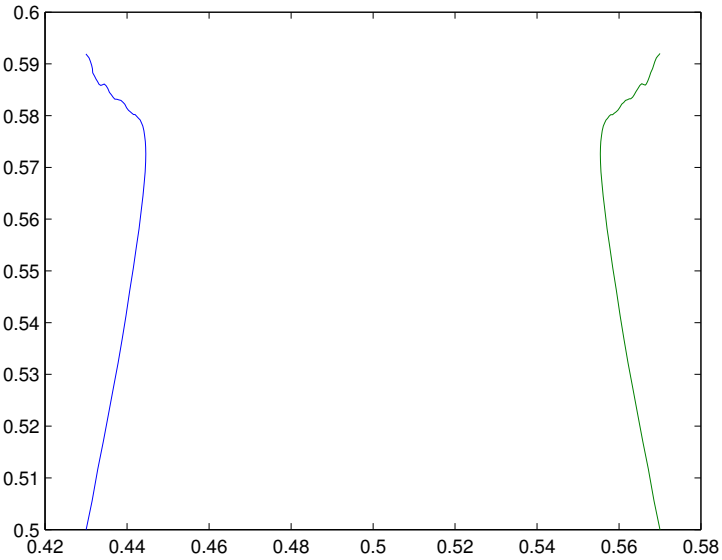

FIGURE 8.8 – Évolution des centres de deux pushers proches

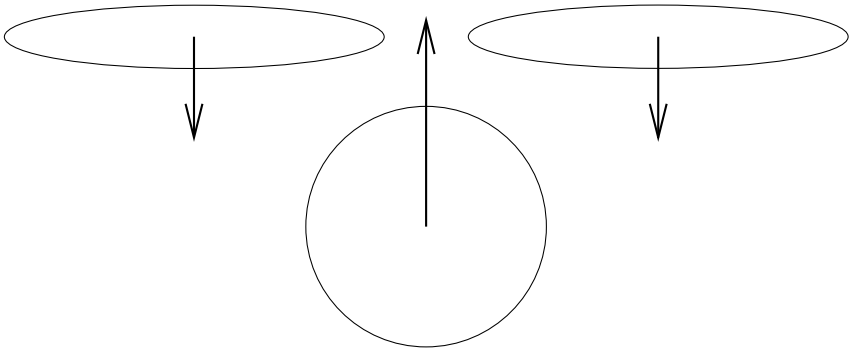

FIGURE 8.9 – Les forces pour un puller

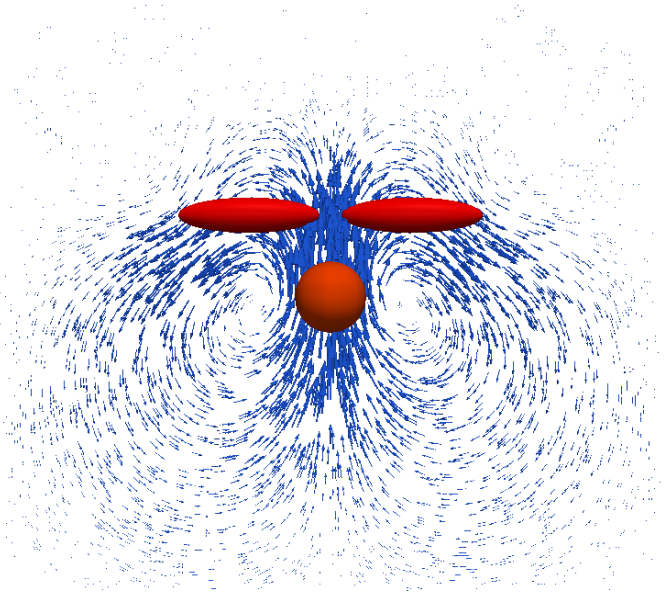

FIGURE 8.10 – Champ de vitesse d'un puller

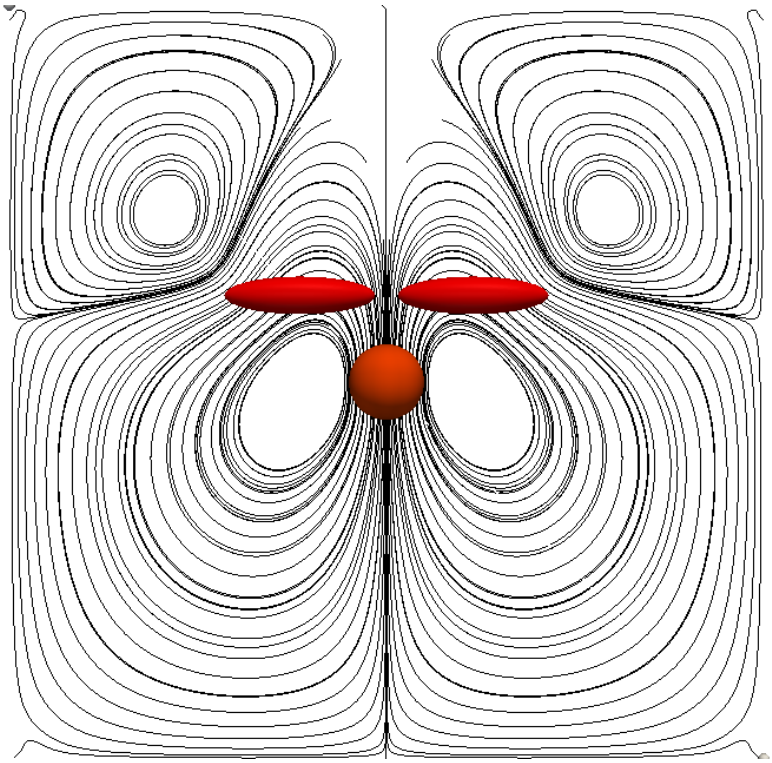

FIGURE 8.11 – Lignes de courant d'un puller

8.1.3 Implémentation dans le code

Pour prendre en compte le type de particules utilisées, on a ajouté une option dans le fichier de configuration permettant de choisir si les particules rigides représentent des particules passives ou bien des pullers ou bien des pushers. On a donc accès au type de particule via un objet de la classe `Config` et on peut donc tester, lors de la construction de l'objet de la classe `Problem`, s'il faut ajouter ou non des flagelles. On a donc créé une classe `Flagelle` qui contient le centre et les rayons de l'ellipse, ainsi qu'un tableau contenant les indices des points de la grille se trouvant dans l'ellipse. Ce tableau est créé de la même façon que celui de la classe `Particles` contenant les indices des points de la grille dans la particule. On commence par prendre une petite boîte contenant l'ellipse afin de ne pas parcourir tous les points de la grille à chaque fois. Ensuite, on parcourt les points de cette boîte et on regarde à chaque fois s'ils se trouvent dans l'ellipse ou non. On ajoute alors une variable nommée `hasFlagelle` qui prend la valeur 1 si le tableau n'est pas vide et 0 sinon. C'est de cette façon que l'on sait si le processus contient une partie de l'ellipse ou non.

Dans la classe `Problem` on ajoute alors une fonction `addFlagelles` qui est appelée dans le constructeur de la classe `Problem` juste après l'appel à la fonction `addParticles` qui construit le tableau de particules. Le constructeur de la classe `Problem` est donc le suivant :

```
Problem::Problem(Config *config, Domain *domain, int dim, Vec solution,
        int nb_particle, int nb_obstacle, double **centers, double *r,
        int *n, double **forces, double **angles, double *rho_solide){

  _nb_parts = nb_particle;
  _nb_obs = nb_obstacle;
  _conf = config;
  _dom = domain;

  if (nb_particle != 0 || nb_obstacle != 0){
    addParticles(centers, r, n, forces, rho_solide);
    if (nb_particle !=0 && _conf->getPartType() > 0){
      addFlagelles(centers, r, angles);
    }
  }

  int idim = _dom->get_dim() == 2 ? 1 : 3;

  _velocities.resize(_dom->get_dim()*_particles.size());
  _angularvel.resize(idim*_particles.size());
  _sol = solution;
}
```

Si on a des particules ou des obstacles (des particules avec une vitesse nulle sur le bord au lieu d'un mouvement rigide), on appelle la fonction `addParticles` qui va allouer et initialiser le tableau `_particles` contenant toutes les particules. Ensuite, si on a des particules rigides et qu'elles ne sont pas passives, alors on ajoute les flagelles en appelant la fonction `addFlagelles`. Les tableaux contenant toutes les vitesses de translation et de rotation des particules, qui seront utlisés pour les déplacer, sont alors alloués à la bonne

taille. La fonction `addFlagelles` se présente sous la forme suivante (on ne met que la partie pusher qui est la plus courte) :

```
void Problem::addFlagelles(double **centers, double *r, double **angles){
  // Add Flagella if needed
  double c[3];
  double radius[3];

  double dP, R;

  if(_conf->getPartType() == 1){
    if(_dom->get_dim() == 2){
      for(int i=0; i<_nb_parts; i++){
        dP = 3.25*r[i];

        c[0] = centers[i][0] - dP*cos(angles[i][0]);
        c[1] = centers[i][1] - dP*sin(angles[i][0]);
        c[2] = 0;

        radius[0] = 2*r[i]; radius[1] = 0.5*r[i]; radius[2] = 0.0;
        _flagelles.push_back(new Flagelles(_dom,_conf,c,radius,angles[i]));
        _flagelles[i]->initFlagelles(_dom, _conf);
      }
    }
    else{
      // Partie 3D
    }
  }

  // Partie pullers
}
```

Le tableau `_flagelles` est un tableau de type `vector<Flagelles*>` que l'on remplit, avec la fonction `push_back`, au fur et à mesure que l'on parcourt les particules rigides. Ce tableau est ensuite utilisé dans la fonction `init_rhs` qui initialise le second membre des équations de Stokes que l'on résout. Pour ajouter la force dans le cas d'un pusher par exemple, on passe par le code suivant :

```
for(int i=0; i<_flagelles.size(); i++){
  if(_flagelles[i]->hasFlagelle){
    forceAmp = _flagelles[i]->get_force();
    angle = _flagelles[i]->get_angle();
    radFlag = _flagelles[i]->get_radius();

    forceAmp /= PI*radFlag[0]*radFlag[1];
    cosTheta = cos(angle[0]);
    sinTheta = sin(angle[0]);

    for(int j=0; j<_flagelles[i]->indicvelloc.size(); j++){
      ind = _flagelles[i]->indicvelloc[j];

      slotu = infou.dof*(ind.i - infou.gxs + infou.gxm*(ind.j - infou.gys));
      pb0u[slotu] += -forceAmp*cosTheta;
      pb0u[slotu + 1] += -forceAmp*sinTheta;
```

```
    }
  }
  if(_particles[i]->hasPart){
    // Ajout de la force opposee dans la particule
  }
}
```

Si le processus contient une partie du flagelle, on calcule tout d'abord la force volumique. On parcourt ensuite tous les points de l'ellipse contenus dans le domaine du processus avec la boucle `for` sur l'indice `j` allant de 0 à `_flagelles[i]->indicvelloc.size()`. On récupère l'indice du point et on ajoute la contribution correspondante dans le second membre. On fait la même chose ensuite sur la particule rigide.

8.2 Un modèle de *squirmer*

8.2.1 Présentation du modèle

On considère ici un modèle de micro nageurs différent des pusher et des pullers, qui s'auto propulsent en générant une vitesse tangentielle sur leur surface sans changer de forme. Ce modèle, appelé *squirmer*, à été introduit par [53] et [17] pour modéliser des micro-organismes sphériques comme les *Opalines*. Ces micro-organismes se déplacent en utilisant des cils dont la taille est très petite devant leur diamètre et recouvrent toute leur surface. Ce modèle a récemment été utilisé pour des colonies comme l'algue verte *Volvox* (voir par exemple [45] et [28]). On considère un squirmer simplifié où le champ de vitesse sur le bord est indépendant du temps dans le référentiel du squirmer et est connu. Les squirmer ne sont de plus soumis à aucunes force ou moment extérieur. Ce modèle est particulièrement bien adapté pour décrire le mouvement de squirmer développé artificiellement consistant en des gouttes de liquide se déplaçant sout l'effet Marangoni. Les gouttes sont plongées dans une solution contenant un surfactant et réagissant sur le bord de la goutte avec un élément chimique contenu dans la goutte. La concentration de surfactant au bord de la goutte est alors modifiée ce qui crée un gradient de tension de surface qui entraîne la goutte.

Le modèle consiste donc à imposer le champ de vitesse dans le référentiel du squirmer sur son bord et à considérer que c'est une particule rigide dans le référentiel de l'expérience. La condition que l'on veut imposer sur le bord des particules n'est donc plus une condition de mouvement rigide mais on cherche à imposer la vitesse du fluide comme la somme d'une vitesse connue (celle dans le référentiel de la goutte) et d'un mouvement rigide. Les

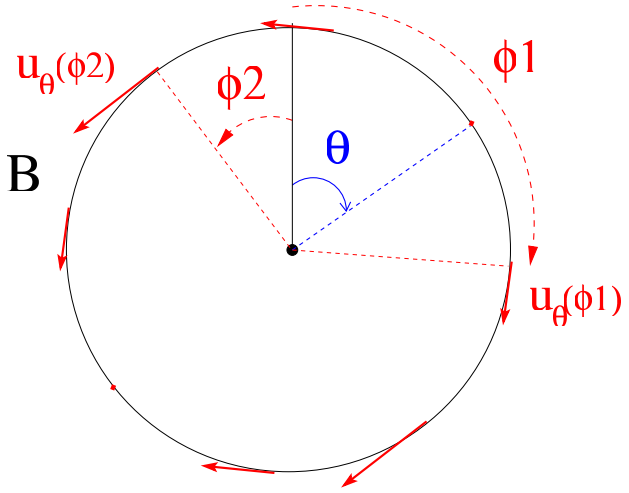

FIGURE 8.12 – Définition des angles pour le squirmer

équations sont donc les suivantes :

$$(8.2)\quad \begin{cases} -2\eta\,\mathrm{div}(D(\mathbf{u})) + \nabla p &= 0\,, & \text{dans } \Omega\setminus\bar{B}\,,\\ \nabla\cdot\mathbf{u} &= 0\,, & \text{dans } \Omega\setminus\bar{B}\,,\\ \mathbf{u} &= 0\,, & \text{sur } \partial\Omega\,,\\ \mathbf{u} &= \mathbf{u}_{\theta_i} + \mathbf{V}_i + \boldsymbol{\omega}_i\wedge\mathbf{r}_i\,, & \text{sur } \gamma_i = \partial B_i\,,\\ \displaystyle\int_{\gamma_i}\sigma(\mathbf{u})\mathbf{n}_i &= 0\,,\\ \displaystyle\int_{\gamma_i}\mathbf{r}_i\wedge\sigma(\mathbf{u})\mathbf{n}_i &= 0\,. \end{cases}$$

Les fonctions $\mathbf{u}_{\theta_i}$ sont définies de la façon suivante :

$$\mathbf{u}_{\theta_i} = \alpha\sin(\phi-\theta_i)\mathbf{e}_\phi\,.$$

Le coefficient α est un réel qui fixe la vitesse du squirmer dans son référentiel. La variable ϕ est la coordonnée polaire et le vecteur $\mathbf{e}_\phi$ est le vecteur tangent au bord du squirmer en polaire (voir figure 8.12). L'angle θ_i est l'orientation du squirmer dans l'espace en dimension 2.

La vitesse utilisée pour déplacer chaque squirmer à la fin du pas de temps est donnée par $\mathbf{V}_i$ et la vitesse de rotation est donnée par $\boldsymbol{\omega}_i$. En milieu infini, un squirmer seul se déplace donc en ligne droite avec notre modèle. Ce sont les intéractions avec les bords du domaine et avec les autres squirmers proches qui lui font changer de direction.

8.2.2 Prise en compte de la nouvelle contrainte

Comme on ne veut plus imposer un mouvement rigide sur le bord des particules, il faut modifier la fonctionnelle à minimiser. On avait avant la fonctionnelle suivante :

$$J(\mathbf{g}) = \sum_{i=1}^{N}\frac{1}{2}\int_{\partial B_i}\left|\mathbf{u}_g - \frac{1}{|\partial B_i|}\int_{\partial B_i}\mathbf{u}_g - \frac{C_d}{{R_i}^2|\partial B_i|}\left(\int_{\partial B_i}\mathbf{r}_i\wedge\mathbf{u}_g\right)\wedge\mathbf{r}_i\right|^2,$$

on voudrait maintenant que ce soit la vitesse $\mathbf{u} - \mathbf{u}_{\theta_i}$ qui soit un mouvement rigide sur le bord de la particule i. On considère donc la fonctionnelle légèrement différente :

$$J(\mathbf{g}) = \sum_{i=1}^{N} \frac{1}{2} \int_{\partial B_i} \left| (\mathbf{u}_g - \mathbf{u}_{\theta_i}) - \frac{1}{|\partial B_i|} \int_{\partial B_i} (\mathbf{u}_g - \mathbf{u}_{\theta_i}) - \frac{C_d}{{R_i}^2 |\partial B_i|} \left(\int_{\partial B_i} \mathbf{r}_i \wedge (\mathbf{u}_g - \mathbf{u}_{\theta_i}) \right) \wedge \mathbf{r}_i \right|^2 .$$

Cette nouvelle fonctionnelle est nulle lorsque chaque $\mathbf{u}_g - \mathbf{u}_{\theta_i}$ est un mouvement rigide sur le bord ∂B_i. Ici, $\mathbf{u}_g$ est la solution du problème de Stokes suivant :

$$\left\{ \begin{array}{rcll} -2\eta \operatorname{div}(D(\mathbf{u}_g)) + \nabla p_g & = & \mathbf{g}\, \chi_B \,, & \text{dans } \Omega \,, \\ \nabla \cdot \mathbf{u}_g & = & 0 \,, & \text{dans } \Omega \,, \\ \mathbf{u}_g & = & 0 \,, & \text{sur } \partial\Omega \,. \end{array} \right.$$

Le gradient de la fonctionnelle est différent. On peut montrer en reprenant les mêmes démonstrations que dans le cas de particules passives, le gradient est la projection de la solution du problème 2.18, en prenant le champ $\mathbf{v}$ égal à $\mathbf{u}_g - \mathbf{u}_{\theta_i}$ sur ∂B_i, sur l'espace K_B :

$$\left\{ \begin{array}{rcll} -2\eta \operatorname{div}(D(\mathbf{w}_g)) + \nabla \boldsymbol{\pi}_g & = & \displaystyle\sum_{i=1}^{N} \left((\mathbf{u}_g - \mathbf{u}_{\theta_i}) - \mathcal{R}_i((\mathbf{u}_g - \mathbf{u}_{\theta_i})) \right) \delta_{\partial B_i} \,, & \text{dans } \Omega \,, \\ \nabla \cdot \mathbf{w}_g & = & 0 \,, & \text{dans } \Omega \,, \\ \mathbf{w}_g & = & 0 \,, & \text{sur } \partial\Omega \,. \end{array} \right.$$

L'opérateur A de la nouvelle fonctionnelle n'a par contre pas changé par rapport au cas de particules passives. Si on utilise une méthode de Krylov type gradient conjugué ou gmres pour minimiser la fonctionnelle, seule l'initialisation de la méthode demande à prendre une distribution simple couche différente du cas de particules passives puisque dans ces méthodes, on a besoin que du produit $A\mathbf{g}$ et non du gradient complet. Dans le code, on ajoute donc un test lors du calcul de la distribution simple couche pour savoir si on utilise des squirmers et si on se trouve dans la phase d'initialisation de la méthode de Krylov pour minimiser la fonctionnelle. Si l'on se trouve en effet dans ce cas, on enlève juste la partie $\mathbf{u}_{\theta_i}$, qui est connue, sur le bord de la particule en question, avant de continuer comme dans le cas de particules passives. On cherche en effet à imposer un mouvement rigide pour $\mathbf{u}_g - \mathbf{u}_{\theta_i}$ pour toutes les particules. La méthode de prolongement régulier nous permet donc de prendre en compte, sans grand changement, le cas des squirmers.

On regarde, comme dans le cas des pushers et des pullers, différentes configurations. On commence par regarder le champ de vitesse obtenu dans le cas d'un squirmer seul situé au milieu du carré unité. Encore une fois, la taille de la particule représentant le squirmer est la même que celle utilisée pour les pushers et les pullers. Les figures 8.13 et 8.14 représentent respectivement le champ de vitesse et les lignes de courant. On remarque que comparé aux pullers ou aux pushers, les squirmers perturbent beaucoup moins le champ de vitesse fluide. On peut donc s'attendre à ce qu'il y ait moins d'intéractions entre les squirmers qu'entre les pushers ou les pullers car ils ont un effet très local.

Comme pour les pushers, on s'intéresse au champ de vitesse d'un squirmer proche d'un bord du domaine. Le champ obtenu montre que le squirmer s'éloigne du bord. Le

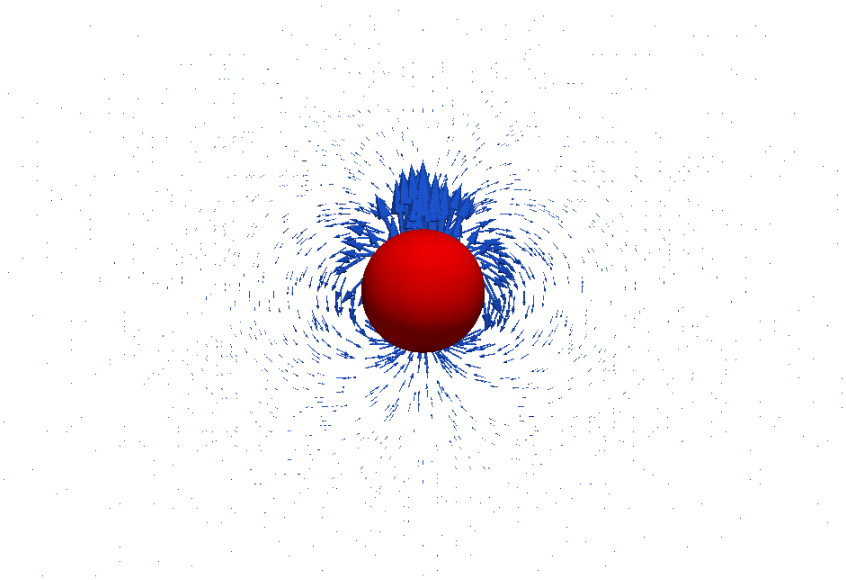

FIGURE 8.13 – Champ de vitesse d'un squirmer seul

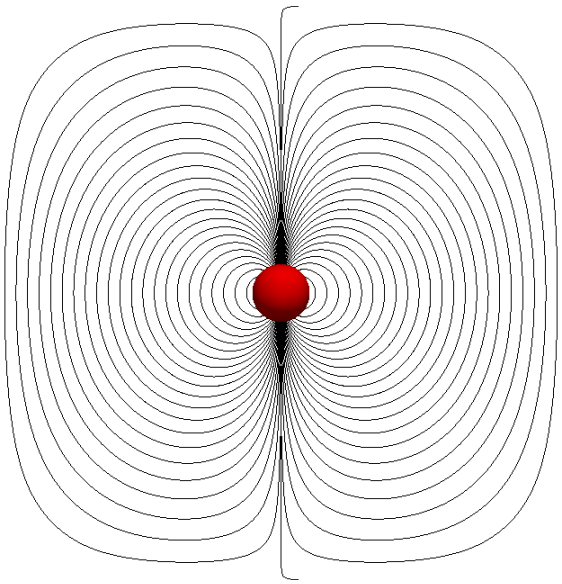

FIGURE 8.14 – Lignes de niveaux d'un squirmer seul

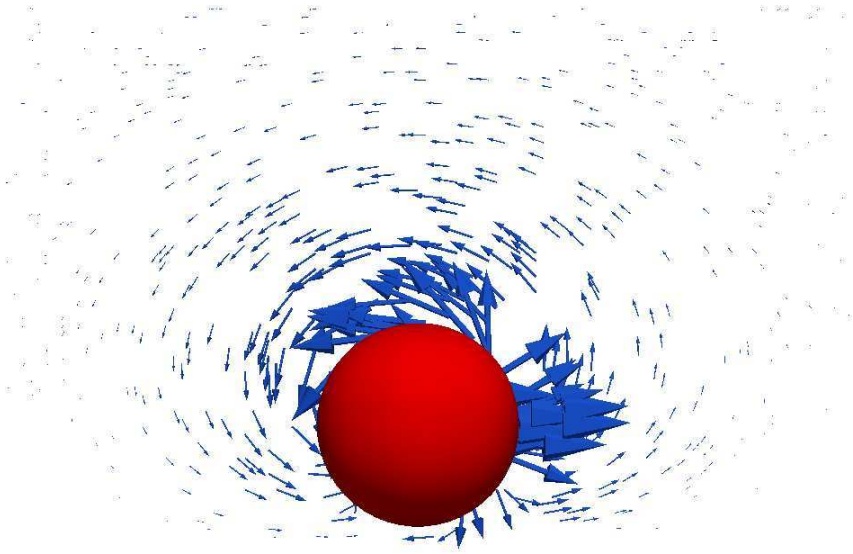

FIGURE 8.15 – Champ de vitesse d'un squirmer proche d'un bord du domaine

champ de vitesse et les lignes de courant associées sont représentées sur les figures 8.15 et 8.16. On a de plus tracé sur la figure 8.17 la trajectoire de ce squirmer pour constater qu'il s'éloigne en effet du bord.

On représente sur les figures 8.18 et 8.19 le champ de vitesse et les lignes de courant dans le cas de deux squirmers proches l'un de l'autre. Le centre et le rayon des particules est le même que dans le cas des pushers. La différence avec les pushers est que le champ de vitesse fluide n'est pas modifié de la même manière et les squirmers s'éloignent directement l'un de l'autre comme on peut le voir sur la figure 8.20 qui représente la trajectoire des deux squirmers. Ils ne commencent pas par se rapprocher comme les pushers le font mais avancent et tournent sur eux mêmes avec une vitesse angulaire opposée.

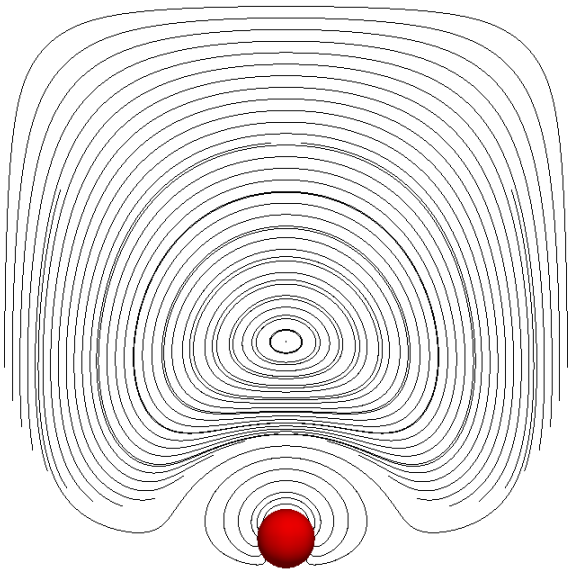

FIGURE 8.16 – Lignes de niveaux d'un squirmer proche d'un bord du domaine

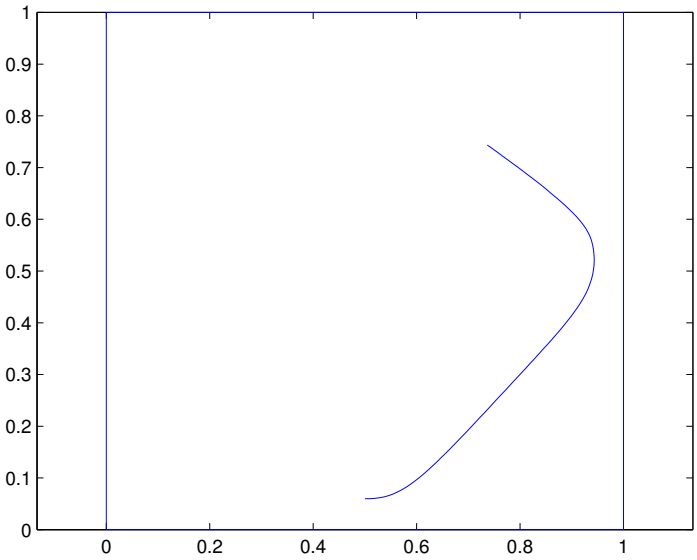

FIGURE 8.17 – Trajectoire d'un squirmer partant près d'un bord

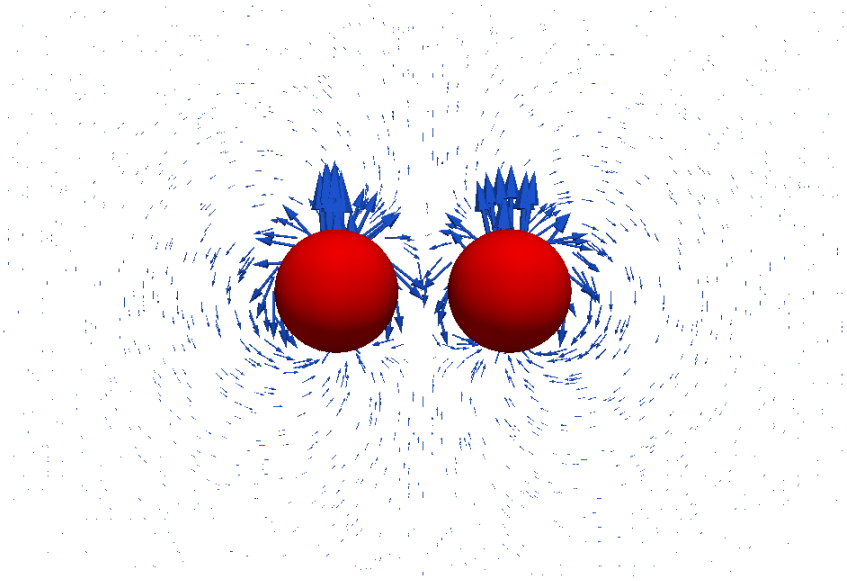

FIGURE 8.18 – Champ de vitesse de deux squirmers proches

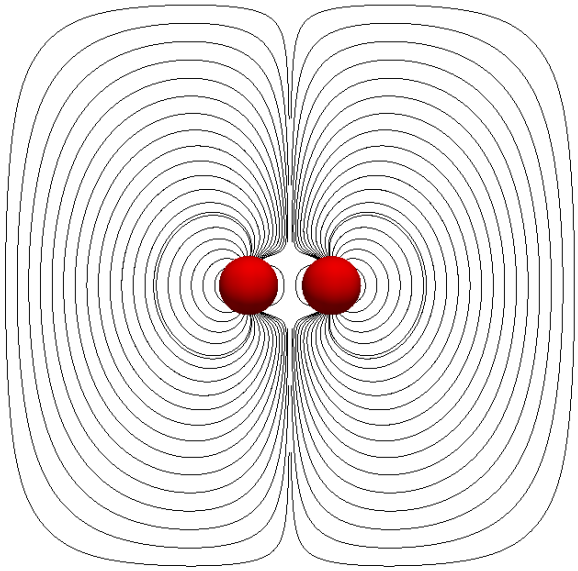

FIGURE 8.19 – Lignes de courant de deux squirmers proches

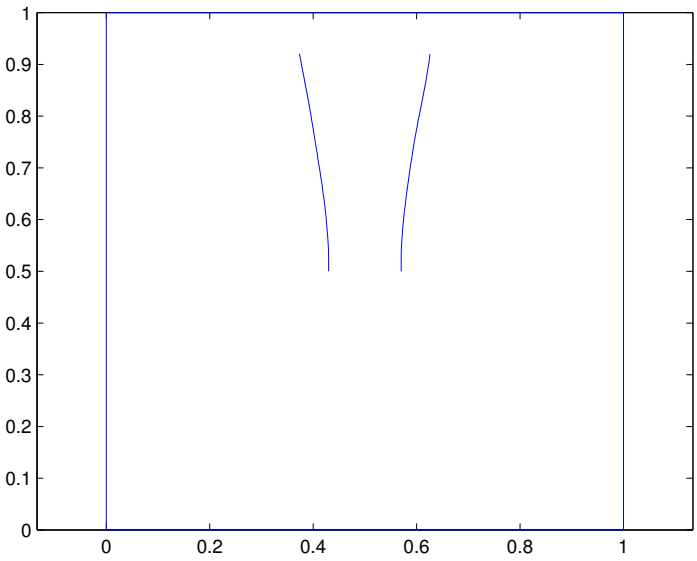

FIGURE 8.20 – Trajectoire de deux squirmers proches

Perspectives

Chapitre 9

Perspectives

Sommaire

Nous présentons dans ce chapitre quelques perspectives. On présente tout d'abord les modifications qu'il faut apporter pour se placer dans le cas des équations de Navier-Stokes et les premières difficultés que l'on rencontre. On perd notamment l'usage de matrices indépendantes de la position des particules. On parle ensuite un peu de bases réduites dans le but de réduire le temps de calcul. La méthode nécessite en effet la résolution de plusieurs problèmes de Stokes où seul le second membre change. Enfin, l'analyse numérique de la méthode dans le cas vectoriel reste à faire, on donne quelques pistes dans la dernière section.

9.1 Vers le régime inertiel

Nous avons considéré uniquement le régime non inertiel mais il peut être intéressant d'étendre l'analyse de la méthode et le code au cas des équations de Navier-Stokes. La simulation du tas de sable s'inscrit très bien dans ce cadre par exemple. Nous présentons dans cette section les idées et problèmes que l'on va rencontrer en voulant passer au régime inertiel.

Tout d'abord les équations sont modifiées et deviennent les suivantes :

$$
(9.1)\quad \left\{ \begin{array}{rcll}
\rho_f \left(\dfrac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} \right) - 2\eta \operatorname{div}(D(\mathbf{u})) + \nabla p &=& \rho_f \mathbf{f}\,, & \text{dans } \Omega \setminus \bar{B}\,, \\
\nabla \cdot \mathbf{u} &=& 0\,, & \text{dans } \Omega \setminus \bar{B}\,, \\
\mathbf{u} &=& 0\,, & \text{sur } \partial\Omega\,, \\
\mathbf{u} &=& \mathbf{V}_i + \boldsymbol{\omega}_i \wedge \mathbf{r}_i\,, & \text{sur } \partial B_i\,, \\
m_i \dfrac{d\mathbf{V}_i}{dt} &=& \mathbf{F}_i - \displaystyle\int_{\gamma_i} \sigma(\mathbf{u})\mathbf{n}_i\,, & \\
\mathbf{I}_i \dfrac{d\boldsymbol{\omega}_i}{dt} &=& \mathbf{T}_i - \displaystyle\int_{\gamma_i} \mathbf{r} \wedge \sigma(\mathbf{u})\mathbf{n}_i\,. &
\end{array} \right.
$$

Les deux dernières équations viennent du principe fondamental de la dynamique comme expliqué dans le premier chapitre. Ce sont exactement les équations 1.6 et 1.7. Une méthode pour discrétiser ces équations est la méthode des caractéristiques. La partie compliquée est de remonter les caractéristiques et de savoir dans quelle maille se trouve le pied. Avec un maillage cartésien, localiser un point dans le maillage est une étape assez simple ce qui en fait une méthode très pratique. Si on utilise un schéma en temps d'Euler explicite pour la dérivée totale, on obtient alors une équation de Stokes généralisée qui consiste en l'ajout d'une matrice de masse multipliée par le terme ρ/dt sur la partie en vitesse. La fonction scalaire ρ est égale à la masse volumique du fluide dans le fluide et à la densité de la particule sinon. Le coefficient dt est simplement le pas de temps utilisé dans la simulation.

Au niveau du code, il faut donc ajouter une matrice à la matrice des termes visqueux et on garde ainsi notre solveur de Stokes. Cette méthode demande donc très peu de changement en ce qui concerne les équations du fluide. On voit néanmoins apparaître un premier problème. Alors qu'avant toutes nos matrices étaient indépendantes de la position des particules, la matrice de masse que l'on ajoute pour prendre en compte les termes

d'inertie dépend de la position des particules par l'intermédiaire de la densité ρ. Il n'est cependant pas nécessaire de reconstruire toute la matrice à chaque itération en temps mais il faut tout de même corriger chaque coefficient de la matrice de masse pour prendre en compte la nouvelle position des particules.

Il reste à voir si on peut appliquer le même raisonnement dans le cas Navier-Stokes et dans le cas Stokes en ce qui concerne la recherche du contrôle. On peut choisir de résoudre l'équation suivante à la place de l'équation 9.1 :

$$(9.2)\quad \begin{cases} \rho\left(\dfrac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u}\right) - 2\eta\,\mathrm{div}(D(\mathbf{u}_g)) + \nabla p_g \\ \qquad = \rho_f \mathbf{f}\,\chi_{\Omega\setminus\bar{B}} + \mathbf{g}\,\chi_B \\ \qquad\quad + \displaystyle\sum_{i=1}^{N} \frac{\mathbf{F}_i}{|B_i|}\,\chi_{B_i} \\ \qquad\quad + \displaystyle\sum_{i=1}^{N} \frac{d}{C_i(d)|B_i|(d-1)}\,(\mathbf{T}_i \wedge \mathbf{r}_i)\,\chi_{B_i} \quad \text{dans } \Omega \\ \nabla\cdot\mathbf{u}_g = 0 \quad \text{dans } \Omega \\ \mathbf{u}_g = 0 \quad \text{sur } \partial\Omega \end{cases}$$

On peut intégrer la première équation sur une particule comme dans le cas de Stokes pour obtenir les contraintes que doit satisfaire le contrôle afin que les équations sur l'équilibre des forces soient satisfaites. Dans le cas Stokes on avait obtenu que le contrôle était cherché dans l'espace K_B, il est possible ici qu'on obtienne des contraintes différentes du fait de la non linéarité et des équations sur l'équilibre des forces qui font intervenir les dérivées en temps.

9.2 Décomposition du contrôle sur une base réduite

L'idée dans cette section est de chercher le contrôle sur une base contenant quelques éléments bien choisis pour en obtenir rapidement un convenable. Si on note $\mathbf{g}_k$ les éléments de la base, on calcule les solutions $\mathbf{u}_k$ du problème de Stokes défini sur tout le domaine :

$$\begin{cases} -2\eta\,\mathrm{div}(D(\mathbf{u}_k)) + \nabla p_k = \rho_f \mathbf{f}\,\chi_{\Omega\setminus\bar{B}} + \mathbf{g}\,\chi_B \\ \qquad\qquad + \displaystyle\sum_{i=1}^{N} \frac{\mathbf{F}_i}{|B_i|}\,\chi_{B_i} \\ \qquad\qquad + \displaystyle\sum_{i=1}^{N} \frac{d}{C_i(d)|B_i|(d-1)}\,(\mathbf{T}_i \wedge \mathbf{r}_i)\,\chi_{B_i}\,, \quad \text{dans } \Omega\,, \\ \nabla\cdot\mathbf{u}_k = 0\,, \quad \text{dans } \Omega\,, \\ \mathbf{u}_k = 0\,, \quad \text{sur } \partial\Omega\,. \end{cases}$$

On note de plus $\tilde{\mathbf{u}}_k$ les solutions des problèmes suivants :

$$\left\{\begin{array}{rcll} -2\eta\,\mathrm{div}(D(\tilde{\mathbf{u}}_k)) + \nabla\tilde{p}_k &=& \mathbf{g}_k\,\chi_B\,, & \text{dans } \Omega\,,\\ \nabla\cdot\tilde{\mathbf{u}}_k &=& 0\,, & \text{dans } \Omega\,,\\ \tilde{\mathbf{u}}_k &=& 0\,, & \text{sur } \partial\Omega\,. \end{array}\right.$$

On exprime alors le contrôle $\mathbf{g}$ recherché comme une combinaison linéaire des contrôles de la base :

$$\mathbf{g} = \sum_{k=1}^{n} \alpha_k \mathbf{g}_k\,,$$

où n est le nombre d'éléments dans la base. La fonctionnelle que l'on minimise devient alors une fonctionnelle sur $\mathbb{R}^n$. En effet, on note $\mathbf{u}_0$ la solution en prenant un contrôle nul du premier problème, alors par linéarité, la solution $\mathbf{u}_g$, pour un contrôle $\mathbf{g}$ qui s'exprime comme ci dessus, s'écrit :

$$\mathbf{u}_g = \mathbf{u}_0 + \sum_{k=1}^{n} \alpha_k \tilde{\mathbf{u}}_k\,.$$

La fonctionnelle devient :

$$J(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \frac{1}{2}\int_{\partial B_i} \left|\mathbf{u}_g - \frac{1}{|\partial B_i|}\int_{\partial B_i}\mathbf{u}_g - \frac{C_d}{{R_i}^2|\partial B_i|}\left(\int_{\partial B_i}\mathbf{r}_i\wedge\mathbf{u}_g\right)\wedge\mathbf{r}_i\right|^2\,,$$

où $\boldsymbol{\alpha}$ est le vecteur dans $\mathbb{R}^n$ des α_k. Le gradient est donc différent et on peut montrer que l'on a :

$$\frac{dJ}{d\alpha_k} = \sum_{i=1}^{N}\int_{\partial B_i}\left(\mathbf{u}_g - \frac{1}{|\partial B_i|}\int_{\partial B_i}\mathbf{u}_g - \frac{C_d}{{R_i}^2|\partial B_i|}\left(\int_{\partial B_i}\mathbf{r}_i\wedge\mathbf{u}_g\right)\wedge\mathbf{r}_i\right)\cdot\tilde{\mathbf{u}}_k\,.$$

Le calcul du gradient est donc plus simple car toutes les fonctions sont connues. L'idée est donc de calculer les solutions $\tilde{\mathbf{u}}_k$ pour tous les éléments dans la base et de minimiser ensuite la fonctionnelle. On gagne en temps de calcul car la minimisation de la fonctionnelle ne demande plus de résoudre un problème de Stokes comme avant. La difficulté est maintenant de bien choisir les éléments de la base qui vont dépendre du problème que l'on souhaite résoudre afin d'en utiliser un minimum.

9.3 Analyse numérique dans le cas vectoriel

L'analyse dans le cas Stokes est plus compliquée que celle faite dans le cas scalaire dans le chapître 4. Dans le cas scalaire on avait montré un lemme du type Aubin-Nitsche pour obtenir une estimation d'erreur sur la norme L^2 de la solution du problèmeÊ4.3 donnant le gradient. On rappelle le problème 2.18 équivalent dans le cas vectoriel :

$$\left\{\begin{array}{rcll} -2\eta\,\mathrm{div}(D(\mathbf{w}_g)) + \nabla\pi_g &=& \boldsymbol{\varphi}_g\delta_{\partial B}\,, & \text{dans } \Omega\,,\\ \nabla\cdot\mathbf{w}_g &=& 0\,, & \text{dans } \Omega\,,\\ \mathbf{w}_g &=& 0\,, & \text{sur } \partial\Omega\,, \end{array}\right.$$

où on a noté $\boldsymbol{\varphi}_g \delta_{\partial B}$ la distribution simple couche. Si on note $\mathbf{w}_h^{\tilde{h}}$ la solution approchée de ce problème en approchant la distribution simple couche par une combinaison de masses de Dirac, on peut refaire la même preuve que celle du lemme 4.2. La différence est que l'on va considérer le problème suivant dans le cas vectoriel :

$$\left\{ \begin{array}{rcll} -2\eta \operatorname{div}(D(\mathbf{v})) + \nabla q &=& \mathbf{w}_g - \mathbf{w}_h^{\tilde{h}}\,, & \text{dans } \Omega\,, \\ \nabla \cdot \mathbf{v} &=& 0\,, & \text{dans } \Omega\,, \\ \mathbf{v} &=& 0\,, & \text{sur } \partial\Omega\,. \end{array} \right.$$

On note $\mathbf{v}_h$ la solution approchée de ce problème. La formulation variationelle de 2.18 avec la fonction test $\mathbf{v}_h$ s'écrie alors :

$$\frac{1}{2}\int_\Omega D(\mathbf{w}_g) : D(\mathbf{v}_h) - \int_\Omega \pi_g \nabla \cdot \mathbf{v}_h = < \boldsymbol{\varphi}_g\,,\, \mathbf{v}_h > \,.$$

De même, la formulation vérifiée par la solution approchée avec une fonction test égale à $\mathbf{v}_h$ est :

$$\frac{1}{2}\int_\Omega D(\mathbf{w}_h^{\tilde{h}}) : D(\mathbf{v}_h) - \int_\Omega \pi_h \nabla \cdot \mathbf{v}_h = < \boldsymbol{\varphi}_h^{\tilde{h}}\,,\, \mathbf{v}_h > \,.$$

La différence entre ces deux formulations donne l'équation suivante :

$$\frac{1}{2}\int_\Omega D(\mathbf{w}_g - \mathbf{w}_h^{\tilde{h}}) : D(\mathbf{v}_h) - \int_\Omega (\pi_g - \pi_h) \nabla \cdot \mathbf{v}_h = < \boldsymbol{\varphi}_g - \boldsymbol{\varphi}_h^{\tilde{h}}\,,\, \mathbf{v}_h > \,.$$

On écrit enfin la formulation variationelle vérifiée par $\mathbf{v}$ avec une fonction test égale à $\mathbf{w}_g - \mathbf{w}_h^{\tilde{h}}$:

$$\frac{1}{2}\int_\Omega D(\mathbf{v}) : D(\mathbf{w}_g - \mathbf{w}_h^{\tilde{h}}) - \int_\Omega q \nabla \cdot (\mathbf{w}_g - \mathbf{w}_h^{\tilde{h}}) = \int_\Omega |\mathbf{w}_g - \mathbf{w}_h^{\tilde{h}}|^2\,.$$

On fait alors la différence entre les deux dernières équations pour obtenir :

$$\begin{aligned} \frac{1}{2}\int_\Omega D(\mathbf{v} - \mathbf{v}_h) : D(\mathbf{w}_g - \mathbf{w}_h^{\tilde{h}}) - \int_\Omega q \nabla \cdot (\mathbf{w}_g - \mathbf{w}_h^{\tilde{h}}) \\ + \int_\Omega (\pi_g - \pi_h) \nabla \cdot \mathbf{v}_h - < \boldsymbol{\varphi}_g - \boldsymbol{\varphi}_h^{\tilde{h}}\,,\, \mathbf{v}_h > = \int_\Omega |\mathbf{w}_g - \mathbf{w}_h^{\tilde{h}}|^2\,. \end{aligned}$$

Dans le cas scalaire, on obtenait alors une estimation sur la norme L^2 de l'erreur. Ici, il reste des termes en pression qui ne sont pas nuls puisque l'on utilise des éléments finis qui ne sont pas à divergence nulle. Pour le moment, l'analyse numérique dans le cas Stokes reste donc à faire, il faut pouvoir prendre en compte ces termes de pression. La comparaison visuelle avec une méthode d'ordre inférieur dans le cas de la sédimentation d'une particule que l'on trouve au chapître 4 laisse toutefois penser que la méthode de prolongement régulier se comporte bien dans le cas vectoriel aussi.

Bibliographie

[1] Page web de scopi, 2012. Available from : http://www.cmap.polytechnique.fr/~lefebvre/SCoPI.htm.

[2] R. A. Adams and J. J. F. Fournier. *Sobolev spaces*. Elsevier, seconde edition, 2003.

[3] C. Amrouche, C. Bernardi, M. Dauge, and V. Girault. Vector potentials in three-dimensional non-smooth domains. *Math. Method Appl. Sci.*, 21(9) :823–864, 1998. doi:10.1002/(SICI)1099-1476(199806)21:9<823::AID-MMA976>3.0.CO;2-B.

[4] C. Atamian, Q. V. Dinh, R. Glowinski, J. He, and J. Périaux. Control approach to fictitious-domain methods. application to fluid dynamics and electro-magnetics. In *Domain decomposition methods for partial differential equations*, 1991.

[5] C. Atamian and P. Joly. Une analyse de la méthode des domaines fictifs pour le problème de helmholtz extérieur. *RAIRO-Math. Model. Num.*, 27(3) :251–288, 1993. Available from : http://www.numdam.org/item?id=M2AN_1993__27_3_251_0.

[6] L. Badea and P. Daripa. On a fourier method of embedding domains using an optimal distributed control. *Numer. Algorithms*, 32(2-4) :261–273, 2003. doi:10.1023/A:1024002802603.

[7] L. Badea and P. Daripa. A domain embedding method using the optimal distributed control and a fast algorithm. *Numer. Algorithms*, 36(2) :95–112, 2004. doi:10.1023/B:NUMA.0000033094.75324.48.

[8] J. Baiges and R. Codina. Approximate imposition of boundary conditions in immersed boundary methods. *Int. J. Numer. Meth. Engng.*, 80(11) :1379–1405, 2009. doi:10.1002/nme.2662.

[9] J. Baiges, R. Codina, F. Henke, S. Shahmiri, and W. A. Wall. A symmetric method for weakly imposing dirichlet boundary conditions in embedded finite element meshes. *Int. J. Numer. Meth. Engng.*, 90(5) :636–658, 2012. doi:10.1002/nme.3339.

[10] S. Balay, J. Brown, , K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. Curfman McInnes, B. F. Smith, and H. Zhang. PETSc users manual. Technical Report ANL-95/11 - Revision 3.3, Argonne National Laboratory, 2012.

[11] S. Balay, J. Brown, K. Buschelman, W. D. Gropp, D. Kaushik, M. G. Knepley, L. Curfman McInnes, B. F. Smith, and H. Zhang. PETSc Web page, 2012. Available from : http://www.mcs.anl.gov/petsc.

[12] S. Balay, W. D. Gropp, L. Curfman McInnes, and B. F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhäuser Press, 1997.

[13] G. K. Batchelor and J. T. Green. The hydrodynamic interaction of two small freely-moving spheres in a linear flow field. *J. Fluid Mech.*, 56(02) :375–400, 1972. doi:10.1017/S0022112072002927.

[14] R. Becker and S. Mao. Quasi-optimality of adaptive nonconforming finite element methods for the stokes equations. *SIAM J. Numer. Anal.*, 49(3) :970–991, 2011. doi:10.1137/100802967.

[15] S. Bertoluzza, M. Ismail, and B. Maury. Analysis of the fully discrete fat boundary method. *Numer. Math.*, 118(1) :49–77, 2011. doi:10.1007/s00211-010-0317-4.

[16] F. Bertrand, P. A. Tanguy, and F. Thibault. A three-dimensional fictitious domain method for incompressible fluid flow problems. *Int. J. for Numer. Methods in Fluids*, 25(6) :719–736, 1997. doi:10.1002/(SICI)1097-0363(19970930)25:6<719::AID-FLD585>3.0.CO;2-K.

[17] J. R. Blake. A spherical envelope approach to ciliary propulsion. *J. Fluid. Mech.*, 46(1) :199–208, 1971. doi:10.1017/S002211207100048X.

[18] D. Boffi and L. Gastaldi. A finite element approach for the immersed boundary method. *Comput. Struct.*, 81(8-11) :491–501, 2003. doi:10.1016/S0045-7949(02)00404-2.

[19] D. Braess. *Finite elements : theory, fast solvers, and application to solid mechanics*. Cambidge University Press, seconde edition, 2001.

[20] H. Brezis. *Analyse fonctionnelle : Théorie et applications*. Dunod, 2005.

[21] E. Burman. Ghost penalty. *C. R. Math. Acad. Sci. Paris*, 348(21-22) :1217–1220, 2010. doi:10.1016/j.crma.2010.10.006.

[22] E. Burman and P. Hansbo. Fictitious domain finite element methods using cut elements : Ii. a stabilized nitsche method. *Appl. Numer. Math.*, 62(4) :328–341, 2012. doi:10.1016/j.apnum.2011.01.008.

[23] F. Chantalat, C.-H. Bruneau, C. Galusinski, and A. Iollo. Level-set, penalization and cartesian meshes : A paradigm for inverse problems and optimal design. *J. Comput. Phys.*, 228(17) :6291Ð6315, 2009. doi:10.1016/j.jcp.2009.05.017.

[24] P. G. Ciarlet. *The finite element method for elliptic problems*. SIAM, 2002.

[25] R. Codina, G. Houzeaux, H. Coppola-Owen, and J. Baiges. The fixed-mesh ale approach for the numerical approximation of flows in moving domains. *J. Comput. Phys.*, 228(5) :1591–1611, 2009. doi:10.1016/j.jcp.2008.11.004.

[26] G.-H. Cottet and E. Maitre. A level set method for fluid-structure interactions with immersed surfaces. *Math. Models Meth. Appl. Sci.*, 16(3) :415–438, 2006. doi:10.1142/S0218202506001212.

[27] A. Decoene, S. Martin, and B. Maury. Microscopic modelling of active bacterial suspensions. *Math. Model. Nat. Phenom.*, 6(5), 2011. doi:10.1051/mmnp/20116506.

[28] K. Dresher, K. C. Leptos, I. Tuval, T. Ishikawa, T. J. Pedley, and R. E. Goldstein. Dancing *Volvox* : hydrodynamic bound states of swimming algae. *Phys. Rev. Lett.*, 102(168101), 2009. doi:10.1103/PhysRevLett.102.168101.

[29] G. Duvaut. *Mécanique des milieux continus*. Masson, 1990.

[30] H. Elman, D. Silvester, and A. Wathen. *Finite Elements and fast iterative solvers : with applications in incompressible fluid dynamics*. Oxford University Press, 2005.

[31] V. Girault and R. Glowinski. Error analysis of a fictitious domain method applied to a dirichlet problem. *Jpn. J. Ind. Appl. Math.*, 12(3) :487–514, 1995. doi:10.1007/BF03167240.

[32] V. Girault, R. Glowinski, H. López, and J.-P. Vila. A boundary multiplier/fictitious domain method for the steady incompressible navier-stokes equations. *Numer. Math.*, 88(1) :75–103, 2001. doi:10.1007/PL00005441.

[33] V. Girault and P.-A. Raviart. *Finite Element Approximation of the Navier-Stokes Equations*. Springer, 1979.

[34] E. Givelberg and J. Bunn. A comprehensive three-dimensional model of the cochlea. *J. of Comput. Phys.*, 191(2) :377–391, 2003. doi:10.1016/S0021-9991(03)00319-X.

[35] R. Glowinski, T.-W. Pan, T. I. Hesla, D. D. Joseph, and J. Périaux. A fictitious domain approach to the direct numerical simulation of incompressible viscous flow past moving rigid bodies : application to particulate flow. *J. Comput. Phys.*, 169(2) :363–426, 2001. doi:10.1006/jcph.2000.6542.

[36] R. Glowinski, T.-W. Pan, and J. Periaux. A lagrange multiplier/fictitious domain method for the dirichlet problem - generalization to some flow problems. *Jpn. J. Ind. Appl. Math.*, 12(1) :87–108, 1995. doi:10.1007/BF03167383.

[37] R. Glowinski, T.-W. Pan, and J. Periaux. A lagrange multiplier/fictitious domain method for the numerical simulation of incompressible viscous flow around moving rigid bodies (i) : The case where the rigid body motions are known a priori. *Cr. Acad. Sci. I-Math.*, 324(3) :361–369, 1997. doi:10.1016/S0764-4442(99)80376-0.

[38] B. M. Haines, I. S. Aranson, L. Berlyand, and D. A. Karpeev. Effective viscosity of dilute bacterial suspensions : a two-dimensional model. *Phys. Biol.*, 5(4), 2008. doi:10.1088/1478-3975/5/4/046003.

[39] A. Hansbo and P. Hansbo. An unfitted finite element method, based on nitscheÕs method, for elliptic interface problems. *Comput. Method Appl. M.*, 191(47-48) :5537–5552, 2002. doi:10.1016/S0045-7825(02)00524-8.

[40] M. Herrmann. A balanced force refined level set grid method for two-phase flows on unstructured flow solver grids. *J. Comput. Phys.*, 227(4) :2674Ð2706, 2008. doi:10.1016/j.jcp.2007.11.002.

[41] H. H. Hu. Direct simulation of flows of solid-liquid mixtures. *Int. J. Multiphas. Flow*, 22(2) :335–352, 1996. doi:10.1016/0301-9322(95)00068-2.

[42] H. Huang and Z. Li. Convergence analysis of the immersed interface method. *IMA J. Numer. Anal.*, 19(4) :583–608, 1999. doi:10.1093/imanum/19.4.583.

[43] F. Ilinca and J.-F. Hétu. A finite element immersed boundary method for fluid flow around rigid objects. *Int. J. Numer. Meth. Fluids*, 65(7) :856–875, 2011. doi:10.1002/fld.2222.

[44] D. M. Ingram, D. M. Causon, and C. G. Mingham. Developments in cartesian cut cell methods. *Math. Comput. Simulat.*, 61(3-6) :561–572, 2003. doi:10.1016/S0378-4754(02)00107-6.

[45] T. Ishikawa and T. J. Pedley. Diffusion of swimming model micro-organisms in a semi-dilute suspension. *J. Fluid Mech.*, 588 :437–462, 2007. doi:10.1017/S0022112007007847.

[46] J. Janela, A. Lefebvre, and B. Maury. A penalty method for the simulation of fluid - rigid body interaction. *ESAIM : Proc.*, 14 :115–123, 2005. doi:10.1051/proc:2005010.

[47] Y. Kim and C. S. Peskin. 3-d parachute simulation by the immersed boundary method. *Comput. & Fluids*, 38(6) :1080–1090, 2009. doi:10.1016/j.compfluid.2008.11.002.

[48] A. Lefebvre. *Mdélisation numérique d'écoulements fluide/particules.* PhD thesis, Université Paris-Sud, 2007.

[49] A. Lefebvre and B. Maury. Apparent viscosity of a mixture of a newtonian fluid and interacting particles. *Comptes rendus mécanique*, 333(12) :923–933, 2005. doi:10.1016/j.crme.2005.10.007.

[50] G. Legendre and T. Takahashi. Convergence of a lagrange-galerkin method for a fluid-rigid body system in ale formulation. *ESAIM-Math. Model. Num.*, 42(4) :609–644, 2008. doi:10.1051/m2an:2008020.

[51] R. J. Leveque and Z. Li. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *SIAM J. Numer. Anal.*, 31(4) :1019–1044, 1994. Available from : http://www.jstor.org/stable/2158113.

[52] Z. Li. An overview of the immersed interface method and its applications. *Taiwan. J. Math.*, 7(1) :1–49, 2003.

[53] M. J. Lighthill. An adaptive augmented lagrangian method for three-dimensional multimaterial flows. *Commun. Pure Appl. Math.*, 5 :109–118, 1952.

[54] J. San Martín, L. Smaranda, and T. Takahashi. Convergence of a finite element/ale method for the stokes equations in a domain depending on time. *J. Comput. Appl. Math.*, 230(2) :521–545, 2009. doi:10.1016/j.cam.2008.12.021.

[55] B. Maury. Direct simulations of 2d fluid-particle flows in biperiodic domains. *J. Comput. Phys.*, 156(2) :325–351, 1999. doi:10.1006/jcph.1999.6365.

[56] B. Maury. A fat boundary method for the poisson problem in a domain with holes. *J. Sci. Comput.*, 16(3) :319–339, 2001. doi:10.1023/A:1012821728631.

[57] B. Maury. Numerical analysis of a finite element/volume penalty method. *SIAM J. Numer. Anal.*, 47(2) :1126–1148, 2009. doi:10.1137/080712799.

[58] R. Mittal and G. Iaccarino. Immersed boundary methods. *Annu. Rev. of Fluid Mech.*, 37 :239–261, 2005. doi:10.1146/annurev.fluid.37.061903.175743.

[59] J. Nitsche. über ein variationsprinzip zur lösung von dirichlet-problemen bei verwendung von teilräumen, die keinen randbedingungen unterworfen sind. *Abh. Math. Sem. Hamburg*, 36(1) :9–15, 1971. doi:10.1007/BF02995904.

[60] T.-W. Pan. A lagrange multipliers / fictitious domain / collocation method for solid-liquid flows. *IMA Vol. in Math. Appl.*, 120 :97–122, 2000.

[61] B. Perret. *Étude de quelques méthodes de domaine fictif et applications aux ondes électromagnétiques*. PhD thesis, Paris 6 Jussieu, 1998.

[62] C. S. Peskin. Numerical analysis of blood flow in the heart. *J. Comput. Phys.*, 25(3) :220–252, 1977. doi:10.1016/0021-9991(77)90100-0.

[63] C. S. Peskin. The immersed boundary method. *Acta numerica*, 11 :479–517, 2002. doi:10.1017/S0962492902000077.

[64] C. S. Peskin and D. M. McQueen. A three-dimensional computational method for blood flow in the heart i. immersed elastic fibers in a viscous incompressible fluid. *J. of Comput. Phys.*, 81(2) :372–405, 1989. doi:10.1016/0021-9991(89)90213-1.

[65] I. Ramière. Convergence analysis of the q_1-finite element method for elliptic problems with non-boundary-fitted meshes. *Int. J. Numer. Meth. Eng.*, 75(9) :1007–1052, 2008. doi:10.1002/nme.2278.

[66] I. Ramière, P. Angot, and M. Belliard. A fictitious domain approach with spread interface for elliptic problems with general boundary conditions. *Comput. Method Appl. M.*, 196(4-6) :766–781, 2007. doi:10.1016/j.cma.2006.05.012.

[67] N. Dos Santos, J.-F. Gerbeau, and J.-F. Bourgat. A partitioned fluid-structured algorithm for elastic thin valves with contact. *Comput. Methods in Appl. Mech. and Eng.*, 197(19-20) :1750–1761, 2008. doi:10.1016/j.cma.2007.03.019.

[68] R. Stenberg. Analysis of mixed finite element methods for the stokes problem : a unified approach. *Math. Comp.*, 42 :9–23, 1984.

[69] Z. Tan, K. M. Lim, and B. C. Khoo. An immersed interface method for stokes flows with fixed/moving interfaces and rigid boundaries. *J. Comput. Phys.*, 228(18) :6855–6881, 2009. doi:10.1016/j.jcp.2009.06.005.

[70] R. Temam. *Navier-Stokes Equations : Theory and Numerical Analysis*. American Mathematical Society, 2001.

[71] L. Tomas. An error estimate for a fictitious domain formulation using volumic lagrange multipliers of a dirichlet problem. *Cr. Acad. Sci. I-Math.*, 325(7) :793–796, 1997. doi:10.1016/S0764-4442(97)80061-4.

[72] R. Tyson, C. E. Jordan, and J. Hebert. Modelling anguilliform swimming at intermediate reynolds number : A review and a novel extension of immersed boundary method applications. *Comput. Methods in Appl. Mech. and Eng.*, 197(25-28), 2008. doi:10.1016/j.cma.2007.07.009.

[73] S. Xu. The immersed interface method for simulating prescribed motion of rigid objects in an incompressible viscous flow. *J. Comput. Phys.*, 227(10) :5045–5071, 2008. doi:10.1016/j.jcp.2008.01.053.