NSIBM : un solveur parallèle de Navier-Stokes avec raffinement automatique basé sur la méthode des frontières immergées

Daniel Durrenberger

THÈSE

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITE DE STRASBOURG

ÉCOLE DOCTORALE MATHÉMATIQUES, SCIENCES DE L'INFORMATION ET DE L'INGÉNIEUR

DOMAINE DE RECHERCHE : MÉCANIQUE DES FLUIDES

présentée par

Daniel DURRENBERGER

NSIBM : un solveur parallèle de Navier-Stokes avec raffinement automatique basé sur la méthode des frontières immergées

---

Soutenue publiquement le 18 décembre 2015
Version du 20 janvier 2016

THÈSE dirigée par :

M. Yannick HOARAU, Professeur, Université de Strasbourg

RAPPORTEURS :

M. Éric GONCALVES , Professeur ISAE-ENSMA

M. Alistair REVELL , Senior Lecturer University of Manchester

EXAMINATEURS :

M. Jan DUŠEK , Professeur Université de Strasbourg

M. Yannick HOARAU , Professeur Université de Strasbourg

Résumé de la thèse

Cette thèse, intitulée *NSIBM : un solveur parallèle de Navier-Stokes avec raffinement automatique basé sur la méthode des frontières immergées*, a été effectuée au sein du laboratoire iCube, département de mécanique, à Strasbourg, dans le quartier de l'Orangerie, sous la direction du professeur Yannick Hoarau. L'essentiel du travail effectué consiste en le développement d'un programme capable de résoudre numériquement l'équation de Navier-Stokes qui régit des fluides en mouvement.

Une attention particulière a été portée à la production de maillages conformes aux géométries proposées et à leur génération. Les moyens mis en œuvre ici pour gérer l'éternel problème de la finesse du maillage opposée au trop grand nombre de cellules sont multiples : le raffinement, la parallélisation et les frontières immergées. Dans un premier temps, j'ai conçu un générateur de maillage en deux et trois dimensions en y intégrant la possibilité de diviser des cellules, et cela de manière automatique, par des critères géométriques, numériques ou physiques. Il permet également de supprimer des cellules, de manière à ne pas mailler le vide ou les parties solides de la géométrie. Dans un deuxième temps, j'ai rendu ce code parallèle en lui donnant la capacité d'utiliser plusieurs processeurs, afin de calculer plus vite et donc d'utiliser davantage de mailles. Cette étape fait appel à deux technologies : *Metis*, qui partage équitablement les mailles sur le nombre choisi de processeurs et *openMPI*, qui est l'interface de communication entre ces processeurs. Enfin, la méthode des frontières immergées a été introduite au code pour gérer les bords non verticaux ou horizontaux dans un maillage cartésien, c'est-à-dire formé de rectangles ou de pavés droits. Elle consiste à donner un caractère hybride à une cellule traversée par une frontière par l'introduction d'un terme numérique de forçage simulant la présence de la paroi.

Ce travail de développement a ensuite été mis à l'épreuve et validé dans une série de cas tests en deux comme en trois dimensions. Des exemples de maillages complexes générés facilement sont donnés.

Remerciements

Je tiens à remercier en premier lieu le chef : Yannick Hoarau, qui m'a permis d'effectuer cette thèse au sein du laboratoire iCube. Sa compétence et sa disponibilité m'ont impressionné tout au long de notre collaboration et sa personnalité enthousiaste rend le travail plus facile et plus détendu. Sa droiture et la confiance qu'il manifeste envers toute son équipe achèvent de faire le lui le meilleur des chefs.

Je remercie les jurés, Éric Goncalves, Allistair Revell, Jan Dušek et bien sûr Yannick Hoarau, qui ont bien voulu rapporter ou examiner ce travail pour leurs remarques constructives et la compréhension qu'ils ont eu de ce document et de l'effort qu'il représente.

J'ai une profonde reconnaissance envers mon canard en plastique, l'inestimable Anthony Ponce qui m'a tant de fois aidé, débloqué, débogué dans mon travail de codage. Il mérite haut la main ce paragraphe de remerciement pour lui tout seul.

Merci à mes parents, qui m'ont encouragé, soutenu, et donné les moyens d'entreprendre ces longues années d'études qui s'achèvent en ce 18 décembre 2015.

Merci à Pauline, la femme de ma vie, que je n'avais pas vu venir en commençant ce travail, d'avoir été tellement là pour moi et surtout d'avoir été tellement patiente, jusqu'à ce que cet épisode soit clos. C'est elle qui a eu la lourde tâche de recueillir mes doutes et m'obliger à tenir mes objectifs d'efficacité datés. Ma gratitude est ici accompagnée d'admiration et de félicitations.

Merci à tous les collègues de l'équipe ITD : Marcin, Dorian, Vincent, Chao-Kun, Wei, Chrystelle, Ali, Anthony, Pierre d'être aussi cools, intéressants et marrants ; travailler quotidiennement à vos côtés a été un plaisir et vous êtes assurément la raison de ma bonne humeur au bureau. Je remercie également tous ceux qui travaillent ou ont travaillé ici dans les autres bureaux, dans les autres équipes pour les atomes crochus que j'ai pu développer avec eux : je pense à Noëlle, Vivien, Sandra, Marie, Charlotte, Maxime, Gilles, Roman.

Merci à mon meilleur frère Simon et à ma meilleure sœur Elisabeth de m'avoir encouragé pendant ces années et d'être fiers d'avoir un grand frère thésard. Ça aide plus que vous ne le croyez !

Ma gratitude va aussi vers tous ceux qui ont participé au bon déroulement de ses années de thèse, de près ou de loin, peut-être même sans s'en rendre compte. Sans relation d'ordre aucune, Knacki Ball, le docteur Christophe Gorensteiner, Maurice, Natacha, Sara, Hélène, Nathan, Sylvain, Jacques, Frédéric, Gabrielle, Jérémy, Mitch, ton altesse Ludovic, Songlin, Romain, Natmouth, Messieurs H et H', Médéric, Anaïs, Claire ; vous avez rendu cette période faste !

Je citerai encore le SUAPS, duquel j'ai bien profité.

Blâmes

La première personne à blâmer est un certain Hans Aplast, ou Dédé, voire carrément moi même : il a eu le mauvais goût d'être imprudent au ski et de se re-blesser deux ans plus tard, retardant ainsi son travail. Je n'aime pas frapper un homme affaibli, mais lui le mérite amplement.

J'ai bien envie de citer dans ce paragraphe tout le travail administratif à réaliser pendant la durée d'une thèse : ces petites tâches qui nous coûtent tant d'énergie et de motivation qu'on les laisse toujours pour la dernière minute.

Un blâme également aux incompatibilités informatiques, aux paquets trop compliqués à installer, bref à tous ces petits problèmes qui font perdre du temps et qui apparaissent sous un système d'exploitation ou un autre.

Ces blâmes sont trop courts ; je pense que, finalement, cela mérite un remerciement.

Il y en tout de même un à qui j'ai énormément à reprocher, mais je préfère pas dire qui c'est...

Table des matières

Liste des figures

Liste des tableaux

Introduction

Lorsque l'on a affaire à un problème trop compliqué pour être résolu de manière analytique, comme c'est le cas pour la mécanique des fluides, on se retrouve face à trois possibilités : abandonner, continuer à chercher une solution exacte ou choisir d'approcher la solution. Démarrer une thèse en suivant la première est inutile et en finir une en optant pour la deuxième me parait impossible.

Le premier défaut d'une solution approchée est d'être partielle en temps et en espace. En effet, la capacité de stockage en mémoire d'une machine est finie et ne peut donc pas sauvegarder un nombre infini, et de surcroît non-dénombrable de valeurs. La première étape est donc de discrétiser l'espace dans lequel on désire résoudre l'équation. Cela revient à choisir des points de l'espace auxquels on résoudra l'équation de manière approchée. Mis ensemble, ces points forment une "toile d'araignée" recouvrant le domaine d'étude. En langage scientifique, c'est un **maillage**.

Essentiellement, chaque valeur en un point du maillage est calculée en s'appuyant sur les valeurs de ces voisins et chacune nécessite un travail de la machine. De ce maillage dépendent donc la précision de la solution numérique ainsi que le temps de calcul nécessaire pour l'obtenir. Comment s'y prendre alors, pour que ces points soient le plus proches possible tout en étant le moins nombreux possible ?

Les équations de Navier-Stokes

Les équations de Navier-Stokes représentent un système d'équations aux dérivées partielles modélisant les mouvements des fluides dans l'approximation des milieux continus. Sa résolution nous échappe encore mathématiquement parlant : nous ne savons pas, à l'heure actuelle, la résoudre de manière théorique et générale. On ne sait le faire que dans certains (rares) cas particuliers. Les équations de Navier-Stokes sont pourtant très importantes pour la communauté scientifique puisqu'elles interviennent, par exemple, dans la prédiction de la météo, l'optimisation du profil d'une aile d'avion, la simulation des océans et même l'amélioration des graphisme de jeux vidéos [1]. Les équations de Navier-Stokes forment d'ailleurs le sixième des sept problèmes du prix du millénaire posés par l'Institut de mathématiques Clay en l'an 2000 et sa résolution est ainsi récompensée d'un million de dollars. Et encore, on ne parle pas de résolution analytique de l'équation différentielle en trouvant une solution explicite ; pour gagner cette somme, il "suffit" de

— démontrer que pour toute condition initiale, il existe une solution régulière et globalement définie (c'est-à-dire définie pour tout t de 0 à l'infini)

OU

— trouver un contre-exemple, c'est-à-dire une condition initiale $u(0)$ telle qu'il n'existe **pas** de solution régulière globalement définie.

Maintenant que l'on commence à saisir la difficulté du problème de Navier-Stokes, on comprend pourquoi mathématiciens, ingénieurs et physiciens s'acharnent depuis 1911 [2] à sa résolution numérique.

Voici enfin les équations de Navier-Stokes, pour un fluide incompressible, qui sont en fait la version fluide de la deuxième loi de la mécanique newtonienne $\sum \boldsymbol{f} = m\boldsymbol{a}$ [3]. La première équation concerne uniquement le champ de vitesse :

$$\operatorname{div} \boldsymbol{u} = 0, \tag{1}$$

tandis que la seconde comprend la vitesse, la pression et un terme non-linéaire :

$$\underbrace{\rho}_{m}\underbrace{\left(\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u}\cdot\nabla\boldsymbol{u}\right)}_{a} = \underbrace{-\nabla p}_{F_{\text{pression}}} + \underbrace{\mu\nabla^2\boldsymbol{u}}_{F_{\text{visqueuse}}} \tag{2}$$

où $\boldsymbol{u}$ est le champ de vitesse, p la pression, ρ la masse volumique du fluide et μ sa viscosité.

Il existe à ce jour de nombreuses méthodes pour la résoudre numériquement et l'objet de cette thèse est d'en mettre une en œuvre et de la choisir en accord avec les besoins et les possibilités de l'équipe **MécaFlu** du laboratoire strasbourgeois **iCube**.

Contenu de cette thèse

Le but de la présente thèse est de munir le laboratoire iCube d'un outil multifonction pour s'attaquer à la simulation d'écoulements de fluides incompressibles et de l'utiliser ensuite sur de nouveaux cas de figure, non traités dans la littérature scientifique à ce jour.

Il s'agit d'un travail méticuleux et ciblé qui consiste principalement en codage en langage Fortran 95. Cette tâche a été accomplie en partant d'un fichier vierge, de quelques notions de Fortran et des objectifs fixés par mon directeur de thèse : le professeur Yannick Hoarau. Ces objectifs étaient de produire un solveur non-structuré des équations de Navier-Stokes capable de créer de manière automatique un maillage adapté à n'importe quelle géométrie décrite par un fichier de stéréolithographie. Il devait également fonctionner en parallèle.

La force d'un tel outil est d'être en mesure de mener à bien des simulations variées en réduisant au maximum l'investissement en temps humain.

Ce "couteau suisse numérique" existe maintenant et est doté des fonctions de base qui lui sont nécessaires. J'ai pris soin de produire un code propre, lisible, commenté et documenté par égard pour mes successeurs. NSIBM est prêt à s'étoffer par l'adjonction de nouvelles fonctionnalités qui lui permettraient de couvrir une gamme encore plus large de problèmes de mécanique des fluides.

En me lançant dans cette thèse, je me disais : "C'est bien, tu codes, donc tu peux progresser linéairement et chaque jour, tu peux mesurer ton avancement en lignes". Force m'est de constater que cette assertion s'est avérée totalement fausse : les jours

passés à accumuler des lignes de codes ont été suivis de semaines à les déboguer et le solveur NSIBM que je laisse derrière moi est la troisième version.

J'ouvre ce manuscrit en passant en revue les voies qui ont déjà été explorées pour venir à bout de ce genre de problème, en orientant le récit vers les options qui ont été mise en œuvre dans NSIBM. Je m'attèle ensuite à expliquer la méthode de résolution numérique et à détailler le fonctionnement du raffinement automatique. La partie suivante contient les informations relatives à la parallélisation du solveur. Enfin, les deux dernières parties sont consacrées à la validation du code sur des cas documentés et à son utilisation sur le cas d'un écoulement libre autour d'un barreau à section carrée monté sur un mur.

Chapitre 1

Bibliographie

« *I am an old man now, and when I die and go to heaven there are two matters on which I hope for enlightenment. One is quantum electrodynamics, and the other is the turbulent motion of fluids. And about the former I am rather optimistic.* »

Sir Horace Lamb

1.1 Techniques de maillage

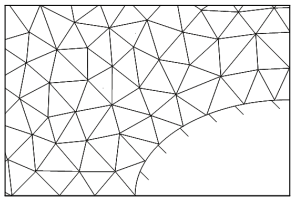
Les modèles physiques décrivant notre monde présument quasiment tous de sa continuité en temps et en espace et mènent à des équations différentielles qui peuvent s'avérer très compliquées. Si tel est le cas, on peut chercher dans un premier temps à simplifier le modèle ; on peut par exemple négliger les frottements ou supposer un liquide incompressible. Malheureusement, cela suffit rarement : notre monde dépasse encore, et de loin, notre avancée mathématique.

Si une solution mathématique continue est hors de portée, l'informatique et la puissance de calcul qui en découle nous permet de nous replier sur une solution mathématique discrète. Il faut alors convertir discrétiser le modèle en temps et en espace. La discrétisation en temps relève du simple choix d'un pas de temps, tandis que la discrétisation en espace mérite une réflexion plus poussée. Le choix du schéma temporel (implicite, explicite, ordre) doit ensuite être fait en fonction des propriétés du maillage.

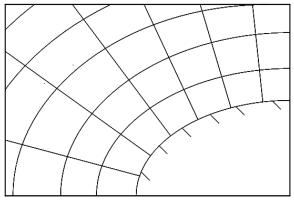
La nécessité d'un maillage ne se limite donc pas au domaine de la mécanique des fluides, mais à tous les domaines de la physique dont les équations différentielles sont trop compliquées pour être résolues analytiquement : en un mot, toutes.

Les maillages ont bien évolué depuis la méthode d'Euler (1707 - 1783) et il est d'à-propos de détailler ce point.

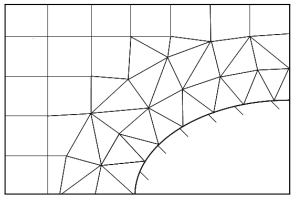
Un maillage peut être formé de rectangles, de triangles, de pavés, de polyèdres, voire d'un mélange de différents types de cellules. Ensuite, il faut choisir la manière dont on gère le maillage ; décider de la manière de numéroter les cellules et de stocker ses informations topologiques. Pour cela, deux solutions peuvent être mises en œuvre : un maillage structuré et un maillage non-structuré. La variété de techniques de maillage est illustrée par quelques exemples tirés de Mentor Graphics Whitepaper [4] sur la figure 1.1 ci-après. L'enjeu est simple : obtenir le résultat numérique le plus précis



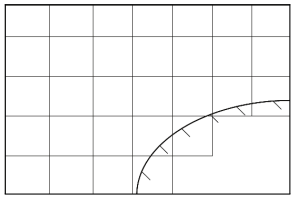
Maillage adapté non structuré



Maillage adapté structuré



Combinaison de maillages



Maillage cartésien non adapté

FIGURE 1.1 – Exemples de techniques de maillage tirés de Mentor Graphics Whitepaper [4]

possible en le moins de temps possible. D'une part, l'équation ou le système d'équations en question doit être résolu à chaque nœud du maillage et cela implique donc une dépendance au mieux linéaire entre le nombre de calculs nécessaires et le nombre

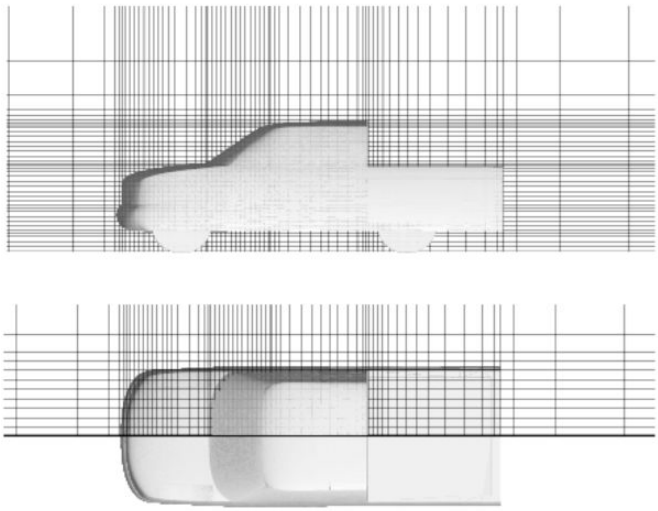


FIGURE 1.2 – Un exemple de maillage cartésien tiré de Kalitzin et al. [5] qui est plus dense au niveau de la frontière. Il convient très bien dans ce cas, mais manque de modularité si des éléments alignés selon des axes nécessitent des précisions de maillage trop hétérogènes.

de nœuds. D'autre part, la précision du résultat dépend directement de la proximité de ces nœuds, et donc de la densité du maillage. Ainsi, le meilleur maillage possible est naïvement celui qui est composé du moins de points tout en étant doté de la plus grande densité, ce qui est un paradoxe. Le comportement numérique du maillage est également à prendre en compte : un maillage non cartésien va nécessiter davantage d'interpolations et de projections qui s'avèrent néfastes à la complexité et au temps de calcul, mais permettra de mieux représenter les réalités physiques que l'on cherche à modéliser. Cette section donne un aperçu des stratégies disponibles à ce jour.

1.1.1 Maillages cartésiens

Les maillages cartésiens, uniquement composés de segments, rectangles ou parallélépipèdes rectangulaires selon que l'on travaille en une, deux ou trois dimensions, sont les plus anciens. C'est le quadrillage par excellence et il constitue la méthode la plus naturelle à l'utilisateur pour discrétiser spatialement un problème. Son principal avantage est la simplicité de sa génération. En effet, les cellules sont toutes du même type et alignées, ce qui permet de définir un ordre de parcours pour les numéroter ; on appelle cela un maillage structuré. Un autre avantage du maillage cartésien est sa simplicité d'utilisation : aucune interpolation n'est nécessaire pour calculer les dérivés des grandeurs en présence. Les points étant alignés selon les axes, il suffit de soustraire deux valeurs voisines et de diviser par la distance. Cela permet également plus de précision et ne nécessite pas de temps de calcul particulier. Enfin, il est également possible de raffiner certaines parties plus que d'autres, comme on peut le voir par exemple dans l'image de pick-up de Kalitzin et al. [5]. Les maillages cartésiens sont plus simples à générer, nécessitent moins de mémoire pour leur stockage et requièrent significativement moins de calculs par cellule que leurs homologues conformes à la paroi [6, 7].

Le principal problème d'un maillage cartésien est qu'il est impossible de dessiner des ronds avec des carrés. Il est donc impossible de représenter précisément la moindre frontière non horizontale ou verticale. Cela peut être réglé en augmentant la

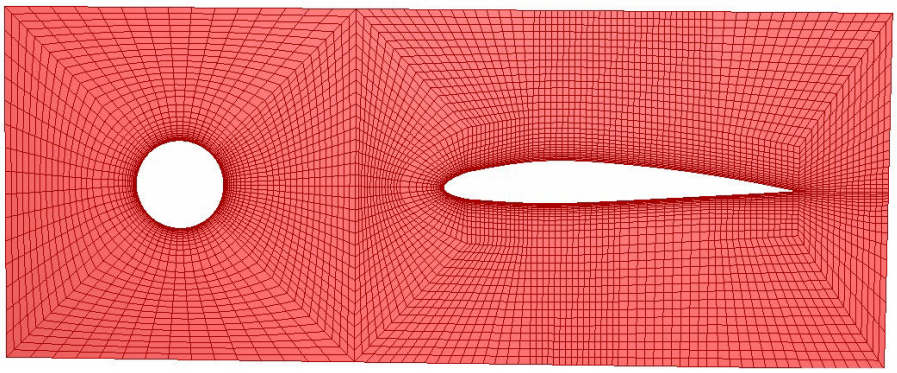


FIGURE 1.3 – Maillages conformes à base de quadrilatères extrait de Bhasker [8]

résolution du maillage, de manière à se contenter d'un bord finement crénelé. Mais si une telle solution est acceptable pour de faibles nombres de Reynolds, elle ne suffit plus à Reynolds élevé. Il est donc nécessaire d'employer des techniques comme la **méthode des frontières immergées** (IBM ou *Immersed Boundary Method*) ou la méthode **level set** pour simuler la présence des frontières à l'intérieur d'une maille cartésienne. l'IBM fait l'objet de la section suivante. Un second bémol peut être relevé : si l'on désire raffiner un maillage cartésien existant, c'est-à-dire lui ajouter des cellules par découpage de mailles existantes, il est nécessaire d'opter pour un maillage non-structuré. Il est en effet impossible d'ajouter des mailles à une grille structurée. Ce dernier point n'est pas propre aux maillages cartésien, mais leur génération en structuré est tellement simple que ce sont eux qui y perdent le plus.

1.1.2 Maillages conformes à la paroi

Il peut s'avérer essentiel de "coller à la paroi" et c'est la principale motivation d'une alternative au maillage cartésien. Les maillages conformes aux parois (*body-fitted*) ont été développés dans ce sens et sont composés de manières diverses et variées. La majorité des maillages conformes aux parois est formée de triangles. On commence par appuyer une première rangée de triangle sur les frontières, puis on remplit l'espace d'autres triangles. En trois dimensions, on imite ce procédé à l'aide de tétraèdres. Mais il existe d'autres formes de cellules possibles : hexagones, trapèzes ou autres quadrilatères, portions de couronne, polyèdres ou mailles cartésiennes déformées peuvent être utilisées. Deux exemples issus de Bhasker [8] permettent d'illustrer cette variété dans le figures 1.3 et 1.4. Les maillages conformes aux parois sont majoritairement utilisés à l'heure actuelle. Grâce à eux, il est possible de placer des nœuds du maillage de manière optimale pour résoudre les caractéristiques géométriques du problème. Ils permettent en effet de localiser très précisément les frontières immergées en y plaçant des points. Un maillage conforme à la paroi se prête aussi particulièrement bien à la gestion des couches limites puisqu'il est possible d'augmenter la densité du maillage près de la paroi pour y assurer une résolution satisfaisante de la couche limite. C'est donc la technique de maillage qui permet le plus haut niveau de personnalisation et d'adaptation. Mais ces avantages ont un coût : il est très difficile d'automatiser la tâche de génération d'un tel maillage. La complexité géométrique des problèmes industriels et la nécessité de contrôler la taille des cellules proches de la paroi pour capturer le développement de la couche limite requièrent un important et souvent long travail de la

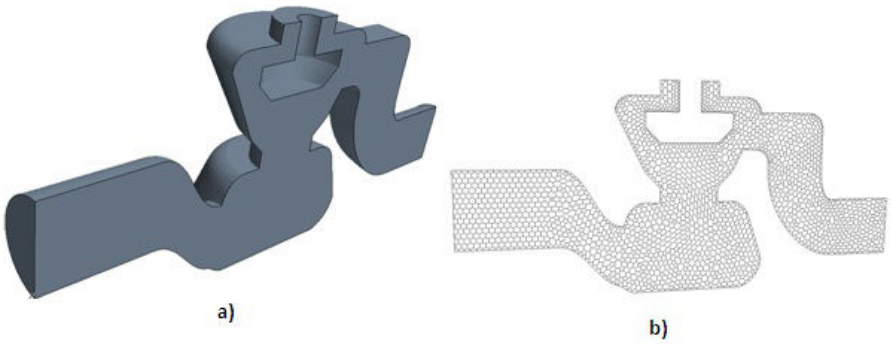


FIGURE 1.4 – Maillage conforme à base de polyèdres extrait de Bhasker [8]

part de l'utilisateur. La gestion des parois mobiles peut également s'avérer problématique. Afin que le maillage reste conforme à une telle frontière, il faudrait re-générer un nouveau maillage à chaque pas de temps, ou au moins le déformer. Cela est évidemment fastidieux et affecte énormément la durée de résolution numérique du problème. Un tel maillage irrégulier donne également lieu à des erreurs d'approximation (ou de troncature locale) causés par la propagation de distances entre les nœuds et la nécessité de recourir systématiquement à des interpolations. Cela est également plus lourd à implémenter et nécessite plus de temps de calcul par nœud que, par exemple, pour un maillage cartésien. On déplore également une perte de robustesse du calcul provoquée par l'interpolation des valeurs aux faces.

1.1.3 Maillages structurés

La structuration ou non du maillage a été évoquée tout au long des paragraphes précédents. Il est donc temps de développer davantage ce point. Un maillage structuré est un maillage qui peut être généré en répétant une maille élémentaire. Cette maille peut-être carrée, rectangulaire, triangulaire, hexagonale, cubique, parallélépipédique rectangulaire, tétraédrique, hexaédrique ou encore bien plus exotique. Si les mailles restent toutes de même taille, le maillage est produit particulièrement facilement. Il est également possible de faire évoluer la taille des mailles sans en modifier le type, comme par exemple sur la figure 1.2, afin d'avoir un maillage localement plus dense dans certaines zones choisies.

En revanche, ce type de maillage ne s'appuie qu'exceptionnellement aux parois du domaine d'étude : seules des configurations assez élémentaires permettent ce luxe. Il est donc nécessaire de ruser en utilisant ici aussi des méthodes comme celle des frontières immergées (*IBM*) ou *level set* pour compenser cette faiblesse.

La principale limitation est le manque d'adaptabilité : s'il est possible d'augmenter sensiblement la densité des mailles dans certaines zones (figure 1.2), il n'est possible de la faire que selon les axes et cela se répercute fortement sur l'ensemble de la grille. Ainsi, en privilégiant une zone bien choisie, par exemple à une abscisse et une ordonnée données, on privilégie en conséquence toute cette abscisse et toute cette ordonnée. Il va de soi que le nombre de mailles peut vite devenir très important et que la précision gagnée ne sera que partiellement pertinente. Cette méthode n'est donc pas idéale si l'on désire une extrême précision localisée.

En dernier lieu, il n'est pas possible d'ajouter ou de supprimer une maille d'un maillage structuré ; les cellules qui se trouvent dans les solides ou à l'extérieur des

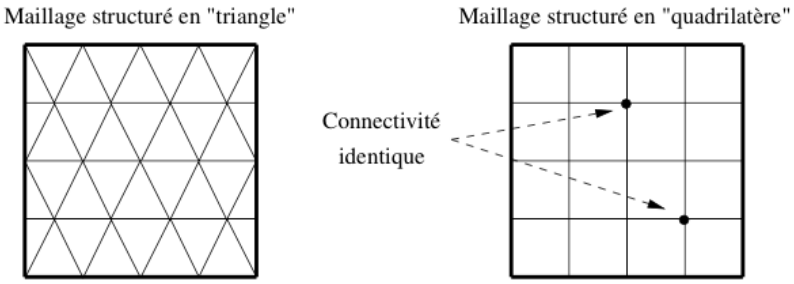


FIGURE 1.5 – Divers maillages structurés illustrés dans Rome [10] : un seul motif est répété et chaque cellule a le même nombre de voisins. Les cellules peuvent avoir des tailles variables, comme sur la figure 1.2, mais les connectivités sont toujours identiques.

géométries peuvent simplement être occultées ou ignorées par le schéma numérique, mais elles restent présentes. Dans une géométrie comme celle des poumons d'un rat (figure 3.3), ces cellules inutiles peuvent représenter bien plus que la moitié du maillage total. Lorsque la géométrie est compliquée, les méthodes basées sur des grilles structurées manquent souvent de pertinence [6, 9].

1.1.4 Maillages non-structurés

La croissante puissance de calcul nous pousse à vouloir obtenir toujours davantage de précision et donc à optimiser les maillages, notamment par une étape de raffinement. Nous avons donc besoin d'une structure de données qui nous permette d'ajouter ou d'ôter des mailles : c'est la raison d'être du maillage non-structuré. Un maillage non-structuré est un maillage au sein duquel la topologie est totalement arbitraire. On peut y assembler des triangles et/ou des quadrilatères en $2D$ et des tétraèdres, prismes, hexaèdres et/ou des pyramides en $3D$. Le principe est que chaque cellule ne connait que ses faces et que chaque face ne connait que les cellules qu'elle sépare. Un tel maillage permet donc d'ajouter ou d'ôter des cellules en n'affectant que ses faces. C'est la manière idéale de raffiner/dé-raffiner de manière localisée. Cela permet d'économiser des cellules par rapport à un maillage structuré, puisque l'on peut mélanger sans difficulté des mailles plus grandes à d'autres plus petites, voire beaucoup plus petites. On prendra tout de même soin de garder une certaine continuité dans le passage de l'une à l'autre. Cette capacité de gestion d'interfaces variées est illustrée dans la figure 1.6 issue de Rome [10].

Bien que la génération d'un tel maillage soit nettement plus complexe que celle d'un maillage structuré, elle reste l'option la plus simple à mettre en œuvre pour des géométries complexes [6, 11]. Elle est modulable à souhait et permet, selon le type de maille choisi, de s'appuyer sur les paroi (de manière à être *body-fitted*). C'est une des raisons pour lesquelles on opte souvent pour un maillage non-structuré à base de triangles.

La conception d'un maillage non-structuré nécessite un temps de développement conséquent : si l'ajout et la suppression de mailles sont parfaitement naturels, il n'en va pas de même du raffinement de cellule. A partir du moment où l'on a affaire à une modification de la connectivité, il est nécessaire de gérer les nouveaux cas topologiques apparaissant aux interfaces. Il est également nécessaire d'adapter le schéma numérique afin qu'il gère ces configurations.

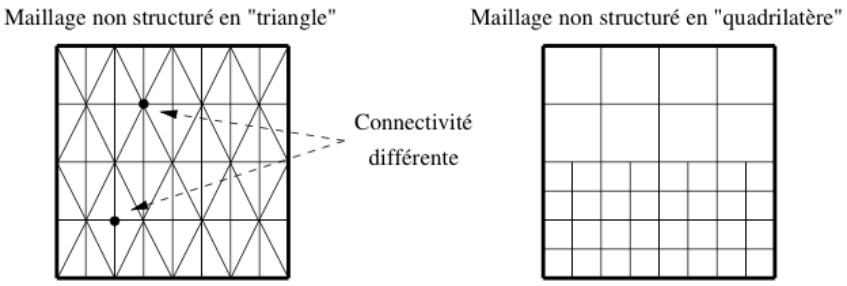


FIGURE 1.6 – Divers maillages non structurés montrant qu'il est possible d'associer des mailles différentes, ou du moins orientées différemment, ou d'avoir un nombre variable de voisins par cellule.

En outre, il est nécessaire de stocker en mémoire toute la structure de données pour chaque maille. Cela peut très vite devenir une limite infranchissable. On peut la contourner en créant différentes variantes plus légères du maillage, ou en multipliant les tests dans le code afin de retrouver certaines informations qui ont été économisées. Néanmoins, une fois les outils de raffinement mis au point, il relativement peu coûteux en temps de développement d'automatiser la personnalisation du maillage.

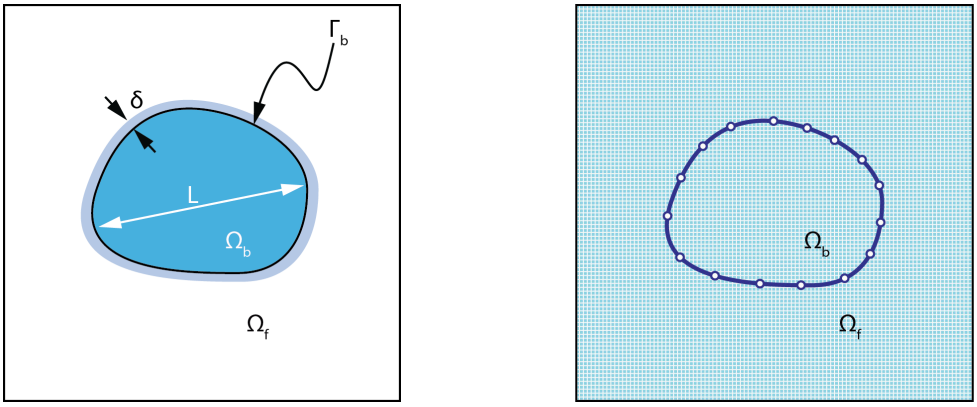
1.1.5 Conclusion

Il n'y a pas de produit miracle. On peut opter pour l'extrême accessibilité d'un maillage cartésien structuré comme pour l'extrême adaptativité d'un maillage conforme non structuré. Il y a un tel écart de mise en œuvre qu'il est essentiel de se demander si l'on a besoin de travailler en profondeur sur peu de cas, et dans ce cas d'investir beaucoup de temps à générer un maillage conforme qui sera peaufiné au fur et à mesure des expérimentations, ou sur une multitude de géométries différentes, sur lesquelles on se contentera d'un très bon maillage, obtenu plus vite, mais moins optimal.

C'est cette dernière ligne de conduite qui figure dans notre cahier des charges : nous souhaitons pouvoir multiplier les simulations sur des géométries complexes variées, y garantir une excellente précision et traiter très attentivement les couches limites. Comme les frontières étudiées peuvent s'avérer mobiles et que les schémas numériques sont pénibles à mettre en œuvre sur des mailles non cartésiennes, nous avons décidé d'utiliser des maillages cartésiens. Mais pour traiter convenablement les couches limites sans faire exploser le nombre le de mailles, nous nous privons de la simplicité du maillage structuré, afin de profiter du raffinement automatique qui n'est disponible que pour un maillage non structuré. Une conséquence agréable en est la précision du calcul des dérivés, qui se fait sans recours systématique à des interpolations.

1.2 La méthode des frontières immergées

La méthode des frontières immergées (IB) permet l'étude de géométries complexes à l'aide de maillages cartésiens en introduisant un terme de forçage dans la discrétisation, aux frontières de l'élément immergé, ou dans les équations. Elle est donc un



(a) **(b)**

FIGURE 1.7 – Un corps immergé.

(a) Schéma montrant un corps immergé dans fluide à simuler. Le corps occupe le volume Ω_b avec la frontière Γ_b. Sa longueur caractéristique est L et une couche limite d'épaisseur δ enveloppe le corps.
(b) Schéma du corps immergé dans un maillage cartésien sur lequel les équations régissant le fluide sont discrétisées.

moyen de gérer les conditions limites pour des surfaces non-alignées avec le maillage. Elle permet de gérer des géométries mobiles dans un maillage fixe et nous épargne l'utilisation de maillages mobiles.

On peut trouver dans Mittal and Iaccarino [12] ou Merlin [13] d'excellents résumés des différentes méthodes IB existantes, que je reproduis partiellement dans cette partie.

Considérons la simulation du comportement d'un fluide autour d'un corps comme sur la figure 1.7 (a). L'approche conventionnelle consiste à employer des maillages adaptés à la forme de l'objet, structurés ou non. Concevoir de tels maillages se fait en deux étapes. D'abord, on génère un maillage de surface couvrant la frontière Γ_b de l'objet. Ensuite, on l'utilise comme condition limite pour générer la grille dans le volume Ω_f occupé par le fluide. Dans le cas d'une méthode de différences finies sur un maillage structuré, la forme différentielle des équations est transformée en un système de coordonnées curvilignes alignées avec les lignes du maillage [14]. Comme le maillage est conforme à la paroi, les équations transformées peuvent être discrétisées dans le domaine de calcul avec une relative facilité. Dans le cas d'une méthode de volumes finis, l'équation sous sa forme intégrale est discrétisée et la géométrie du maillage est directement incorporée à la discrétisation. Si un maillage non structuré est employé, on peut utiliser soit la méthode des différences finies, soit celle des éléments finis. Les trois approches tiennent compte de la topologie des cellules et n'ont pas recours à des transformations de maillage.

Considérons maintenant que, comme dans la figure 1.7 (b), on décide de préférer un maillage cartésien non conforme aux paroi du corps. Dans cette approche, les bords de l'objet doivent toujours être représentés par un quelconque moyen, comme une grille de surface, mais le maillage cartésien est généré sans en tenir compte. En conséquence, la limite solide va venir couper à travers les cellules du maillage. Étant donné que le maillage n'est pas conforme au corps, l'incorporation des conditions limites va nécessiter des modifications des équations au voisinage de la frontière. Ces

modifications sont précisément l'objet de la discussion qui va suivre. Cependant, en supposant qu'une telle procédure est disponible, les équations seraient ensuite discrétisées par une technique de différences, de volumes ou d'éléments finis et n'auraient recours ni à des changements de coordonnées, ni à des opérateurs complexes de discrétisation.

À ce point, il est utile de comparer les avantages et inconvénients de ces deux approches. L'imposition des conditions limites dans la méthodes IB n'est clairement pas triviale et ses conséquences sur la précision et la conservation des propriétés du schéma numérique ne sont pas évidentes. De plus, l'alignement du maillage avec le corps permet de mieux contrôler la résolution du maillage au voisinage des frontières, ce qui permet la gestion de nombres de Reynolds plus élevés. En effet, un nombre de Reynolds plus élevé nécessite une meilleure résolution proche paroi et cela implique une augmentation de la résolution sur tout le maillage cartésien ou localement sur un maillage adapté au corps. C'est pourquoi la taille du maillage augmente beaucoup plus vite dans le cas cartésien. Cependant, remarquons que cela n'augmente pas nécessairement la complexité dans la même proportion puisqu'une partie substantielle de ces points se trouve à l'intérieur du corps et n'a pas besoin d'être calculée. De plus, le coût de calcul d'un point est inférieur en maillage cartésien du fait de l'absence de termes additionnels associés à la transformation du maillage.

L'avantage principal de la méthode IB reste la simplicité de la génération du maillage. Construire un maillage adapté au corps, structuré ou non, est en général très lourd. L'objectif est de créer un maillage ayant une résolution adéquate et comprenant le nombre minimum de points. À l'exception des cas les plus simples, ce critère contradictoire peut mener à une baisse de la qualité du maillage et donc affecter la précision et les propriétés de convergence des solveurs [14]. Ainsi, même pour des géométries simples, créer un maillage conforme de qualité peut être un processus itératif nécessitant un nombre significatif d'entrées de la part de la personne chargée de la génération du maillage. Plus la géométrie se complexifie, plus la difficulté de ce travail augmente. Dans l'approche structurée sur une géométrie complexe, il est commun de décomposer le domaine en sous-domaines et d'y générer des maillages séparés [15]. En plus de la complexité introduite dans l'algorithme, due à la présence de multiple sous-domaines, la régularité du maillage peut être affectée, notamment à l'interface entre les sous-domaines. L'approche non structurée est intrinsèquement mieux adaptée aux géométries compliquées, mais là aussi, la qualité du maillage a tendance à diminuer lorsque la géométrie se complexifie. A l'inverse, pour une simulation effectuée sur un maillage cartésien non conforme, ni la qualité, ni la complexité du maillage ne sont affectées significativement par la complexité de la géométrie.

L'avantage de la méthode IB sur un maillage cartésien est également particulièrement évident dans le cas d'écoulements à frontières mobiles. Une telle simulation sur un maillage conforme nécessiterait en effet une mise à jour du maillage à chaque pas de temps, ainsi qu'une procédure de projection de la solution sur le nouveau maillage [16] ou bien une prise en compte de la vitesse du maillage dans les termes advectifs. Ces deux étapes peuvent impacter négativement la simplicité, la précision, la robustesse et le temps de calcul de la solution, en particulier dans les cas impliquant des mouvements importants. A l'opposé, l'intégration d'un corps mobiles dans le domaine via la méthode IB est relativement simple, puisqu'elle se base sur un maillage cartésien fixe qui ne se déforme pas. Aussi, malgré les importants progrès faits dans la simulation d'écoulement à frontières mobiles sur des maillages cartésiens [16, 17, 18], la méthode IB représente une alternative très attractive pour ce type d'écoulement, du fait de sa simplicité.

1.2.1 Imposition des conditions limites

L'imposition des conditions limites sur les frontières immergées est la clé dans le développement d'une méthode IB. C'est également ce qui distingue une méthode d'une autre. Considérons la simulation d'un écoulement incompressible passant autour du corps de la figure 1.7(a), qui est régi par les équations de Navier-Stokes incompressibles :

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u}\cdot\nabla\boldsymbol{u} + \frac{1}{\rho}\nabla p - \frac{\mu}{\rho}\nabla^2\boldsymbol{u} = 0 \qquad \text{et} \tag{1.1}$$

$$\nabla\cdot\boldsymbol{u} = 0 \qquad \text{dans } \Omega_f$$

avec

$$\boldsymbol{u} = \boldsymbol{u}_\Gamma \qquad \text{dans } \Gamma_b, \tag{1.2}$$

où $\boldsymbol{u}$ est la vitesse du fluide, p est la pression, et ρ et μ en sont respectivement la masse volumique et la viscosité. Le solide occupe le volume Ω_b, avec une frontière notée Γ_b et le domaine fluide les entourant est noté Ω_f. Le bord extérieur du domaine de calcul est passé sous silence dans cette discussion. Afin d'alléger les notations, on couple les équations des moments et de continuité en

$$\mathcal{L}(\underline{U}) = 0 \quad \text{dans} \quad \Omega_f \tag{1.3}$$

$$\text{avec} \quad \underline{U} = \underline{U}_\Gamma \quad \text{sur} \quad \Gamma_b \tag{1.4}$$

où $\underline{U} = (\boldsymbol{u}, p)$ et $\mathcal{L}$ est l'opérateur représentant le système d'équation de Navier-Stokes comme dans l'équation 1.1. Notons que dans le contexte du système de Navier-Stokes incompressible, la pression est déterminée par la contrainte de continuité et que par conséquent, l'équation de continuité est considérée comme une équation implicite pour la pression.

Les méthodes conventionnelles consistent à discrétiser l'équation 1.3 sur un maillage adapté à la forme de l'objet dans lequel l'équation 1.4 est implémentée directement. Dans la méthode IB, l'équation 1.1 est discrétisée sur un maillage cartésien non adapté aux frontières du corps et les conditions limites sont imposées indirectement via une modification de l'équation 1.3. En général, cette modification prend la forme d'une fonction de forçage (ou terme source) dans les équations qui reproduit l'effet d'une frontière. L'introduction d'une telle fonction de forçage, dont la nature précise fait l'objet de la discussion des paragraphes suivants, peut être implémentée de deux manières différentes, ce qui mène à une scission fondamentale entre les méthodes IB. Dans la première implémentation, la fonction de forçage, notée $\underline{f}_b$, est insérée dans l'équation de continuité 1.3, menant à l'équation $\mathcal{L}(\underline{U}) = \underline{f}_b$, qui s'applique alors à tout le domaine $(\Omega_f + \Omega_b)$. Notons que $\underline{f}_b = (\boldsymbol{f}_m, f_p)$, où $\boldsymbol{f}_m$ est f_p sont les fonctions de forçage appliquées respectivement aux moments et à la pression. Cette équation est discrétisée ensuite sur un maillage cartésien, ce qui mène au système d'équations discrètes

suivant :

$$[L]\{\underline{U}\} = \{\underline{f}_b\}, \tag{1.5}$$

qui est résolu sur la totalité du domaine.

Dans la seconde approche, l'équation est d'abord discrétisée sur le maillage cartésien sans préoccupation des frontières immergées, ce qui mène au système d'équations discrétisé $[L]\{\underline{U}\} = 0$. La discrétisation dans les cellules proches de la frontière immergée est ensuite ajustée pour tenir compte de sa présence. Il en résulte le système d'équations modifié $[L']\{\underline{U}\} = \{\underline{r}\}$ qui est résolu sur le maillage cartésien. Dans cette équation, $[L']$ est l'opérateur discret modifié et $\{\underline{r}\}$ représente les termes connus associés aux conditions limites au niveau de la surface immergée. Ce précédent système peut ainsi être réécrit

$$[L]\{\underline{U}\} = \{\underline{f}'_b\}, \tag{1.6}$$

où $\{\underline{f}'_b\} = \{\underline{r}\} + [L]\{\underline{U}\} - [L']\{\underline{U}\}$. La comparaison les équations 1.5 et 1.6 met en évidence le lien entre ces deux méthodes. Dans la première approche, appelée "forçage continu", le terme de forçage est introduit avant la discrétisation des équations, alors que dans la seconde, appelée "forçage discret", il est introduit après la discrétisation des équations. Une caractéristique attractive du forçage continu est d'être formulé indépendamment de la discrétisation spatiale sous-jacente. A l'opposé, le forçage discret dépend très fortement de la méthode de discrétisation utilisée. Cependant, cela permet un contrôle direct de la précision et de la stabilité numérique, ainsi que des propriétés de conservation du solveur. Des méthodes qui font parties de chacune de ces deux catégories sont décrites dans les paragraphes suivants.

1.2.2 Le forçage continu

Les parois élastiques et rigides nécessitant des traitements différents avec les méthodes de cette catégorie, elles seront traitées séparément dans cette section.

Écoulements avec parois élastiques

La méthode IB originale de Peskin [19, 20] a été développée pour une simulation couplée de flux sanguin et de contraction musculaire dans le cœur et est généralement convenable pour traiter des écoulements avec parois élastiques. Dans cette méthode, l'écoulement du fluide régi par le système d'équation de Navier-Stokes incompressible est résolu sur un maillage cartésien stationnaire. La frontière immergée est représentée par un ensemble de fibres élastiques dont les positions sont suivies de manière lagrangienne par un certain nombre de points sans masse se déplaçant à la vitesse locale du fluide. Ainsi, les coordonnées $\boldsymbol{X}_k$ du $k^{\text{ème}}$ point lagrangien sont commandées par l'équation

$$\frac{\partial \boldsymbol{X}_k}{\partial t} = \boldsymbol{u}(\boldsymbol{X}_k,\, t). \tag{1.7}$$

La déformation du solide est liée à la force (notée $\boldsymbol{F}$) via la loi de Hooke. L'effet des frontières immergées sur le fluide alentour est essentiellement capté par un terme de forçage localisé dans l'équation des moments, qui est donné par

$$\boldsymbol{f}_m(\boldsymbol{x},\ t) = \sum_k \boldsymbol{F}_k(t)\delta(|\boldsymbol{x} - \boldsymbol{X}_k|), \tag{1.8}$$

où δ est la fonction de Dirac. Comme les fibres ne sont en général pas situées sur des

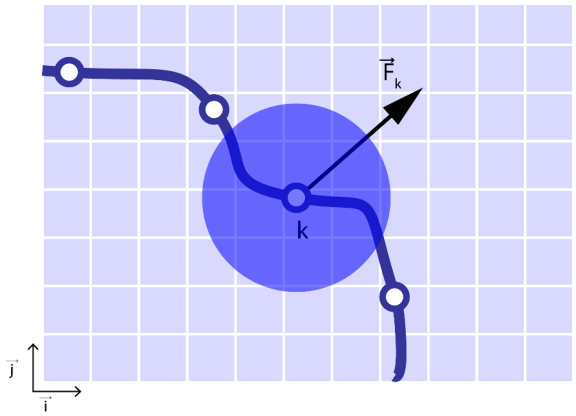


FIGURE 1.8 – Zone de forçage. Image reproduite d'après Mittal and Iaccarino [12]

Transfert du terme de forçage $\boldsymbol{F}_k$ depuis le point lagrangien vers les points du fluides alentours. La zone ombrée montre l'extension de la distribution de la force.

nœuds du maillage cartésien, le forçage est distribué sur une bande de cellules autour de chaque point lagrangien (voir figure 1.8) et cette force distribuée est imposée dans l'équation des moments des nœuds alentours. Ainsi, la fonction delta est remplacée par une fonction de distribution continue, notée d, plus convenable pour un espace discret. Le terme de forçage devient donc, en un point $\boldsymbol{x}_{i,j}$,

$$\boldsymbol{f}_m(\boldsymbol{x}_{i,j},\ t) = \sum_k \boldsymbol{F}_k(t)d(|\boldsymbol{x}_{i,j} - \boldsymbol{X}_k|). \tag{1.9}$$

La vitesse de la fibre dans l'équation 1.7 est également obtenue en utilisant la même fonction lissée. Le choix de la fonction de distribution d est un ingrédient clé dans cette méthode. Plusieurs fonctions différentes ont déjà été utilisées par le passé [19, 21, 22, 23]. Pour un problème unidimensionnel simple, Beyer and Leveque [24] ont montré qu'il est possible de construire une fonction de distribution qui préserve la précision de la discrétisation spatiale. Les méthodes de cette catégorie ont été utilisées avec succès pour une large variété de problèmes, dont la mécanique cardiaque [20], la dynamique cochléaire [21], la locomotion aquatique animale [25], la dynamique des bulles [26] et l'écoulement à travers des filaments flexibles [27].

Écoulements avec parois rigides

La méthode précédente est naturellement bien adaptée pour des corps élastiques, mais son application aux écoulements avec des corps rigides pose des problèmes parce

que les lois de comportement utilisées pour les parois élastiques ne sont généralement pas bien posées dans les parois rigides. Ce problème peut être contourné en considérant le corps comme élastique, mais extrêmement raide. Une seconde approche consiste à considérer que la structure est attachée à une position d'équilibre [22, 24] par un ressort dont la force $\boldsymbol{F}$ est donnée par

$$\boldsymbol{F}_k(t) = -\kappa(\boldsymbol{X}_k - \boldsymbol{X}_k^e(t)) \tag{1.10}$$

où κ est un coefficient de raideur positif et $\boldsymbol{X}_k^e$ est la position d'équilibre du $k^{\text{ème}}$ point lagrangien. L'imposition précise de la condition limite sur la frontière immergée requiert de grandes valeurs de κ. Cependant, cela se traduit par un système rigide d'équations qui est soumis à de graves contraintes de stabilité [22, 28]. D'autre part, de plus petites valeurs de κ peuvent mener à des effets élastiques parasites tels que la déformation excessive de la position d'équilibre, comme dans les simulations du sillage d'un cube à faible Reynolds dans Lai and Peskin [22].

Cette dernière approche peut être vue comme un cas particulier du modèle développé par Goldstein et al. [29] pour simuler les écoulements autour de corps rigides. Dans ce modèle, l'effet de la paroi rigide sur le flux est modélisé par le terme de forçage

$$\boldsymbol{F}(t) = \alpha \int_0^t \boldsymbol{u}(\tau)d\tau + \beta \boldsymbol{u}(t), \tag{1.11}$$

où les coefficients α et β sont choisis de manière à imposer le mieux possible les conditions limites aux frontières du corps immergé. L'intention de départ de l'équation 1.11 était de fournir un contrôle rétroactif sur la vitesse près de la surface [29], mais d'un point de vue physique, il peut aussi représenter un oscillateur amorti [30]. Cette méthode a été utilisée pour simuler le démarrage d'un écoulement autour d'un cylindre à Reynolds modéré et l'écoulement turbulent à Reynolds faible au sein d'un canal à rainures [29]. Les résultats sont généralement prometteurs pour de faibles nombres de Reynolds, mais l'imposition précise des conditions limites nécessite de grandes valeurs de α et β qui peuvent conduire à des problèmes de stabilité, notamment pour des écoulements hautement turbulents.

Une autre méthode de ce type a été développée par Angot et al. [31] et Khadra et al. [32]. Elle consiste à considérer un écoulement dans un milieu poreux et de ce fait régi par l'équation de Navier-Stokes/Brinkman [33]. Elle a été utilisée pour simuler l'écoulement autour d'un cylindre pour des nombres de Reynolds allant jusqu'à 200 et pour un écoulement descendant une marche [32].

Conclusions

Le forçage continu est une manière très attractive de gérer les écoulements autour de corps élastiques immergés. Pour ces écoulements, cette méthode proche de la physique est facile à implémenter. C'est pour cette raison que l'on la retrouve souvent en biologie [20, 21, 25] et en écoulements multiphasiques [26], où les parois élastiques abondent. Cependant, son usage pour les parois rigides pose davantage de problèmes, qui sont liés au fait que les termes de forçage ne se comportent pas bien à la limite rigide. Ils sont essentiellement résolus en utilisant des modèles simplifiés pour imiter l'effet de la paroi rigide sur l'écoulement. Malgré tout, ces paramètres ont des consé-

quences numériques négatives, tant sur la précision que sur la stabilité. La fonction de lissage inhérente à cette technique nuit également à la netteté des parois rigides. Cela peut s'avérer particulièrement désagréable pour les écoulements à Reynolds élevés. Enfin, le forçage continu nécessite de résoudre les équations de Navier-Stokes à l'intérieur du corps immergé. Lorsque le nombre Reynolds s'élève, le maillage doit se densifier et cela augmente la quantité de points à résoudre au sein du corps. L'obligation de résoudre les équations à l'intérieur du solide peut être une épée de Damoclès.

1.2.3 Le forçage discret

Les méthodes de forçage discret se divisent en deux catégories : celles qui imposent les conditions limites (BC pour "boundary conditions") au flux de manière indirecte et celles qui les imposent directement sur les parois.

Imposition indirecte des conditions limites

Il arrive que, dans certains problèmes intégrables unidimensionnels, on puisse directement dériver formellement le terme de forçage qui impose une condition spécifique sur une paroi à l'intérieur du domaine de calcul [21]. Mais cela n'est en général pas applicable aux équations de Navier-Stokes parce qu'elles ne peuvent pas être intégrées de manière analytique pour déterminer la fonction de forçage. En conséquence, toutes les méthode du paragraphe précédent utilisent des modèles simplifiés pour implémenter cette fonction de forçage. Pour éviter cela, Mohd-Yusof [34], Verzicco et al. [35] ont développé une méthode qui extrait le terme de forçage directement de la solution numérique, pour laquelle une valeur *a priori* peut être déterminée. En partant de la discrétisation de Navier-Stokes, sans aucune modification due à la présence des parois immergées, et en utilisant les même notations que dans la section 1.2.2, le système $[L]\{\underline{U}^*\} = 0$, où $\{\underline{U}^*\}$ représente une prédiction du champ de vitesse, est résolu à chaque pas de temps. Le terme de forçage $\{\underline{f'}_b\}$ de l'équation 1.6 est alors défini par

$$\{\underline{f'}_b\} \approx \{\underline{r}\} + [L]\{\underline{U}^*\} = \{\underline{r}\} - [L']\{\underline{U}^*\} \tag{1.12}$$

où $\{\underline{r}\} = \{\underline{U}_\Gamma - \underline{U}^*\}d(|\boldsymbol{X}_k - \boldsymbol{x}_{i,j}|)$ et $[L'] = [L] + ([I] - [L])\delta(|\boldsymbol{X}_k - \boldsymbol{x}_{i,j}|)$, $[I]$ étant la matrice identité. Comme précédemment, la fonction de Dirac delta est remplacée par une distribution lisse d et l'équation 1.6 pour cette méthode devient

$$[L]\{\underline{U}\} = \{\underline{U}_\Gamma - \underline{U}^*\}d(|\boldsymbol{X}_k - \boldsymbol{x}_{i,j}|) + [L]\{\underline{U}j\}d(|\boldsymbol{X}_k - \boldsymbol{x}_{i,j}|) \tag{1.13}$$

et cela représente formellement l'imposition de la condition limite au point lagrangien $\boldsymbol{X}_k$ de la surface immergée.

L'avantage majeur de l'approche discrète est l'absence de paramètres propres à l'utilisateur et des contraintes de stabilité numériques qui vont avec. Cependant, le forçage s'étend toujours sur une région fluide à cause de la fonction de distribution et les détails de l'implémentation dépendent fortement du schéma numérique utilisé pour discrétiser les équations du fluide. Cette technique a été appliquée à plusieurs problèmes, y compris un écoulement turbulent à l'intérieur d'un moteur à combustion interne [35] et des écoulements autour de corps escarpés en 2D [36] et en 3D [37]

et dans un réservoir cylindrique sous agitation [38]

Imposition directe des conditions limites

Bien que l'application des méthodes IB aux faibles et moyens nombres de Reynolds a été une réussite, leur extension aux nombres de Reynolds plus élevés est un défi, à cause du surcroît de précision requis dans les cellules proches des parois. Dans ces cellules, il est primordial d'être précis et l'effet dispersant de la fonction de distribution devient très indésirable. Pour cette raison, il existe d'autres approches, où les frontières sont clairement marquées, qui ne font pas intervenir de dispersion et qui sont davantage focalisées sur la précision proche des parois. Cela est habituellement fait en modifiant l'incorporation des frontières immergées de manière à leur imposer directement la condition limite. Deux méthodes procédant ainsi sont décrites ici.

Cellule fantôme en différences finies ou volumes finis Le principe de la méthode de la cellule fantôme (GCM ou *Ghost-Cell Method*) est de remplacer le terme de forçage dans les équations de Navier-Stokes par une interpolation [39, 40]. On l'appelle également méthode de reconstruction (*reconstruction method*) ou d'interpolation [6, 41, 42, 43]. Une formule d'interpolation remplace les équations de transport à proximité de la frontière immergée et a pour but de reconstruire localement le champ de vitesse proche à la paroi immergée. Les paramètres de l'écoulement sont uniquement déterminés dans la partie fluide. La partie solide ne nécessite aucun traitement particulier, ce qui permet d'économiser du temps de calcul CPU et évite l'utilisation de descriptions non physiques (telles que celles intrinsèques aux méthodes de forçages continus). Les cellules fantômes sont définies comme des cellules dans le solide qui ont au moins un voisin dans le fluide. Par exemple, la cellule G sur la figure 1.9 est une cellule fantôme. L'étape clé consiste à interpoler la valeur ϕ_G d'une variable générique ϕ du flux au point fantôme G. Plusieurs méthodes existent pour y parvenir ; elles sont choisies en corrélation avec le type de maillage utilisé et différent par leur schéma ou leur ordre. Elles peuvent par exemple être linéaires (Ferzinger and Perić [14]), bilinéaires, linéaire dans une direction et quadratique dans l'autre (Majumdar et al. [39]), trilinéaires, quadratiques [38, 39, 44]...

Quelque soit la méthode d'interpolation utilisée, la valeur de la variable ϕ au nœud fantôme G peut s'exprimer comme

$$\phi_G = \sum \omega_i \phi_i, \tag{1.14}$$

où la sommation s'étend sur tous les points nécessaires à l'interpolation, incluant un ou plusieurs autres points fantômes et les ω_i sont des coefficients dépendant de la géométrie.L'équation ci-dessus représente l'équation 1.6 pour les cellules fantômes et peut maintenant être résolue directement simultanément avec l'équation de Navier-Stokes discrétisée pour les points fluides. Cette méthode a été utilisée pour résoudre une large variété d'écoulements, dont un écoulement compressible autour d'un cylindre et d'une aile [44] à des nombres de Reynolds allant jusqu'à $O(10^5)$, de la propulsion aquatique [12], un écoulement dans un passage serpenté [30] et un écoulement turbulent autour d'un pick-up [5].

Cellule coupée en volumes finis Aucune des méthodes IB présentées jusqu'à présent ne satisfait les lois de conservations sous-jacentes pour les cellules voisines de la

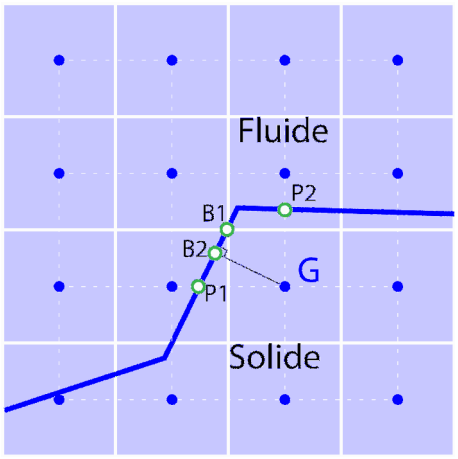


FIGURE 1.9 – Représentation des points aux environs d'une surface immergée pour l'approche par cellules fantômes. G est le point fantôme et B_i et P_i sont les points où la condition limite peut être imposée. B_1 est sur la paroi, équidistant de P_1 et P_2 et B_2 est le projeté orthogonal de G sur la paroi. Image reproduite d'après Mittal and Iaccarino [12]

frontière immergée. La conservation stricte de la masse et du moment, au niveau local et global, ne peut se faire que via une méthode des volumes finis. C'est justement la raison principale de la méthode de la cellule coupée [6, 45]. Dans cette méthode, introduite pour des maillages cartésiens, les cellules traversées par la frontière immergée sont repérées et les intersections de leurs faces avec la frontière sont déterminées. Ensuite, ces cellules coupées dont le centre appartient au fluide sont remodelées en leur ôtant la portion appartenant au solide. Les cellules dont le centre est solide sont absorbées par les cellules environnantes. On forme ainsi une limite basée sur des trapèzes [46] comme dans la figure 1.10(a). Des configurations plus précises et donc plus complexes à mettre en œuvre sont également envisageables.

La discrétisation des équations de Navier-Stokes en volumes finis nécessite une estimation sur les faces de chaque cellules de la masse, des flux diffusifs et convectifs et du gradient de pression. La difficulté est de les évaluer sur les faces des cellules trapézoïdales. L'approche proposée par Ye et al. [46] est d'exprimer une variable de flux ϕ par une fonction d'interpolation polynomiale de degré 2 dans une région appropriée et d'évaluer les flux f en se basant sur cette fonction. Par l'exemple, pour approximer le flux sur la face sud-ouest, f_{sw} de la figure 1.10(b), ϕ est exprimée en termes d'une fonction linéaire en x et quadratique en y :

$$\phi = C_1 x y^2 + C_2 y^2 + C_3 x y + C_4 x + C_5 y + C_6, \tag{1.15}$$

où C_1 à C_6 sont six coefficients inconnus qui sont exprimés en fonction des valeurs de ϕ aux six points encadrés en rouges dans la figure 1.10(b) et une expression similaire à 1.14 est développée pour f_{sw}. On procède de manière analogue pour calculer les flux f_e et f_i. Ce schéma de discrétisation est précis à l'ordre 2 et conserve la masse et la quantité de mouvement de manière exacte quelque soit la résolution de la grille.

Cette méthode a surtout été utilisée en 2D pour des problèmes variés tels que des écoulements avec des frontières fixes ou mobiles [47], des feuilles battantes [48], des objets en chute libre dans un fluide [49] et des jets dirigés par une membrane [50]. L'extension de cette méthode à la 3D n'est pas triviale ; elle nécessite un découpage des

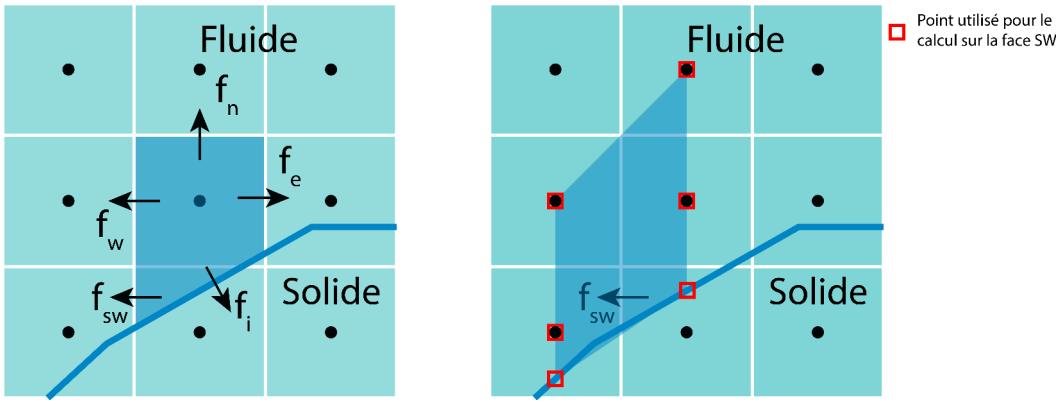


FIGURE 1.10 – Schéma de principe de la méthode de la cellule coupée.

(a) Volume fini trapézoïdal formé près de la frontière immergée pour laquelle f représente le flux sur une face d'une variable physique générique. **(b)** Région d'interpolation est points utilisés pour approximer le flux f_{sw} sur la face sud-ouest du trapèze. Image reproduite d'après Mittal and Iaccarino [12]

cellules en polyèdres complexes et la discrétisation de l'équation de Navier-Stokes sur de telles cellules serait compliquée. Cela reviendrait quasiment à convertir un maillage cartésien en maillage conforme, à l'image de ce qui est fait dans Berger and Aftosmis [51].

Conclusions

Les méthodes vues dans ce paragraphes introduisent les conditions aux limites directement dans les équations discrétisées. Le forçage discret est donc intimement lié aux détails de la méthode de discrétisation et son implémentation n'est pas aussi facile que pour le forçage continu. Cependant, il permet une représentation nette de la surface immergée, ce qui est particulièrement apprécié à Reynolds élevé. En outre, le forçage discret n'introduit pas de contraintes de stabilités supplémentaires dans la représentation de solides immergés. Enfin, cette approche découple les équations pour les nœuds fluides de celles des nœuds solides, évitant ainsi la résolution des équations à l'intérieur des corps immergés. C'est un avantage énorme lorsque le nombre de Reynold est grand. En revanche, l'inclusion de frontières mobiles n'est pas aisée avec une méthode de forçage discret. Souvent, il est nécessaire d'imposer une condition limite de pression sur la surface immergée, comme par exemple dans Udaykumar et al. [52], tandis cela est inutile lors de l'utilisation de méthodes de forçage continus.

1.3 Raffinement de maillage cartésien

Les techniques de raffinement de maillage ont été introduites par Berger and Oliger [53], Beyer and Leveque [54], Powell et al. [55] pour raffiner des maillages cartésiens. Cette section a pour but de faire un tour d'horizon des diverses manières de raffiner un maillage cartésien non structuré en vue d'une utilisation en parallèle. Bien que les maillages cartésiens datent des années 1970, il ne sont vraiment utilisables que depuis l'adjonction d'une option de raffinement adaptatif (AMR pour *Adaptative Mesh Refinement*) [6, 53, 56]. Ces maillages sont très utilisés en CFD (*Computional Fluid Dynamics*), notamment parce qu'il est plus simple d'y mettre en place des schémas numériques d'ordres élevés que sur des grilles qui ne sont pas à base de rectangles ou de pavés. La subdivision des cellules peut sembler triviale, mais génère de nombreux cas particuliers qu'il convient de traiter soigneusement. La figure 1.11 illustre un raffinement typique, pour calculer l'écoulement autour d'un F16XL [57]

1.3.1 Principe

L'idée de base est de partir d'un maillage cartésien carré ou cubique qui soit suffisamment grand pour que la géométrie à étudier puisse y être plongée et relativement grossier, puis de supprimer toutes les cellules qui sont à l'extérieur de cette géométrie. On raffine ensuite une première fois les cellules pour lesquelles le procédé est pertinent en les divisant, *a priori* en quatre ou en huit, selon que qu'il s'agisse d'un problème en deux ou en trois dimensions. Ces nouvelles cellules sont mémorisées dans un arbre comme étant les fils de la cellule divisée. On recommence ensuite le processus de suppression/raffinement jusqu'à obtenir la précision de maillage souhaitée, ou plutôt jusqu'à atteindre les limites de mémoire ou de capacité de calcul. Cela permet d'obtenir le maillage de départ sur lequel on commence à lancer le calcul. On peut ensuite diviser le maillage en autant de parties que l'on a de processeurs à disposition, afin de résoudre le problème en parallèle. Si l'on a affaire à des géométries mobiles, il est possible de définir un pas de temps de mise à jour du maillage. Lors d'une telle mise à jour, il est alors possible de **dé-raffiner**, c'est-à-dire ré-assembler des cellules filles pour reconstruire le parent, ou de raffiner soit davantage aux mêmes endroits, soit ailleurs. Ceci n'est pas sans poser quelques difficultés. Tout d'abord, il est nécessaire d'interpoler la solution depuis l'ancien maillage vers le nouveau, mais surtout, si l'on travaille en parallèle, cela va entraîner un déséquilibre des nœuds de calculs et donc du temps de calcul perdu entre la fin des calculs des nœuds les moins et les plus chargés. De plus, ajouter des cellules localement sur un processeur peut affecter les processeurs voisins au niveau des interfaces. Cela augmente considérablement la durée des communications entre les processeurs. Ces communications sont le goulot d'étranglement de cette méthode.

1.3.2 Critère de raffinement

Le but du raffinement est de pouvoir représenter finement ce qu'il se passe aux endroits les plus sensibles, c'est-à-dire ceux où les gradients sont les plus élevés [58]. Avant de lancer le calcul sur un maillage, ces endroits ne sont pas connus avec certitude. C'est pourquoi il est nécessaire de les anticiper. Dans un premier temps, il convient de raffiner le maillage aux abords des objets immergés et des parois, de manière à être capable de les décrire le plus précisément possible. Ensuite, à l'aide, par exemple,

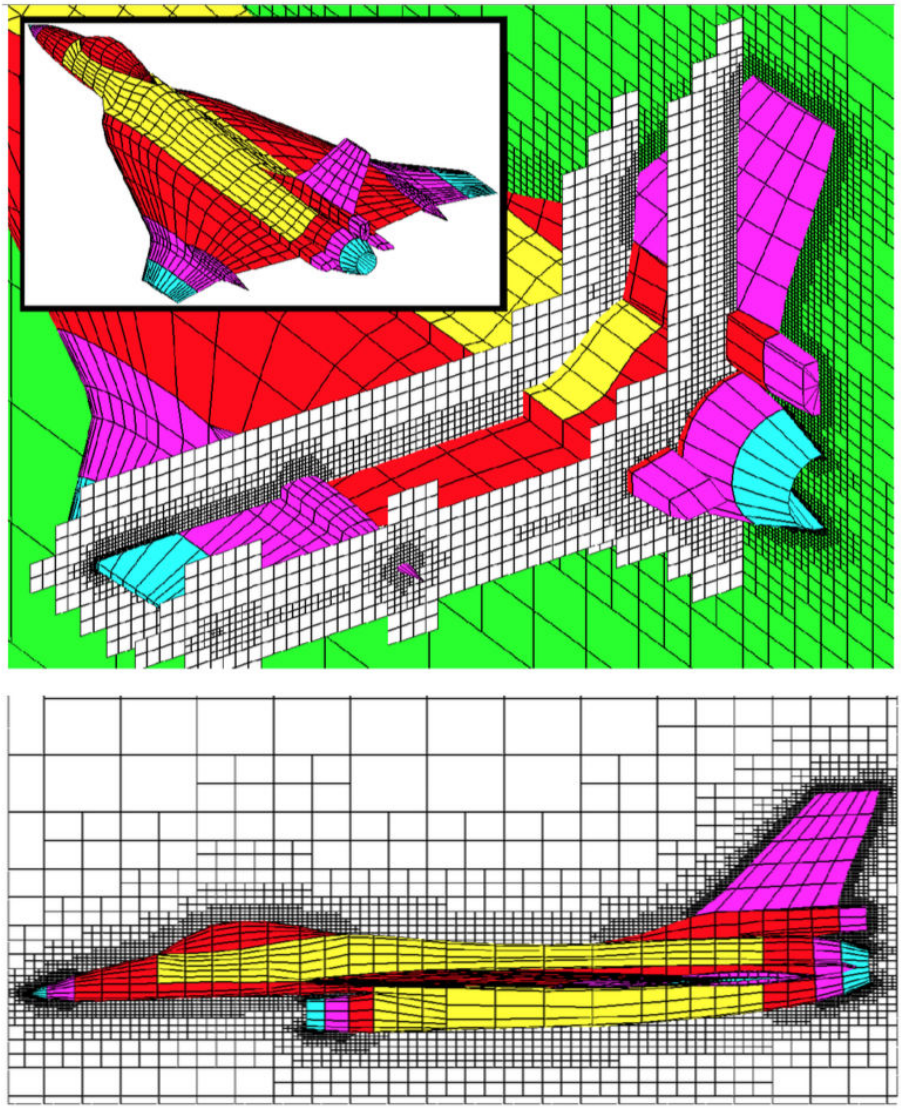


FIGURE 1.11 – Maillage cartésien raffiné autour d'un F16XL issu de [57]

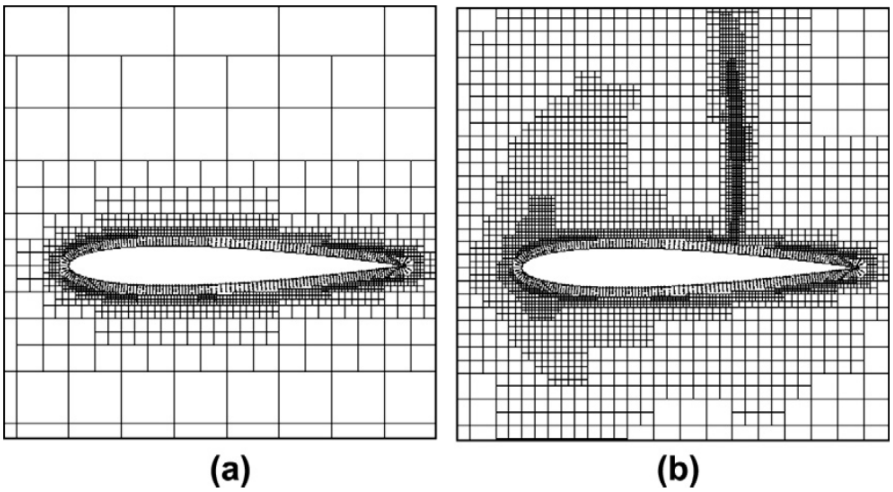


FIGURE 1.12 – Maillages initial et après quatre niveaux de raffinements d'une aile NACA 0012 issu de Jahangirian and Hashemi [6]

d'anciennes simulations moins précises, on choisit les zones à raffiner davantage. Des exemples de maillages cartésiens raffinés issus de Jahangirian and Hashemi [6] sont donnés aux figures 1.12 et 1.14a. On peut constater l'augmentation de la précision sur le profil des des pressions entre les maillages de la figure 1.12 sur la figure 1.13 L'article de Antepara et al. [59] détaille son processus de sélection des cellules à raffiner. Il repose sur le concept des multi-échelles variationnelles (*VMS* pour *variational multi-scale*) introduit dans Hughes et al. [60]. Ce concept est basé sur une compréhension physique de l'écoulement afin d'identifier les zones critiques du problème. Pour déterminer les mailles à raffiner ou dé-raffiner, on applique à la vitesse un filtre qui a pour but d'en lisser les valeurs sur le maillage.On l'appelle la vitesse filtrée et on la note $.\bar{u}$. On calcule ensuite la vitesse résiduelle u' :

$$u' = u - \bar{u} \tag{1.16}$$

où u représente la vitesse instantanée. La mesure dont on se sert comme critère de raffinement est

$$\phi_c = \|u'\| \tag{1.17}$$

dont le maximum

$$\phi_{\max^{n_0}} = \max[\phi_c] \tag{1.18}$$

est la valeur maximale du critère de raffinement. On en calcule la moyenne temporelle afin d'obtenir un maillage final dont la montée en degré de raffinement est lisse :

$$\phi_{\max^n} = [\phi_{\max^n}(t) + \phi_{\max^n}(\Delta t)]\frac{1}{t+\Delta t}. \tag{1.19}$$

On en calcule ensuite la valeur moyenne sur les cellules où la valeur est supérieure à 10% de cette dernière valeur. Cette opération permet de ne pas tenir compte des

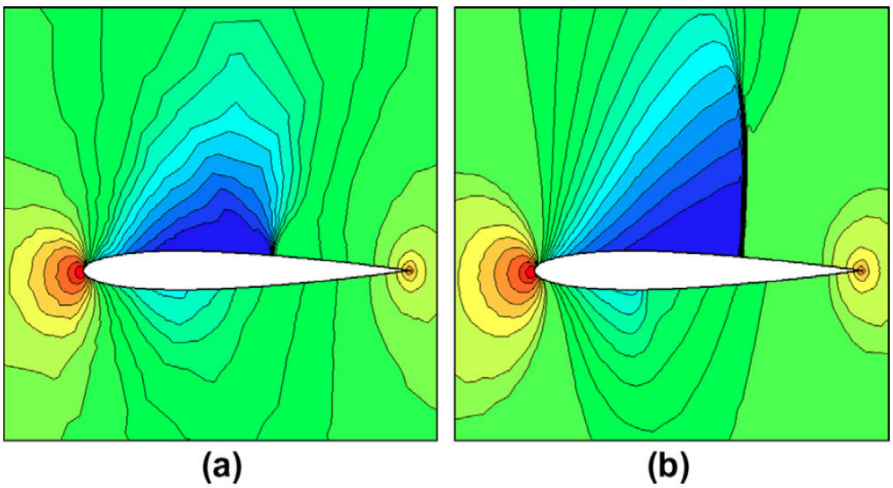


FIGURE 1.13 – Profils des pressions autour d'une aile NACA 0012 avec les maillages de la figure 1.12 issu de Jahangirian and Hashemi [6]

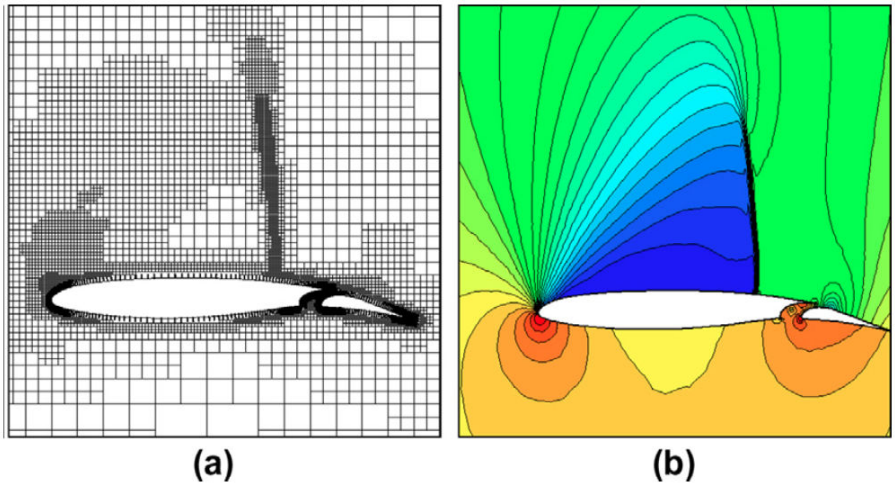


FIGURE 1.14 – Maillage adapté et profil des pressions autour d'une aile SKF1 tiré de Jahangirian and Hashemi [6]

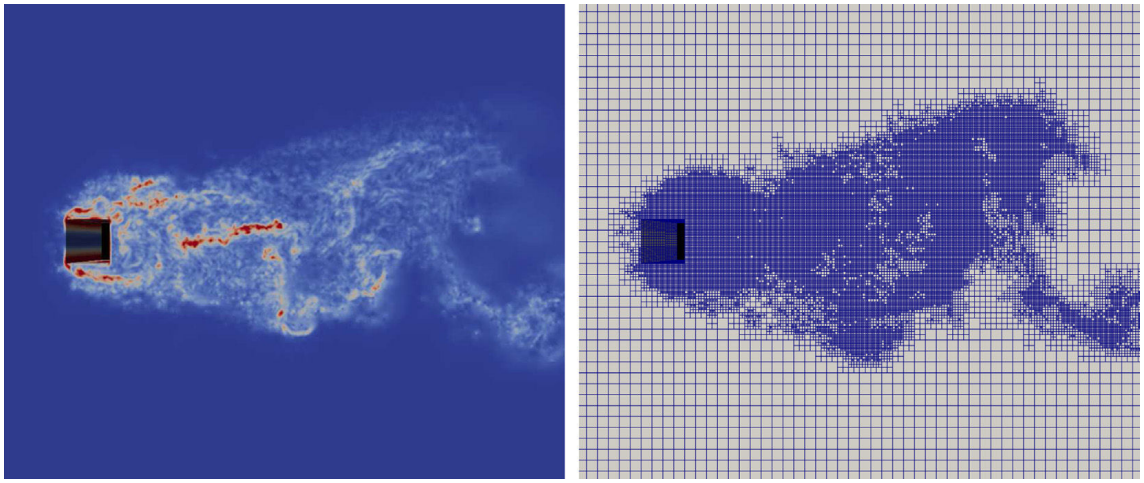


FIGURE 1.15 – Raffinement basé sur la vitesse résiduelle tiré de Antepara et al. [59] : champ de vorticité autour d'un barreau à section carrée à Re = 22000

cellules où la vitesse résiduelle est proche de zéro.

$$\phi_{\text{avg}^n} = \frac{\sum_{\text{cellules}} f(\phi_c) \cdot \phi_c}{\sum_{\text{cellules}} f(\phi_c)} \tag{1.20}$$

où $f(\phi_c)$ est donné par

$$f(\phi_c) = \begin{cases} 1 & \text{si } \phi_c \geq 0.1\phi_{\text{max}^n} \\ 0 & \text{si } \phi_c < 0.1\phi_{\text{max}^n} \end{cases} \tag{1.21}$$

Le seuil peut ainsi être défini :

$$\epsilon_c = \frac{\phi_{\text{avg}^n}}{\phi_c}. \tag{1.22}$$

Avec ce paramètre, les cellules destinées à être raffinées sont celles avec $\epsilon_c \geq 2$. L'expérience montre qu'il est plus consistant de raffiner également les cellules avec $\epsilon_c < 2$ si elles sont voisines de cellules marquées comme étant à raffiner. Les figures 1.15 à 1.18 issues de Antepara et al. [59] montrent les résultats auxquels ce procédé peut aboutir.

1.3.3 Construction à partir d'une seule cellule

Une manière d'obtenir un maillage cartésien raffiné de manière automatique est de partir d'un seul cube assez grand pour contenir le problème et de le raffiner suffisamment : cette *Building-Cube Method* (BCM) a été introduite par Nakahashi and Kim [61] et est utilisée pour une génération de maillage en parallèle dans Lintermann et al. [58]. Le principe est de construire une première cellule cubique, ou plus généralement en forme de pavé droit, et de concevoir une fonction capable de diviser cette cellule en huit. Le processus de division est ensuite appliqué aux cellules filles. Après chaque

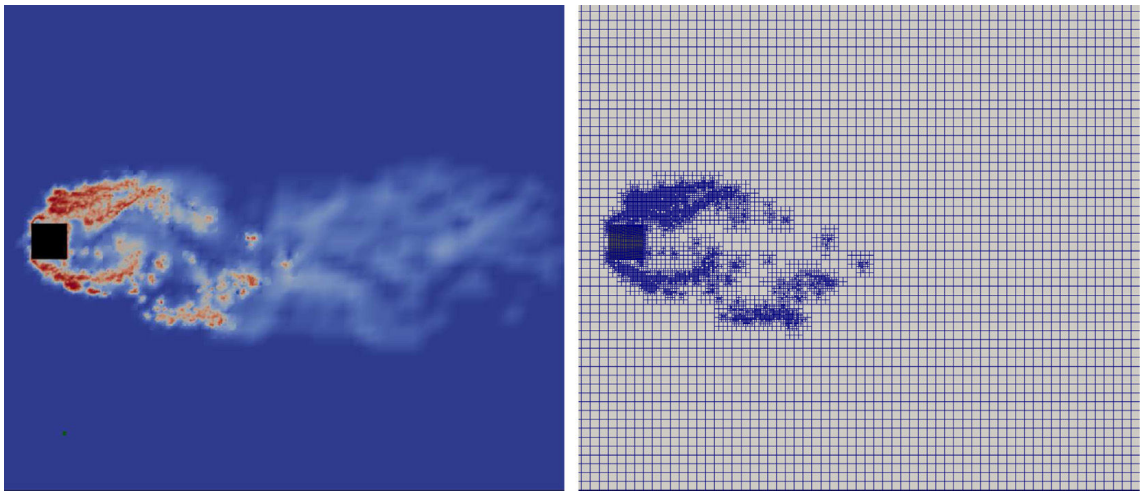


FIGURE 1.16 – Raffinement basé sur la vorticité tiré de Antepara et al. [59] : champ de vorticité autour d'un barreau à section carrée à Re = 22000

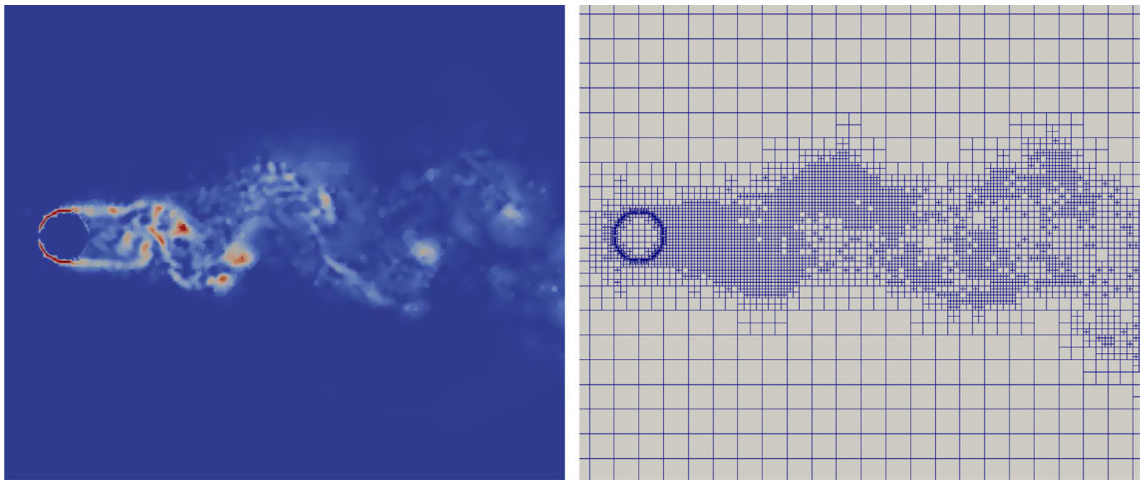


FIGURE 1.17 – Autre raffinement basé sur la vorticité tiré de Antepara et al. [59] : champ de vorticité autour d'un barreau à section circulaire à Re = 3900

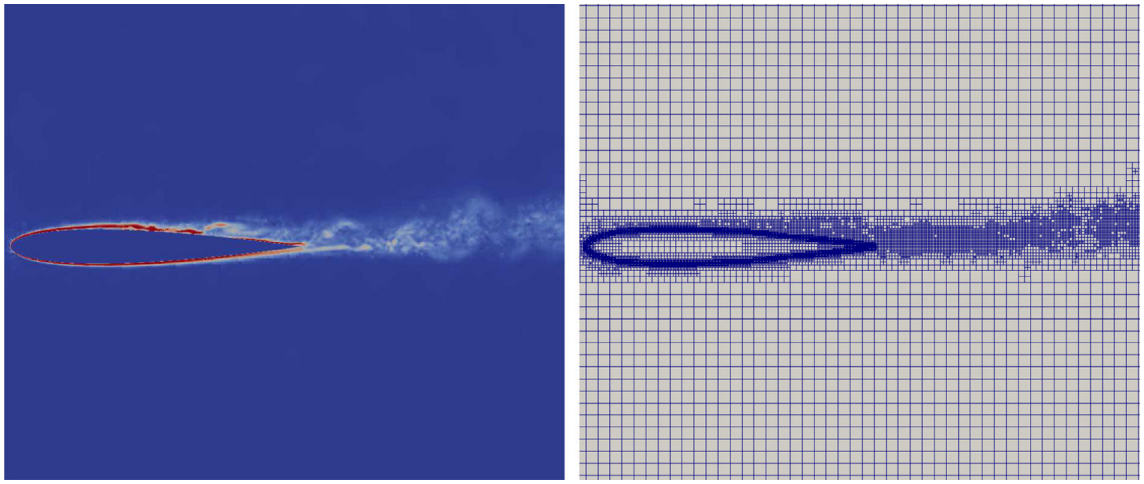


FIGURE 1.18 – Autre raffinement basé sur la vitesse résiduelle tiré de Antepara et al. [59] : champ de vorticité d'un écoulement turbulent autour d'une aile NACA 0012 à Re $= 5 \cdot 10^5$

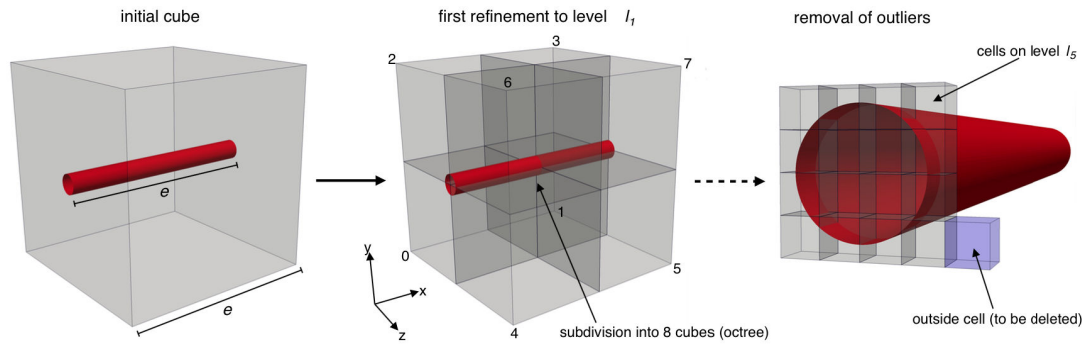


FIGURE 1.19 – Principe de la *Building-Cube Method* illustré dans Nakahashi and Kim [61].

étape de raffinement, les cellules qui sont en-dehors de la géométrie sont supprimées. Un fichier décrivant le domaine a mailler est nécessaire. Il s'agit habituellement d'un fichier STL. La figure 1.19 résume cet algorithme.

1.3.4 Arborescence

Considérons à titre d'exemple un problème de givrage sur un profil d'aile d'avion. Au temps $t = 0$, il n'y a pas de givre et nous avons besoin d'un maillage capable de décrire précisément le contour de l'aile. Mais au fur et à mesure que la couche de givre se forme, l'interface fluide/solide évolue et un nombre grandissant de cellules qui étaient fluides se retrouvent solides. Par conséquent, la puissance de calcul est amoindrie ou au mieux mésutilisée. L'idéal est alors de déraffiner, voire de supprimer, ces mailles converties à l'état solide et de raffiner le maillage au niveau de la nouvelle interface. Cette mise à jour du maillage est destinée à être effectuée régulièrement. La suppression pure et simple des cellules nous priverait de suivre la fonte éventuelle du givre. Le déraffinement, quant à lui, exige un historique des divisions effectués ; mettre ensemble des cellules qui ne proviennent pas d'une même cellule d'origine ne constitue pas un algorithme pérenne. En effet, on risquerait d'une part d'assembler des cellules

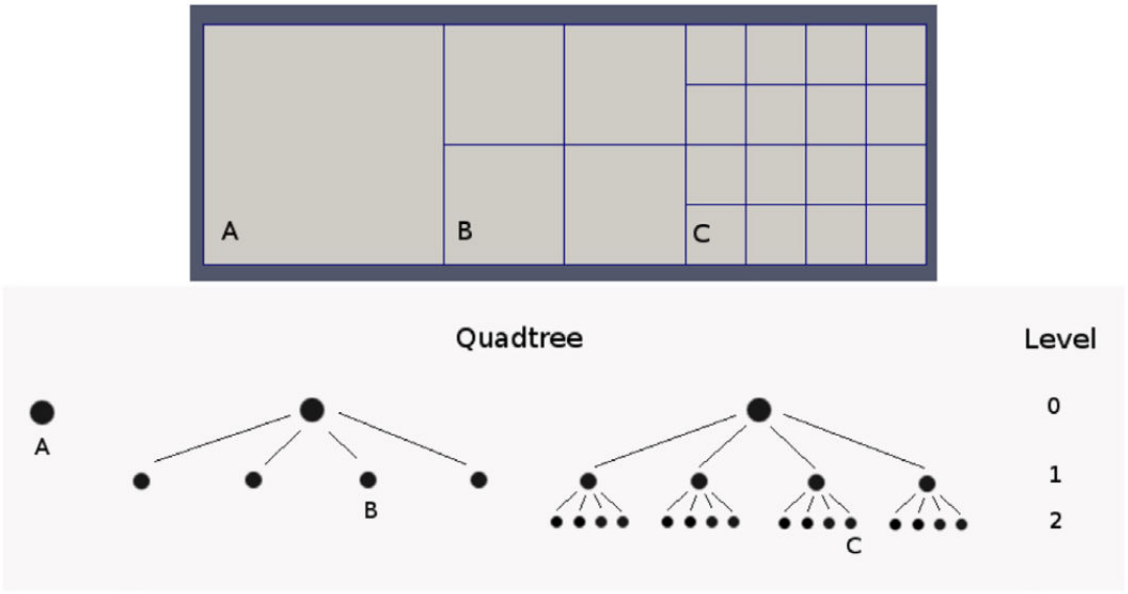


FIGURE 1.20 – Un arbre quaternaire en 2D tiré de Antepara et al. [59]

de tailles différentes et d'autre part d'aboutir à des configurations de maillage non prévues par le solveur utilisé. Afin de pouvoir reconstruire les cellules divisées, il est donc nécessaire de sauvegarder la filiation des cellules raffinées. On procède en général par arborescence en utilisant des arbres quaternaires (*quadtree*) en deux dimensions et des arbres octaires (*octree*) en trois dimensions. Les cellules raffinées (pères) sont divisées en 2^{dim} nouvelles cellules (filles) jusqu'à ce que le niveau de raffinement voulu soit atteint. Un exemple d'arbre quaternaire est donnée dans la figure 1.20 issue de l'article de Antepara et al. [59].

Chapitre 2

Méthode de résolution numérique

« *On résout les problèmes qu'on se pose et non les problèmes qui se posent.* »

Henri Poincaré

Ce chapitre décrit les équations générales gouvernant un fluide visqueux en situation isovolume et isotherme. Nous expliquons ensuite l'algorithme général de résolution de ce système d'équations. Enfin, nous détaillons la méthode d'interpolation ainsi que les conditions aux limites utilisées dans notre code.

2.1 Mise en équation de Navier-Stokes

2.1.1 Principe de conservation

En mécanique du solide, les lois de conservation s'obtiennent par dérivation d'une propriété physique extensive sur une quantité de matière donnée, souvent une *masse de contrôle* (CM pour *control mass* dans la littérature anglophone [14]). Lorsqu'il s'agit de fluides, il est plus difficile de suivre une particule de matière. Il convient donc de raisonner sur le fluide contenu dans un *volume de contrôle* (CV pour *control volume* dans la littérature anglophone [14]). Deux quantités extensives vont nous intéresser dans un premier temps : la masse et la quantité de mouvement (*momentum*). Dans les écoulements que nous étudierons, la masse n'est ni créée, ni détruite : il en résulte donc que pour une masse de contrôle :

$$\frac{\mathrm{d}m}{\mathrm{d}t} = 0. \tag{2.1}$$

Quant à la quantité de mouvement, elle est altérée par les forces qui agissent sur le fluide en suivant la deuxième lois de Newton :

$$\frac{\mathrm{d}(m\boldsymbol{u})}{\mathrm{d}t} = \sum \boldsymbol{f}, \tag{2.2}$$

où t représente le temps, m la masse, $\boldsymbol{u}$ la vitesse et $\boldsymbol{f}$ les forces agissantes sur la masse de contrôle. Nous allons transformer ces équations en formulation volumique en utilisant des grandeurs intensives comme la masse volumique ρ et la vitesse $\boldsymbol{u}$ qui représente la quantité de mouvement par unité de masse. Si ϕ est une variable intensive conservée et Φ sa variable extensive associée, on a toujours :

$$\Phi = \int_{\Omega_{\mathrm{CM}}} \rho\phi \mathrm{d}\Omega, \tag{2.3}$$

où Ω_{CM} représente le volume de contrôle occupé par la CM. Le choix $\phi = 1$ correspond à la conservation de la masse, $\phi = \boldsymbol{u}$ à la conservation de la quantité de mouvement. En utilisant cette notation, on obtient :

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega_{\mathrm{CM}}} \rho\phi \mathrm{d}\Omega = \frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega_{\mathrm{CV}}} \rho\phi \mathrm{d}\Omega + \oint_{S_{\mathrm{CV}}} \rho\phi(\boldsymbol{u} - \boldsymbol{u}_b)\cdot\boldsymbol{n}\mathrm{d}S, \tag{2.4}$$

où Ω_{CV} est le CV, S_{CV} est la surface fermée englobant le CV, $\boldsymbol{n}$ est le vecteur orthonormal à S_{CV} est dirigé vers l'extérieur, $\boldsymbol{u}$ est le vecteur vitesse du fluide et $\boldsymbol{u}_b$ est la vitesse à laquelle la surface de la CV se déplace. Pour un CV fixe, ce qui représente la plupart des cas, $\boldsymbol{u}_b = \boldsymbol{0}$ et la première dérivée du membre de droite devient partielle. Cette égalité traduit le fait que la variation de la propriété sur la masse de contrôle est égale à la somme de sa variation au sein du volume occupé par le fluide et de son flux à travers la surface l'englobant. Nous nous contenterons à présent du cas où le CV est fixe et

noterons donc Ω pour Ω_{CV}. et S pour S_{CV}.

2.1.2 Conservation de la masse

La forme intégrale de la conservation de la masse, également appelée **équation de continuité**, est obtenue directement en choisissant $\phi = 1$ dans le second membre de l'équation 2.4 et en l'injectant dans l'équation 2.1 :

$$\frac{\partial}{\partial t}\int_{\Omega} \rho \mathrm{d}\Omega + \oint_{S} \rho \boldsymbol{u} \cdot \boldsymbol{n} \mathrm{d}S = 0. \tag{2.5}$$

En appliquant le théorème de Green-Ostrogradski au terme de convection, on change l'intégrale de surface en intégrale de volume. En considérant des volumes de contrôle infinitésimaux, on obtient une forme différentielle sans coordonnées de l'équation de continuité :

$$\frac{\partial \rho}{\partial t} + \mathrm{div}\,(\rho \boldsymbol{u}) = 0. \tag{2.6}$$

Dans le cas d'un écoulement incompressible, la masse volumique ρ est constante et annule le premier terme de l'équation. Il en résulte que :

$$\mathrm{div}\,(\rho \boldsymbol{u}) = 0. \tag{2.7}$$

Ce qui donne, en coordonnées cartésiennes et en utilisant la convention d'Einstein où un indice apparaissant au moins deux fois dans un même terme doit être sommé sur l'ensemble de ses valeurs :

$$\frac{\partial \rho u_i}{\partial x_i} = \frac{\partial \rho u_x}{\partial x} + \frac{\partial \rho u_y}{\partial y} + \frac{\partial \rho u_z}{\partial z} = 0, \tag{2.8}$$

où x_i $(i = 1,2,3)$ ou (x,y,z) sont les coordonnées cartésiennes et u_i ou (u_x, u_y, u_z) sont les composantes cartésiennes du vecteur vitesse $\boldsymbol{u}$.

2.1.3 Conservation de la quantité de mouvement

Une manière d'obtenir l'équation intégrale de conservation de la quantité de mouvement est de procéder comme pour l'équation de conservation de la masse, mais en choisissant $\phi = \boldsymbol{u}$:

$$\frac{\partial}{\partial t}\int_{\Omega_{CV}} \rho \boldsymbol{u} \mathrm{d}\Omega + \oint_{S_{CV}} \rho \boldsymbol{u}\boldsymbol{u} \cdot \boldsymbol{n} \mathrm{d}S = \sum \boldsymbol{f}. \tag{2.9}$$

Afin de détailler le terme de droite, il est nécessaire de considérer les efforts auxquels le fluide est soumis :

- les forces surfaciques (pression, contraintes sur la surface, etc...) ;
- les forces distantes ou *body forces* (gravitation, champs électro-magnétiques, Coriolis, etc...).

À ce point, il est nécessaire de faire de nouvelles approximations afin de pouvoir fermer le système ; c'est-à-dire avoir autant d'équations que d'inconnues indépendantes. Nous allons donc considérer le fluide comme étant newtonien, ce qui est le cas pour un bon nombre des fluides étudiés à ce jour. Pour un fluide newtonien, le tenseur de contrainte T, qui est taux de transport moléculaire de la quantité de mouvement, s'écrit :

$$\mathsf{T} = -\left(p + \frac{2}{3}\mu \operatorname{div} \boldsymbol{u}\right)\mathsf{I} + 2\mu\mathsf{D}, \tag{2.10}$$

où μ est la viscosité dynamique, I le tenseur unité, ρ la pression statique et D le tenseur de déformation :

$$\mathsf{D} = \frac{1}{2}\left[\mathbf{grad}\,\boldsymbol{u} + \left(\mathbf{grad}\,\boldsymbol{u}\right)^{\mathrm{T}}\right]. \tag{2.11}$$

Ces deux dernières équations deviennent, en coordonnées cartésiennes :

$$T_{ij} = \left(p + \frac{2}{3}\mu\frac{\partial u_j}{\partial x_j}\right)\delta_{ij} + 2\mu D_{ij}, \tag{2.12}$$

$$D_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right), \tag{2.13}$$

où δ_{ij} est le symbole de Kronecker, valant 1 si et seulement $i = j$ et 0 sinon. Pour un écoulement incompressible, le second terme entre parenthèses du tenseur T est nul en vertu de l'équation de conservation de la masse 2.7. En choisissant la notation fréquemment utilisée dans la littérature, on peut écrire :

$$\tau_{ij} = 2\mu D_{ij} - \frac{2}{3}\mu\delta_{ij} \operatorname{div} \boldsymbol{u}. \tag{2.14}$$

Avec les forces distantes par unité de masse (*body forces*) représentée par le vecteur $\boldsymbol{b}$, la forme intégrale de l'équation de conservation de la quantité de mouvement s'écrit :

$$\frac{\partial}{\partial t}\int_{\Omega_{\mathrm{CV}}} \rho\boldsymbol{u}\mathrm{d}\Omega + \oint_{S_{\mathrm{CV}}} \rho\boldsymbol{u}\boldsymbol{u}\cdot\boldsymbol{n}\mathrm{d}S = \oint_{S_{\mathrm{CV}}} \mathsf{T}\cdot\boldsymbol{n}\mathrm{d}S + \int_{\Omega_{\mathrm{CV}}} \rho\boldsymbol{b}\mathrm{d}\Omega. \tag{2.15}$$

Une version vectorielle indépendante du choix des coordonnées de l'équation de conservation de la quantité de mouvement 2.15 est obtenue en appliquant directement le théorème de Gauss-Ostrogradsky aux termes de flux diffusif et convectif :

$$\frac{\partial}{\partial t}(\rho \boldsymbol{u}) + \operatorname{div}(\rho \boldsymbol{u}\boldsymbol{u}) = \operatorname{div} \mathsf{T} + \rho \boldsymbol{b}. \tag{2.16}$$

L'équation correspondante pour la $i^{\text{ème}}$ coordonnée cartésienne est :

$$\frac{\partial}{\partial t}(\rho u_i) + \operatorname{div}(\rho u_i \boldsymbol{u}) = \operatorname{div} \boldsymbol{t}_i + \rho b_i. \tag{2.17}$$

Cette dernière équation 2.17 est dite sous forme conservative puisque tous ses termes sont exprimés en tant que divergence d'un vecteur ou d'un tenseur. Une forme non-conservative peut-être obtenue en y intégrant l'équation de conservation de la masse. Comme, par dérivation,

$$\operatorname{div}(\rho \boldsymbol{u} u_i) = u_i \operatorname{div}(\rho \boldsymbol{u}) + \rho \boldsymbol{u} \cdot \mathbf{grad}\, u_i \tag{2.18}$$

et que $\operatorname{div}(\rho \boldsymbol{u}) = 0$, il en suit que

$$\rho \frac{\partial u_i}{\partial t} + \rho \boldsymbol{u} \cdot \mathbf{grad}\, u_i = \operatorname{div} \boldsymbol{t}_i + \rho b_i. \tag{2.19}$$

Le terme de pression contenu dans t_i peut alors être écrit

$$\operatorname{div}(p \boldsymbol{i}_i) = \mathbf{grad}\, p \cdot \boldsymbol{i}_i. \tag{2.20}$$

Le terme de pression est alors vu comme une force distante. Si l'équation 2.14 est substituée dans 2.17, on a :

$$\frac{\partial(\rho u_i)}{\partial t} + u_j \frac{\partial(\rho u_i)}{\partial x_j} = \frac{\partial \tau_{ij}}{\partial x_i} - \frac{\partial p}{\partial x_i} + \rho g_i, \tag{2.21}$$

où g_i est la composante selon le $i^{\text{ème}}$ vecteur du vecteur d'accélération gravitationnelle $\boldsymbol{g}$. Dans le cas que nous étudierons dans la suite, l'équation 2.21 devient :

$$\frac{\partial(\rho u_i)}{\partial t} + u_j \frac{\partial(\rho u_i)}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \nu \frac{\partial^2(\rho u_i)}{\partial x_j^2}, \tag{2.22}$$

où $\nu = \frac{\mu}{\rho}$ est la viscosité cinématique.

2.2 Schéma de Braza

La méthode de résolution utilisée dans NSIBM repose sur la méthode SMAC introduite par Amsden and Harlow [62] qui résout l'équation de Navier-Stokes écrite en variables premières (pression - vitesse) par une formulation explicite. Cette méthode a été étendue à une formulation implicite et semi-implicite par Braza [63] et est également utilisée dans la thèse de Hoarau [64]. Cette extension est basée sur un découplage entre la pression et la vitesse rendu possible par une méthode de prédiction-correction de la pression.
Le système d'équation à résoudre est composé par les équations 2.7 et 2.22 :

- l'équation de continuité ou de conservation de la masse :

$$\mathrm{div}\,(\rho \boldsymbol{u}) = 0, \tag{2.23}$$

- l'équation de conservation de la quantité de mouvement :

$$\frac{\partial(\rho u_i)}{\partial t} + u_j \frac{\partial(\rho u_i)}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \nu \frac{\partial^2(\rho u_i)}{\partial x_j^2} \tag{2.24}$$

Les inconnues, la vitesse et la pression, interviennent dans la même équation et ne peuvent donc pas être déterminées séparément.

On suppose connues la vitesse et la pression au pas de temps n et on cherche à les déterminer au pas $n+1$. L'écoulement étant considéré comme incompressible, ρ est constant. Les précédentes équations peuvent alors être discrétisées de la manière suivante :

$$\mathrm{div}\,(\boldsymbol{u}^{n+1}) = 0, \tag{2.25}$$

et

$$\frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^n}{\Delta t} + \mathbf{div}(\boldsymbol{u}^{n+1} \otimes \boldsymbol{u}^{n+1}) = -\frac{1}{\rho}\mathbf{grad}\; p^{n+1} + \nu\,\mathbf{div}(\mathbf{grad}\; \boldsymbol{u}^{n+1}) \tag{2.26}$$

où **grad** est la notation utilisée soit pour le gradient d'un champs scalaire, soit pour le gradient d'un champs vectoriel, auquel cas le résultat est un tenseur d'ordre 2. La notation div correspond à la divergence d'un vecteur, aboutissant à un scalaire, tandis que **div** correspond à l'opérateur divergence d'un tenseur d'ordre 2, aboutissant à un vecteur. Le terme convectif est linéarisé en approchant la quantité $\boldsymbol{u}^{n+1}$ par $\boldsymbol{u}^n$, soit :

$$\frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^n}{\Delta t} + \mathbf{div}(\boldsymbol{u}^{n} \otimes \boldsymbol{u}^{n+1}) = -\frac{1}{\rho}\mathbf{grad}\; p^{n+1} + \nu\,\mathbf{div}(\mathbf{grad}\; \boldsymbol{u}^{n+1}). \tag{2.27}$$

La pression au pas de temps $n+1$ étant inconnue, on en introduit une estimation notée p^*. Afin de continuer à travailler avec un couple pression-vitesse solution de l'équation de quantité de mouvement, on lui associe la prédiction de vitesse $\boldsymbol{u}^*$ de manière à ce

que le champ $(\boldsymbol{u}^*, p^*)$ vérifie également l'équation de quantité de mouvement, soit :

$$\frac{\boldsymbol{u}^* - \boldsymbol{u}^n}{\Delta t} + \mathbf{div}(\boldsymbol{u}^n \otimes \boldsymbol{u}^*) = -\frac{1}{\rho}\mathbf{grad}\ p^* + \nu\,\mathbf{div}(\mathbf{grad}\ \boldsymbol{u}^*). \tag{2.28}$$

Le pas prédicteur suppose, par exemple que

$$p^* = p^n. \tag{2.29}$$

On est ainsi amené à résoudre l'équation linéaire d'inconnue $\boldsymbol{u}^*$ suivante :

$$\frac{\boldsymbol{u}^* - \boldsymbol{u}^n}{\Delta t} + \mathbf{div}(\boldsymbol{u}^n \otimes \boldsymbol{u}^*) = -\frac{1}{\rho}\mathbf{grad}\ p^n + \nu\,\mathbf{div}(\mathbf{grad}\ \boldsymbol{u}^*). \tag{2.30}$$

$\boldsymbol{u}^*$ vérifie les mêmes conditions aux limites que $\boldsymbol{u}^{n+1}$. En prenant le rotationnel de l'équation 2.27 et de l'équation 2.30, on obtient de plus :

$$\mathbf{rot}\ \boldsymbol{u}^* = \mathbf{rot}\ \boldsymbol{u}^{n+1}. \tag{2.31}$$

Le champ $(\boldsymbol{u}^*, p^*)$ transporte donc le montant exact du rotationnel à l'instant $n+1$. Cependant, $\boldsymbol{u}^*$ ne satisfait plus l'équation de continuité. Il est donc nécessaire d'introduire un champ corrigé tel que :

$$\mathrm{div}\,(\boldsymbol{u}^{**}) = 0, \tag{2.32}$$

qui est définit comme

$$\boldsymbol{u}^{**} = \boldsymbol{u}^n - \mathbf{grad}\,\phi, \tag{2.33}$$

où ϕ est une fonction potentielle auxiliaire qui doit encore être identifiée. En appliquant l'opérateur divergence à cette dernière équation et en considérant la propriété de $\boldsymbol{u}^{**}$ donnée dans l'équation 2.32, on obtient :

$$\Delta\,\phi = \mathrm{div}\ \boldsymbol{u}^* \tag{2.34}$$

qui est une équation de Poisson pour ϕ. La fonction ϕ est donc déduite directement des valeurs de $\boldsymbol{u}^*$. La vitesse corrigée $\boldsymbol{u}^{**}$ est ensuite obtenue à partir de l'équation 2.33. Par ailleurs, comme on a $\mathrm{div}\,(\boldsymbol{u}^{**}) = 0$, et que d'après 2.31 et 2.33, on a aussi

$$\mathbf{rot}\ \boldsymbol{u}^{**} = \mathbf{rot}\ \boldsymbol{u}^* = \mathbf{rot}\ \boldsymbol{u}^{n+1}$$

et puisque les conditions limites de $\boldsymbol{u}^{**}$ sont identiques à celles de $\boldsymbol{u}^{n+1}$, on peut démontrer (Braza [65]) que $\boldsymbol{u}^{**}$ s'identifie au champ de vitesse corrigée à l'instant $n+1$.

La pression à l'instant $n+1$ s'obtient en remplaçant $\boldsymbol{u}^{n+1}$ par l'expression précé-

demment obtenue 2.33 dans l'équation 2.27 et en soustrayant le résultat à l'équation 2.30. On a alors :

$$\mathbf{grad}\ p^{n+1} = \mathbf{grad}\ (p^n + \frac{\phi}{\Delta t}) + \mathbf{div}(\boldsymbol{u}^n \otimes \mathbf{grad}\ \phi) - \nu\nabla^2(\mathbf{grad}\ \phi). \tag{2.35}$$

Cette relation peut se simplifier en :

$$p^{n+1} = p^n + \frac{\phi}{\Delta t}. \tag{2.36}$$

Les test numériques effectués dans l'article de Braza et al. [66] ont montré que la précision et la stabilité du code ne sont pas affectées par cette dernière formulation de p^{n+1}. Cette relation fournissant des résultats sensiblement proches de ceux de la relation complète 2.35, c'est celle qui est couramment utilisée.

En résumé, à chaque pas de temps, les équations suivantes sont résolues dans cet ordre :

$$\begin{cases} \frac{\boldsymbol{u}^* - \boldsymbol{u}^n}{\Delta t} + \mathbf{div}(\boldsymbol{u}^n \otimes \boldsymbol{u}^*) = -\frac{1}{\rho}\mathbf{grad}\ p^n + \nu\,\mathbf{div}(\mathbf{grad}\ \boldsymbol{u}^*) \\ \Delta\,\phi = \mathrm{div}\ \boldsymbol{u}^* \\ \boldsymbol{u}^{**} = \boldsymbol{u}^n - \mathbf{grad}\ \phi \\ p^{n+1} = p^n + \frac{\phi}{\Delta t}. \end{cases} \tag{2.37}$$

L'algorithme ainsi utilisé est résumé sur la figure 2.1.

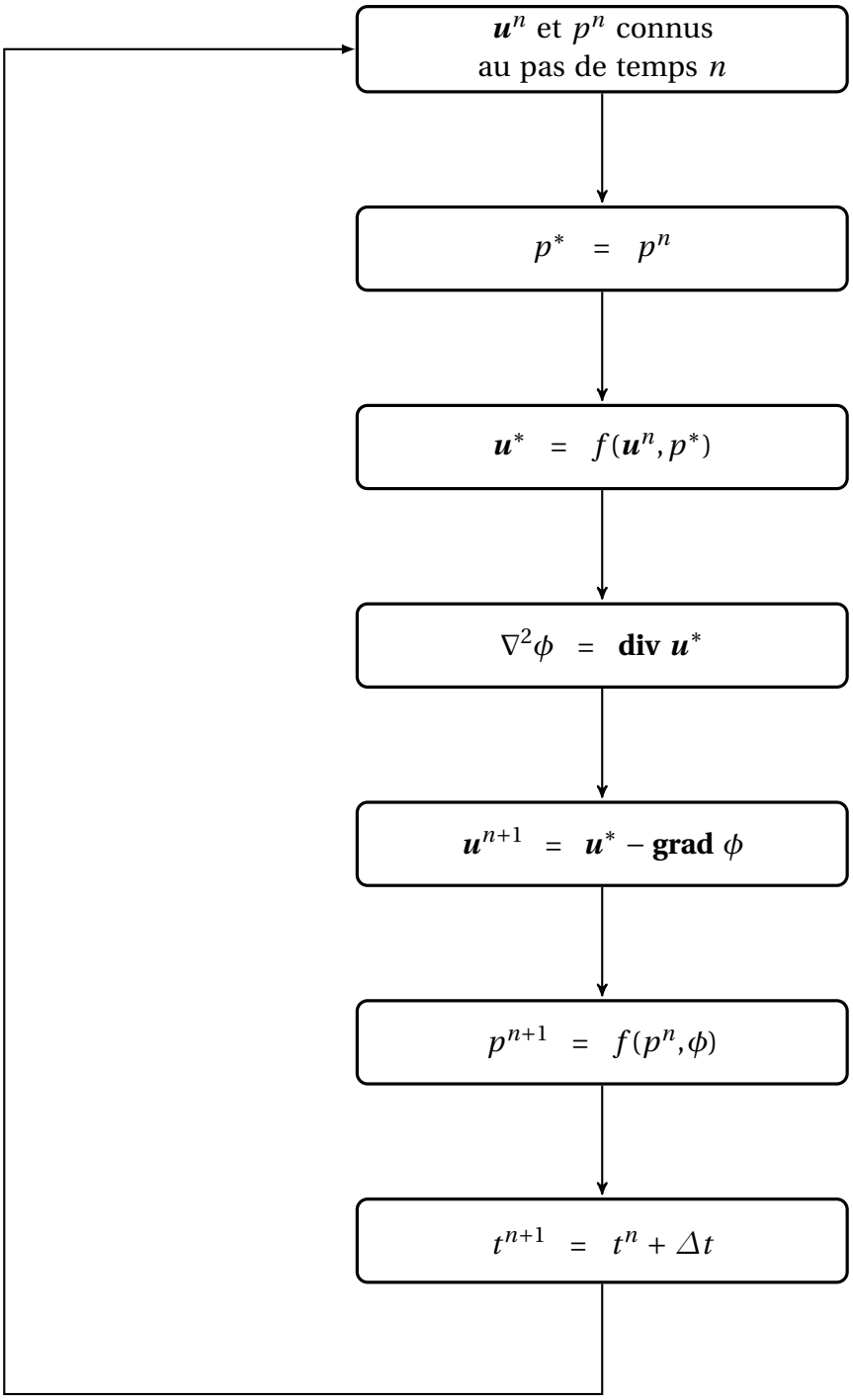


FIGURE 2.1 – Algorithme de résolution

2.3 La méthode *SIMPLE*

La schéma de résolution numérique *SIMPLE* a été développé sur pour maillages décalés *(staggered grids)*, où les variables comme la pression et la masse volumique sont stockées aux centres des cellules, tandis que les vitesses sont stockées sur les faces, puis adaptée à des maillages standards *(collocated grids)*, où toutes les variables sont stockées aux centres des cellules, via l'interpolation de Rhie & Chow [67, 68] pour obtenir les valeurs nécessaires aux faces. En notant les corrections de pression et de vitesse

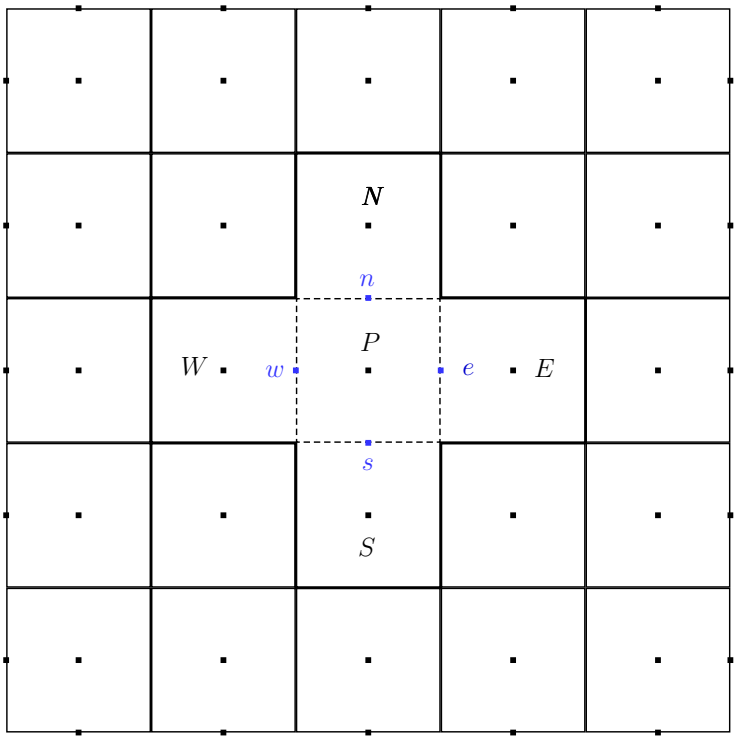


FIGURE 2.2 – Maillage décalé pour *SIMPLE* : Les points sont placés au centre des cellules, sauf pour les frontières. On introduit les notations suivantes pour les volumes de contrôle voisins de la cellule centrale bleue notée P : N, S, W et E. Ses faces en pointillés bleus sont notées en minuscules : n, s, w et e et leurs surfaces sont respectivement écrites S_n, S_s, S_w et S_e.

p' et u_i' comme les différences entre les champs de pression ou de vitesse p et u_i à l'itération en cours (*nouveau*) et les champs de pression ou de vitesse p^* et u_i^* à l'itération précédente (*ancien*), on a :

$$\begin{aligned} p &= p^* + p' \\ u_i &= u_i^* + u_i'. \end{aligned} \tag{2.38}$$

Dans la grille décalée, on peut maintenant réécrire l'équation de conservation de la quantité de mouvement 2.22

$$a_e u_{ie}^* = \sum a_{\text{nb}} u_{\text{inb}}^* + (p_P^* - p_E^*) S_{ie} + b_{u_i e} \tag{2.39}$$

au point e pour les *anciennes* vitesses, où les (a_i) dépendent des schéma numériques spatio-temporels choisis, et la soustraire à celle des *nouvelles* vitesses, i.e. :

$$a_e u_{ie} = \sum a_{\text{nb}} u_{i\text{nb}} + (p_P - p_E) S_{ie} + b_{u_i e} \tag{2.40}$$

de manière à obtenir

$$a_e u'_{ie} = \sum a_{\text{nb}} u'_{i\text{nb}} + (p'_P - p'_E) S_{ie}. \tag{2.41}$$

L'omission du terme $\sum a_{\text{nb}} u'_{i\text{nb}}$ constitue la principale approximation du schéma *SIMPLE*. Elle nous donne :

$$u'_{ie} = (p'_P - p'_E) \frac{S_{ie}}{a_e} \tag{2.42}$$

Cet oubli n'affecte cependant pas le résultat final puisque les corrections de vitesse $u'_{i\text{nb}}$ s'annulent une fois que la solution a convergé. Cependant, cette approximation nuit à la consistance du schéma puisque l'ordre du terme ignoré est le même que celui du terme de gauche dans l'équation 2.41. Une variante plus consistante de ce schéma existe sous le nom de *SIMPLEC (Semi-Implicite Method for Pressure-Linked Equations, Consistent)*. Elle est obtenue en soustrayant le terme $\sum a_{\text{nb}} u'_{ie}$ des deux côtés de l'équation 2.41. Il en résulte

$$\left(a_e - \sum a_{\text{nb}}\right) u'_{ie} = \sum a_{\text{nb}} (u'_{i\text{nb}} - u'_{ie}) + (p'_P - p'_E) S_{ie} \tag{2.43}$$

L'omission du terme $\sum a_{\text{nb}} (u'_{i\text{nb}} - u'_{ie})$ représente l'approximation consistante de l'algorithme *SIMPLEC*, donnant :

$$u'_{ie} = (p'_P - p'_E) \frac{S_{ie}}{a_e - \sum a_{\text{nb}}}. \tag{2.44}$$

Finalement, on obtient l'expression de la vitesse sur la face est à partir de l'équation 2.38 :

$$u_{ie} = u^*_{ie} + d_e (p'_P - p'_E) \tag{2.45}$$

et, par analogie :

$$\begin{aligned}
u_{iw} &= u^*_{iw} + d_w (p'_W - p'_P) \\
u_{in} &= u^*_{in} + d_n (p'_P - p'_N) \\
u_{is} &= u^*_{is} + d_s (p'_S - p'_N) \\
u_{if} &= u^*_{if} + d_f (p'_P - p'_F) \\
u_{ir} &= u^*_{ir} + d_r (p'_R - p'_P).
\end{aligned} \tag{2.46}$$

En insérant ces vitesses corrigées dans le l'équation de continuité discrétisée d'un volume de contrôle non décalé, on obtient

$$\sum_{\mathrm{CS}}(\rho u_{i\mathrm{nb}} S_{i\mathrm{nb}}) = 0 \tag{2.47}$$

et en identifiant les coefficients, on obtient une équation discrétisée pour la correction de pression :

$$b_P p'_P = b_N p'_N + b_S p'_S + b_E p'_E + b_W p'_W + RHS_{p'} \tag{2.48}$$

où

$$\begin{aligned} b_P &= b_E + b_W + b_N + b_S \\ b_E &= \rho_e \frac{S_e^2}{a_e} \\ &\vdots \\ RHS_{p'} &= -\textstyle\sum_{\mathrm{CS}}(\rho u^*_{i\mathrm{nb}} S_{i\mathrm{nb}}). \end{aligned} \tag{2.49}$$

2.4 Interpolation de Rhie & Chow

L'interpolation de Rhie & Chow, introduite dans [67] permet d'émuler un maillage décalé (*staggered grid*) à partir d'un maillage standard (*collocated grid*). Sa principale force est d'augmenter la robustesse sur schéma par rapport à une interpolation linéaire classique, tout en étant très simple à mettre en place numériquement. Il évite le découplage pression-vitesse et les oscillations de pression qui y sont associées. La vitesse sur la face e est habituellement obtenue de la manière suivante :

$$u_e = \frac{1}{2}(u_E + u_P) \tag{2.50}$$

Cependant, dans un maillage non décalé, ceci peut engendrer des oscillations de pression. Afin d'éviter cela, les vitesses aux faces sont calculées en ajoutant et en ôtant le gradient de pression :

$$u_e = \frac{1}{2}(u_E + u_P) - \frac{V_e}{a_P}\left(\frac{\partial p}{\partial x}\right)_e + \frac{V_e}{a_P}\left(\frac{\partial p}{\partial x}\right)_e. \tag{2.51}$$

Ce gradient de pression peut être calculé de deux manières : d'une part comme moyenne des gradients de pression aux point P et E

$$\begin{aligned} \left(\frac{\partial p}{\partial x}\right)_e &= \tfrac{1}{2}\left(\left(\frac{\partial p}{\partial x}\right)_P + \left(\frac{\partial p}{\partial x}\right)_E\right) \\ &= \tfrac{1}{2}\left(\tfrac{p_E - p_W}{d_{WE}} + \tfrac{p_{EE} - p_P}{d_{PEE}}\right) \end{aligned} \tag{2.52}$$

et d'autre part en le calculant directement sur la face

$$\left(\frac{\partial p}{\partial x}\right)_e = \frac{p_E - p_P}{d_{PE}}. \tag{2.53}$$

En cas d'équidistance, $d_{WE} = d_{PEE} = 2d_{PE}$ et

$$u_e = \frac{1}{2}(u_E + u_P) + \frac{S_e}{4d_{PE}a_P}(p_{EE} + 3p_P - 3p_E - p_W) \tag{2.54}$$

qui est utilisé pour calculer les convections $conv_e$ à travers la face est. En cas de non-équidistance, le premier terme est calculé comme une moyenne pondérée tandis que le second est conservé à l'identique, puisqu'il s'agit d'un terme d'ordre quatre pour réduire les oscillations. Selon la complexité du maillage, le terme en u_E et u_P peut s'avérer relativement difficile à calculer. Son obtention est expliquée au paragraphe 3.4.

2.5 Conditions initiales

Les calculs faits à ce jour avec le solveur *NSIBM* pour des nombres de Reynolds modérés sont initialisés avec un champ uniforme pour la vitesse et la pression, soit pour un écoulement incident d'angle α :

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix}_{\text{init}} = \begin{pmatrix} u_\infty \cos\,\alpha \\ v_\infty \sin\,\alpha \\ 0 \end{pmatrix}$$

et

$$p = \text{constante}.$$

Il est également possible d'utiliser comme conditions initiale le résultats d'un calcul précédemment effectué. Cela permet de continuer le calcul pour accéder à des pas de temps ultérieurs sans avoir à recommencer du début.

2.6 Conditions aux limites

La gestion des frontières du domaine se fait par adjonction d'une cellule fictive (ou fantôme) à chaque face limitrophe. La cellule est toujours symétrique à la cellule intérieure par rapport à la face qui les sépare. Elle contient l'information de son type, de manière à pouvoir appliquer les bonnes conditions limites au niveau de la face. Ces conditions sont détaillées ci-dessous.

2.6.1 Pression

En instationnaire, des conditions limites de Neumann sont appliquées à toutes les frontières pour en assurer la nullité du gradient :

$$\frac{\partial p}{\partial \boldsymbol{n}} = 0 \tag{2.55}$$

où $\boldsymbol{n}$ est un vecteur unitaire normal à la frontière.

En stationnaire, la condition de Neumann

$$\frac{\partial p}{\partial \boldsymbol{n}} = 0 \tag{2.56}$$

est également imposée partout, sauf à la sortie, où on impose :

$$p = \text{constante} \tag{2.57}$$

2.6.2 Vitesse

Pour le champ de vitesse, les conditions aux limites sont de type Dirichlet à l'entrée et aux parois. Dans le cas général d'un écoulement incident d'angle α, on a :

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix}_{\text{inorwall}} = \begin{pmatrix} u_\infty \cos\,\alpha \\ v_\infty \sin\,\alpha \\ 0 \end{pmatrix}. \tag{2.58}$$

Á la sortie, on impose à la vitesse une condition de Neumann

$$\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{n}} = 0, \tag{2.59}$$

puis on vérifie que la conservation de la masse soit respectée entre l'entrée et la sortie pour s'assurer qu'il n'y a ni perte, ni création de masse.

Chapitre 3

Raffinement de maillage

« *Rendez les choses aussi simples que possible, mais pas plus simples.* »

Albert Einstein

Ce chapitre résume l'algorithme de raffinement mis en place dans NSIBM. Il s'apparente à une documentation du code et sera utile à celles et ceux qui seront amenés à s'en servir et à en poursuivre le développement.

3.1 Introduction

Le code *NSIBM*, acronyme de *Navier-Stokes Immersed Boundary Method* est un programme développé par mes soins au laboratoire strasbourgeois iCube, département de mécanique, en langage Fortran. Son but est de résoudre l'équation de Navier-Stokes incompressible sur une large variété de problèmes numériques, en minimisant le temps d'utilisation. Il dispose d'un générateur automatique de maillage en deux et en trois dimensions, fonctionnant en séquentiel, et utilise *OpenMPI* pour gérer le calcul en parallèle.

3.2 Génération du maillage

Nous avons opté pour un maillage cartésien, non structuré et non conforme, ce qui veut dire qu'il sera constitué de rectangles en 2D et de pavés droits en 3D, que les cellules voisines n'auront pas nécessairement une numérotation suivie et qu'elles n'auront pas obligatoirement la même taille. De cette manière, nous sommes capables de générer facilement, et surtout automatiquement, de bons maillages pour des géométries compliquées en utilisant le raffinement de maillage automatique (*AMR* pour Automatic Mesh Refinement). Les mailles sont générées selon l'organigramme 3.1.
Afin d'expliquer proprement le processus, il m'est nécessaire de décrire certains éléments du code.

3.2.1 Structure des données

Le but de cette sous partie est de donner la structure choisie pour les éléments de base du maillage.

Point

L'objet **point** ne contient qu'une information : ses coordonnées cartésiennes. C'est un triplet de réels double précision stocké dans l'attribut **xyz** résumé dans le tableau 3.1.

Type pointData			
attr.	information	type	dim.
xyz	coordonnées du point	réel	3

TABLEAU 3.1 – Attribut du type point

Face

L'objet **face** est l'élément clé de la géométrie puisqu'il lie les mailles entre elles : c'est par les faces que l'on accède aux cellules voisines. On y trouve également des informations nécessaires à l'interpolation de la valeur d'une grandeur physique ϕ sur

la face, à partir de ses valeurs dans les cellules à proximité. Le type face est décrit dans le tableau 3.2.

Type faceData			
attr.	information	type	dim.
facetype	type de face	entier	1
cell	numéros des cellules voisines	entier	(0 :1,0 :5)
vertex	numéro des points composant la cellule	entier	9
number	numéro global, des processeurs et locaux	entier	5
direction	indice de vecteur normal	entier	1
area	surface de la face	réel	1
lambda	coefficients d'interpolation	réel	(0 :1,0 :5)
xyz	coordonnées du centre de la face	réel	3
celldist	la distance entre les cellules mitoyennes	réel	1

TABLEAU 3.2 – Attributs du type face

L'attribut **facetype** du tableau 3.2 ou **celltype** du tableau 3.9, prend les valeurs du tableau 3.3.

Facetype ou celltype	
valeur	type
1	fluide
2	mur
3	entrée
4	sortie
5	symétrie
6	connexion entre deux processeurs
7	cellule fantôme
8	cellule d'interpolation

TABLEAU 3.3 – Indexation des types de face et de cellule

L'attribut **cell** est le plus important. C'est celui qui contient toutes les informations concernant la connectivité des cellules et les indices nécessaires à l'interpolation. L'interpolation est nécessaire pour le calcul des grandeurs physiques au niveau des faces, puisque celles-ci sont calculées au centre des cellules par le schéma numérique. Si les deux cellules de part et d'autre de la face ont le même niveau de raffinement, il suffit de prendre la valeur moyenne de la grandeur physique dans ces deux cellules. Si l'une est plus raffinée que l'autre, différents cas de figure sont susceptibles d'apparaître. Ils sont expliqués au paragraphe 3.4 et illustrés sut les figures 3.7 et 3.8. Ces cas requièrent l'intervention de plus de cellules que seulement celles à l'interface. Cela implique leur stockage, ainsi que celui des coefficient d'interpolation qui doivent leur être attribué. L'attribut **cell** les contient. L'explication la plus claire est celle de l'exemple du tableau 3.4.

Tableau cell						
Côté	nbr.	cellule 1	cellules 2 et +			
Gauche	1	42	0	0	0	0
Droit	3	512	34	743	0	0

TABLEAU 3.4 – Le tableau contenant les cellules d'interpolation

"Gauche" correspond à l'indice 0 du premier attribut (indexé par zéro) du tableau et à la cellule située à une abscisse inférieure sur l'axe normal à la face. "Droit" correspond à l'indice 1 du premier attribut du tableau et à la cellule située à une abscisse supérieure sur l'axe normal à la face. La face décrite ici fait l'interface entre la cellule 42, à sa gauche, et la 512, à sa droite. La colonne "nbr." nous indique que la cellule de gauche est plus raffinée que celle de droite et que les cellules 34 et 743 servent à calculer la valeur à droite de la face. Cette configuration correspond au premier cas de la figure 3.8.

Ce tableau cell est associé à l'attribut **lambda** qui stocke les coefficients d'interpolation à appliquer à chacune des cellules. L'indexation est la même que précédemment, à ceci près que l'indice nul de le seconde dimension ne donne plus le nombre de cellules à utiles à l'interpolation, mais les coefficients d'interpolation globaux pour les blocs de gauche et de droite. Les réels suivants donnent explicitement les coefficients relatifs aux cellules dont l'indice de **cell** correspond. Expliquons-le avec l'exemple du tableau 3.5 associé au tableau 3.4 ci-dessus :

Tableau lambda						
Côté	coef. côté	cellule 1	cellules 2 et +			
Gauche	2/3	1	0	0	0	0
Droit	1/3	1/2	1/4	1/4	0	0

TABLEAU 3.5 – Le tableau contenant les coefficients d'interpolation

Notons Φ_g la valeur à gauche de la face et Φ_d celle à droite. Considérons que l'on veuille connaître la vitesse sur la face. Les vitesses étant stockées dans le tableau **U** et par cellule, on a :

$$\Phi_g = \mathbf{U}(42) \tag{3.1}$$

et

$$\Phi_d = \frac{1}{2}\mathbf{U}(512) + \frac{1}{4}\mathbf{U}(34) + \frac{1}{4}\mathbf{U}(743) \tag{3.2}$$

La vitesse sur la face Φ_f est alors :

$$\Phi_f = \frac{2}{3}\Phi_g + \frac{1}{3}\Phi_d. \tag{3.3}$$

Cellule

La cellule est l'élément de base du maillage. Elle se compose de faces et de ses huit sommets. Les numéros de cellules négatifs sont dédiés aux cellules fantômes et les numéros positifs aux cellules standards. Le type **cell** a pour attributs :

Type cellData			
attr.	information	type	dim.
celltype	type de cellule	entier	1
facelist	liste des faces	entier	(4,2) ou (6,4)
number	numéro global, du processeur et local	entier	3
amrlevel	degré de raffinement	entier	5
corners	points aux coins de la cellule	entier	8
innercell	cellule interne d'une cellule fantôme	entier	1
innerface	face interne d'une cellule fantôme	entier	1
xyz	coordonnées du centre de la cellule	réel	3
vol	volume de la cellule	réel	1

TABLEAU 3.6 – Attributs du type cell

L'attribut **facelist** mérite que l'on s'y attarde. Un exemple en 2D est donné dans le tableau 3.7 et un exemple 3D se trouve dans le tableau 3.8.

Facelist 2D				
	ouest	est	sud	nord
face primaire	122	134	144	152
face secondaire	0	325	0	0

TABLEAU 3.7 – Stockage de la liste des faces en 2D

Facelist 3D						
	ouest	est	avant	arrière	sud	nord
face primaire	122	134	144	152	153	154
face secondaire	765	325	0	0	0	0
face secondaire	766	326	0	0	0	0
face secondaire	767	327	0	0	0	0

TABLEAU 3.8 – Stockage de la liste des faces en 3D

Le nombre maximal de côtés divisés en plusieurs faces peut être imposé dans le code. Cette condition est détaillée à la section 3.3.2. En trois dimensions, une face dont un voisin est plus raffiné que l'autre est divisée en quatre.

Cellule fantôme

Les cellules fantômes sont **indexées négativement** dans le tableau **cell**. Elles contiennent trois principales informations : le type (entrée/sortie ou mur, par exemple), le numéro de la face adjacente, dans **innerface** et le numéro de la cellule interne adjacente, dans **innercell**.

Type cellData pour les indices négatifs			
attr.	information	type	dim.
celltype	type de cellule	entier	1
innercell	cellule interne d'une cellule fantôme	entier	1
innerface	face interne d'une cellule fantôme	entier	1

TABLEAU 3.9 – Attributs du type cell utilisés pour les cellules fantômes

En résumé, chaque cellule connait directement sa position et les faces qui la composent et trouve ses voisins en passant par les informations contenues dans ses faces. Chaque face connait directement sa position et les numéros des cellules dont elle est l'interface.

3.2.2 Processus de génération du maillage

La première étape consiste à créer un maillage cartésien non structuré rectangulaire ou parallélépipédique rectangle, selon qu'il s'agisse d'un problème en deux ou trois dimensions, voir la figure 3.2. Ce rectangle ou pavé droit doit avoir été choisi suffisamment grand pour pouvoir y plonger le domaine que vous nous désirons étudier. Il faut ensuite en ôter tout ce qui "dépasse" en supprimant des mailles. Cela permet d'optimiser leur nombre et donc la complexité du solveur. La figure 3.3, qui représente un poumon de rat en $3D$, illustre cette action. Enfin, on raffine automatiquement près des embranchements ou des entrées/sorties par exemple. La figure 3.4 illustre cela. Le générateur de maillage est conçu de manière à pouvoir recommencer et combiner ces deux dernières actions à l'envi, afin d'obtenir un résultat aussi fin que voulu.

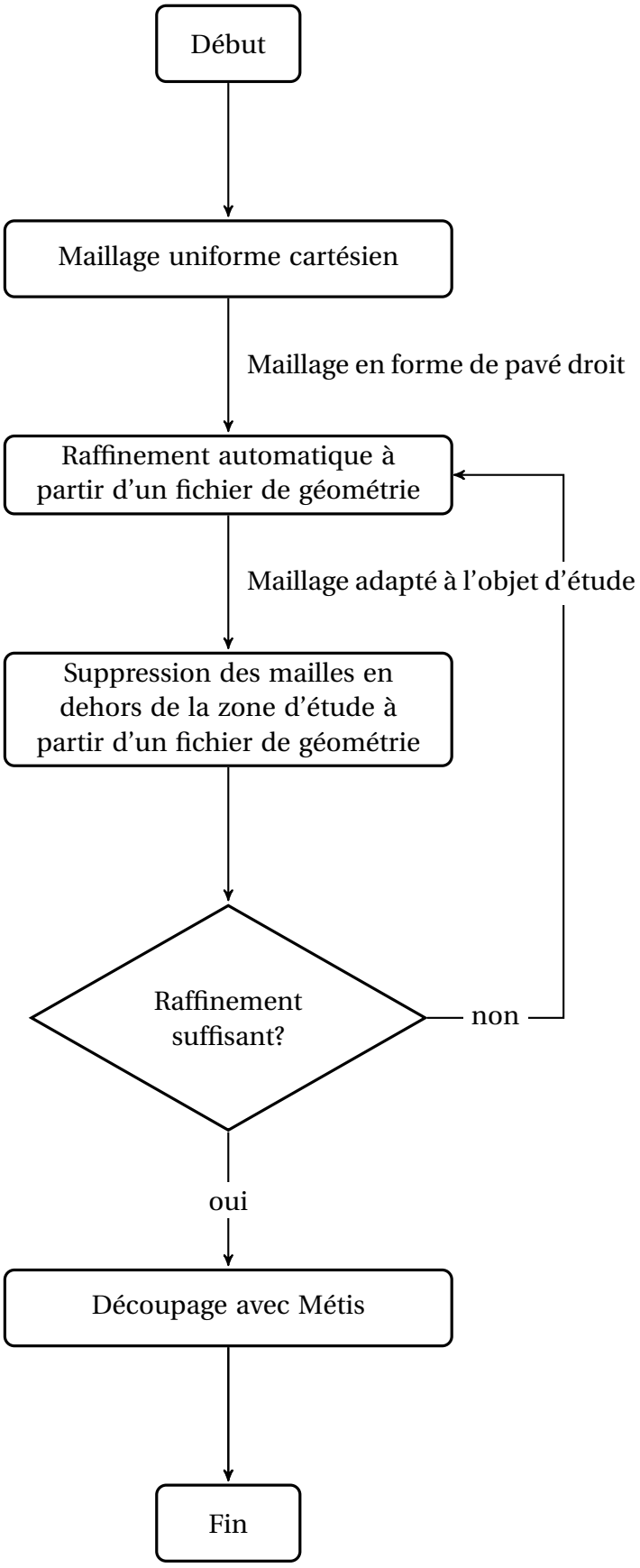
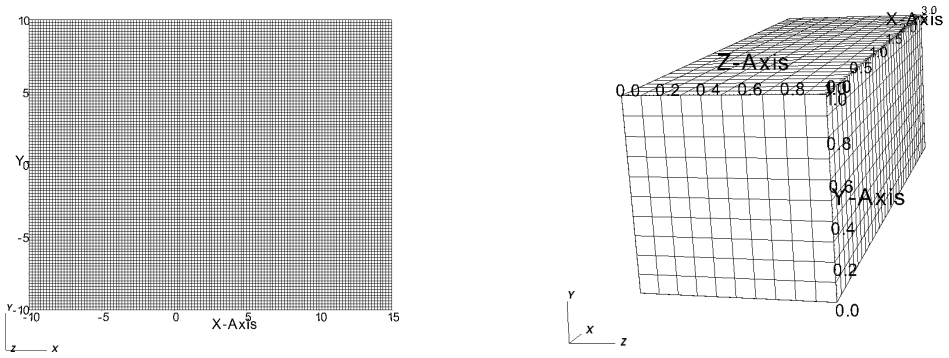


FIGURE 3.1 – Organigramme de la génération du maillage



2D 3D

FIGURE 3.2 – Exemples de maillages de départ

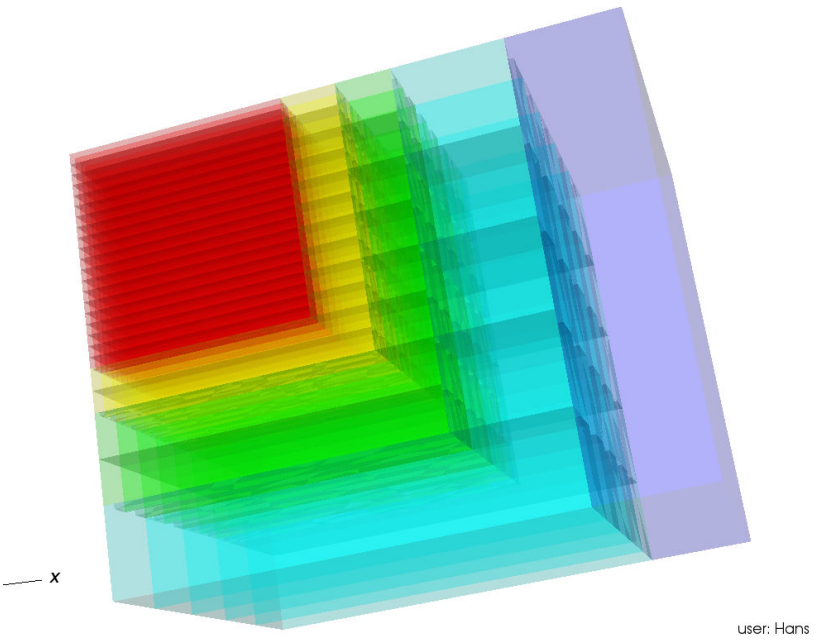


FIGURE 3.3 – Un maillage montrant différents niveaux de raffinement

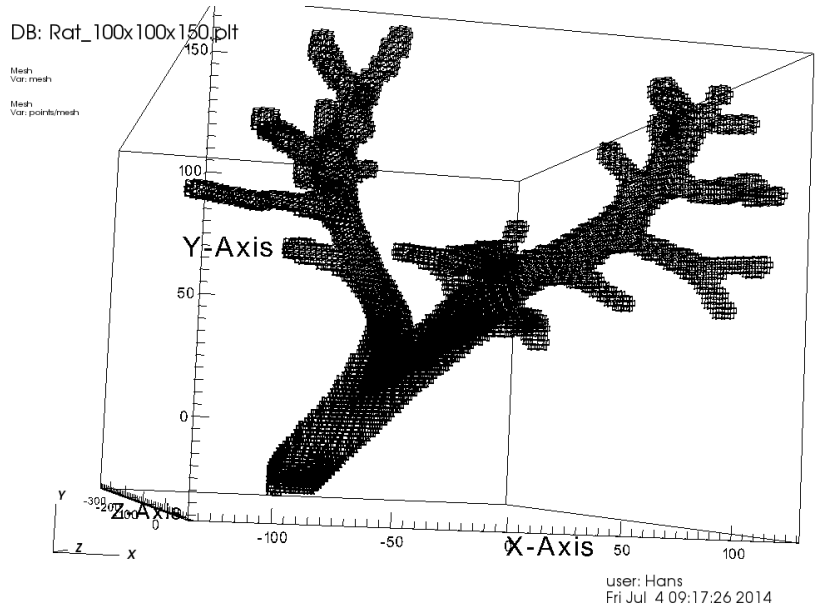


FIGURE 3.4 – Poumon de rat raffiné

3.3 Raffinement

3.3.1 Principe

Le principe du raffinement de maillage est tout simplement de découper une cellule en plusieurs cellules de même forme. Comme nous travaillons avec un maillage cartésien, nous partageons les rectangles en quatre et des pavés droits en huit. La non-structuration nous permet d'ajouter des cellules en modifiant les attributs des cellules et faces qui sont affectées et en créant celles qui manquent. La première étape est de remplir un tableau de la taille du nombre de cellules, initialisé à zéro et incrémenté à un si la cellule doit être raffinée. Ce tableau sera l'argument de la subroutine récursive **refine_mesh** qui se chargera de modifications nécessaires, en accord avec le cahier des charges qui suit. La seconde étape consiste à utiliser la subroutine **delete_extra_geometry**, qui les supprimera les cellules extérieures à la géométrie et ajoutera de nouvelles cellules fantômes aux nouvelles frontières. Le choix des cellules à raffiner et à supprimer se fait de diverses manières :

- géométriquement : en fonctions des coordonnées des cellules. Possible si la géométrie étudiée est simple (composée de cuboïdes, sphères, etc...) et si l'on sait où auront lieu les perturbations à surveiller ;
- par la méthodes des frontières immergées : se référer au chapitre suivant ;
- numériquement : en fonction de résultats moins fins obtenus précédemment. On peut par exemple raffiner 5% des cellules en choisissant celles où les vitesses sont les plus importantes. La valeur de div **u** est également un critère utilisable.

3.3.2 Conditions de raffinement

Afin que le raffinement soit le plus homogène possible, le cahier des charges suivant a été mis en place :

- deux cellules voisines ne peuvent avoir plus d'un niveau de raffinement d'écart : cela rend plus douces les zones de transitions entre les parties du maillages les plus raffinées et les plus grossières, limite les configurations d'interpolation et de raffinement possibles et concoure à la stabilité numérique du solveur ;
- une cellule ne peut avoir plus d'un certain nombre de côtés raffinés. Ce nombre est un paramètre du raffinement. Cette condition limite la fréquence recours à l'interpolation et évite de se retrouver avec une cellule raffinée isolée.

Si on pose ce maximum à une seule face, deux cellules voisines d'une même troisième cellule ne peuvent différer de plus d'un niveau de raffinement. Ceci permet de réduire les configurations d'interpolation possible. Le paragraphe suivant détaille ce point.

3.3.3 Récursivité

Le procédé de raffinement est récursif et se fait degré par degré. Avant tout, un maillage de base non raffiné, en pavé ou en rectangle, est fabriqué. Le niveau de raffinement de chacune de ses cellules est donc nul. Un programme est ensuite lancé pour remplir un tableau *stat* de la taille du nombre de cellules de zéros et de uns, indiquant respectivement qu'il faut ou non raffiner la cellule correspondante. Ce programme très

variable : il peut s'agir d'une simple caractérisation par coordonnées ou de l'usage d'un fichier STL, détaillé en 3.5. Ces cellules sont alors raffinées par l'appel de la subroutine *refine_mesh(stat)* et leur niveau de raffinement est incrémenté.

Afin de passer à un niveau de raffinement supérieur, il faut d'abord effectuer un travail de pré-raffinement pour s'assurer que le maillage est prêt à accueillir ce nouveau niveau de raffinement. En effet, si une cellule notée c est sélectionnée comme étant à raffiner et qu'elle est déjà au niveau de raffinement k, il faut s'assurer que ses voisins directs soient à ce même niveau k de raffinement avant de procéder à la division de la cellule c. Cette non-commutativité permet de diminuer le nombre de cas de figure susceptible d'apparaître au moment du découpage de la cellule. Comme il n'est de toute façon pas prévu que des raffinement de cellules voisines diffèrent de plus d'un niveau, cela n'affecte en rien la souplesse du mailleur. Cette phase de pré-raffinement est également l'occasion de s'assurer que les cellules ayant un sommet (2D) ou une arête (3D) en commun avec la cellule c aient un niveau de raffinement d'au moins k, de manière à ce qu'une fois c raffinée, l'écart de raffinement n'excède pas un. Là encore, il s'agit d'une mesure visant à réduire le nombre de cas particuliers à gérer lors de l'interpolation des valeurs aux faces. Cette préparation du maillage s'opère récursivement en appelant la même subroutine *refine_mesh*. Il est à noter que j'ai implémenté une fonction récursive qui prend en argument un tableau de taille différente à chaque appel.

3.3.4 Mise en œuvre

Gestion de la non structuration

Ces transformations simples et très visuelles cachent en réalité toute la complexité de travailler avec un maillage non structuré. Diviser une cellule implique de mettre à jour tous les attributs de la structure de données. Il faut commencer par ré-attribuer les tableaux de points, faces et cellules à leurs nouvelles dimensions, puis d'ajouter, dans cet ordres, tous les nouveaux points, toutes les nouvelles faces, puis toutes les nouvelles cellules. A l'ajout des faces, il est donc encore impossible de prédire les numéros des cellules dont elles sont l'interface.

Gestion des cellules fantômes

On gère les bords du domaine grâce des cellules fantômes. Dans le maillage de départ, celles-ci sont toujours les symétriques des cellules adjacentes par rapport à la face qui les sépare. Cette symétrie est conservée même si la cellule vient à être raffinée. Les cellules fantômes sont totalement ignorées lors d'une phase de raffinement ou de suppression des cellules extérieures au domaine et sont recrées en fin de processus. L'information du type d'une cellule fantôme n'est qu'une copie de l'information sur le type de la face adjacente. Ce doublon ne coûte pas de mémoire puisqu'il s'agit d'un attribut du type CellData qui est présent de toute manière pour les cellules standards.

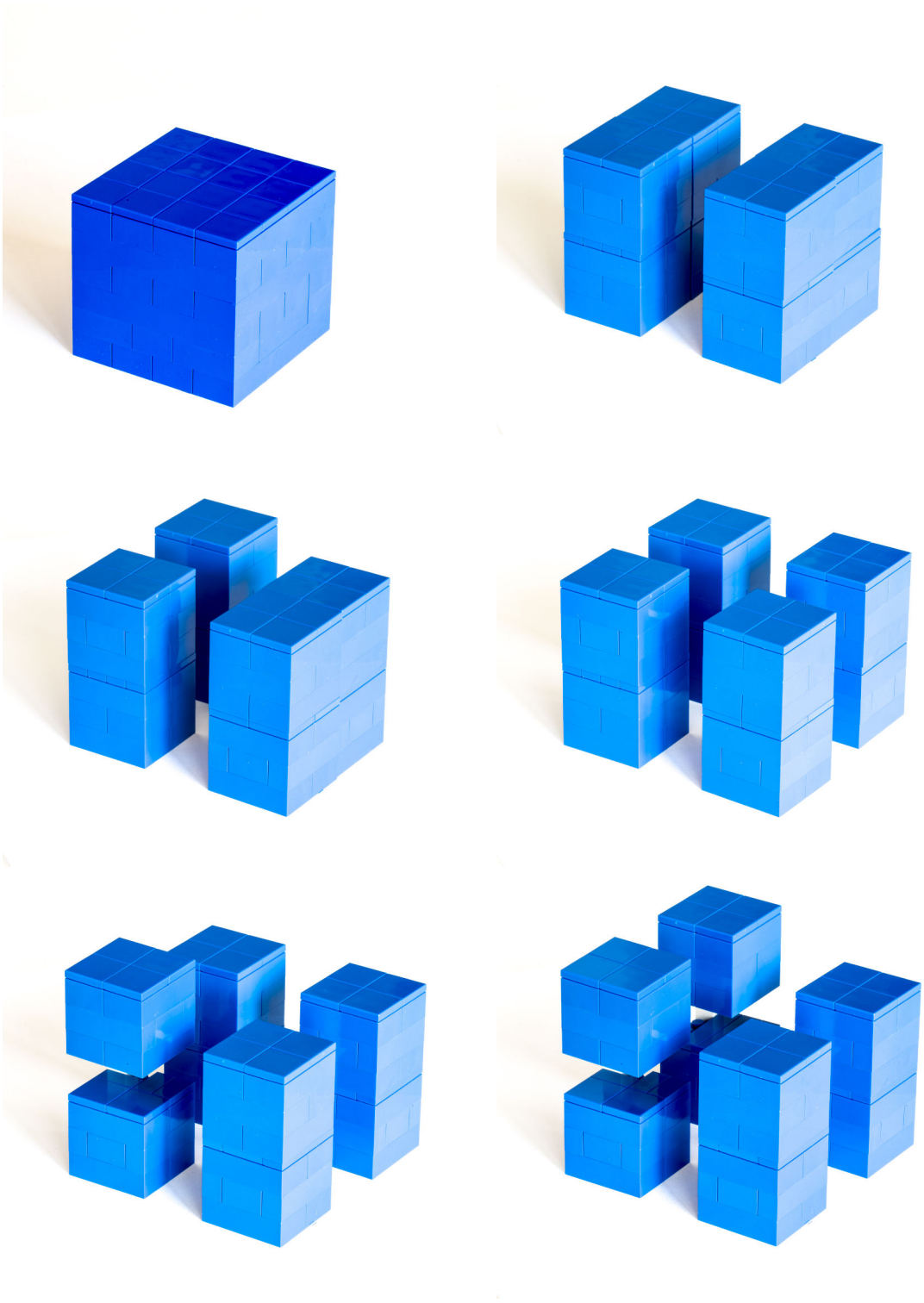


FIGURE 3.5 – Raffinement de cellule tridimensionnelle : illustration sommaire du processus de création des nouvelles faces et cellules. La manière de numéroter ces nouveaux éléments est commentée par des dessins au sein même des fichiers de code de NSIBM.

3.4 Interpolation

La vitesse est connue à chaque pas de temps au centre des cellules et nous en avons également besoin sur chaque face. Cette valeur a donc besoin d'être interpolée sur chaque face à l'aide des cellules situées de part et d'autre de celle-ci. Lorsque les deux mailles voisines ont le même niveau de raffinement, il est évidemment facile de faire la moyenne des vitesses qui s'y trouvent. Malheureusement tout raffinement génère une interface entre cellules de tailles différentes qu'il convient de traiter avec soin. Deux points méritent d'être rappelés ici :

- deux cellules voisines ne peuvent avoir plus d'un niveau de raffinement d'écart ;
- deux cellules ayant une troisième cellule en commun ne peuvent avoir plus d'un niveau de raffinement d'écart.

Ces deux propriétés permettent de limiter le nombre de configurations possibles au niveau de chaque face, et ainsi de réduire la complexité de l'interpolation, et de rendre le recours à l'interpolation moins fréquent. En effet, interpoler la valeur à l'interface de voisins déséquilibrés est plus couteux en temps machine qu'entre deux voisins de même taille et, à choisir, il convient mieux d'augmenter le nombre de cellules. Plusieurs cas de figure sont susceptibles d'apparaître.

3.4.1 Cas de cellules de même taille

C'est le cas où les deux cellules qui se partagent la face ont la même taille. Pour interpoler la valeur d'une grandeur physique Φ sur cette face, il suffit de moyenner les valeurs de cette grandeur aux centres des deux faces :

$$\Phi_{face} = \lambda \Phi_g + (1-\lambda)\Phi_d.$$

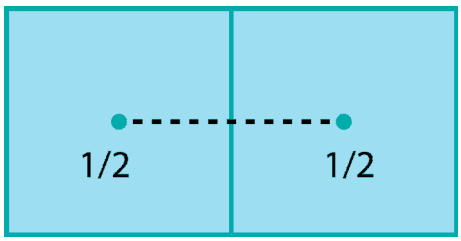


FIGURE 3.6 – Cas un pour un en 2D

3.4.2 Cas 2D de cellules de tailles différentes

Deux sous-cas sont à relever. En effet, pour calculer la valeur à l'interface entre une grande et une petite cellule, il est nécessaire d'interpoler une sous-valeur de la grande cellule qui soit alignée avec le centre de la face selon sa normale. Cette valeur est obtenue en utilisant les autres voisines de la grande cellule, avec une bonne pondération. Les schémas de la figure 3.7 résument ces valeurs. La principale difficulté est d'identi-

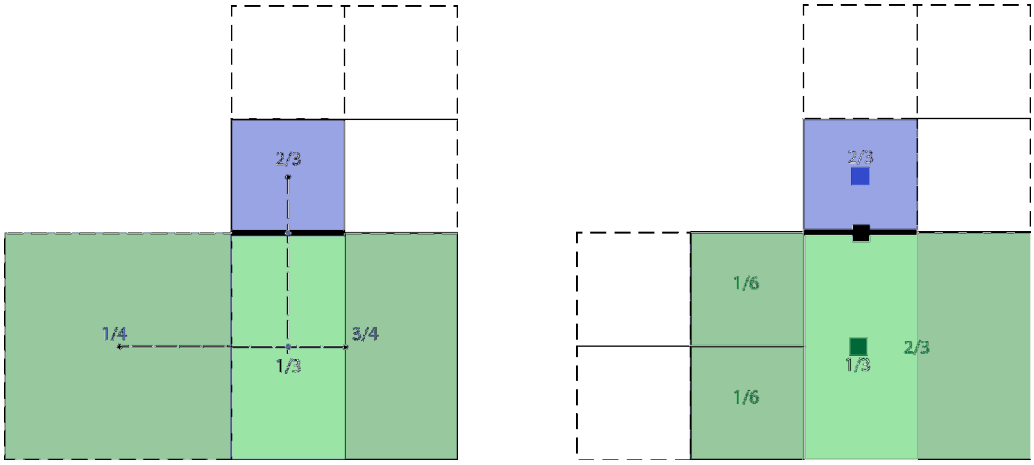


FIGURE 3.7 – Cas d'interpolation 2D

fier les bons voisins. Les coefficients sont stockés dans le tableau de réels **lambda** et les numéros des cellules dans le tableau d'entiers **cell** qui sont des attributs du type face. Ces éléments sont détaillés dans les tableaux 3.4 et 3.5.

3.4.3 Cas 3D de cellules de tailles différentes

Trois sous-cas sont à relever. Les schémas de la figure 3.8 résument ces valeurs. Afin d'obtenir un maillage le plus homogène possible, nous interdisons à deux cellules ayant un voisin en commun un écart de raffinement strictement supérieur à un. Un bénéfice important de ce critère est l'économie d'autres cas plus compliqués, qui nécessiteraient l'interpolation de valeurs au sein de deux cellules au moins. Cela rendrait également bien plus complexe la parallélisation du solveur et son optimisation. La valeur interpolée obtenue ici constitue la valeur moyenne sur la face.

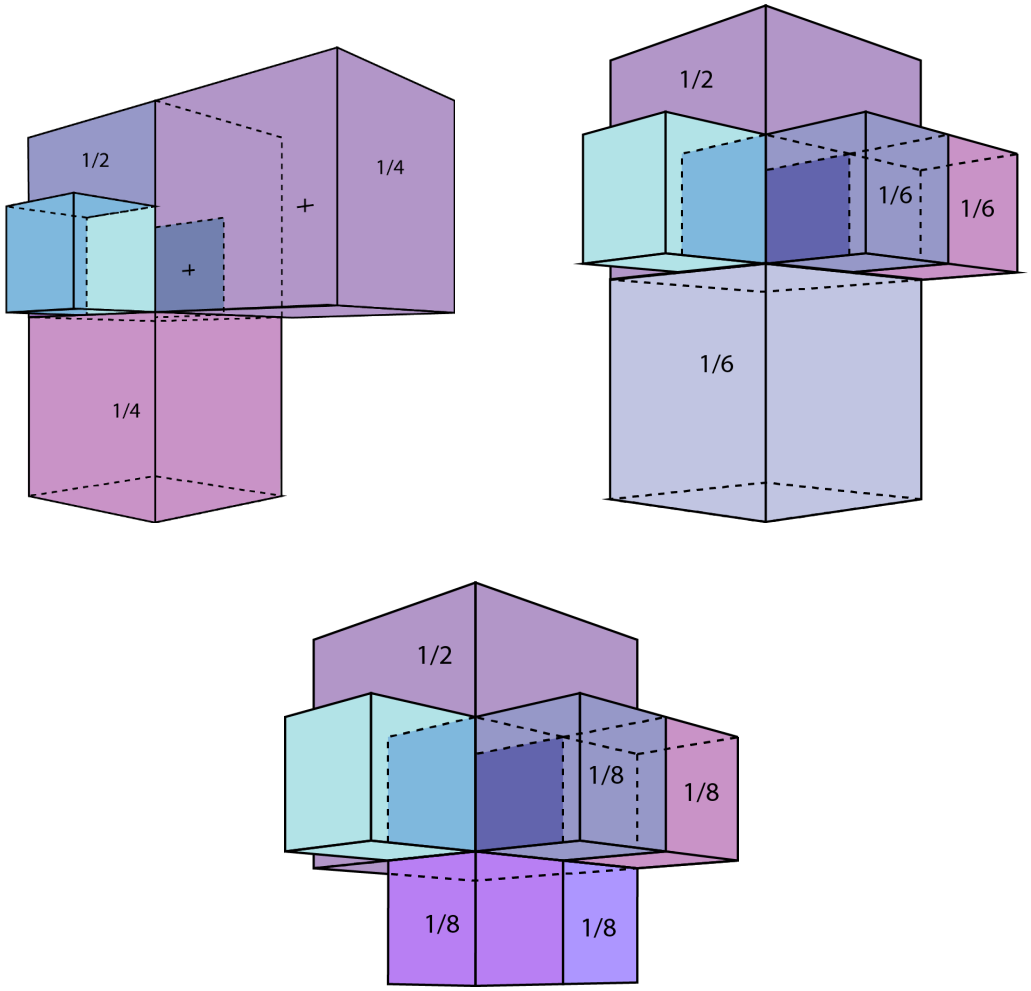


FIGURE 3.8 – Cas d'interpolation 3D ; Le premier cas sert d'exemple dans le tableaux 3.4 et 3.5.

3.5 Raffinement à partir d'un fichier source

Toute géométrie complexe doit être décrite afin de pouvoir être utilisée dans un solveur. Le format stl est le plus commun à cet effet.

3.5.1 Le fichier de stéréolithographie

Le fichier de stéréolithographie ou fichier STL est en fait un maillage à base de triangles d'une surface. La description qui suit est tirée de [69]. Un fichier STL ASCII commence toujours par

solid *name*

où *name* est une chaîne facultative (mais si *name* est omis il doit toujours y avoir un espace après le code solid). Le fichier continue avec n'importe quel nombre de triangles, chacun représenté comme suit :

facet normal n_i n_j n_k

outer loop
vertex $v1_x$ $v1_y$ $v1_z$
vertex $v2_x$ $v2_y$ $v2_z$
vertex $v3_x$ $v3_y$ $v3_z$
endloop
endfacet

où chaque n ou v est un nombre à virgule flottante au format e-signe-exposant comme par exemple : $-2.648000e-002$. Le fichier est clôturé par :

endsolid *name*

Les réels qui suivent facet normal sont les coordonnées du vecteur normal au triangle. Pour pouvoir utiliser le fichier STL dans notre implémentation de l'IBM, il est nécessaire que ces vecteurs normaux soient consistants : c'est-à-dire qu'ils pointent tous vers l'extérieur ou vers l'intérieur, de manière à pouvoir définir un intérieur et un extérieur à la surface. On trouve assez facilement des fichiers STL libres de droits, mais peu d'entre eux sont consistants. Il faut donc les éditer pour les rendre utilisables et ceci n'est pas toujours facile.

3.5.2 Lecture du fichier IBM

Nous nous servons d'un fichier STL pour décrire les géométries complexes. Une subroutine **ReadIBM** est chargée de son décryptage. Son rôle est de remplir un tableau de la taille du nombre de cellules nommé *ibmstat* d'indices correspondant aux types des cellules en question. L'indexation est :

- -1 pour une cellule à l'extérieur de la géométrie ;
- 0 pour une cellule à l'intérieur de la géométrie ;
- des valeurs supérieures à 1 pour les cellules à raffiner.

Une fois le tableau *ibmstat* rempli, les cellules indexées strictement positivement sont raffinées au degré choisi.

Chapitre 4

Parallélisation du code

« *Si les faits ne correspondent pas à la théorie, changez les faits.* »

Albert Einstein

Ce chapitre est consacré à la parallélisation du code. Il détaille l'utilisation de *Metis* et d'*OpenMPI*.

4.1 Partition du maillage

Une fois le maillage créé, il est nécessaire de le partitionner en le nombre de processeurs dont on dispose, afin de pouvoir leur partager le travail. Nous utilisons pour cela le programme **Metis** [70, 71, 72, 73, 74, 75] qui dispose d'une subroutine effectuant cette décomposition. La subroutine en question a besoin des informations d'entrée suivantes : liste des cellules, listes du nombre de nombre de voisins par par cellule, liste de ces voisins et de leurs pondération. En se basant sur la théorie des arbres, Metis construit un partage optimal, qui assure un équilibre entre les travaux des différents processeurs et une minimisation des communications nécessaires entre les processeurs.

4.1.1 Notion de voisinage

Metis a besoin de données concernant les liens entre les cellules pour opérer son algorithme. Il est nécessaire d'associer à chaque cellules ses voisins. Dans notre cas, la notion de voisinage est étendue aux cellules utiles aux interpolations sur les faces. Est donc un voisin d'une cellule toute cellule apparaissant dans la liste des cellules relatives à l'une des faces composant la cellule en question. Une fois un processeur attribué à chaque cellule, il est donné à cette dernière un numéro local. L'attribut **number** de CellData contient les trois informations numéro global, numéro de processeur et numéro local. Des tableaux locaux sont créés pour les faces et les cellules. Ils sont d'abord remplis avec des copies des cellules et des faces, puis mis à jour : les numéros globaux des faces et cellules dans les divers attributs des type CellData et FaceData sont remplacés par les numérotations locales. Ces tableaux locaux sont écrits dans des fichiers qui sont lus par les différents et les cellules fantômes sont créées.

4.1.2 Algorithme

Cette section détaille le processus de partition du solveur NSIBM, une fois que chaque cellule est associée à un processeur. L'algorithme d'attribution des numéros de processeurs effectué par Metis n'est pas détaillé ici. Son évolution fait l'objet des articles suivants : Karypis and Kumar [70, 71, 72, 73, 74], Karypis [75].

Initialisation

Á l'issue de la génération du maillage, raffiné ou non, on dispose de trois fichiers binaires contenant les données relatives aux points, aux faces et aux cellules. Ils constituent les données d'entrée de la subroutine *decomposeMetis* chargé de produire des fichiers semblables pour chaque processeur. Dans un premier temps, *decomposeMetis* demande à l'utilisateur combien de processeurs il désire utiliser. Ce nombre est noté $nproc$. Suit ensuite une phase de lecture de fichiers cités précédemment. Celui concernant les points est ignoré car inutile pour cette fonctionnalité.

Appel à Metis

Il faut maintenant construire les vecteurs contenant les informations nécessaires à Metis :

- un vecteur contenant à la suite les uns des autres les voisins de la cellule 1, puis 2, etc... ;
- un vecteur contenant le nombre de voisins de la cellule 1, puis 2, etc...

Ceux-ci sont obtenus en parcourant la liste des cellules, puis celles de leurs faces, puis l'attribut *cells* qui s'y trouve. On alloue un vecteur de la taille de la liste des cellule. La subroutine fait ensuite appel à Metis qui stocke dans ce troisième vecteur le numéro de processeur attribué à la cellule en question.

Production des zones du maillage

Dans un premier temps, *decomposeMetis* compte le nombre de cellules et de faces pour chaque processeur. Les faces sont ré-indexées localement et sont susceptibles d'avoir maintenant cinq indices puisque susceptible d'être à l'interface entre deux processeurs :

$$(n\_global, n\_proc\_1, n\_on\_proc\_1, \ n\_proc\_2, n\_on\_proc\_2).$$

Á partir d'ici commence le fonctionnement en parallèle. Dans tableaux sont alloués aux bonnes dimensions par chaque processeur puis les cellules et faces y sont concernées y sont recopiées. Il reste à remplacer les indices globaux par les indices locaux dans les attributs *cell* et *facelist* des types dérivés *FaceData* et *CellData*.

Cellules fantômes

Jusqu'à présent, chaque face situé au bord du domaine de calcul était pourvue d'une cellule fantôme symétrique à la cellule intérieure adjacente. Ces cellules traitées comme des cellules réelles dans la phase précédentes sont conservées dans les différentes zones du maillage. Ces nouvelles zones produisent de nouveaux bords fictifs. Ceux-ci sont traités de la même façon. Il y a donc maintenant davantage de cellules fantômes que dans le maillage d'origine. Les deux types de cellules fictives cohabitent et dans l'indexation négatives des tableaux locaux de cellules. Leur usage diffère cependant au moment du calcul en parallèle.

Limitation et solution à déployer

Certaines informations sont ignorées lors de la lecture du maillage global : seules importent les données qui concernent la topologie du maillage, c'est-à-dire les faces constituant les cellules et les cellules nécessaires aux interpolations des valeurs sur lesdites faces. Cette économie de mémoire est très vite essentielle : un maillage non structuré est toujours très gourmand en mémoire comparé à un maillage structuré du même nombre de mailles. Même avec cette précaution, il est indispensable de disposer d'au moins 16 Go de RAM pour décomposer un maillage de plusieurs millions de cellules. Cela représente un axe de développement de NSIBM qui est en tête de liste. La solution consiste à passer l'attribut *cell* du type *FaceData* au format dynamiquement allouable (*allocatable*). En effet, seule une faible proportion des faces requiert à plus de deux cellules pour l'interpolation, mais l'attribut *cell* contient constamment douze entiers. Le tableau contenant les faces est déjà allouable et la gestion d'allouables allouables se révèle assez capricieuse en Fortran. La gestion des accès mémoire n'est

justement pas triviale avec Fortran et de nombreux problèmes de débogage liés au passage au format allouable de l'attribut *facelist* du type *CellData*, lui-même faisant l'objet d'un tableau allouable, nous ont convaincus de remettre l'implémentation de cette évolution à une date ultérieure.

Ce procédé peut également être appliqué aux numéros des faces et permettrait de passer de cinq entiers systématiques par face à trois pour la plupart d'entre elles.

Une fois ces évolutions implémentées, il ne sera plus possible de réduire la consommation en mémoire du maillage puisque toutes les autres données sont indispensables et sont stockées de manière optimale.

La phase de preprocessing s'achève ici.

4.2 Communication entre les processeurs

4.2.1 Principe

Ici commence la phase de calcul : on lance le module de résolution numérique sur autant de processeurs que l'on a de zones du maillage. Tout d'abord, les fichiers de géométrie sont lus, intégralement cette fois-ci, et mis en mémoire. Des fichiers d'initialisation détaillant la modélisation de l'écoulement et les paramètres de résolution numérique comme le schéma numérique sont également lus.

Une subroutine est ensuite appelée pour que les processeurs s'informent mutuellement des cellules qu'il doivent échanger. En effet, le maillage est divisé en $nproc$ zones qui correspondent chacune à un maillage local sur un processeur. Les frontières entre ces zones, formés par des faces, dessinent une frontières entre les données accessibles depuis chacun des processeurs.

Les cellules de part et d'autre de chaque face frontière ne sont donc pas accessibles depuis un même processeur. Or, calculer les valeurs des grandeurs physiques sur une telle face requiert l'accès à ces deux cellules, et probablement même certaines autres. Il s'avère donc indispensable de pouvoir y accéder depuis une même nœud de calcul.

4.2.2 Exemple explicatif

Considérons le processeur $proc\_2$ et imaginons qu'il partage une interface avec les processeurs $proc\_1$, $proc\_3$, $proc\_6$ et $proc\_12$. Le terme *interface* ne désigne pas simplement une face commune, mais plus généralement, signifie que pour pouvoir interpoler des grandeurs sur l'une ou l'autre de ses faces, le $proc\_2$ requiert des données issues de mailles résidant dans les zones attribuées aux processeurs $proc\_1$, $proc\_3$, $proc\_6$ et $proc\_12$, en plus de celles dont il dispose évidemment au sein de sa propre zone.

Chaque processeur boucle ses faces pour déterminer les ses besoins et prévoit par la même occasion les données dont il dispose et qu'un de ses voisins lui demandera. Il stocke ses voisins dans le tableau *neighbours* et les tailles des interfaces dans le tableau *connectivity_size*. Les échanges sont effectués via *OpenMPI* [76] par l'intermédiaire de tableaux d'entiers. Des exemples de ces tableaux sont données sur les figures 4.1 et 4.2.

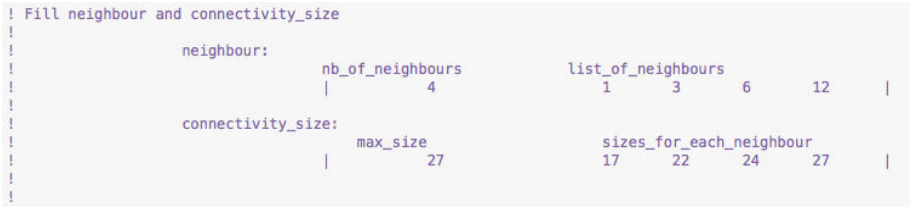


FIGURE 4.1 – Tableau des voisins : ceci est une capture d'écran des commentaires présents dans le code.

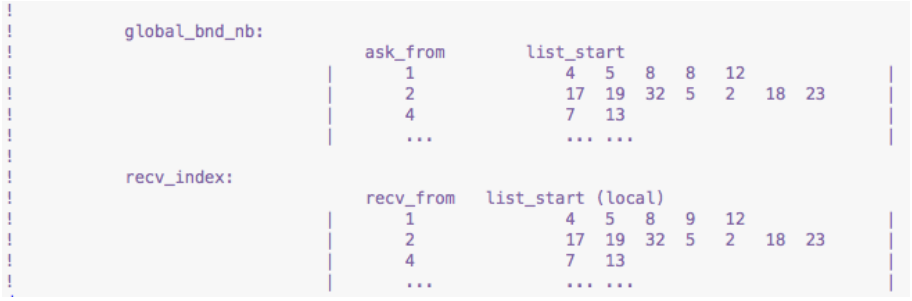


FIGURE 4.2 – Tableau des tailles d'interface : ceci est une capture d'écran des commentaires présents dans le code.

4.2.3 Gestion des interfaces

Les cellules distantes sont intégrées à chaque zones comme des cellules fantômes, dans la partie des indices négatifs du tableau des cellules. Elles sont du type *iConn*, représenté par l'entier 6 (voir tableau 3.3).

4.2.4 Déroulement d'un calcul en parallèle

Á chaque pas de temps, le système est résolu localement par chaque processeur sur la zone qui lui a été attribuée. Avant de passer au pas de temps suivant, on attend que chaque processeur ait envoyé sa mise à jour des cellules à partager. En fin de simulation, toutes les données sont rapatriées sur le premier processeur et écrites dans un fichier de visualisation. Un fichier est également crée pour chaque zone, de manière à pouvoir continuer la simulation au cas échéant.

4.3 Exemples de maillages partitionnés

4.3.1 En 2D

La figure 4.3 montre un maillage 2D de la marche descendante raffiné quatre fois au niveau de la principale recirculation. Ce maillage ne s'est cependant pas avéré être un choix judicieux : l'importance du raffinement est accrue au niveau des parois et de la marche, et non au niveau de la recirculation. Les vitesses sont en effet faibles au sein du tourbillon et ce sont les interactions fluide/paroi qui en sont à l'orgine. Le maillage finalement utilisé pour traiter ce cas ce trouve au chapitre suivant, sur la figure 5.11.

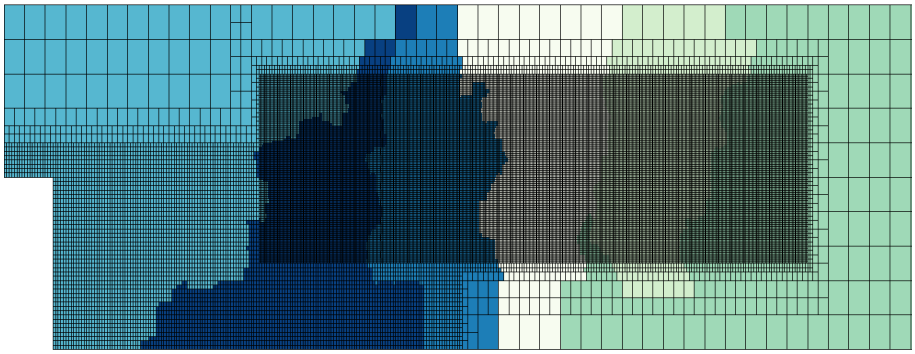


FIGURE 4.3 – Raffinement sur la marche descendante : ce maillage de 50000 de cellules doté de 4 niveaux de raffinement a été décomposé sur 6 processeurs.

Les 50000 mailles qui le composent ont été partitionnées sur six processeurs, chacun représenté par un couleur. Les frontières entre les processeurs sont moins rectilignes que ce à quoi on aurait pu s'attendre, mais il va nous falloir nous en contenter car c'est ce que *Metis* nous fournit. Il pourrait être intéressant (et long) d'explorer plus profondément les possibilités de *Metis*, voire de tester d'autres algorithmes similaires, comme *Scotch*.

4.3.2 En 3D

La figure 4.4 exhibe un maillage 3D d'une sphère raffinée quatre fois à sa surface. Le million de mailles qui le compose a été partitionné sur vingt-cinq processeurs, chacun représenté par un couleur.

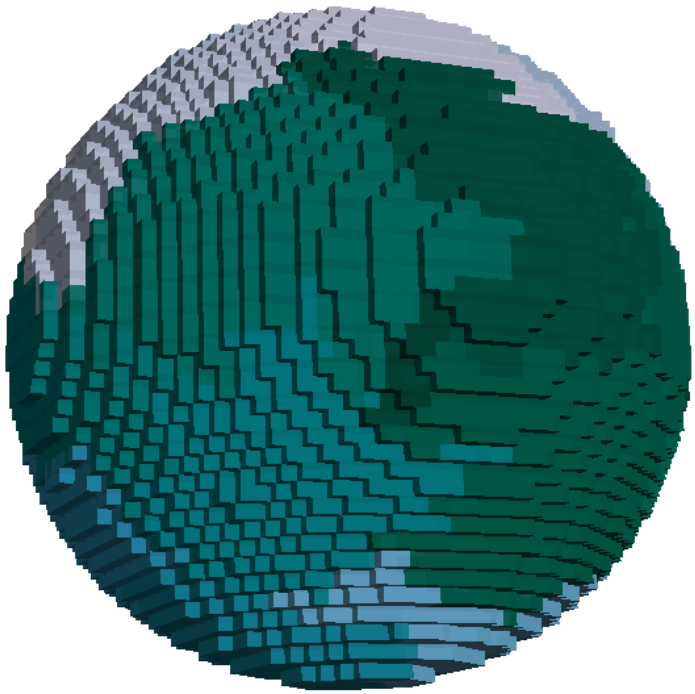


FIGURE 4.4 – Raffinement d'une sphère : ce maillage d'un million de cellules doté de 4 niveaux de raffinement a été décomposé sur 25 processeurs

4.4 Gain de temps

4.4.1 Principe

Le but de ce procédé est évidemment de gagner du temps, l'idéal étant de diviser le temps de calcul par le nombre de processeurs. Pour avoir une scalabilité linéaire, il faudrait que tous les processeurs soient en action à tout moment de la simulation et que le nombre total de calculs ne soit pas plus important en parallèle qu'en séquentiel. Le temps de preprocessing n'est pas comptabilisé dans la durée du calcul. L'opération de partitionnement est cependant assez rapide. Les opérations les plus longues y sont les écritures en mémoire et sur des fichiers.

Les opérations calculatoires faites en parallèles prennent le même temps qu'en séquentiel pour chaque processeur : les manipulations à faire pour résoudre le système d'équations à chaque itération temporelle sont en effet identiques. Á ce niveau-là, il n'y a pas de perte d'efficacité.

Le principal ralentissement se produit pendant la phase de communication entre les nœuds. Premièrement, une fois obtenue la solution à la $n^{\text{ième}}$ sur le premier processeur à finir les calculs, celui-ci se met en attente jusqu'à ce que tous les autres soient arrivés à son avancement et soient prêts pour l'échange des données mises à jour. On comprend ici l'importance de synchronisés les unités de calculs en leur attribuant des travaux de même durée. On peut envisager d'utiliser des processeurs de puissances diverses, à condition de bien calibrer la proportion du maillage à leur confier. Cela n'a en général rien de trivial. Il est préférable de partager équitablement le travail à des processeurs de même puissance.

Considérons par exemple une simulation de $k = 20000$ itérations temporelles sur $nproc = 1000$ cœurs identiques et imaginons que la distribution des mailles soit telle que l'un des processeurs mette $d = 1/100^{\text{ième}}$ de seconde de plus que la moyenne des autres processeurs à résoudre un pas de temps. Le temps perdu par les processeur au chômage technique sur l'ensemble de la simulation est alors :

$$delay = k\,d(nproc-1) = 199800\text{s} = 55{,}5\text{h} \tag{4.1}$$

soit plus de deux jours.

Il est malheureusement impossible de synchroniser parfaitement les travaux. Du temps sera perdu ici en comparaison du cas idéal.

Deuxièmement, la phase de partage des données entre les processeurs constitue un dose d'opérations à effectuer en plus de ceux nécessaires en cas de résolution séquentielle et condamnent l'utopique scalabilité linéaire.

4.4.2 Scalabilité de NSIBM

Les résultats de la scalabilité (*scaling*) ssont détaillés dans cette section : il s'agit de la proportion de temps de calcul gagnée relativement au nombre de processeurs. Trois tests ont été réalisés.

Le premier (figure 4.5) montre le *speed up* obtenu en passant une simulation de la cavité entrainée en 3D en parallèle sur un HPC.

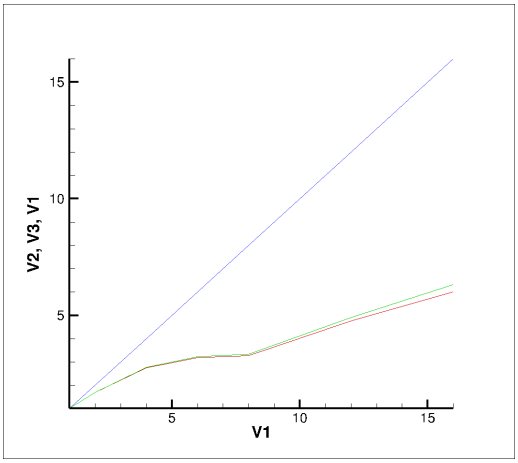


FIGURE 4.5 – Speed up sur un noeud de CFD : scaling parfait en bleu, boucle principale en vert et PSBLAS en rouge

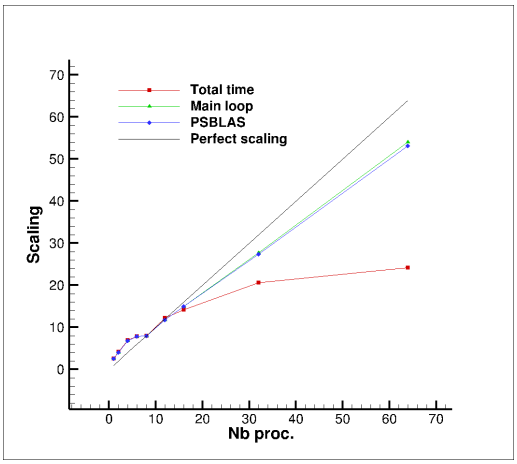


FIGURE 4.6 – Speed up sur 2 millions de mailles

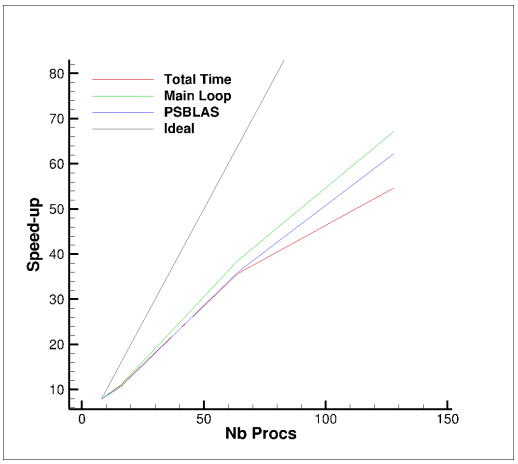


FIGURE 4.7 – Speed up sur 15 millions de mailles

Le second (figure 4.6) montre le *speed up* obtenu en passant une simulation de la cavité entrainée en 3D sur un maillage à deux millions de cellules en parallèle sur HPC.

Le dernier (figure 4.7) montre le *speed up* obtenu en passant une simulation de la cavité entrainée en 3D sur un maillage à quinze millions de cellules en parallèle sur HPC.

Les courbent présentent une bonne scalabilité. Le meilleur moyen d'encore la faire progresser est diminuer encore la tailles des interfaces entres les zones, c'est-à-dire en ayant de meilleurs résultats de *Metis*.

Chapitre 5

Applications

« *La rigueur vient toujours à bout de l'obstacle.* »

Léonard de Vinci

5.1 Production de maillage

Le principal apport de NSIBM par rapport aux codes utilisés jusque-là au laboratoire iCube, notamment NSMB, est le mailleur non-structuré avec raffinement basé sur l'IBM. Il est capable de produire facilement des maillages pertinents pour des géométries complexes. Á ce jour, le solveur n'est pas encore prêt à résoudre toutes les simulations les concernant, mais cela ne saurait tarder. Cette section est destinée à illustrer les qualité de la production de maillages dans NSIBM.

5.1.1 Poumon de rat

Le premier test de génération de maillage concerne un poumon de rat décrit par un fichier STL. Un aperçu du maillage est donné sur la figure 5.1. NSIBM n'est pas encore capable de résoudre ce cas pour des raisons indépendantes du maillage : notre schéma numérique ne permet actuellement que gérer les conditions limites sous forme explicite. Cela ne pose pas de problème au entrées ou sur les parois où des conditions de Dirichlet sont appliquées, mais génère trop de perte aux sorties où des conditions de Neumann ont cours.

Le maillage du rat a été obtenu en partant d'un pavé dont les longueurs sont obtenues par lecture du fichier STL avec les dimensions suivante :

— $nx = 65 \times 4$;

— $ny = 55 \times 4$;

— $nz = 115 \times 2$.

L'option de suppression des cellules est primordiale pour ce genre de géométrie.

La manière dont le mailleur est implémenté permettrait également de produire ce style de maillage en partant d'une seule cellule et en la raffinant à répétition. Cependant, cette méthode ne nous semble pas idéale en séquentiel puisque la génération du pavé de départ ainsi que la suppression de cellules font partie des opérations les moins coûteuses en temps, tandis que le raffinement est l'opération la plus lourde. En effet, chaque augmentation de l'ordre de raffinement implique une recopie du maillage dans des tableaux plus grands. Il manque malheureusement à Fortran une méthode pour ré-allouer dynamiquement des tableaux.

5.1.2 Poumon humain

Nous avons ensuite réussi à mailler des poumons humains, bien plus complexes que ceux du rat, visible sur la figure 5.2. Grâce à l'IBM qui interprète le fichier STL et sélectionne automatiquement les cellules à supprimer et à raffiner, obtenir ce genre de maillage ne prend que peu de temps à l'utilisateur, qui n'a besoin que de modifier quelques lignes dans le programme de preprocessing et d'un fichier STL bien orienté.

Le temps de production d'un tel maillage est tout à fait raisonnable (15 minutes pour le poumon humain et 10 minutes pour celui du rat) et en tous les cas négligeable devant le temps de calcul total que requiert une géométrie aussi complexe. La majeure partie du temps est consommée lors du test destiné à décider quelles cellules initiales sont à l'intérieur et lesquelles sont à l'extérieur. Pour ces deux exemples, environ 80% des mailles sont supprimer lors de cette phase.

FIGURE 5.1 – Le poumon de rat

5.1.3 Profil d'aile de NACA 0012

Nous avons également discrétiser l'espace bi-dimensionnel autour d'un profil d'aile de NACA 0012 avec une incidence cd 12ř. La figure 5.3 illustre ce cas. La simulation de écoulement passant l'aile a été réalisée pour des nombre de Reynolds allant de 200 à 1200 et donne de bons résultats. Ceux-ci ne sont pas discutés de manière approfondie dans ce présent manuscrit.

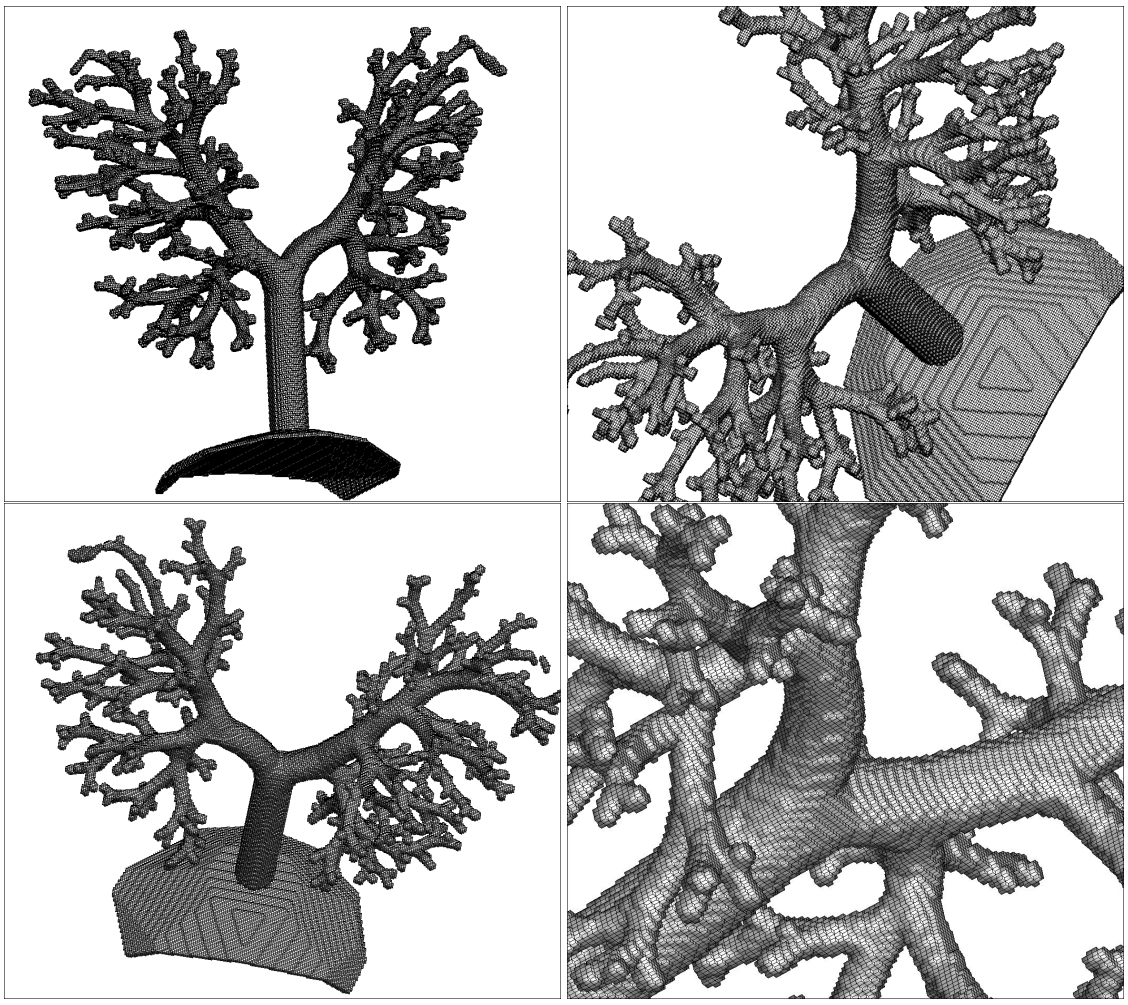


FIGURE 5.2 – Le poumon humain

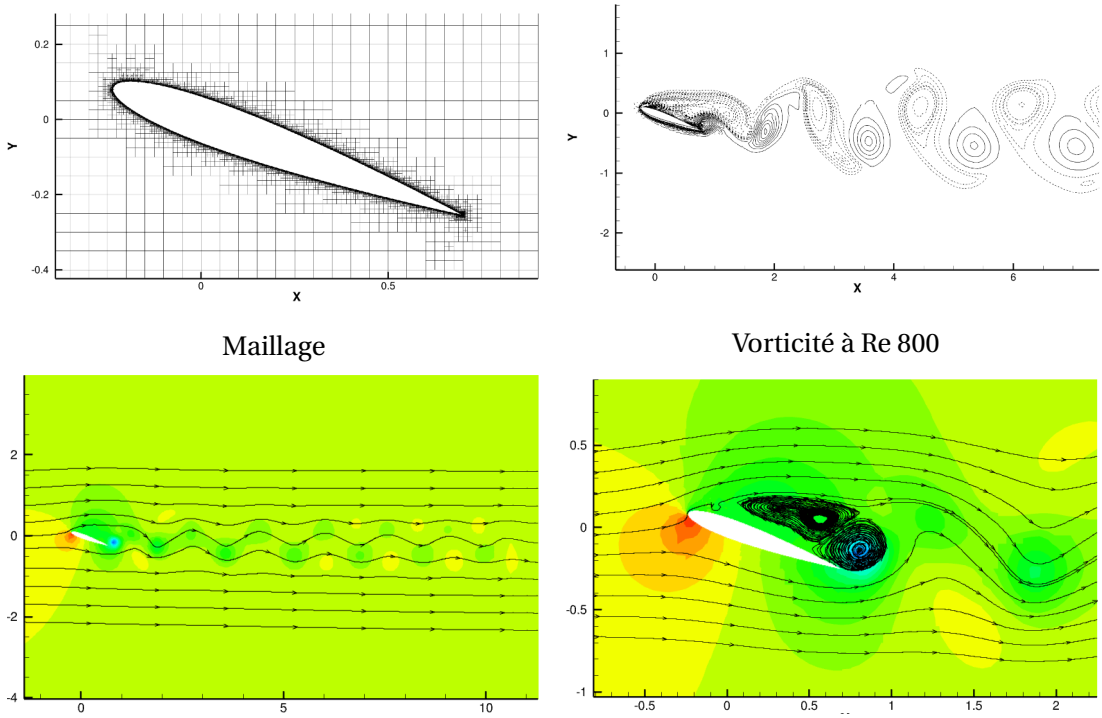


FIGURE 5.3 – Le NACA 0012

5.2 La cavité entraînée 2D

5.2.1 Description

La cavité entraînée (*Lid-driven cavity*) sert de cas test de base pour vérifier la bonne implémentation d'un solveur de Navier-Stokes depuis l'article de Ghia et al. [77]. Il s'agit d'une configuration particulièrement simple : un carré bidimensionnel dont les trois murs inférieurs sont fixes et le mur supérieur est mobile, de vitesse constante u_{lid}, entrainant une recirculation principale du fluide au milieu de la cavité et, selon la valeur du nombre de Reynolds, des recirculations secondaires dans les coins. Facile à implémenter, puisque le maillage et les conditions initiales sont triviaux, elle est également révélatrice puisqu'elle manifeste pleinement la non-linéarité de l'équation de Navier-Stokes.

La séparation et le ré-attachement de fluides turbulents sont en effet très fréquents dans le domaine de l'ingénierie, que ce soit dans des systèmes comme des diffuseurs, combusteurs ou canaux à forte expansion ou comme l'écoulement autour d'une aile d'avion. Il est connu que plus le nombre de Reynolds augmente, plus ces phénomènes sont accentués. Le nombre de Reynolds est calculé comme suit :

$$\text{Re} = \frac{u_{\text{lid}} L}{\nu}. \tag{5.1}$$

Un schéma numérique implicite d'ordre deux a été utilisé pour cette simulation. La

figure 5.4 illustre ce cas.

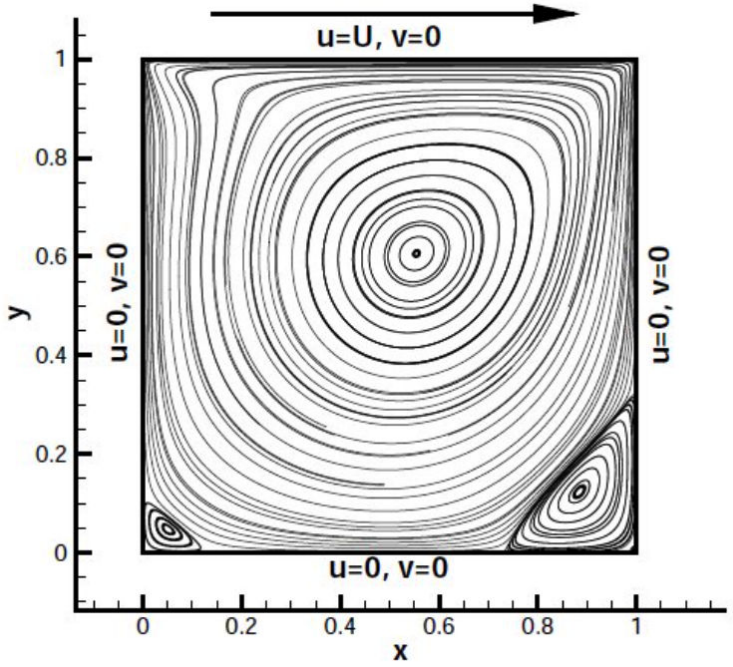
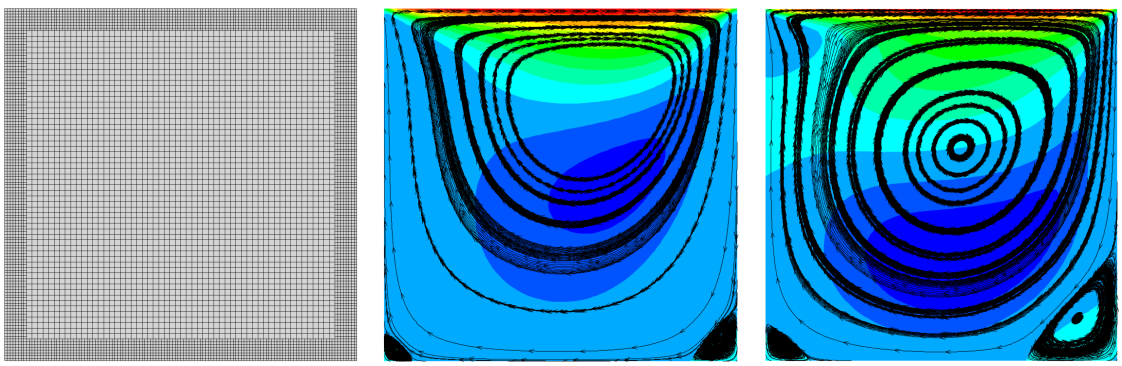


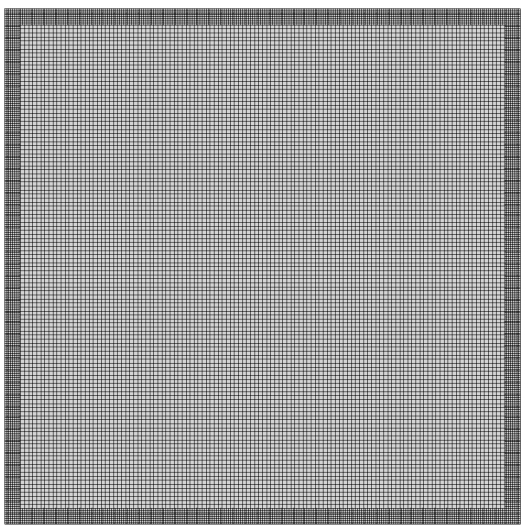
FIGURE 5.4 – Configuration de la cavité entraînée en 2D

5.2.2 Validation

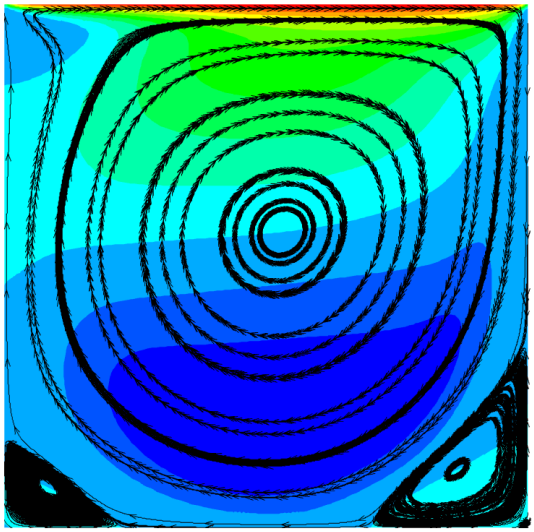


Le maillage de taille 64 × 64 Re 100 Re 400

FIGURE 5.5 – La cavité entrainée résolue avec un niveau de raffinement au bord à Re 100 et 400



Le maillage de taille 128 × 128



Re 1000

FIGURE 5.6 – La cavité entrainée résolue avec un niveau de raffinement au bord à Re 1000

On trouve dans l'article de Ghia et al. [77] une description aussi bien qualitaive que quantitative des résultats de la cavité entrainée, notamment les données dimensionnelles des diverses recirculations. La figure 5.7 est issue de l'article et illustre ce propos.

Les profils de vitesses dans les plans médians sont comparés avec l'article de Ghia et al. [77] dans les figures 5.8 et 5.9. On conclut à de bons résultats pour NSIBM.

Cette étude permet la validation du code dans le cas suivant : écoulement bidimensionnel avec raffinement.

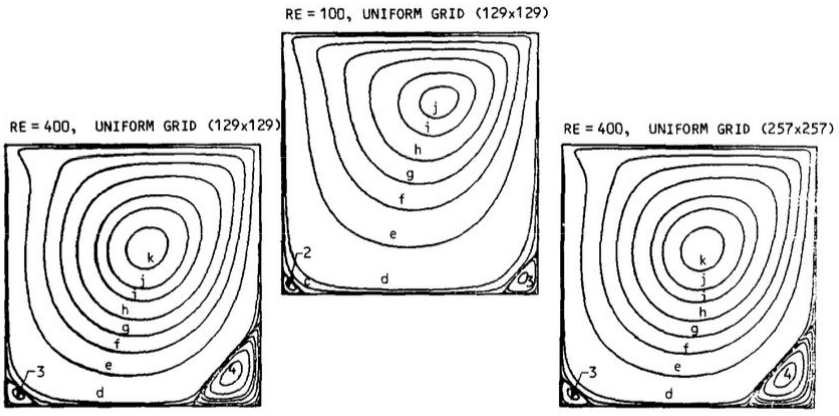


FIGURE 5.7 – Cavité entrainée : données extraites de Ghia et al. [77]

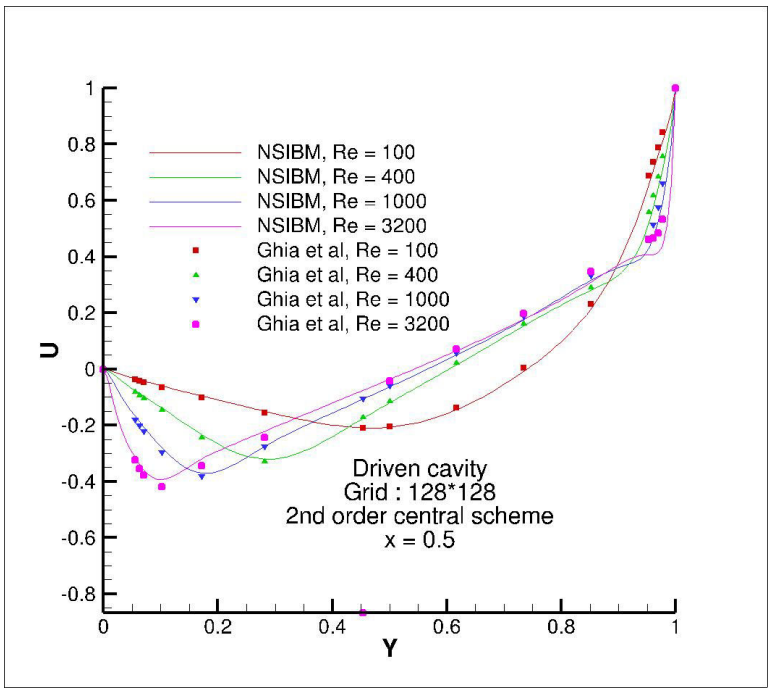


FIGURE 5.8 – Cavité entrainée : comparaison des vitesses selon l'axe x pour $x = 0.5$

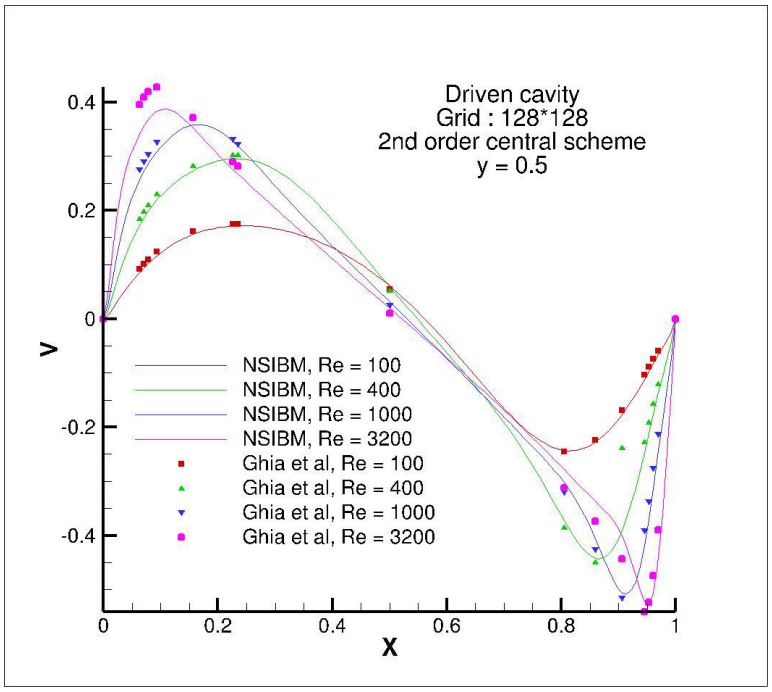


FIGURE 5.9 – Cavité entrainée : comparaison des vitesses selon l'axe y pour $y = 0.5$

5.3 Écoulement laminaire 2D passant une marche descendante

5.3.1 Description

On se base sur l'article de Armaly et al. [78], décrivant l'expérience illustrée sur la figure 5.10 ci-dessous. Le fluide en écoulement horizontal dans un canal est soumis à une brusque expansion en passant une marche descendante. Le tunnel d'entrée est suffisamment long pour que l'écoulement soit stationnaire et le tunnel de sortie suffisamment long pour observer toutes les perturbations générées par l'expansion. Les dimensions sont celles de l'article de Armaly et al. [78] cité précédemment :

- hauteur de la marche : $s = 0,0049m$
- hauteur du tunnel d'entrée : $h = 0,0051m$
- longueur du tunnel d'entrée : $l = 6s$
- longueur du tunnel expansé : $L = 0,6m = 60 \cdot (s+h)$

Elles sont résumées sur la figure 5.10, tirée de la thèse de Bennetsen [79]. Une recirculation collée à la marche est observée pour tout nombre de Reynolds et sa longueur $x_{sép}$ est mesurée expérimentalement et calculée numériquement dans l'article de Armaly et al. [78]. Les simulations faites avec NSIBM pour mesurer ce phénomène sont illustrées par les figures 5.12 à 5.14. Á Reynolds 450 et au-delà, on observe également une zone de recirculation secondaire, au niveau de la paroi supérieure, qui débute à peu près à l'abscisse à laquelle la recirculation primaire s'arrête. L'article de Armaly et al. [78] contient également des données concernant cette seconde zone. Les figures 5.15 et 5.16 montrent les résultats obtenus avec NSIBM concernant ces zones. Une triple comparaison est faite pour valider notre solveur NSIBM ce cas : les données de Armaly et al. [78], valeurs numériques et expérimentales, et les données de la thèse plus récente de Bennetsen [79] sont utilisées. Les calculs sont faits à Reynolds 200, 450 et 1000 et des différentes longueurs, normalisées par la hauteur de la marche, sont résumées dans les tableaux 5.1 et 5.2. Dans le cas de la marche descendante, les recirculations présentes sont la conséquence du choc du fluide sur les parois, ce qui rend essentielle leur bonne gestion. J'ai donc généré un maillage précis aux bords du domaine et plus grossier à l'intérieur. Un aperçu en est donné sur la figure 5.11.

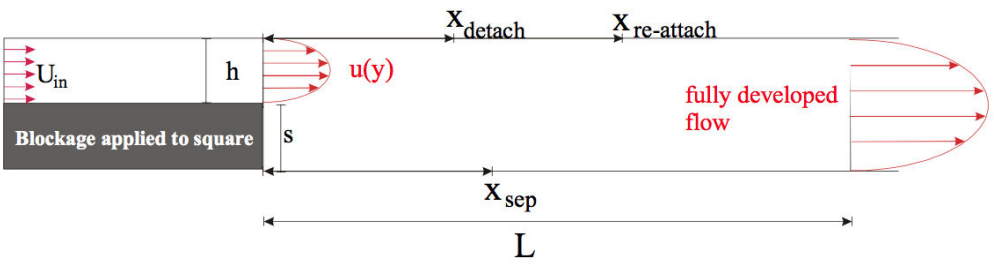


FIGURE 5.10 – Configuration de la marche descendante en 2D

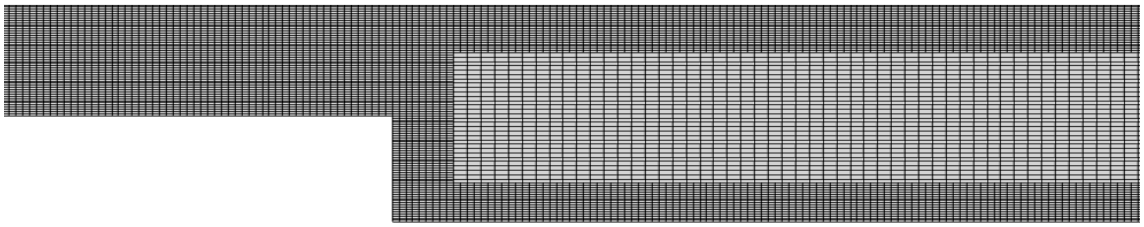


FIGURE 5.11 – Maillage de la marche descendante en 2D

5.3.2 Résultats

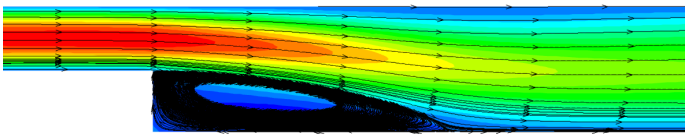


FIGURE 5.12 – La marche descendante à Re 200

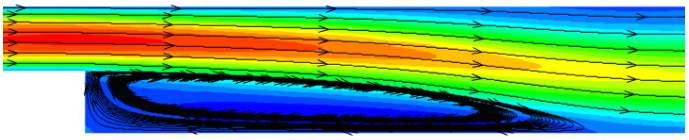


FIGURE 5.13 – La marche descendante à Re 450

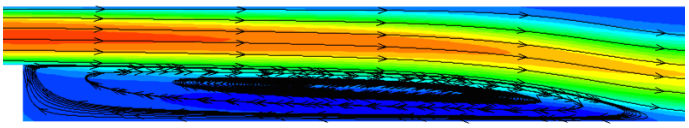


FIGURE 5.14 – La marche descendante à Re 1000

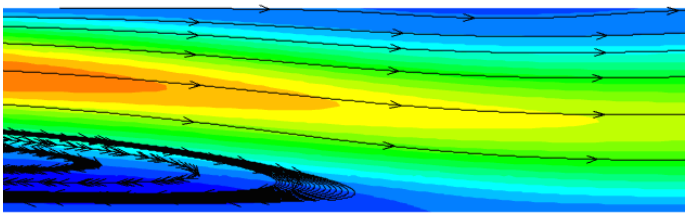


FIGURE 5.15 – Recirculation secondaire à Re 450

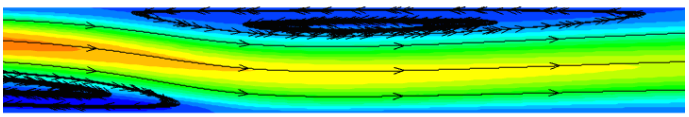


FIGURE 5.16 – Recirculation secondaire à Re 1000

Pour un nombre de Reynolds allant jusqu'à 450, les résultats de notre code NSIBM sont cohérents avec les valeurs expérimentales et les valeurs expérimentales faisant référence à l'heure actuelle. Au-delà, la taille de la recirculation décline en comparaison des observations expérimentales faites par Armaly et al. [78]. Différents essais effectués avec plusieurs maillages ont tous aboutis au même résultat, même avec un maillage particulièrement fin et raffiné. L'article de Kim and Moin [80] mentionne ces écarts entre les résultats numériques et les observations expérimentales qui apparaissent entre Re = 500 et Re = 600. Une étude de raffinement de maillage montre que cette différence entre les résultats numériques et expérimentaux n'est pas la conséquence d'erreurs numériques. Cet écart commun aux trois résultats numériques considérés ici peut s'expliquer par le fait que l'écoulement ne reste pas strictement bidimensionnel au-delà de Re = 400, comme le font remarquer Armaly et al. [78]. Une étude 3D de ce cas devrait donc être réalisée pour obtenir des résultats plus proche de la réalité à Reynolds plus élevé.

Cepend ant, le but n'est pas ici l'étude complète du cas physique mais la validation de notre solveur NSIBM. Les résultats obtenus sont en accord avec l'expérience à Reynolds inférieur à 500 et avec les autres solveurs bidimensionnels à Reynolds plus élevé. Cette étude nous montre que NSIBM n'est pas mis en défaut de manière numérique et permet la validation du code dans le cas suivant : écoulement bidimensionnel avec raffinement et suppression de cellules.

Recirculation primaire	**Armaly & al.** 1983 Expérimental	**Armaly & al.** 1983 Numérique	**Bennetsen** 1999 Numérique	**NSIBM** 2015 Numérique
Re = 200	5.29	5.12	5.20	5.10
Re = 450	9.43	8.40	8.70	8.47
Re = 1000	16.43	7.40	11.60	11.43

TABLEAU 5.1 – Comparaison de la longueur de réattachement normalisée $x_{sép}/s$ de la recirculation primaire

Recirculation secondaire	**Armaly & al.** (1983)		**Bennetsen** (1999)		**NSIBM** (2015)	
	$x_{\text{détach}}/s$	$x_{\text{ré-attach}}/s$	$x_{\text{détach}}/s$	$x_{\text{ré-attach}}/s$	$x_{\text{détach}}/s$	$x_{\text{ré-attach}}/s$
Re = 450	7.84	11.30	7.70	11.60	7.76	10.61
Re = 1000	13.50	21.90	9.40	21.70	9.18	20.61

TABLEAU 5.2 – Comparaison des longueurs de détachement et de réattachement normalisées de la recirculation secondaire

5.4 Écoulement laminaire 2D autour d'un barreau cylindrique

Description du cas

Il s'agit d'un écoulement bidimensionnel horizontal de gauche à droite autour d'un obstacle carré de dimension $D = 1$. Le côté gauche est l'entrée du fluide dont on a imposé $u = 1$. Les bords haut, bas et droit du domaine sont des sorties. Aucune symétrie n'est mise en place ici. La figure 5.17 montre le maillage utilisé et illustre la description du cas. Le centre du carré est placé en $(0,0)$ et le domaine est prévu suffisamment grand autour du carré : $10D$ devant, au-dessus et en-dessous du carré et $15D$ derrière, de manière à bien pouvoir suivre les tourbillons générés par la présence de l'obstacle. Les simulations sont faites à Re = 200, 250, 300, 400 et 500.

Simulation

Le maillage suivant est utilisé :

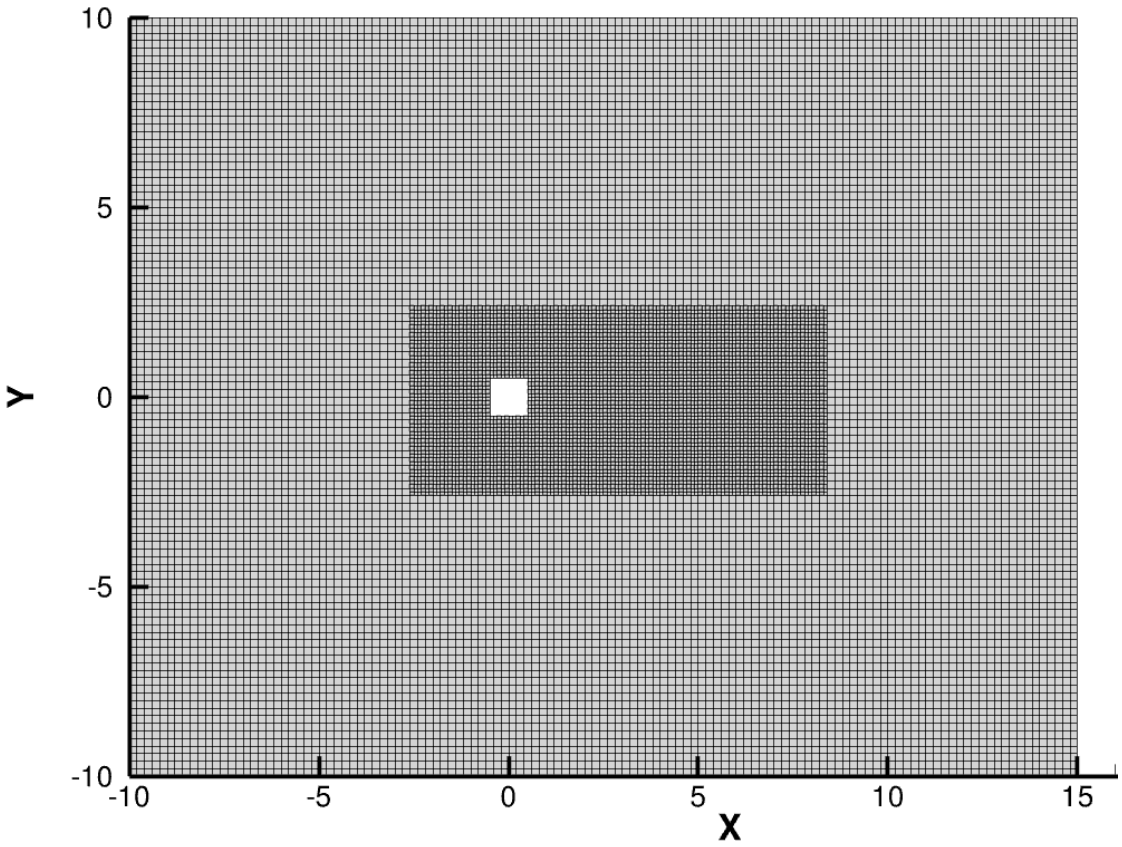


FIGURE 5.17 – Maillage du "cylindre carré" en 2D

Il a été généré à partir d'un rectangle en 125x100 en raffinant la partie centrale et en supprimant les cellules à l'intérieur du carré. C'est un maillage raisonnable qui permet un temps de calcul assez court. La simulation a été faite en parallèle, sur 12 processeurs, en instationnaire. Les résultats présentés montrent le temps $t = 20s$, après 20 000 itérations.

Bibliographie du "cylindre carré"

Ce cas de figure, répertorié en tant que "cylindre carré", tire son origine dans son analogie avec le cas d'un écoulement (tridimensionnel) autour d'un cylindre, au sens mathématique du terme. Les deux cas sont si proches que la dénomination de cylindre a été conservée dans la littérature pour ce qui est en réalité un pavé droit, souvent à base carrée.

Bien que d'apparence simpliste, ce cas est en réalité primordial en pratique : les structures intégrant des éléments (quasi-)rectangulaires sont légions : éléments architecturaux, bâtiments, poutres, clôtures apparaissent dans de nombreux écoulements intérieurs ou extérieurs. En pratique, à partir d'un certain nombre de Reynolds, appelé "critique", l'écoulement autour d'une vraie structure en pavé droit mènent à la formation de tourbillons. Ces tourbillons provoque une variation des forces s'appliquant à la structure et sont susceptibles de provoquer une vibration de la structure, du bruit, voire une entrée en résonance pouvant altérer la structure [81].

Plusieurs études numériques similaires d'écoulement autour d'un pavé droit ont déjà été effectuée qui, tout en conservant des conclusions proches, présentent des différences qui révèlent leur dépendance au maillage choisi ([82, 83, 84]).

D'après l'article de Sohankar et al. [85], l'écoulement passe de stationnaire à instationnaire à $Re \sim 50$.

Résultats

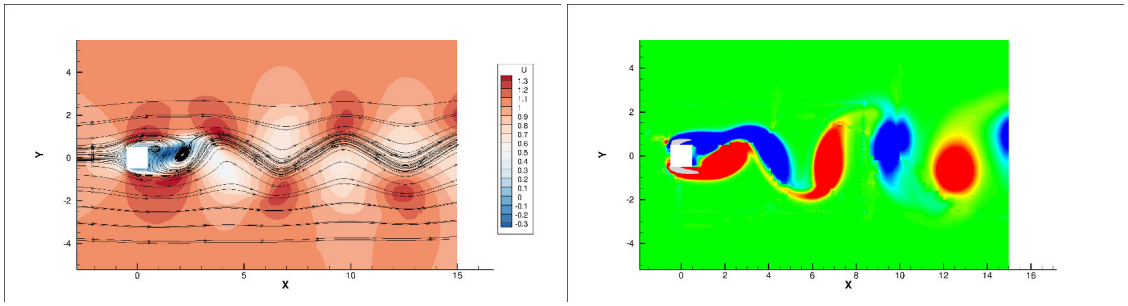


FIGURE 5.18 – $t = 20s$ à Re=200 - Vitesse u et vorticité selon z.

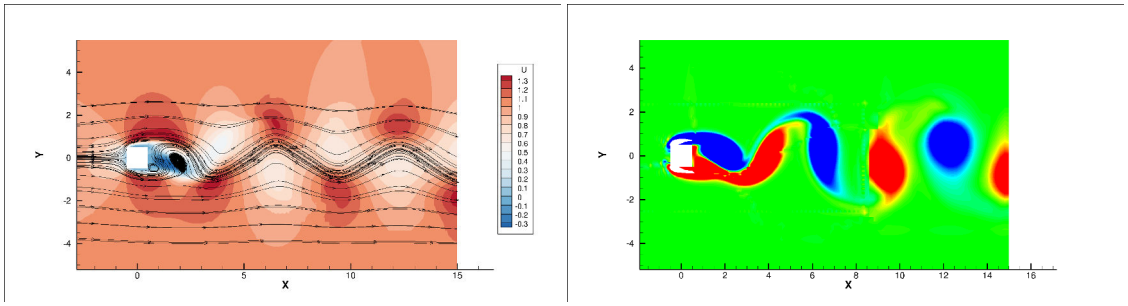


FIGURE 5.19 – $t = 20s$ à Re=250 - Vitesse u et vorticité selon z.

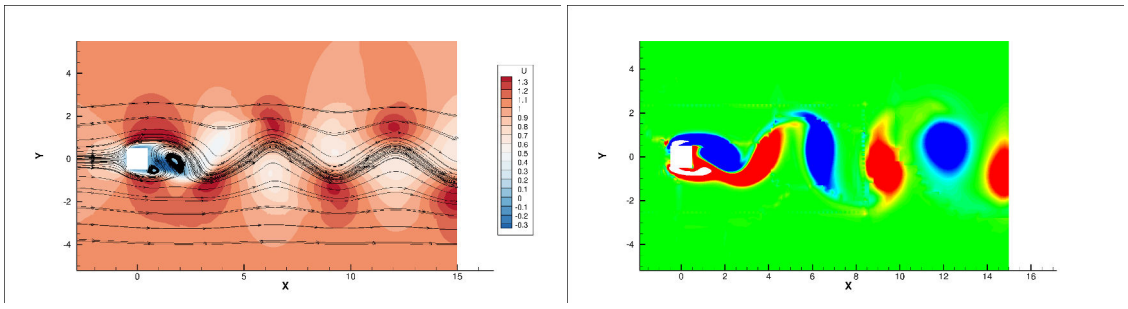


FIGURE 5.20 – $t = 20s$ à Re=300 - Vitesse u et vorticité selon z.

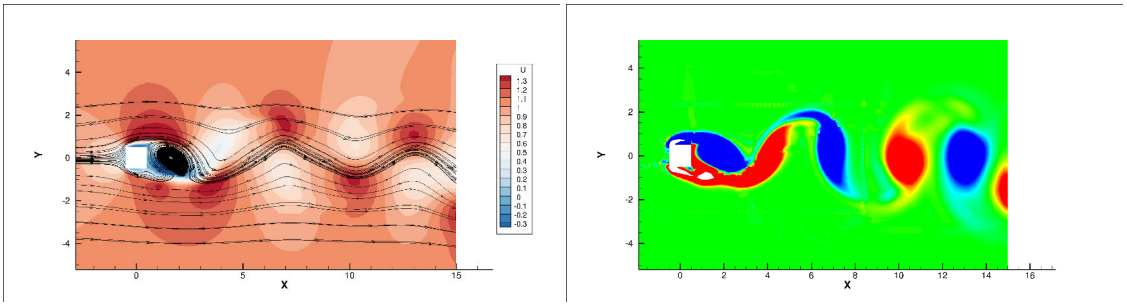


FIGURE 5.21 – $t = 20s$ à Re=400 - Vitesse u et vorticité selon z.

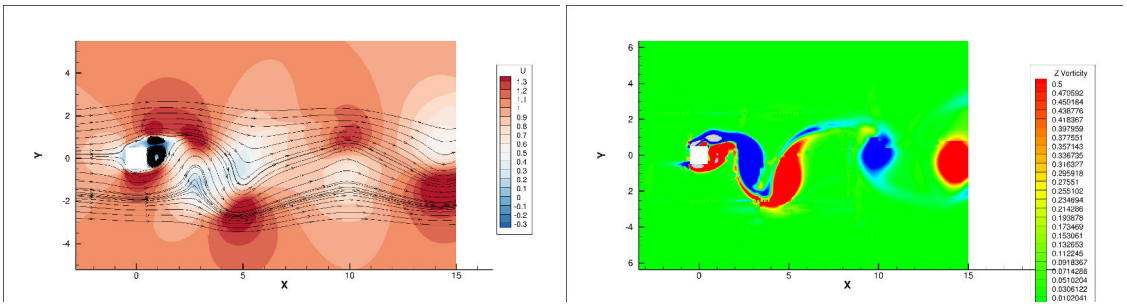


FIGURE 5.22 – $t = 20s$ à Re=500 - Vitesse u et vorticité selon z.

5.4.1 Comparaison et conclusions

L'article de Doolan [84] compare deux simulations numériques dont seul le raffinement de maillage diffère : un premier maillage (A) est doté de cellules plus grandes aux abords du carré et reste régulier derrière celui-ci, tandis que le second (B) est plus précis sur les contours mais se disperse ensuite.

Les résultats obtenus, bien que proches, révèlent ces différences : des tourbillons épars sont visibles au niveau des angles du carré sur le cas A, mais sont absent du cas B, où le tourbillon principal s'éloigne doucement des arêtes du carrés. À l'inverse, le dessin des tourbillons est plus nette sur le maillage A, les écarts de tailles de cellules du cas B entraînant une dispersion des bords au fur et à mesure que l'on s'éloigne du carré.

Les résultats obtenus ici avec NSBIM sont semblables à ceux de l'article de Doolan [84] : les longueurs de tourbillons sont du même ordre, les petits tourbillons aux arêtes du cas A commencent à faire leur apparition et le dessin des grands tourbillons est net comme au cas B. La figure 5.23 résume cela.

Source	Re	St	Cx moyen
NSIBM	200	0.155	1.47
Sohankar et al. [83]	200	0.170	1.46

TABLEAU 5.3 – Comparaison à Re 200

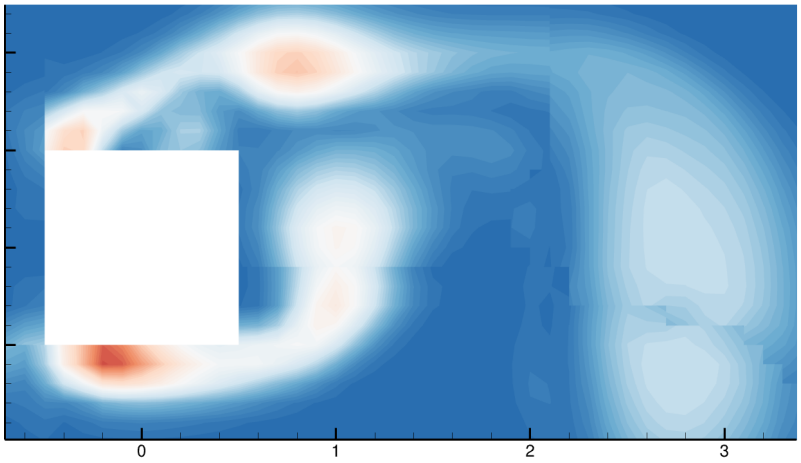


FIGURE 5.23 – $t = 20s$ à Re=500 : zoom sur les abords du carré révélant les petits tourbillons au niveau de arêtes ainsi que des grands tourbillons nets

5.5 Écoulement laminaire 2D autour d'un barreau cylindrique

5.5.1 Description

On étudie ici le cas d'un écoulement laminaire bidimensionnel autour d'un barreau cylindrique à nombre de Reynolds modéré. La simulation est réalisée avec une section de diamètre $D = 1$. Le maillage s'étend sur $10D$ avant le barreau et $15D$ derrière pour en suivre la traînée. Dessus et dessous, une marge de $10D$ est prise afin qu'il n'ait pas de perte de fluide au niveau de la perturbation générée par le cylindre. Le bord gauche est une entrée de fluide où la vitesse est :

$$\begin{pmatrix} u \\ v \end{pmatrix}_{\text{left}} = \begin{pmatrix} u_\infty \\ 0 \end{pmatrix}$$

avec $u_\infty = 1\text{m/s}$. Les autres bords du domaine sont tous des sorties. Le maillage utilisé ici est illustré par la figure 5.24. Il est admis dans la littérature que cet écoulement est stationnaire jusqu'à Reynolds 49,9 et qu'une phase d'instabilité de von Karman débute ensuite.

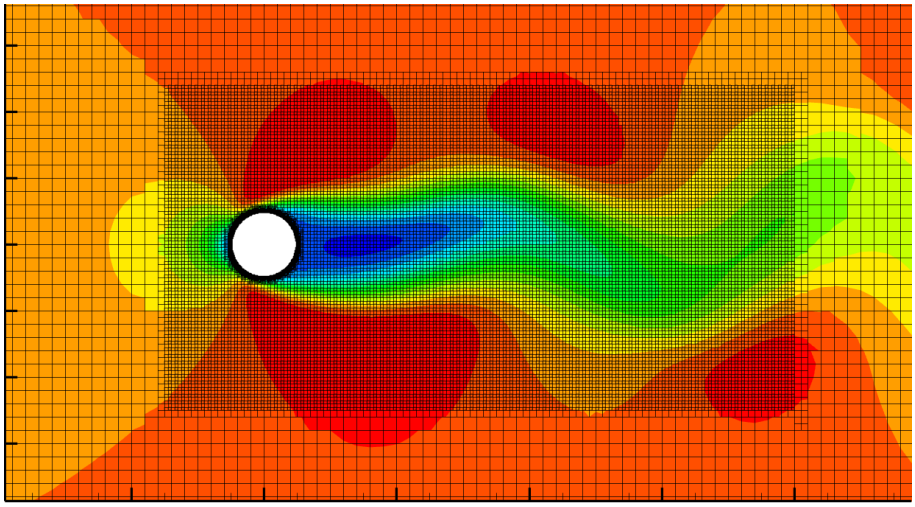
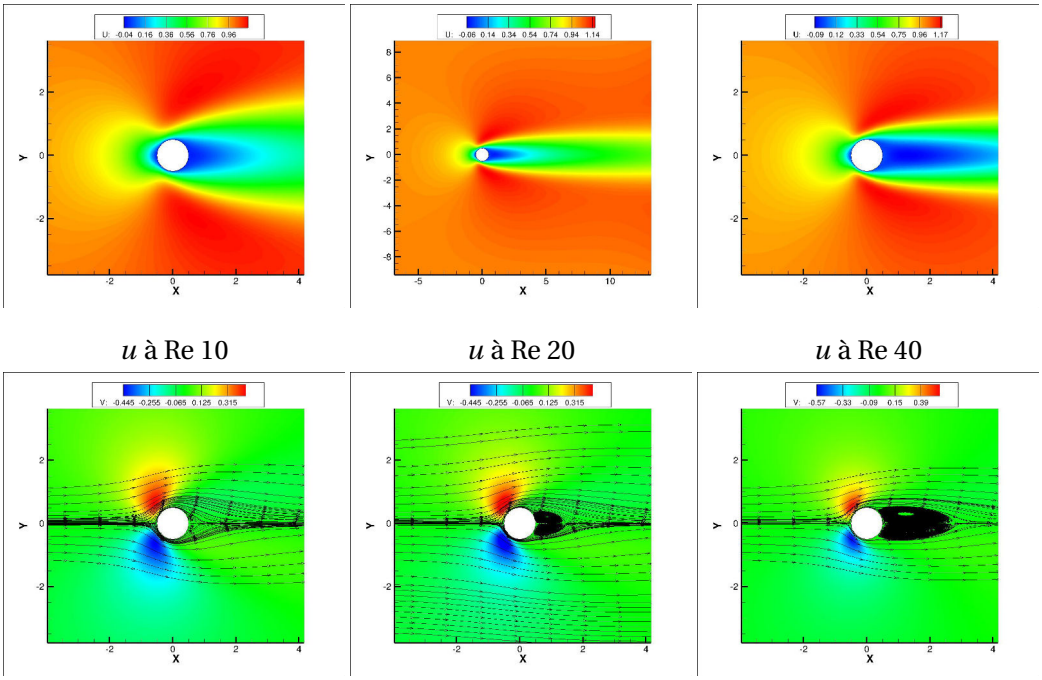


FIGURE 5.24 – Maillage du barreau cylindrique 2D : il a été généré en partant d'un maillage rectangle de 125 × 100 mailles et en raffinant jusqu'à quatre fois la zone autour de la paroi du cylindre. Les cellules solides à l'intérieur du cylindre ont été supprimée de la géométrie. Un fichier STL n'a pas été utile pour produire ce maillage, mais sert au moment de la simulation dans l'IBM pour compenser le crénelage de la section du cylindre par les mailles cartésiennes.

5.5.2 État stationnaire

Une première série de tests a été effectuée à faible Reynolds, entre 10 et 40, où l'écoulement est stationnaire. Cet état est documenté depuis longtemps et les résultats produit par NSIBM son comparés à la littérature existante. Les données généralement commentées sont la longueur de la recirculation, l'angle de décollement et le coefficient de traînée. La figure 5.25 illustre les simulations de l'écoulement stationnaire.

Les comparaisons sont compilées ci-après dans les graphes 5.28, 5.27 et 5.26. Elles permettent de valider le comportement du solveur NSIBM dans le cas d'une simulation d'un écoulement stationnaire bidimensionnel avec un maillage raffiné dont parois sont géréés par la méthode des frontières immergées.



u à Re 10 | *u* à Re 20 | *u* à Re 40

v à Re 10 | *v* à Re 20 | *v* à Re 40

FIGURE 5.25 – État stationnaire du barreau cylindrique 2D : les profils de vitesses selon les axes x et y sont représentés respectivement sur la première et la deuxième ligne. Les lignes de courant montrent une recirculation axisymétrique conforme aux attentes et à la littérature existante sur ce cas.

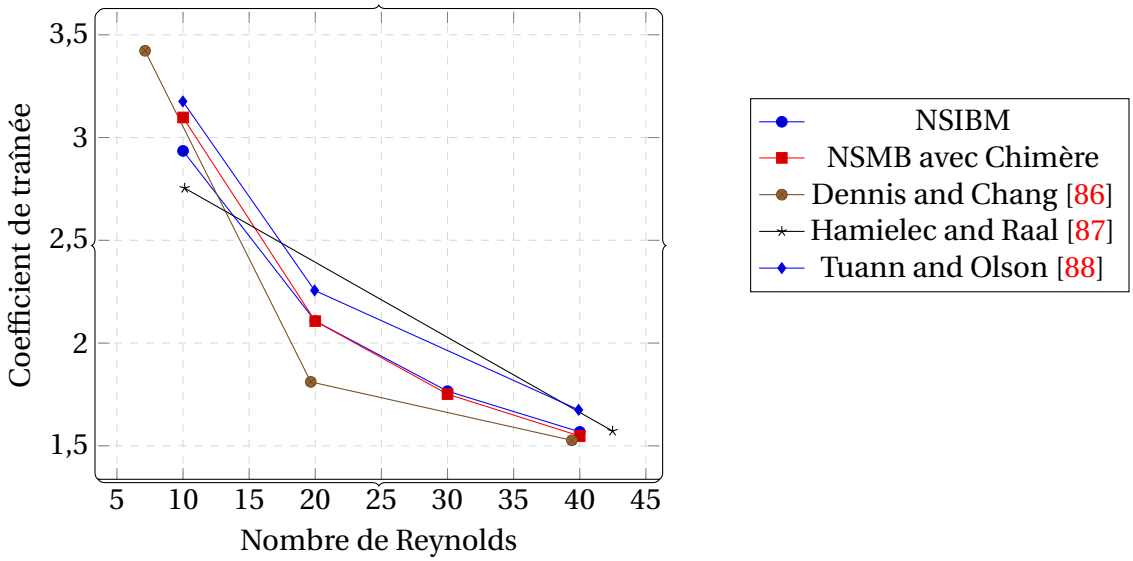


FIGURE 5.26 – Cylindre 2D : comparaison des coefficients de traînée. Les résultats obtenus avec NSIBM sont comparés aux autres résultats numériques obtenus par NSMB et la méthode chimère et ceux trouvés dans Dennis and Chang [86], Hamielec and Raal [87] et Tuann and Olson [88].

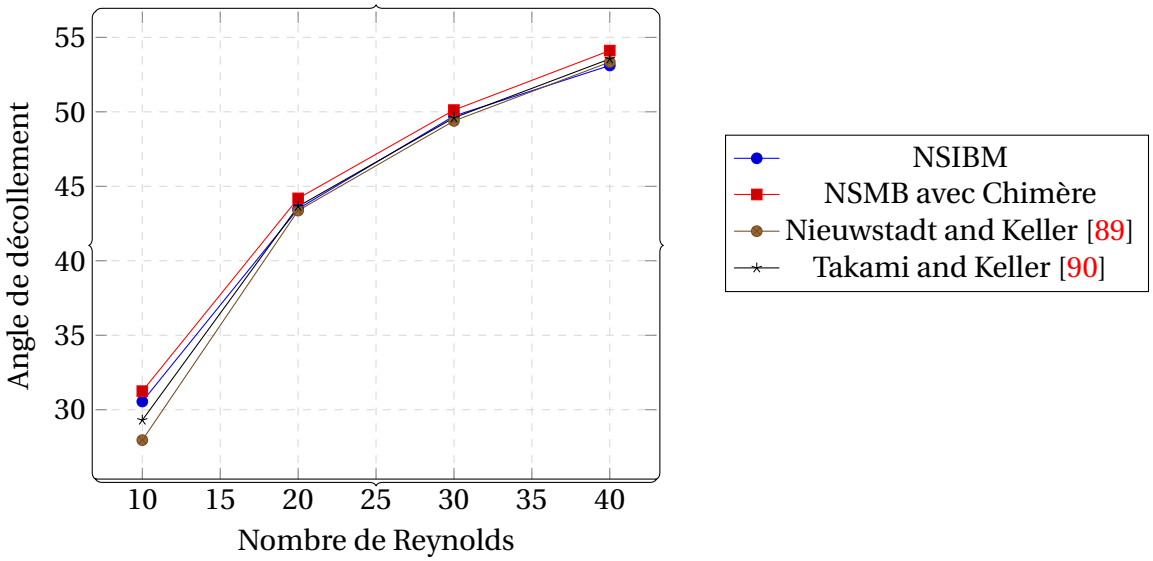


FIGURE 5.27 – Cylindre 2D : comparaison des coefficients de traînée. Les résultats obtenus avec NSIBM sont comparés aux autres résultats numériques obtenus par NSMB et la méthode chimère et ceux trouvés dans Nieuwstadt and Keller [89] et Takami and Keller [90].

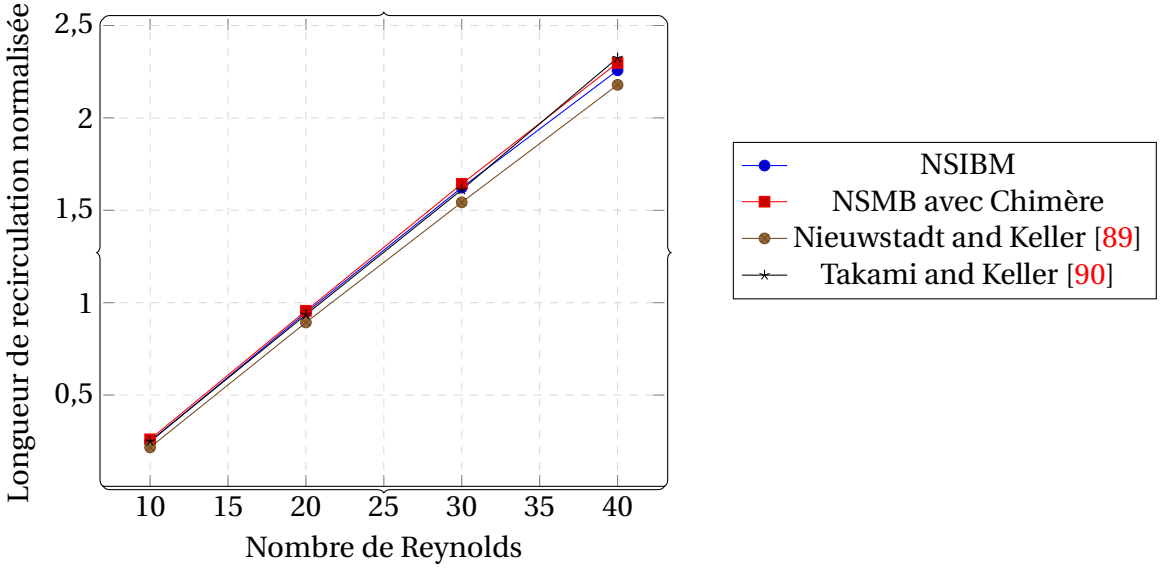


FIGURE 5.28 – Cylindre 2D : comparaison des longueurs de recirculation. Les résultats obtenus avec NSIBM sont comparés aux autres résultats numériques obtenus par NSMB et la méthode chimère et ceux trouvés dans Nieuwstadt and Keller [89] et Takami and Keller [90].

5.5.3 État instationnaire

Une seconde série de tests a été réalisée à Reynolds entre 100 et 300. Dans cette plage, il est consensuellement admis que l'écoulement est instationnaire avec un lâcher périodique de tourbillons dans le sillage de l'obstacle. Il n'est donc plus pertinent de se focaliser sur les mêmes données que dans le cas stationnaire. Nous allons plutôt nous concentrer sur la fréquence d'émission tourbillonnaire. Á cet effet, il est judicieux de considérer le nombre de Strouhal, nommé d'après le physicien tchèque Vincent Strouhal qui l'a introduit en 1878, défini par

$$St = \frac{fD}{u}, \tag{5.2}$$

où

f est la fréquence d'émission des tourbillons ;

D est la longueur caractéristique, ici le diamètre de la section du cylindre ;

u est la vitesse de l'écoulement non perturbé.

Le nombre de Strouhal correspond donc à la fréquence des lâchers tourbillonnaires normalisée.

Le graphe 5.34 compare les résultats de notre solveur NSIBM aux données obtenues avec le code NSMB via la méthode Chimère et celles issues des articles de Williamson and Govardhan [91] et de Persillon and Braza [92]. Il permet de s'assurer que NSIBM donne de bons résultats dans le cas d'un écoulement instationnaire bidimensionnel avec un maillage raffiné.

La figure 5.29 illustre le lâcher tourbillonnaire à Reynolds 100, 200 et 300.

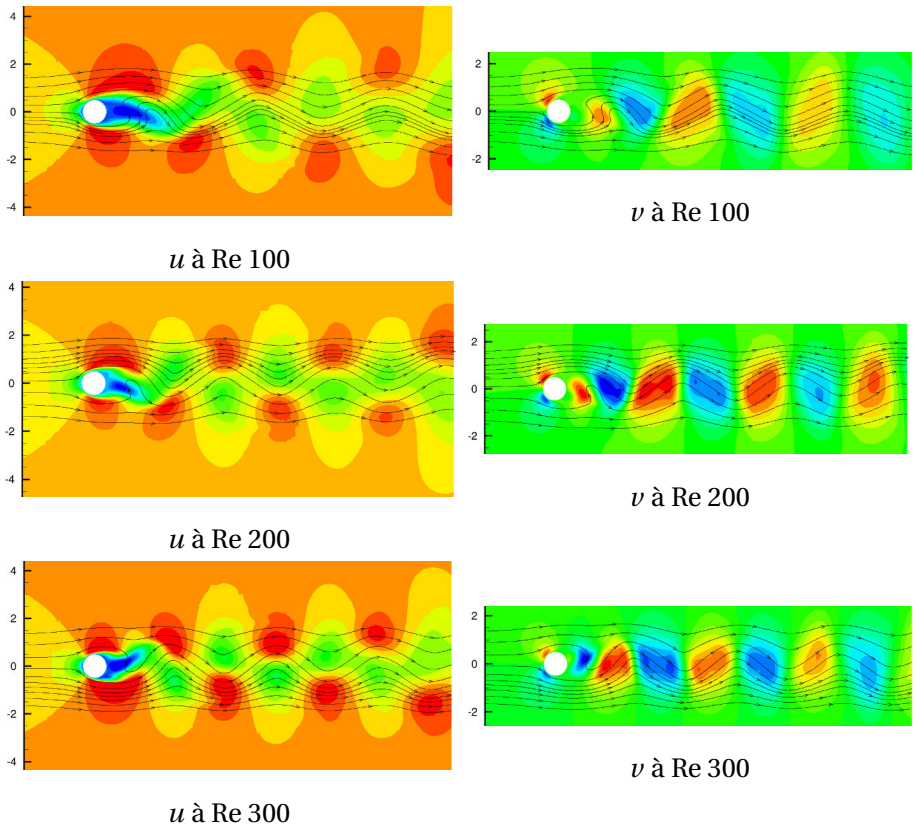


FIGURE 5.29 – État instationnaire du barreau cylindrique 2D : les profils de vitesses selon les axes x et y sont représentés respectivement sur la première et la deuxième colonne. Les lignes de courant montrent une le lâcher tourbillonnaire escompté.

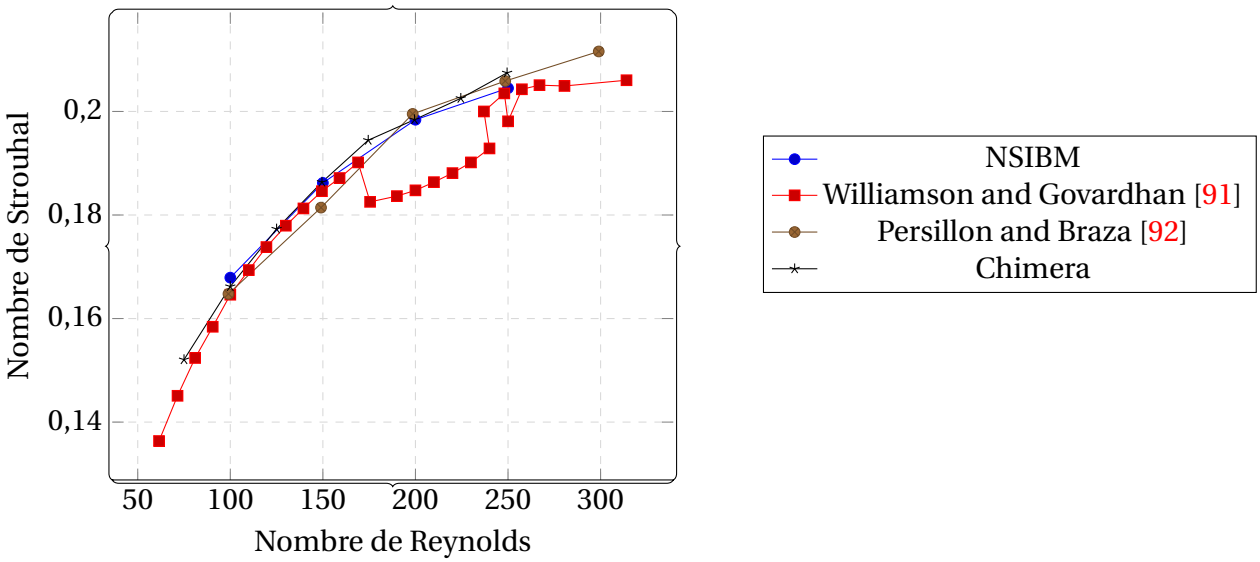
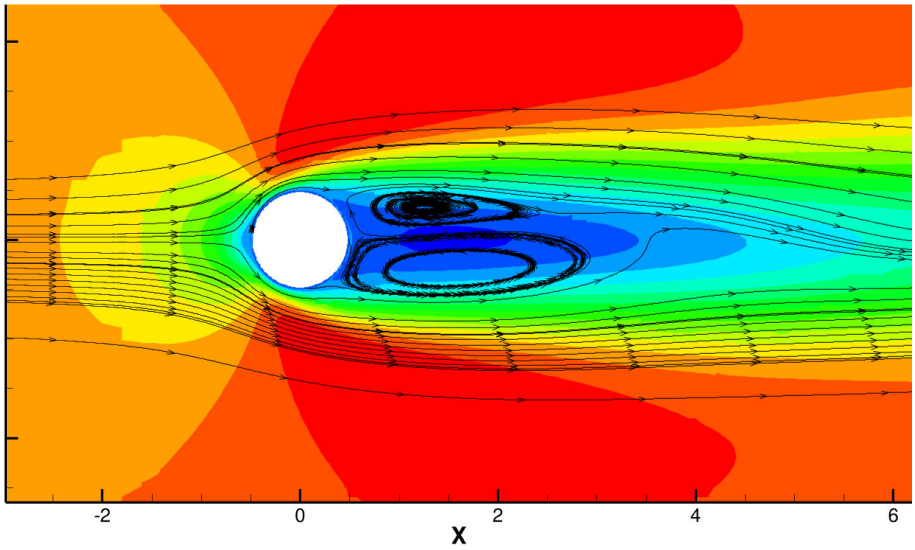
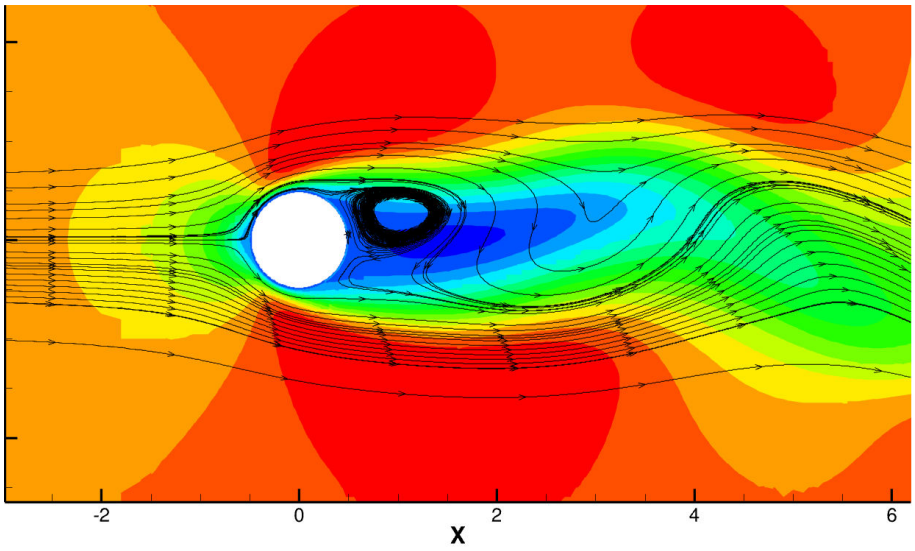


FIGURE 5.30 – Sphère 3D : comparaison des nombres de Strouhal. Les résultats obtenus avec NSIBM sont comparés aux autres résultats numériques obtenus avec la méthode Chimère dans NSMB ou trouvés dans Williamson and Govardhan [91] et Persillon and Braza [92].



Re 50



Re 60

FIGURE 5.31 – Écoulement stationnaire autour du cylindre

5.6 Écoulement laminaire 3D d'une sphère

Le cas d'un écoulement laminaire autour d'une sphère fixe est largement documenté à ce jour ; la plupart des études s'intéressent aux différents états que cet écoulement peut prendre et aux plages du nombre de Reynolds sur lesquels ces états ont lieu. La classification suivante est largement admise aujourd'hui [93, 94] :

- $Re \ll 1$: écoulement rampant avec une axisymétrie longitudinale et une symétrie amont/aval ;
- $Re \leq 20$: écoulement laminaire avec perte de symétrie amont/aval ;
- $20 \leq Re \leq 212$: écoulement décollé stationnaire axisymétrique possédant une recirculation en forme de tore en aval de la sphère ;
- $212 < Re \leq 273$: écoulement décollé stationnaire non-axisymétrique possédant une recirculation ayant perdu son axisymétrie ;
- $273 < Re$: écoulement décollé instationnaire avec lâcher tourbillonnaire périodique.

Ce cas nous sert de test pour NSIBM ; il n'est donc pas pas pertinent d'en donner reproduire une étude physique et numérique profonde. Au besoin, il est préférable de se reporter au manuscrit de thèse de Deloze [93]. Nous nous contenterons ici de comparer la plage des nombres de Reynolds allant de 50 à 250 par pas de 50 à la documentation actuelle. La simulation est réalisée avec une sphère de diamètre $D = 1$. Le maillage s'étend sur $10D$ avant le barreau et $15D$ derrière pour en suivre la traînée. Dessus, dessous devant et derrière, une marge de $10D$ est prise afin qu'il n'ait pas de perte de fluide au niveau de la perturbation générée par la sphère. Le bord gauche est une entrée de fluide dont la vitesse est :

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix}_{\text{left}} = \begin{pmatrix} u_\infty \\ 0 \\ 0 \end{pmatrix}$$

avec $u_\infty = 1\text{m/s}$. Les autres bords du domaine sont tous des sorties. Le maillage utilisé ici est illustré par la vue de coupe de la figure 5.32.

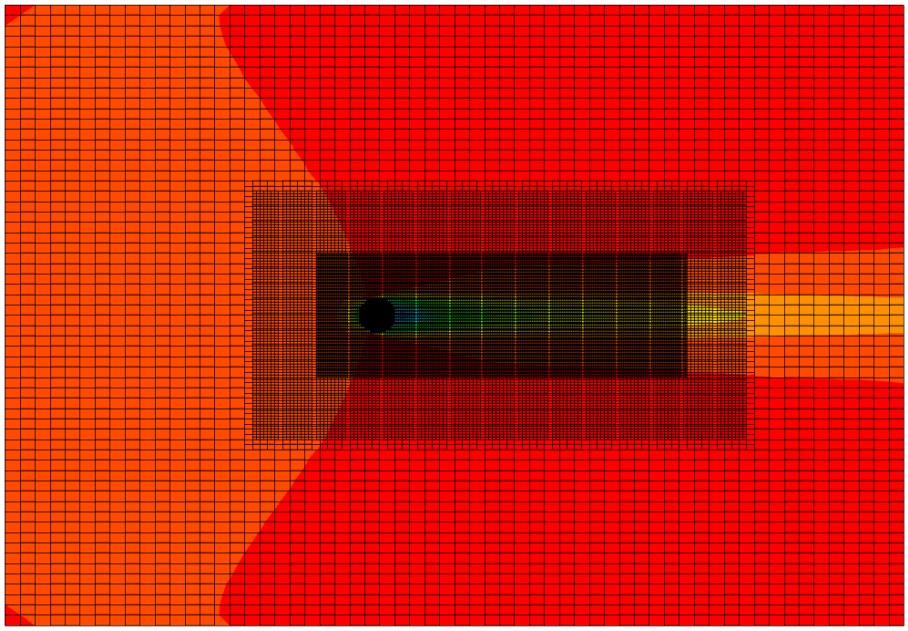


FIGURE 5.32 – Coupe du maillage de la sphère 3D mettant en évidence les quatre zones de raffinement ; il s'agit du plan $z = 0$ passant par le centre de la sphère situé en $(0,0,0)$.

Les nombres de Reynolds 50 à 200 simulés ici sont tous dans la plage de où l'écoulement est encore stationnaire. Il est donc pertinent de mesurer la longueur de la recirculation ainsi que l'angle de décollement. Nous les allons comparer avec les données recueillies expérimentalement par Taneda [95] et celles obtenues numériquement par Johnson and Patel [96].

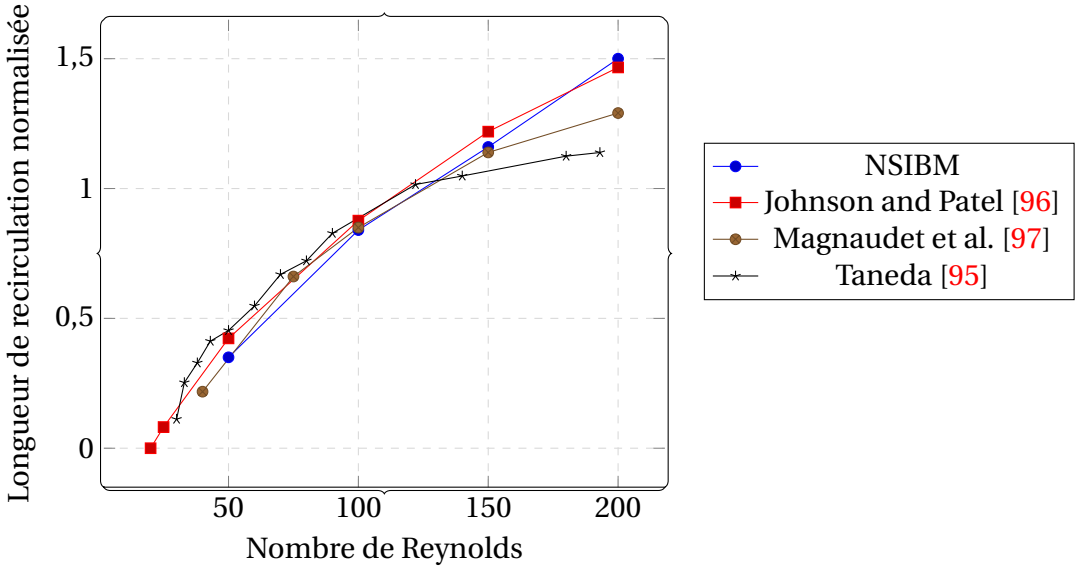


FIGURE 5.33 – Sphère 3D : comparaison des longueurs de recirculation. Les résultats obtenus avec NSIBM sont comparés aux autres résultats numériques trouvés dans Taneda [95], Magnaudet et al. [97] et Johnson and Patel [96].

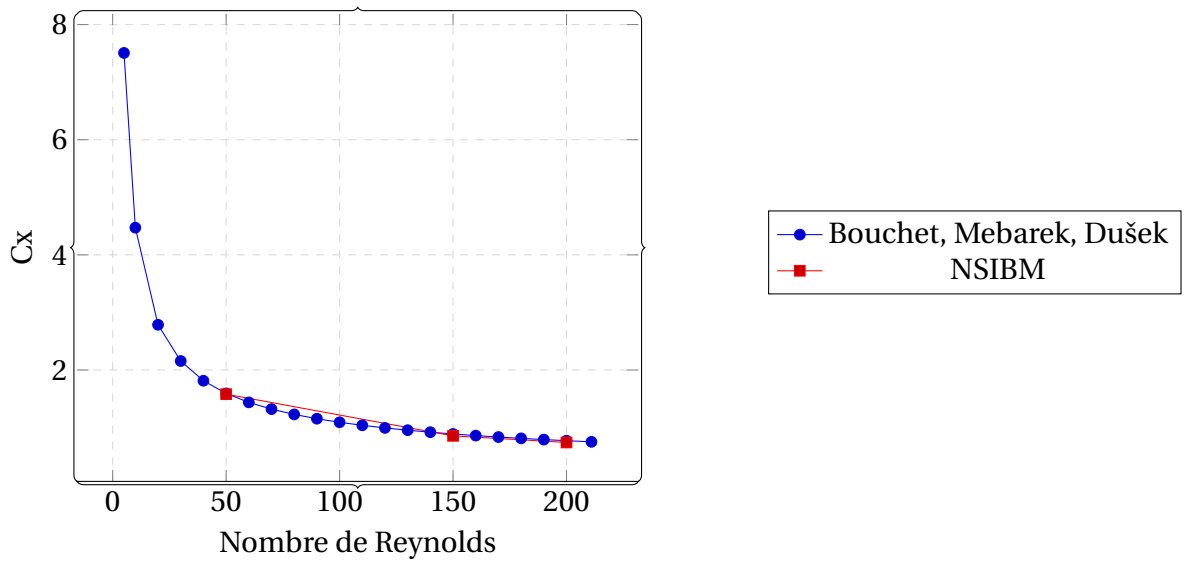


FIGURE 5.34 – Sphère 3D : comparaison des coefficients de traînée

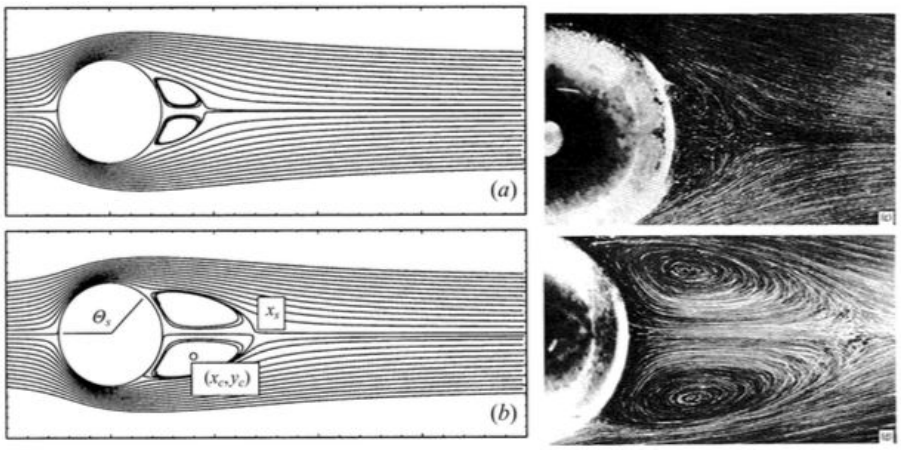


FIGURE 5.35 – Visualisation des lignes de courant pour la sphère à Re 50 et 100 dans le plan longitudinal à Re 50 pour la première ligne et à Re 100 pour la seconde. la recirculation est courte et bien symétrique. Les images de la colonne de gauche sont issues des résultats numériques de Johnson and Patel [96] et celles de la colonne de droite sont extraites des résultats expérimentaux de Taneda [95].

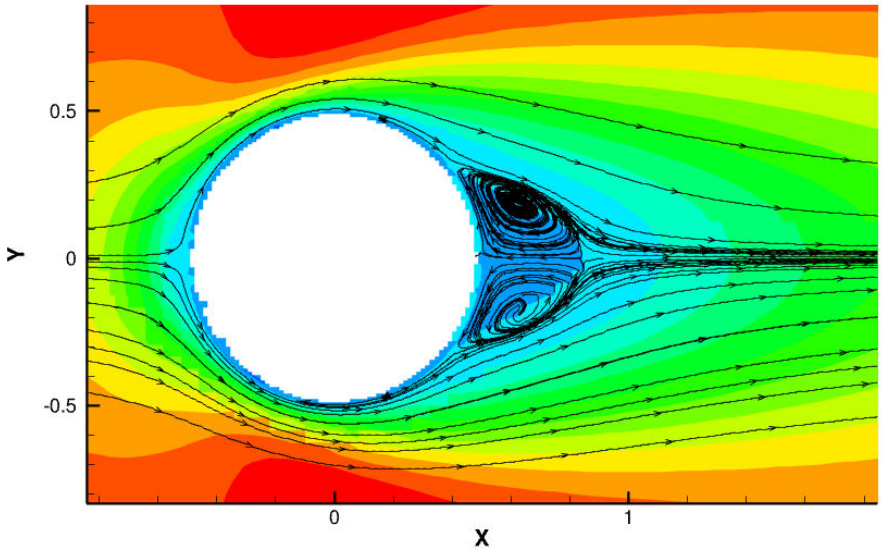


FIGURE 5.36 – Sphère 3D à Re 50 ; la recirculation est courte et bien symétrique. L'état de l'écoulement est donc bien celui annoncé dans l'étude. Visuellement, on a un résultat sensiblement identique à l'étude numérique de Johnson and Patel [96] illustré dans la figure 5.35.

Le graphe comparatif 5.33 montre que NSIBM se comporte de manière idéale dans ce cas test pour les nombres de Reynolds entre 50 et 200.

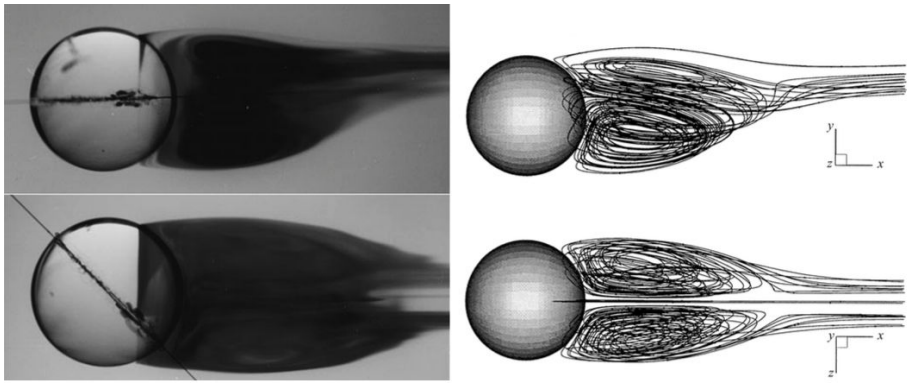


FIGURE 5.37 – État non-axisymétrique de la sphère à Re 250 dans les plans longitudinaux. Les images de la colonne de gauche sont issues des résultats numériques de Bouchet et al. [94] et celles de la colonne de droite sont extraites des résultats numériques de Johnson and Patel [96].

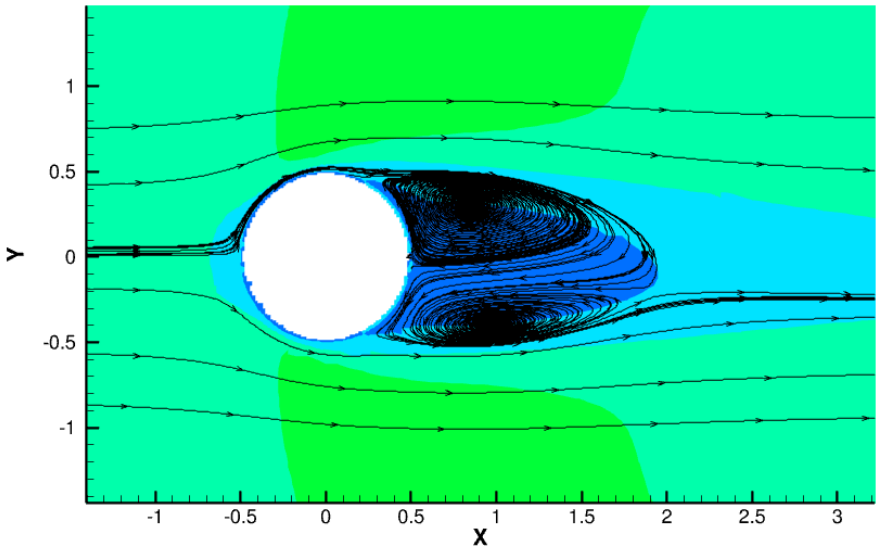


FIGURE 5.38 – Sphère 3D à Re 250 ; la recirculation est longue et a perdu sa symétrie. L'état de l'écoulement est donc bien celui annoncé dans l'étude. Visuellement, on a un résultat sensiblement identique des l'études numériques de Johnson and Patel [96] et de Bouchet et al. [94] illustrés dans la figure 5.37.

Chapitre 6

Conclusion

Ce travail mené au sein de l'équipe *Instabilité et Turbulences Diphasiques* du laboratoire iCube, département de mécanique contribue au développement d'outils destinés à l'analyse par simulation numérique d'écoulements instationnaires décollés autour de structures portantes bi- et tri-dimensionnelles pour un fluide incompressible.

Nous souhaitions doter le laboratoire d'un outil neuf, puissant et souple capable de s'attaquer rapidement à des écoulements dans ou autour de de géométries complexes : un générateur de maillages cartésiens non-structurés doté d'un raffinement automatique fonctionnant grâce à la lecture de fichiers de stéréolithographie décrivant le domaine à étudier. Nous sommes donc partis de rien pour construire ce dispositif. Fort de 6 mois de Fortran et d'études de Mathématiques ne faisant pas mention de mécanique des fluides, je me suis attelé au développement du générateur de maillage. À l'arrivée de cette thèse, il reste beaucoup à faire sur ce jeune code, mais l'essentiel y figure.

Nous avons conçu un mailleur bi-dimensionnel cartésien non-structuré avec raffinement automatique capable de prendre en compte une géométrie complexe via un fichier STL. Nous avons conçu son alter ego tri-dimensionnel de manière à ce que les fichiers de codes ne soient pas dupliqués mais compatibles entre eux afin de minimiser le travail de maintenance. Nous avons conçu un solveur capable de fonctionner avec les maillages produits grâce à ces outils. Ce solveur, nommé NSIBM, utilise un schéma implicite d'ordre deux sur une discrétisation spatiale en volumes finis et gère les bords du domaine par la méthode des frontières immergées.

Nous avons ensuite rendu ce code parallèle afin de pouvoir profiter de la puissance informatique disponible de nos jours. Nous avons pour cela fait appel à des ressources logicielles extérieures en usant de $Metis$ et de $OpenMPI$.

Cet outil a enfin été testé dans différents cas bi- et tri-dimensionnels et s'est révélé à la hauteur en termes de résultats numériques, de speed up et de maniabilité : nous avons ainsi fait tourner des calculs sur la cavité entrainée 2D, la marche descendante 2D, l'écoulement autour d'un barreau à section carrée ou circulaire, toujours en 2D, et enfin l'écoulement autour d'une sphère en 3D.

Certaines géométries pour lesquelles la génération de maillage a été réalisée n'ont pas encore pu être résolues, parce qu'il manque encore encore des options de gestions des bords du domaine, comme un schéma implicite pour les sorties qui aurait permis la simulation des poumons du rat ou de l'homme ou des frontières cycliques qui auraient permis la résolution des écoulements autour des barreaux à section carré et circulaire en 3 dimensions.

En perspective, NSIBM mérite de s'étoffer et de se voir doté de nouveaux modèles d'écoulements, afin de pouvoir simuler des cas de givrage ou de fonte.

Nomenclature

Les caractères gras représentent des vecteurs.

Alphabet latin

a	coefficient
$\boldsymbol{a}$	vecteur accélération
C	coefficient
$conv$	convection à travers une face
d	distance
$\boldsymbol{f}$	vecteur force
$\boldsymbol{i}_i$	$i^{\text{ème}}$ vecteur de la base canonique de l'espace
m	masse
$\boldsymbol{n}$	vecteur normal unitaire à une face, dirigé vers l'extérieur
p	pression
S	terme source ou surface englobant un volume
t	temps
u, u_1, u_x	composante de vitesse selon x
v, u_2, u_y	composante de vitesse selon y
w, u_3, u_z	composante de vitesse selon z
$\boldsymbol{u}$	vecteur vitesse
V	volume d'une cellule
P	centre de la cellule courante
E	centre de la cellule à l'est de la cellule courante
W	centre de la cellule à l'ouest de la cellule courante
N	centre de la cellule au nord de la cellule courante
S	centre de la cellule au sud de la cellule courante
F	centre de la cellule à l'avant de la cellule courante en 3D (*Front*)
R	centre de la cellule à l'arrière de la cellule courante en 3D (*Rear*)
e	face est de la cellule courante
w	face ouest de la cellule courante
n	face nord de la cellule courante
s	face sud de la cellule courante
f	face avant de la cellule courante en 3D
r	face arrière de la cellule courante en 3D
x_i	$i^{\text{ème}}$ coordonnée cartésienne
x	première coordonnée cartésienne ou coordonnée cartésienne générique
y	deuxième coordonnée cartésienne

Alphabet grec

α	coefficient de sous-relaxation
δ_{ij}	symbole de Kronecker valant 1 si est seulement si $i = j$ et 0 sinon
Φ	une variable universelle, vectorielle ou non
μ	viscosité dynamique
ν	viscosité cinématique valant $\frac{\mu}{\rho}$
ρ	masse volumique
ω	coefficient
Ω	domaine

Indices

CV volume de contrôle

nb cellule voisine (*neighbor*)

P valeur au centre de la cellule courante

E valeur au centre de la cellule à l'est de la cellule courante

W valeur au centre de la cellule à l'ouest de la cellule courante

N valeur au centre de la cellule au nord de la cellule courante

S valeur au centre de la cellule au sud de la cellule courante

F valeur au centre de la cellule à l'avant de la cellule courante en 3D (*Front*)

R valeur au centre de la cellule à l'arrière de la cellule courante en 3D (*Rear*)

e valeur au centre de la face est de la cellule courante

w valeur au centre de la face ouest de la cellule courante

n valeur au centre de la face nord de la cellule courante

s valeur au centre de la face sud de la cellule courante

f valeur au centre de la face avant de la cellule courante en 3D

r valeur au centre de la face arrière de la cellule courante en 3D

i indice de coordonnée cartésienne

x première coordonnée cartésienne ($i = 1$)

y deuxième coordonnée cartésienne ($i = 2$)

z troisième coordonnée cartésienne ($i = 3$)

Exposants

$'$ temps précédent

$*$ valeur à l'itération précédente

Bibliographie

[1] J. Stam. Real-Time Fluid Dynamics for Games. In *Proceedings of the Game Developer Conference*, 2003. 1

[2] L.F. Richardson. The approximate arithmetical solution by finite differences of physical problems involving differential equations, with an application to the stresses in a masonry dam. *Royal Society of London Philosophical Transactions Series A*, 210 :307–357, 1911. doi : 10.1098/rsta.1911.0009. 2

[3] D. Louapre. La mystérieuse équation de Navier-Stokes. http://sciencetonnante.wordpress.com/2014/03/03/la-mysterieuse-equation-de-navier-stokes//, 2014. [En ligne le 05/11/2014]. 2

[4] Mentor Graphics Whitepaper. Advanced immersed boundary cartesian meshing technology in FloEFD. www.mentor.com, 2011. 5

[5] G. Kalitzin, G. Iaccarino, and B. Khalighi. Towards an immersed boundary solver for rans simulations. *AIAA Pap*, 770(2003) :33, 2003. 6, 18

[6] A. Jahangirian and M. Y. Hashemi. Adaptive cartesian grid with mesh-less zones for compressible flow calculations. *Computers & Fluids*, 54 :10–17, 2012. 6, 9, 18, 19, 21, 23, 24

[7] M. J. Aftosmis. Solution adaptive cartesian grid methods for aerodynamic flows with complex geometries. *VKI Lecture Series*, 2 :1–105, 1997. 6

[8] C. Bhasker. *Simulation of Three Dimensional Flows in Industrial Components using CFD Techniques*. INTECH Open Access Publisher, 2011. 7, 8

[9] J.F. Thompson, F.C. Thomas, and C.W. Mastin. Automated numerical generation of body fitted curvilinear coordinate system for field containing any number of arbitrary 2d bodies. *Journal of Computational Physics*, 1(15) :299–319, 1974. 9

[10] C. Rome. *Une méthode de raccordement de maillages non-conformes pour la résolution des équations de Navier-Stokes*. PhD thesis, Université Sciences et Technologies - Bordeaux I, 2006. 9

[11] O Hassan, EJ Probert, and K Morgan. Unstructured mesh procedures for the simulation of three-dimensional transient compressible inviscid flows with moving boundary components. *International journal for numerical methods in fluids*, 27 (1-4) :41–55, 1998. 9

[12] R. Mittal and G. Iaccarino. Immersed boundary methods. *Annual Review of Fluid Mechanics*, 37 :239–61, 2005. doi : 10.1146/annurev.fluid.37.061903.175743. 11, 15, 18, 19, 20

[13] C. Merlin. *Simulation numérique de la combustion turbulente : Méthode de frontères immergées pour les écoulements compressibles, application à la combustion en aval d'une cavité*. PhD thesis, INSA de Rouen, 2011. 11

[14] J.H. Ferzinger and M. Perić. *Computational Methods for Fluid Dynamics*. Springer, 1998. ISBN 9783642976513. URL http://books.google.fr/books?id=mbd3NAEACAAJ. 11, 12, 18, 30

[15] A. Quarteroni and A. Valli. *Domain Decomposition Methods for Partial Differential Equations*. Numerical mathematics and scientific computation. Clarendon Press, 1999. ISBN 9780198501787. URL http://books.google.fr/books?id=CswwZ9Ca9IIC. 12

[16] T.E. Tezduyar. Finite element methods for flow problems with moving boundaries and interfaces. *Archives of Computational Methods in Engineering*, 8 :83–130, 2001. 12

[17] J. D. Baum, H. Luo, and R. Loehner. The numerical simulation of strongly unsteady flows with hundreds of moving bodies. In *36th AIAA Aerospace Sciences Meeting and Exhibit*, volume 788, 1998. 12

[18] R. Ramamurti and W.C. Sandberg. Simulation of flow about flapping airfoils using a finite element incompressible flow solver. *AIAA J.*, 39 :253–352, 2001. 12

[19] C.S. Peskin. *Flow patterns around heart valves : a digital computer method for solving the equations of motion*. PhD thesis, Albert Einstein College of Medicine of Yeshiva University, 1972. 14, 15

[20] C.S. Peskin. The fluid dynamics of heart valves : experimental, theoretical and computational methods. *Annual Review of Fluid Mechanics*, 14 :235–59, 1981. 14, 15, 16

[21] R.P. Beyer. A computational model of the cochlea using the immersed boundary method. *Journal of Computational Physics*, 98 :145–62, 1992. 15, 16, 17

[22] M.C. Lai and C.S. Peskin. An immersed bound- ary method with formal second-order accuracy and reduced numerical viscosity. *Journal of Computational Physics*, 160 :709–19, 2000. 15, 16

[23] E.M. Saiki and S. Biringen. Numerical simulation of a cylinder in uniform flow : application of a virtual boundary method. *Journal of Computational Physics*, 123 : 450–65, 1996. 15

[24] R. P. Beyer and R. J. Leveque. Analysis of a one-dimensional model for the immersed boundary method. *SIAM Journal on Numerical Analysis*, 29 :332–64, 1992. 15, 16

[25] L.J. Fauci and A. McDonald. Sperm motility in the presence of boundaries. *Bulletin of Mathematical Biology*, 57 :679–99, 1994. 15, 16

[26] S. Unverdi and G. Tryggvason. A front-tracking method for viscous, incompressible, multifluid flows. *Journal of Computational Physics*, 100 :25–42, 1992. 15, 16

[27] L. Zhu and C.S. Peskin. Interaction of two filaments in a flowing soap film. *Physics of Fluids*, 15 :128–36, 2003. 15

[28] J.M. Stockie and B.R. Wetton. Analysis of stiffness in the immersed boundary method and implications for time-stepping schemes. *Journal of Computational Physics*, 154 :41–64, 1998. 16

[29] D. Goldstein, R. Handler, and L. Sirovitch. Modeling a no-slip flow boundary with an external force field. *Journal of Computational Physics*, 105 :354–66, 1993. 16

[30] G. Iaccarino, G. Kalitzin, and G. J. Elkins. Numerical and experimental investigation of the turbulent flow in a ribbed serpentine passage. Technical report, DTIC Document, 2003. 16, 18

[31] P. Angot, C. H. Bruneau, and P. Frabrie. A penalization method to take into account obstacles in viscous flows. *Numerische Mathematik*, 81 :139–65, 1999. 16

[32] K. Khadra, P. Angot, and S. Parneix. Fictitious domain approach for numerical modeling of navier-stokes equations. *International Journal for Numerical Methods in Fluids*, 34 :651–84, 2000. 16

[33] H. C. Brinkmann. A calculation of the viscous force exerted by a flowing fluid on a swarm of particles. *Applied Science Research*, 1 :27–34, 1947. 16

[34] J. Mohd-Yusof. Combined immersed-boundary/b-spline methods for simulations of ow in complex geometries. *Center for Turbulence Research. Annual Research Briefs.*, pages 317–327, 1997. 17

[35] R. Verzicco, J. Mohd-Yusof, P. Orlandi, and D. Haworth. LES in complex geometries using boundary body forces. *Center for Turbulence Research Proceedings of the Summer Program. NASA Ames. Stanford University*, pages 171–186, 1998. 17

[36] E. Balaras. Modeling complex boundaries using an external force field on fixed cartesian grids in large-eddy simulations. *Computers and Fluids*, 33 :375–404, 2004. 17

[37] R. Verzicco, M. Fatica, G. Iaccarino, P. Moin, and B. Khalighi. Large eddy simulation of a road vehicle with drag-reduction devices. *AIAA journal*, 40(12) :2447–2455, 2002. 17

[38] Gianluca Iaccarino and Roberto Verzicco. Immersed boundary technique for turbulent flow simulations. *Applied Mechanics Reviews*, 56(3) :331–347, 2003. 18

[39] S. Majumdar, G. Iaccarino, and P.A. Durbin. Rans solver with adaptive structured boundary non-conforming grids. *Annual Research Briefs, Center for Turbulence Research*, pages 353–64, 2001. 18

[40] EA Fadlun, R Verzicco, P_ Orlandi, and J Mohd-Yusof. Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations. *Journal of Computational Physics*, 161(1) :35–60, 2000. 18

[41] A Gilmanov, F Sotiropoulos, and E Balaras. A general reconstruction algorithm for simulating flows with complex 3d immersed boundaries on cartesian grids. *Journal of Computational Physics*, 191(2) :660–669, 2003. 18

[42] Anvar Gilmanov and Fotis Sotiropoulos. A hybrid cartesian/immersed boundary method for simulating flows with 3d, geometrically complex, moving bodies. *Journal of Computational Physics*, 207(2) :457–492, 2005. 18

[43] H Ding, C Shu, KS Yeo, and D Xu. Development of least-square-based two-dimensional finite-difference schemes and their application to simulate natural convection in a cavity. *Computers & Fluids*, 33(1) :137–154, 2004. 18

[44] Reza Ghias, Rajat Mittal, and Thomas S Lund. A non-body conformal grid method for simulation of compressible flows with complex immersed boundaries. *AIAA paper*, 80 :2004, 2004. 18

[45] W. J. Coirier and K. G. Powell. Solution-adaptive cartesian cell approach for viscous and inviscid flows. *AIAA Journal*, 34(5) :938–45, 1996. 19

[46] T. Ye, R. Mittal, H.S. Udaykumar, and W. Shyy. An accurate cartesian grid method for viscous incompressible flows with complex immersed boundaries. *Journal of computational physics*, 156(2) :209–240, 1999. 19

[47] R. Mittal, C. Bonilla, and H.S. Udaykumar. Cartesian grid methods for simulating flows with moving boundaries. In P Anagnostopoulous CA Brebbia, GM Carlomagno, editor, *Computational Methods and Experimental Measurements XI*, 2003. 19

[48] R. Mittal, Y. Utturkar, and H. S. Udaykumar. Computational modeling and analysis of biomimetic flight mechanisms. *AIAA Paper*, 865 :2002, 2002. 19

[49] R. Mittal, V. Seshadri, and H.S. Udaykumar. Flutter, tumble and vortex induced autorotation. *Theoretical and Computational Fluid Dynamics*, 17(3) :165–70, 2004. 19

[50] Y. l. Utturkar, R. Mittal, P. Rampunggoon, and L. Cattafesta. Sensitivity of synthetic jets to the design of the jet cavity. *AIAA paper*, 124 :2002, 2002. 19

[51] M.J. Berger and M.J. Aftosmis. Aspects (and aspect ratios) of cartesian mesh methods. In *Proc. 16th Int. Conf. Numer. Methods Fluid Dyn*, 1998. 20

[52] H.S. Udaykumar, R. Mittal, P. Rampunggoon, and A. Khanna. A sharp interface cartesian grid method for simulating flows with complex moving boundaries. *Journal of Computational Physics*, 174(1) :345–380, 2001. 20

[53] M. J. Berger and J. Oliger. Adaptive mesh refinement for hyperbolic partial differential equations. *Journal of Computational Physics*, 53(3) :484–12, March 1984. 21

[54] R. P. Beyer and R. J. Leveque. An adaptive cartesian mesh algorithm for the euler equations in arbitrary geometries. In *9TH COMPUTATIONAL FLUID DYNAMICS CONFERENCE*, 1989. 21

[55] K.G. Powell, P.L. Roe, and J. Quirk. Adaptive-mesh algorithms for computational fluid dynamics. *Algorithmic trends in computational fluid dynamics*, 53 :303–37, 1993. 21

[56] A. Jahangirian and Y. Shoraka. Adaptive unstructured grid generation for engineering computation of aerodynamic flows. *Mathematics and Computers in Simulation*, 78(5) :627–644, 2008. 21

[57] Joe F Thompson, Bharat K Soni, and Nigel P Weatherill. *Handbook of grid generation*, chapter 22. CRC press, 1998. 21, 22

[58] A. Lintermann, S. Schlimpert, J.H. Grimmen, C. Günther, M. Meinke, and W. Schröder. Massively parallel grid generation on hpc systems. *Compter Methods in Appplied Mechanics and Engineering*, 277 :131–53, 2014. 21, 25

[59] O. Antepara, O. Lehmkuhl, R. Borrell, J. Chiva, and A. Oliva. Parallel adaptive mesh refinement for large-eddy simulations of turbulent flows. *Computers & Fluids*, 110 :48–61, 2015. 23, 25, 26, 27, 28

[60] T. Hughes, L. Mazzei, and K. E. Jansen. Large eddy simulation and the variational multiscale method. *Computing and Visualization in Science*, 3(1-2) :47–59, 2000. 23

[61] K. Nakahashi and L. Kim. Building-cube method for large-scale high resolution flow conputations. In *42nd AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, 2004. 25, 27

[62] A. Amsden and F. Harlow. The SMAC method : A numerical technique for calculating incompressible fluid flows. Technical report, Los Alamos Scientific Lab., N. Mex., 1970. 34

[63] M. Braza. *Simulation numérique du décollement instationnaire externe par une formulation vitesse-pression : application à l'écoulement autour d'un cylindre*. PhD thesis, Toulouse, INPT, 1981. 34

[64] Y. Hoarau. *Analyse physique par simulation numérique et modélisation des écoulements décollés instationnaires autour de surfaces portantes*. PhD thesis, Toulouse, INPT, 2002. 34

[65] M. Braza. *Analyse physique du comportement dynamique d'un écoulement externe, décollé, instationnaire en transition laminaire-turbulente : application : cylindre circulaire*. PhD thesis, Toulouse, INPT, 1986. 35

[66] M. Braza, P. Chassaing, and H. Ha-Minh. Numerical study and physical analysis of the pressure and velocity fields in the near wake of a circular cylinder. *Journal of fluid mechanics*, 165 :79–130, 1986. 36

[67] C. M. Rhie and W. L. Chow. Numerical study of the turbulent flow past an airfoil with trailing edge separation. *Aiaa Journal*, 21 :1525–1532, 1983. doi : 10.2514/3.8284. 38, 40

[68] S. Zhang, X. Zhao, and S. Bayyuk. Generalized formulations for the rhie–chow interpolation. *Journal of Computational Physics*, 258(0) :880–914, 2 2014. doi : http://dx.doi.org/10.1016/j.jcp.2013.11.006. URL http://www.sciencedirect.com/science/article/pii/S0021999113007523. 38

[69] Fichier de stéréolithographie, 2015. URL https://fr.wikipedia.org/wiki/Fichier_de_stéréolithographie. 57

[70] George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on scientific Computing*, 20(1) :359–392, 1998. 60

[71] George Karypis and Vipin Kumar. Analysis of multilevel graph partitioning. In *Proceedings of the 1995 ACM/IEEE conference on Supercomputing*, page 29. ACM, 1995. 60

[72] George Karypis and Vipin Kumar. Multilevel algorithms for multi-constraint graph partitioning. In *Proceedings of the 1998 ACM/IEEE conference on Supercomputing*, pages 1–13. IEEE Computer Society, 1998. 60

[73] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on scientific Computing*, 20(1) :359–392, 1998. 60

[74] George Karypis and Vipin Kumar. Parallel multilevel series k-way partitioning scheme for irregular graphs. *Siam Review*, 41(2) :278–300, 1999. 60

[75] G. Karypis. Multi-constraint mesh partitioning for contact/impact computations. In *Proceedings of the 2003 ACM/IEEE conference on Supercomputing*, page 56. ACM, 2003. 60

[76] Edgar Gabriel, Graham E. Fagg, George Bosilca, Thara Angskun, Jack J. Dongarra, Jeffrey M. Squyres, Vishal Sahay, Prabhanjan Kambadur, Brian Barrett, Andrew Lumsdaine, Ralph H. Castain, David J. Daniel, Richard L. Graham, and Timothy S. Woodall. Open MPI : Goals, concept, and design of a next generation MPI implementation. In *Proceedings, 11th European PVM/MPI Users' Group Meeting*, pages 97–104, Budapest, Hungary, September 2004. 62

[77] UKNG Ghia, Kirti N Ghia, and CT Shin. High-re solutions for incompressible flow using the navier-stokes equations and a multigrid method. *Journal of computational physics*, 48(3) :387–411, 1982. 72, 75

[78] B.F. Armaly, F. Durst, J. C. F. Pereira, and B. Schönung. Experimental and theoretical investigation of backward-facing step flow. *Journal of Fluid Mechanics*, 127 : 473–496, 1983. 77, 80

[79] J. C. Bennetsen. *Numerical Simulation of Turbulent Airflow in Livestock Buildings*. PhD thesis, The Department of Mathematical Modelling, The Technical University of Denmark, The Department of Agricultural Engineering Research Centre Bygholm, Danish Institute for Agricultural Sciences, 1999. 77

[80] J. Kim and P. Moin. Application of a fractional-step method to incompressible navier-stokes equations. *Journal of Computational Physic*, 59 :308–323, 1985. 80

[81] A. Sohankar, C. Norberg, and L. Davidson. Numerical simulation of flow past a square cylinder. In *Proceedings of FEDSM99 3rd ASME/JSME Joint Fluids Engineering Conference. July*, pages 18–23, 1999. 82

[82] M. Sukri M. Ali, C. J. Doolan, and V. Wheatley. Grid convergence study for a two-dimensional simulation of flow around a square cylinder at a low reynolds number. In *Seventh International Conference on CFD in The Minerals and Process Industries, CSIRO (CSIRO Australia, Melbourne, Australia, 2009)*, 2009. 83

[83] A. Sohankar, C. Norberg, and L. Davidson. Low-reynolds-number flow around a square cylinder at incidence : study of blockage, onset of vortex shedding and outlet boundary condition. *International journal for numerical methods in fluids*, 26(1) :39–56, 1998. 83, 84

[84] C. Doolan. Flat-plate interaction with the near wake of a square cylinder. *AIAA journal*, 47(2) :475–479, 2009. 83, 84

[85] A. Sohankar, C. Norberg, and L. Davidson. Numerical simulation of unsteady low-reynolds number flow around rectangular cylinders at incidence. *Journal of Wind Engineering and Industrial Aerodynamics*, 69 :189–201, 1997. 83

[86] S.C.R. Dennis and G.Z. Chang. Numerical solutions for steady flow past a circular cylinder at reynolds numbers up to 100. *Journal of Fluid Mechanics*, 42(03) :471–489, 1970. 88

[87] A.E. Hamielec and J.D. Raal. Numerical studies of viscous flow around circular cylinders. *Physics of Fluids (1958-1988)*, 12(1) :11–17, 1969. 88

[88] S. Tuann and M. D. Olson. Numerical studies of the flow around a circular cylinder by a finite element method. *Computers & Fluids*, 6(4) :219–240, 1978. 88

[89] F. Nieuwstadt and H.B. Keller. Viscous flow past circular cylinders. *Computers & Fluids*, 1(1) :59–71, 1973. 88, 89

[90] H. Takami and H. B. Keller. Steady two-dimensional viscous flow of an incompressible fluid past a circular cylinder. *Physics of Fluids (1958-1988)*, 12(12) :II–51, 1969. 88, 89

[91] C.H.K. Williamson and R. Govardhan. Dynamics and forcing of a tethered sphere in a fluid flow. *Journal of Fluids and Structures*, 11(3) :293–305, 1997. 89, 91

[92] H. Persillon and M. Braza. Physical analysis of the transition to turbulence in the wake of a circular cylinder by three-dimensional navier–stokes simulation. *Journal of Fluid Mechanics*, 365 :23–88, 1998. 89, 91

[93] T. Deloze. *Couplage fluide-solide appliqué à l'étude de mouvement d'une sphère libre dans un tube vertical*. PhD thesis, Université de Strasbourg, 2011. 93

[94] G. Bouchet, M. Mebarek, and J. Dušek. Hydrodynamic forces acting on a rigid fixed sphere in early transitional regimes. *European Journal of Mechanics-B/Fluids*, 25(3) :321–336, 2006. 93, 97, 98

[95] S. Taneda. Experimental investigation of the wake behind a sphere at low reynolds numbers. *Journal of the Physical Society of Japan*, 11(10) :1104–1108, 1956. 94, 96

[96] T. Johnson and V. Patel. Flow past a sphere up to a reynolds number of 300. *Journal of Fluid Mechanics*, 378 :19–70, 1999. 94, 96, 97, 98

[97] J. Magnaudet, M. Rivero, and J. Fabre. Accelerated flows past a rigid sphere or a spherical bubble. part 1. steady straining flow. *Journal of fluid mechanics*, 284 : 97–135, 1995. 94

Résumé

L'essentiel du travail effectué consiste en le développement d'un programme capable de résoudre numériquement l'équation de Navier-Stokes qui régit des fluides en mouvement. Dans un premier temps, j'ai conçu un générateur de maillage en deux et trois dimensions en y intégrant la possibilité de diviser des cellules de manière automatique, par des critères géométriques, numériques ou physiques. Il permet également de supprimer des cellules, de façon à ne pas mailler le vide ou les parties solides de la géométrie. Dans un deuxième temps, j'ai rendu ce code parallèle afin d'utiliser davantage de mailles. Cette étape fait appel à deux technologies : *Metis*, qui partage équitablement les mailles sur le nombre choisi de processeurs et *openMPI*, qui est l'interface de communication entre ces processeurs. Enfin, la méthode des frontières immergées a été introduite au code pour gérer les bords non verticaux ou horizontaux dans un maillage cartésien. Elle consiste à donner un caractère hybride à une cellule traversée par une frontière par l'introduction d'un terme numérique de forçage simulant la présence de la paroi. Ce travail de développement a ensuite été mis à l'épreuve et validé dans une série de cas tests en deux comme en trois dimensions. Des exemples de maillages complexes générés facilement sont donnés.

Mots clés : CFD, Navier-Stokes, OpenMPI, raffinement de maillage, IBM

Summary

This work mainly deals with coding a program able to solve the Navier-Stokes equations that governs moving fluids, in a numerical way. Particular attention was paid to the production of meshes that suit given geometries and their generation. The means used here to handle the eternal problem of the fineness of the mesh opposed to too many cells are several : refinement, parallelization and the immersed boundary method. Initially, I designed a two and three-dimensional mesh generator that includes the possibility of dividing cells, in an automatic way, by geometrical, numerical or physical criteria. It also allows to remove cells, where there is no point keeping it. Secondly, I parallelized the program by giving him the ability to use multiple processors to calculate faster and therefore use bigger meshes. This step uses two available libraries : *Metis*, which gives a optimal mesh partition, and *openMPI*, which deals with communication between nodes. Finally, the immersed boundary method has been implemented to handle non-vertical or non-horizontal edges in a cartesian grid. Its principle is to confer a hybrid status to a cell which is crossed by an edge by adding a numerical force term simulating the presence of the boundary. This development work was then tested and validated in a serie of test cases in two and three dimensions. Examples of complex meshes easily generated are given.

Keywords : CFD, Navier-Stokes, OpenMPI, mesh refinement, IBM