


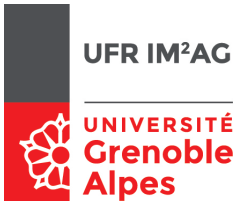

Grenoble INP – ENSIMAG
Université Grenoble Alpes – UFR IMAG

Rapport de PFE

Cerfacs - ONERA

Développement d'une méthode de suivi lagrangien de particules dans un code CFD d'ordre élevé pour la LES

Nicolas Gindrier
M2MSIAM

5 mars 2018 – 31 août 2018

Cerfacs
42, avenue Gaspard Coriolis
31100 Toulouse

Maîtres de stage
Jean-Mathieu Senoner
Bénédicte Cuenot
Tuteur Ensimag
Emmanuel Maître

Nom : Nicolas Gindrier
e-mail : nicolas.gindrier@grenoble-inp.org
Maîtres de stage : Bénédicte Cuenot (cuenot@cerfacs.fr)
Jean-Mathieu Senoner (Jean-Mathieu.Senoner@onera.fr)

Table des matières

Résumé

Les termes techniques mentionnés ci-dessous seront définis au fil du rapport.

L'objectif de ce stage de 6 mois (de début mars à fin août) était de développer une extension au code JAGUAR. JAGUAR est un code résolvant les équations de Navier-Stokes instationnaires, compressibles et monophasiques (gaz avec une loi d'état de type gaz parfait). Le code utilise une approche de type différences finies spectrales, basée sur une discrétisation polynomiale dans chaque maille de calcul mais autorisant la présence de discontinuités entre elles. Afin de pouvoir traiter des géométries non structurées, ce solveur utilise des transformations géométriques des mailles sur un élément isoparamétrique.

Parmi les applications visées pour ce code, on peut notamment citer la combustion aérobie, typiquement pour les turboréacteurs d'avion ou d'hélicoptère. Afin de pouvoir simuler ce type d'applications, il est nécessaire de modéliser l'injection de carburant liquide. Lorsque le carburant est dispersé sous forme d'un brouillard de gouttes, sa représentation par une méthode de suivi lagrangienne semble judicieuse, à la fois en terme de coût calcul et de modélisation. À ce titre, il semblait très intéressant d'examiner la transposition d'algorithmes de repérage lagrangien dans un contexte de transformation isoparamétrique. Bien que ce type de transformations soit assez répandu, notamment pour les approches de type Galerkin discontinue, une brève étude bibliographique n'a pas permis de trouver de travaux antérieurs sur ce sujet.
Ce stage s'est déroulé au Cerfacs à Toulouse avec Bénédicte Cuenot (encadrante Cerfacs) et Jean-Mathieu Senoner (encadrant Onera).

Remerciements

Je tiens à remercier JF. Boussuge et G. Puigt pour leurs conseils sur JAGUAR. Je remercie chaleureusement mes deux maîtres de stage B. Cuenot et JM. Senoner pour m'avoir accueilli et fait confiance. Je tiens en particulier à remercier JM. Senoner pour l'énorme quantité de temps et d'énergie qu'il m'a accordée, notamment pour la chasse aux bugs ou pour la relecture de ce rapport.

Je remercie du fond du cœur les trois stagiaires qui ont partagé le bureau I20 avec moi et ont fortement égayé le stage, et souhaite à Nicolas et Jonathan une bonne thèse au cerfacs, et à Thomas un bon séjour à Georgia Tech.

Notations et glossaire

LES : Large Eddy Simulation ou simulation aux grandes échelles (SGE) en français. Approche de modélisation pour les écoulements turbulents, utilisée dans JAGUAR. En considérant l'écoulement turbulent comme constitué de tourbillons répartis sur une gamme d'échelles de longueur très disparate, le principe de la LES est de résoudre explicitement les plus grands tourbillons / échelles turbulentes. En effet, le comportement de ces derniers est dicté par la géométrie et le type d'écoulement simulés, ce qui rend leur modélisation délicate. Afin de relâcher les contraintes de raffinement du maillage de calcul et de réduire le coût de calcul, les plus petites échelles turbulentes sont modélisées, ce qui semble judicieux d'un point de vue modélisation puisque ces échelles ont *a priori* un caractère plus universel.

CFD : Computational Fluid Mechanics, domaine de la mécanique des fluides qui utilise l'analyse numérique pour résoudre un système d'équations différentielle, généralement les équations de Navier-Stokes, pour décrire la dynamique d'écoulements fluides.

Elément et **cellule** : ces deux termes seront utilisés indifféremment pour désigner une entité du maillage (dans JAGUAR un quadrangle en 2D ou un hexaèdre en 3D). Dans le code JAGUAR on utilisera davantage élément (`element` dans le code). Un élément servira de **volume de contrôle** pour les méthodes utilisées. Enfin on pourra aussi utiliser le terme de maille à consonance davantage géométrique.

Ecoulement diphasique : écoulement contenant soit un fluide existant sous deux phases différentes, par exemple lors de l'évaporation d'un liquide dans une conduite, soit deux fluides distincts existant sous deux phase distinctes, par exemple un carburant liquide et un écoulement d'air dans une chambre de combustion, le stage traitant du second cas. On insistera sur le fait qu'on traite ici un cas très simplifié d'écoulement diphasique. Ainsi, il n'est pas question ici de résoudre les équations de Navier-Stokes dans chaque phase puis de les coupler via des relations d'interface (écoulements diphasiques dits denses). On se contente ici de simuler de façon approchée le comportement d'un brouillard de goutelettes dispersé dans un écoulement porteur fluide gazeux, on parle d'écoulements diphasiques dispersés.

Méthode de Galerkin : méthode utilisée dans les éléments finis consistant à définir une solution polynomiale à une équation différentielle dans chaque cellule. Dans une **méthode de Galerkin discontinue**, les valeurs des solutions aux nœuds entre deux cellules les partageant peuvent être différentes, ce qui implique un nombre de degrés de liberté plus important.

1 Introduction

Après avoir effectué un stage de 6 semaines dans une start-up (entre ma première et ma seconde année à l'Ensimag), et un stage de 3 mois dans une université canadienne (McMaster, à Hamilton), j'ai voulu essayer une structure de taille intermédiaire et à mi-chemin entre les mondes industriel et académique. Le Cerfacs travaillait sur certains domaines qui m'intéressaient, tels que la mécanique des fluides et la simulaton numérique. La perspective était d'obtenir une thèse après le stage si il apportait satisfaction. J'ai fait partie des trois élèves de l'Ensimag ayant effectués leur stage au Cerfacs, mais je suis le seul ayant travaillé sur de la modélisation physique.
Une partie de mon stage se déroulait également à l'ONERA, parce qu'un de mes deux maîtres de stage y travaille et que JAGUAR est développé conjointement à l'ONERA et au Cerfacs. Cependant le Cerfacs restait mon employeur.

1.1 Cerfacs

Le Cerfacs (Centre Européen de Recherche et de Formation Avancée en Calcul Scientifique) est situé à Toulouse, dans le site du météopole. C'est un organisme de recherche travaillant principalement sur des simulations numériques et de la modélisation physique. Le Cerfacs emploie des physiciens, des mathématiciens et des informaticiens pour étudier les problématiques du climat, du calcul haute performance et de l'aéronautique, notamment en combustion, acoustique et aérodynamique. Le Cerfacs collabore avec sept actionnaires qui sont Airbus, le CNES, EDF, Météo France, l'ONERA, Safran et Total, et a noué des partenariats avec le CNRS, le CEA et l'INRIA.
Il est constitué de plusieurs équipes :

- Algorithmique Parallèle (ALGO)
- Aviation et Environnement (AE)
- Modélisation du climat (GLOBC)
- Computational Fluids Dynamics (CFD, là où je travaille, et qui constitue l'équipe la plus importante du Cerfacs en terme d'effectif)
- Equipe Informatique et Support Utilisateur (CSG)

En 2018 y travaillent 65 membres permanents et 45 doctorants ou post-doctorants.
Le Cerfacs offre également beaucoup de formations et la possibilité d'assister à des conférences. Par exemple, chaque lundi après-midi, un doctorant doit présenter en anglais un article de recherche et un logiciel.

1.2 Objectif du stage

L'objectif de ce stage est d'implémenter un algorithme de suivi particulaire lagrangien dans un code de mécanique des fluides numérique, ou computational fluid dynamics (CFD) en anglais, appelé JAGUAR et développé conjointement par le Cerfacs et l'ONERA. L'application visée est la combustion dans des foyers aéronautiques, pour lesquels le carburant est injecté sous forme d'un liquide continu. Une fois que ce liquide s'est pulvérisé en un fin brouillard de goutelettes, des approches de type phase dispersée sont utilisées pour simuler ce dernier. Les approches les plus répandues sont l'approche eulérienne et lagrangienne, c'est le deuxième type d'approche qui sera traité ici.

L'approche lagrangienne suit les particules dans leur propre référentiel et les déplace ainsi sur le maillage de calcul. L'algorithme de repérage des particules sur le maillage de calcul constitue donc un élement algorithmique essentiel de l'approche lagrangienne.

Dans le cadre du stage, on se fixe pour objectif d'effectuer ce repérage dans l'espace isoparamétrique et non dans l'espace dit physique. Premièrement, le passage dans l'espace isoparamétrique permettrait de traiter de façon naturelle les volumes de contrôle ayant des faces courbes, pour lesquels le repérage semble difficile à traiter dans l'espace physique avec les algorithmes usuels de la littérature. De plus, le traitement exclusif du repérage particulaire dans l'espace isoparamétrique permettrait de rendre l'algorithme sous-jacent performant d'un point de vue coût calcul. En effet, alors que la transformation de l'espace isoparamétrique vers l'espace physique est naturelle et directe, la réciproque n'est pas vraie puisqu'une méthode itérative doit être utilisée pour trouver les coordonnées isoparamétriques d'un point dans une maille connaissant ses coordonnées physiques.

2 Le code JAGUAR

2.1 Introduction

JAGUAR (proJect of an Aerodynamic solver using General Unstructured grids and high ordeR schemes) est un solveur CFD résolvant les équations de Navier-Stokes instationnaires, multi-espèce et compressibles. Il est basé sur une approche de type **différences spectrales** (que l'on notera méthode SD) reposant sur une représentation polynomiale à l'intérieur de chaque volume de contrôle. Ceci permet d'atteindre des ordres de discrétisation spatiale élevés. Cette propriété est très intéressante pour la simulation haute fidélité d'écoulements turbulents instationnaires, typiquement les approches Large Eddy Simulation (LES) et Direct Numerical Simulations (DNS). En effet, une discrétisation spatiale d'ordre élevé permet de grandement limiter les phénomènes de dissipation et dispersion numériques, c'est à dire que les structures physiques cohérentes ne sont pas amorties de façon artificielle.

La parallélisation est gérée de façon classique via une approche de décomposition de domaine, où chaque processeur simule une partition du maillage (mémoire distribuée). Les partitionneurs sont Metis et Parmetis. La mise à jour des cellules situées aux frontières entre partitions nécessite des échanges entre processeurs gérés via la librairie MPI. Le code JAGUAR possède d'excellentes propriétés de parallélisation et une scalabalité faible quasi-parfaite sur plusieurs dizaines de milliers de processeurs. Des fonctionnalités de gestion de mémoire partagée via OpenMP sont également disponibles mais peu utilisées en pratique.

2.2 Equations d'Euler et de Navier Stokes

Le code JAGUAR a pour vocation de résoudre les équations d'Euler et de Navier Stokes. Ces équations décrivent toutes deux la conservation de la masse, de la quantité de mouvement et de l'énergie d'un fluide. Les équations d'Euler négligent les effets visqueux alors que les équations de Navier-Stokes tiennent compte de ces derniers et sont donc plus générales. Pour un fluide constitué d'une seule espèce, les équations de Navier-Stokes s'écrivent :

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{u}) = 0 \tag{1}$$

$$\frac{\partial \rho \boldsymbol{u}}{\partial t} + \nabla \cdot (\rho \boldsymbol{u} \otimes \boldsymbol{u} + \nabla P) = \nabla \cdot \underline{\boldsymbol{\tau}} \tag{2}$$

$$\frac{\partial \rho E}{\partial t} + \nabla \cdot (\rho E \boldsymbol{u} + \nabla \cdot P \boldsymbol{u}) = -\nabla \cdot \boldsymbol{q} \tag{3}$$

Avec t la variable temporelle, ρ la densité, $\boldsymbol{u}$ le vecteur vitesse fluide, P le terme de pression, $\underline{\boldsymbol{\tau}}$ le tenseur des contraintes visqueuses et $\boldsymbol{q}$ le vecteur des flux de chaleur et E l'énergie totale du fluide. Plus de détails sur cette équation seront donnés en annexe section A.

Conformément à la classification des flux décrite précédemment, on peut écrire les équations de Navier-Stokes sous la forme suivante :

$$\frac{\partial \boldsymbol{\mathcal{Q}}}{\partial t} + \nabla \cdot \boldsymbol{\mathcal{F}}_e(\boldsymbol{\mathcal{U}}) = \nabla \cdot \boldsymbol{\mathcal{F}}_{ns}(\nabla \boldsymbol{\mathcal{U}}) \tag{4}$$

avec $\boldsymbol{\mathcal{Q}}, \boldsymbol{\mathcal{U}}$, $\boldsymbol{\mathcal{F}}_e$ et $\boldsymbol{\mathcal{F}}_{ns}$ respectivement le vecteur des variables dites conservatives, le vecteur des variables primitives, le vecteur des flux Euler et le vecteur des flux Navier-Stokes :

$$\boldsymbol{\mathcal{Q}} = \begin{cases} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{cases} \quad \text{et} \quad \boldsymbol{\mathcal{U}} = \begin{cases} \rho \\ u \\ v \\ w \\ E \end{cases} \tag{5}$$

$$\boldsymbol{\mathcal{F}}_e = \begin{cases} \boldsymbol{F} = (\rho u, P + \rho u^2, \rho uv, \rho uw, u(\rho E + P))^T \\ \boldsymbol{G} = (\rho v, \rho uv, P + v^2, \rho vw, v(\rho E + P))^T \\ \boldsymbol{H} = (\rho w, \rho uw, \rho vw, P + \rho w^2, w(\rho E + P))^T \end{cases} \tag{6}$$

$$\boldsymbol{\mathcal{F}}_{ns} = \begin{cases} 0 \\ \underline{\boldsymbol{\tau}} \\ \boldsymbol{q} \end{cases} \tag{7}$$

Ces équations seront résolues grâce à la méthode des différences spectrales décrite annexe C.

3 Approche lagrangienne

3.1 Equations de la dynamique particulaire

L'objectif est d'implémenter un algorithme de repérage particulaire ainsi qu'une ébauche de solveur lagrangien dans le code JAGUAR. Pour cela, il est d'abord nécessaire de présenter les équations régissant la dynamique de particules immergées dans un écoulement porteur fluide. Il est important de préciser que les particules ne sont pas explicitement résolues sur le maillage de calcul, mais considérées comme ponctuelles, c'est-à-dire petites devant la taille de maille caractéristique locale.

Ainsi, l'évolution dynamique des particules peut être décrite par la position de leur centre de gravité $\boldsymbol{x_p}$ et leur vecteur vitesse $\boldsymbol{u_p}$. On obtient alors le système d'équations suivant :

$$\frac{d\boldsymbol{x_p}(t)}{dt} = \boldsymbol{u_p}(\boldsymbol{x_p}(t)) \tag{8}$$

$$m_p \frac{d\boldsymbol{u_p}(\boldsymbol{x_p}(t))}{dt} = \sum \boldsymbol{F_p}(t) = \int_{\partial S} p\boldsymbol{n} + \underline{\boldsymbol{\tau}} \cdot \boldsymbol{n} \partial S \tag{9}$$

où m_p désigne la masse de la particule considérée. Les dérivées apparaissant dans les membres de gauche sont des dérivées totales, c'est-à-dire décrivant des variations temporelles dans le référentiel mobile des particules. La première équation exprime une simple

condition cinématique : l'évolution de la position du centre de gravité est dictée par la vitesse de ce point. L'équation (9) exprime quant à elle la seconde loi de Newton : le produit de la masse et de l'accélération particulaire est égal à la somme des forces agissant sur cette dernière. Pour une particule immergée dans un écoulement fluide, les forces résultent de l'intégrale de la pression fluide et du tenseur des contraintes visqueuses évaluée à la surface de la particule. Puisque la particule n'est pas explicitement résolue sur le maillage de calcul, cette intégrale surfacique ne peut pas être calculée numériquement et doit être approchée en l'exprimant via des propriétés moyennes du fluide et de la particule. Cette dérivation est possible lorsque le nombre de Reynolds particulaire, qui traduit le rapport entre forces inertielles et visqueuses, est faible devant l'unité :

$$Re_p = \frac{\rho_f \|\boldsymbol{u_f} - \boldsymbol{u_p}\| d_p}{\mu_f} \ll 1 \tag{10}$$

où les indices f et p désignent respectivement les propriétés fluide et particulaire. μ est la viscosité dynamique. Les dérivations associées ont été successivement effectuées et corrigées par Basset [1], Boussinesq [2] et Oseen [3] et l'équation résultante porte les initiales de ces trois contributeurs majeurs : ce sont les équations dites BBO. Plus récemment, Maxey et Riley [4] ainsi que Gatignol [5] ont corrigé de façon indépendante certaines erreurs dans la dérivation des équations BBO. Leurs dérivations de l'expression des forces agissant sur une goutte à faible Reynolds particulaire font encore référence aujourd'hui, on parle des équations MRG.

Dans le cas de figure étudié, la majorité des forces s'exerçant sur les particules peut être négligée en vertu des ordres de grandeur en présence :

- on vise des applications où les particules sont soit liquides, soit solides et évoluent dans un fluide gazeux. Ainsi la densité des particules est très largement supérieure à celle du fluide, typiquement d'un rapport de l'ordre du millier : $\rho_p >> \rho_f$
- la fraction volumique $\alpha_p = V_p/V_{ref} = V_p/(V_p+V_f)$, définie comme le volume occupé par les particules dans un volume de référence, typiquement une cellule du maillage de calcul, est faible : $10^{-6} < \alpha_p < 10^{-2}$.
- Le diamètre des particules d_p est compris entre quelques microns et une centaine de microns : $5\,\mu\text{m} < d_p < 100\,\mu\text{m}$. Ainsi, les particules sont supposées petites devant les échelles caractéristiques de l'écoulement.

En utilisant les hypothèses précédentes, on peut montrer que les forces prépondérantes agissant sur la particule sont la force de traînée et la gravité :

$$\begin{aligned} \sum \boldsymbol{F_p}(t) &\approx \boldsymbol{F_d}(t) + \boldsymbol{F_g} \\ &= \frac{\pi}{8}\rho_f d_p^2 C_d \|\boldsymbol{u_f}(\boldsymbol{x_p}(t)) - \boldsymbol{u_p}(\boldsymbol{x_p}(t))\|(\boldsymbol{u_f}(\boldsymbol{x_p}(t)) - \boldsymbol{u_p}(\boldsymbol{x_p}(t))) \\ &+ \left(1 - \frac{\rho_f}{\rho_p}\right)\boldsymbol{g} \end{aligned} \tag{11}$$

avec C_d le coefficient de traînée. Ainsi, la vitesse fluide doit être évaluée à la position de la particule. La force de traînée peut également être exprimée comme la différence des vecteurs vitesse fluide et particulaire divisée par un temps de relaxation τ_p. En effet, la comparaison des équations (9) et (11) indique que l'ensemble des scalaires précédent le différentiel des vecteurs vitesse fluide et particulaire du terme de traînée dans l'équation (11) est homogène à l'inverse d'un temps. Ce temps caractérise alors la durée requise pour que la particule atteigne la vitesse fluide. Ainsi on obtient :

$$\boldsymbol{F_d}(t) = m_p \frac{\boldsymbol{u_f}(\boldsymbol{x_p}(t)) - \boldsymbol{u_p}(\boldsymbol{x_p}(t))}{\tau_p(Re_p)} \tag{12}$$

avec :

$$\tau_p = \frac{\rho_p d_p^2}{18\mu_f \beta(Re_p)} \tag{13}$$

où β représente un facteur correctif fonction du Reynolds particulaire. On peut montrer que :

$$\lim_{Re_p \to 0} \beta = 1 \tag{14}$$

Cependant, le nombre de Reynolds particulaire est loin d'être négligeable dans la plupart des applications, avec des valeurs typiques de l'ordre de quelques dizaines. Aucune expression analytique n'étant connue à ce jour pour le facteur correctif β dans de tels cas de figure, des corrélations empiriques sont utilisées. On peut citer la corrélation proposée par Schiller et Naumann [6] à titre d'exemple :

$$\beta = 1 + 0.15 Re_p^{0.687} \tag{15}$$

ou de façon équivalente :

$$C_d = \frac{24}{Re_p}\left(1 + 0.15 Re_p^{0.687}\right) \tag{16}$$

La corrélation de Schiller et Naumann [6] est valable jusqu'à des nombres de Reynolds particulaires de 800 environ, ce qui est largement suffisant pour la plupart des applications.

3.2 Résolution numérique

Maintenant que les équations permettant de décrire la dynamique particulaire ont été présentées, la résolution numérique de ces équations peut être traitée. Puisque chaque particule est suivie dans son propre référentiel, la résolution numérique du système (8-9) implique seulement une intégration temporelle :

$$\boldsymbol{x_p}(t) = \boldsymbol{x_p}(t_0) + \int_{t_0}^{t} \boldsymbol{u_p}(\boldsymbol{x_p}(t))\,\mathrm{d}t \tag{17}$$

$$\boldsymbol{u_p}(\boldsymbol{x_p}(t)) = \boldsymbol{u_p}(\boldsymbol{x_p}(t_0)) + \int_{t_0}^{t} \sum \boldsymbol{F_p}(t)\,\mathrm{d}t \tag{18}$$

On peut alors calculer les intégrales temporelles de façon approchée par des méthodes explicites de type Runge-Kutta. L'utilisation de méthodes implicites pour la résolution temporelle des équations lagrangiennes n'est pas aisée puisque les propriétés particulaires ne sont pas définies de façon continue. Ainsi, le calcul d'une jacobienne des flux nécessaire à l'implicitation n'est pas naturel dans l'approche lagrangienne.

Un autre point important est à souligner ici. Conformément aux équations (11) et (12), l'évaluation de la force de traînée nécessite la connaissance de la vitesse fluide à la position de la particule. Ainsi, bien qu'elle n'implique pas de dérivée spatiale, l'approche lagrangienne nécessite une interpolation de propriétés fluides à la position de la particule. Naturellement, la précision de la méthode lagrangienne dépendra en partie de l'ordre de cette interpolation.

Si on considère que l'équation (19) est intégrée via une approche explicite, le résultat se traduira par une relation algébrique permettant la mise à jour de la position de la particule :

$$\boldsymbol{x_p}(t_{n+1}) = \boldsymbol{x_p}(t_n) + u_p(\boldsymbol{x_p}(t_n))(t_{n+1} - t_n) = \boldsymbol{x_p}(t_n) + \Delta \boldsymbol{x_p}^{n,n+1} \tag{19}$$

La particule devra donc être déplacée sur le maillage de calcul. Afin de pouvoir procéder à l'interpolation des données du fluide porteur à la position des particules, les particules doivent être localisées sur le maillage de calcul, c'est-à-dire que l'indice de la cellule contenant une particule donnée doit être connu à chaque itération. Ainsi, lorsqu'une particule est déplacée, la cellule contenant son point d'arrivée doit être déterminée connaissant sa position et sa cellule à l'itération précédente ainsi que son vecteur déplacement. La section suivante décrira les algorithmes de localisation les plus courants trouvés dans la littérature pour un contexte non structuré.

3.3 Algorithme de localisation particulaire

De nombreux algorithmes de localisation particulaire existent dans la littérature. On se contentera de présenter deux concepts de repérage majeurs qui permettent d'englober de nombreuses variantes d'algorithmes présentés dans la littérature :

— Une approche basée sur des produits scalaires entre vecteurs déplacement et normales aux faces, proposée entre autres par Haselbacher *et al.* [7].
— Une autre basée sur des produits vectoriels pour situer le vecteur déplacement de la particule par rapport aux arêtes d'une cellule, initialement par Chorda *et al.* [8]

Ces deux approches seront brièvement décrites par la suite.

3.3.1 Algorithme de type Chorda

Nous présentons ici la méthode 2D de l'algorithme de repérage particulaire proposé par Chorda *et al.* [8]. L'algorithme de type Chorda se base sur deux tests : P2L et T2L (et sa variante T2R), consistant à observer le signe de produits vectoriels. L'idée de l'algorithme est de déterminer de quel côté d'une certaine ligne ou d'un certain plan (trajectoire, face...) la particule se trouve. La demi-droite du vecteur déplacement peut intersecter une face si celle-ci passe entre ses sommets, ce qui implique un changement de signe du produit vectoriel entre le vecteur déplacement et le vecteur reliant le point de départ de la trajectoire et les sommets. Si le test est positif, on vérifie que la particule est dans la cellule voisine en vérifiant si le point d'arrivée est situé à droite de toutes les arêtes via un autre produit vectoriel. Le concept est suffisamment général pour être transposable en trois dimensions, on testera alors si la demi-droite du vecteur déplacement passe à gauche ou à droite des arêtes d'une face.

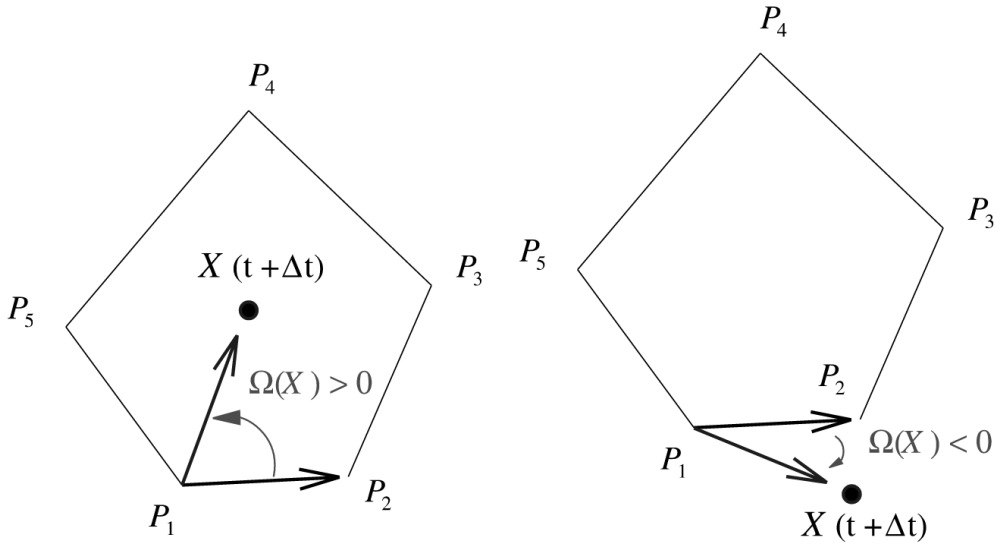

FIGURE 1 – Ce test appelé P2L étudie le signe de $(P2 - P1) \wedge (X(t + \Delta t) - P1)$ avec $X(t+\Delta t)$ la position de la particule à l'instant $t+\Delta t$. P2L permet de savoir si la particule est à l'intérieur de la cellule considérée. On dit qu'il est validé si le signe est positif. Image tirée de Chorda*et al.* [8]

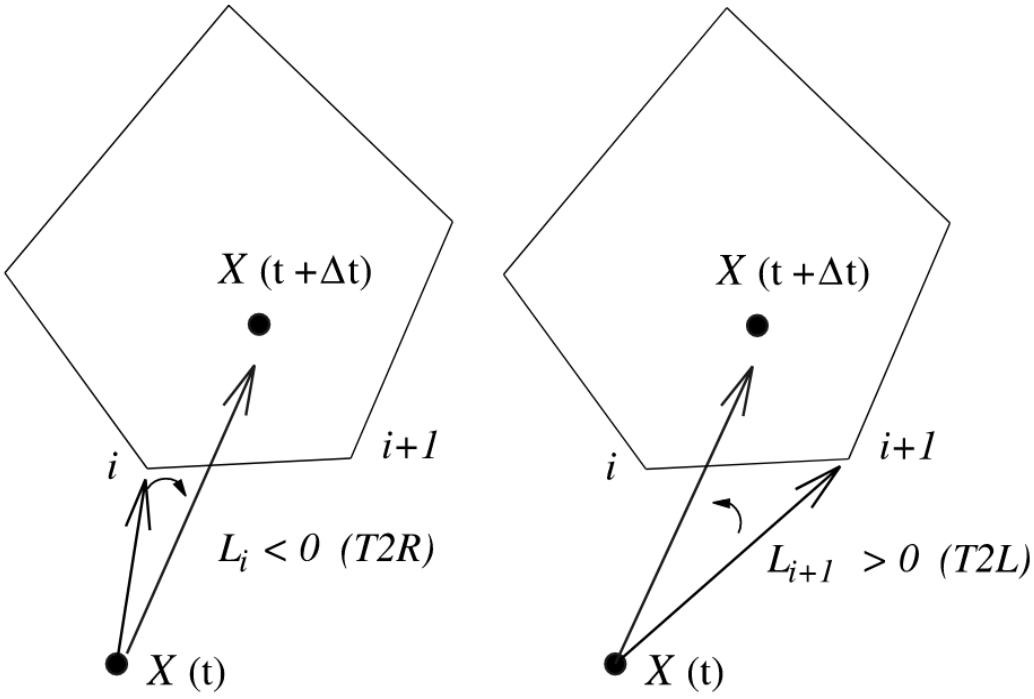

FIGURE 2 – Ces tests appelés T2L et T2R étudient le signe de $(X(t) - P) \wedge (X(t + \Delta t) - X(t))$ avec $X(t + \Delta t)$ la position de la particule à l'instant $t + \Delta t$ et P la position du sommet i ou i+1. P2L permet de savoir si la demi-droite du vecteur déplacement intersecte la face, ce qui sera le cas si T2L et T2R ont un signe différent. Image tirée de Chorda*et al.* [8]

Algorithm 1: Algorithme de Chorda

Input: X(t) la position de la particule à t, X(t+dt) la position de la particule à t+dt, CurrCell l'élément contenant X(t)
Output: cellule contenant X(t+dt)

```
Vérifier si la trajectoire X(t)X(t + Δt) traverse une face de la cellule en comparant
 le signe des tests T2L et T2R
if test T2L/T2R vérifié pour une face then
    Aller à la cellule connexe
    Appliquer le test P2L pour chaque face
    if P2L < 0 pour chaque face then
        La cellule d'arrivée a été trouvée
    end
end
else
    La particule n'a pas changé de cellule entre t et t + Δt
end
```

3.3.2 Algorithme type Haselbacher

Cet algorithme est plus proche de celui qui sera utilisé dans JAGUAR. Il se base sur l'utilisation du produit scalaire pour déterminer à la fois si une particule est bien contenue dans une cellule donnée et pour repérer les faces d'une cellule que la particule peut potentiellement traverser lors de son déplacement. Le test de présence dans une cellule s'effectuera sur le signe de $H_{j,i} = \langle \boldsymbol{X}_j - \boldsymbol{G}_i, \boldsymbol{n}_i \rangle$ avec $\boldsymbol{X}_j$ la position de la particule j, $\boldsymbol{G}_i$ le centre de gravité de la face i (simplement le milieu de l'arête i en 2D) et $\boldsymbol{n}_i$ le centre de gravité de la face i. Si $\forall i, H_i > 0$ la particule est localisée dans la cellule, on appellera cela le test de Haselbacher *et al.* [7], cf. fig. 3. Le test de sortie d'une cellule consiste à étudier le signe du produit scalaire entre son vecteur déplacement et les vecteurs des normales aux faces : $\langle (\boldsymbol{X}(t + \Delta t) - \boldsymbol{X}(t), \boldsymbol{n}_i \rangle$. Les normales étant considérées comme sortantes, la particule ne peut quitter la cellule que par une face pour laquelle ce produit scalaire est positif. Finalement, parmi les faces pour lesquelles un produit scalaire positif est obtenu, la particule sort de la cellule par la face qui minimise la distance entre le point de départ de la particule et ce point d'intersection.

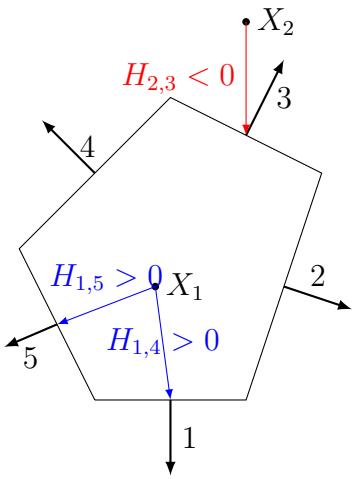

FIGURE 3 – le test d'Haselbacher, consistant à étudier le signe du produit scalaire entre d'un côté les vecteurs reliant le point de départ du vecteur déplacement aux centres de gravité des faces et de l'autre les normales à ces faces, est valide pour X_1 mais pas pour X_2 : seule la particule 1 est dans la cellule.

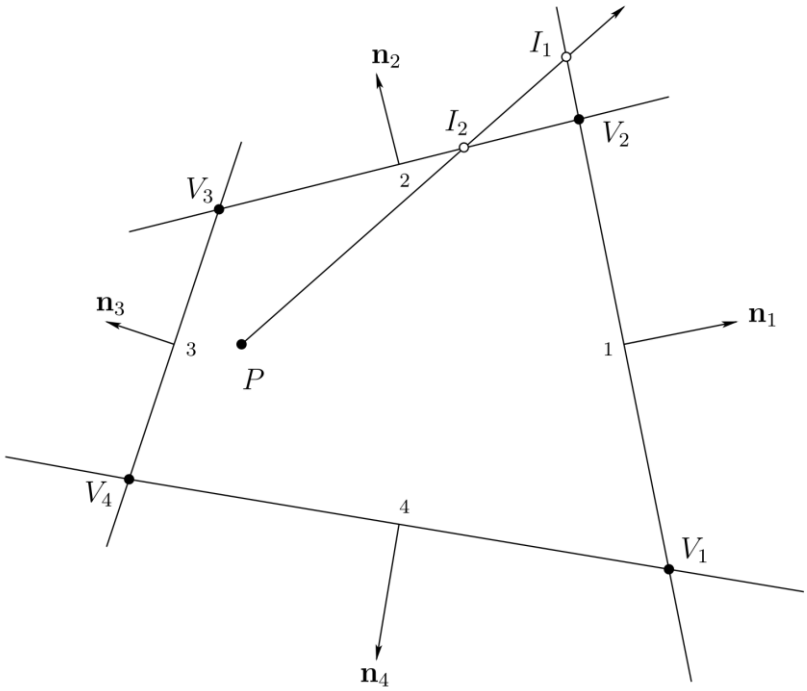

FIGURE 4 – Exemple de la méthode d'Haselbacher. Les conditions de sortie $\langle(\boldsymbol{X}(t+\Delta t)-\boldsymbol{X}(t), \boldsymbol{n}_i\rangle$ sont vérifiées pour les faces 1 et 2. Deux intersections I_1 et I_2 sont calculées Figure tirée de Haselbacher *et al.* [7]

Algorithm 2: Algorithme de type Haselbacher*et al.* [7]

Input: X(t) la position de la particule à t, X(t+dt) la position de la particule à t+dt, CurrCell l'élément contenant X(t)
Output: cellule contenant X(t+dt)

```
d=X(t+dt)-X(t)
while le test d'Haselbacher échoue do
    for chaque face i do
        if d · n_i > 0 then
            calcul de l'intersection I_i entre d et la face i
        end
    end
    I = min_i(I_i)
    d = d − I
    Mettre à jour CurrCell avec la cellule connexe à la face contenant I
end
```

Dans l'algorithme de Chorda comme dans celui d'Haselbacher, plusieurs variantes existent pour la mise en oeuvre des différentes étapes mais le plan reste le même : chercher une face par laquelle la particule est potentiellement passée, aller dans la cellule connexe et recommencer l'algorithme tant que la particule n'est pas localisée dans la cellule actuelle.

4 Transformations isoparamétriques

4.1 Utilité des transformations isoparamétriques

Comme décrit annexe C , la méthode SD nécessite des transformations vers un élément de référence dit isoparamétrique afin de calculer les dérivées spatiales intervenant dans les équations de Navier-Stokes. Par ailleurs, la gestion des polynômes décrivant la solution se fait également dans l'espace isoparamétrique.

Pour l'algorithme de localisation lagrangienne, cela offre deux possibilités : la localisation des particules peut être effectuée sur le maillage d'origine / l'espace physique ou l'espace isoparamétrique.

Il semble plus avantageux de gérer la localisation particulaire dans l'espace isoparamétrique. Tout d'abord, cette gestion permettrait de naturellement traiter les éléments d'ordre élevé, pour lesquels les algorithmes décrits précédemment semblent inadaptés. De plus, l'interpolation de propriétés fluides à la position des particules nécessiterait de toute façon des transformations de l'espace physique vers l'espace isoparamétrique. Cependant, la transformation de l'espace physique vers l'espace isoparamétrique repose sur une méthode itérative assez coûteuse, cf. algorithme 4 section 5.1.2. Ainsi, il a été choisi de dériver un algorithme permettant de traiter le repérage particulaire dans l'espace isoparamétrique. À la connaissance de l'auteur, un tel algorithme n'a pas été publié dans la littérature à ce jour.

On distingue de nombreux types de maillages (voir Annexes section B.1), dont certains sont bien plus complexes que des maillages cartésiens (par exemple les maillages avec éléments courbes). Dans l'espace isoparamétrique , tout quadrangle devient un carré et tout hexaèdre un cube (et ce, même si les éléments sont courbes), ce qui simplifie grandement les calculs liés à la localisation une fois la transformation dans l'espace effectuée.

On définira $\boldsymbol{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$ les coordonnées dans l'espace physique et $\boldsymbol{\xi} = \begin{pmatrix} \xi \\ \eta \\ \psi \end{pmatrix}$ les coordonnées dans l'espace isoparamétrique.

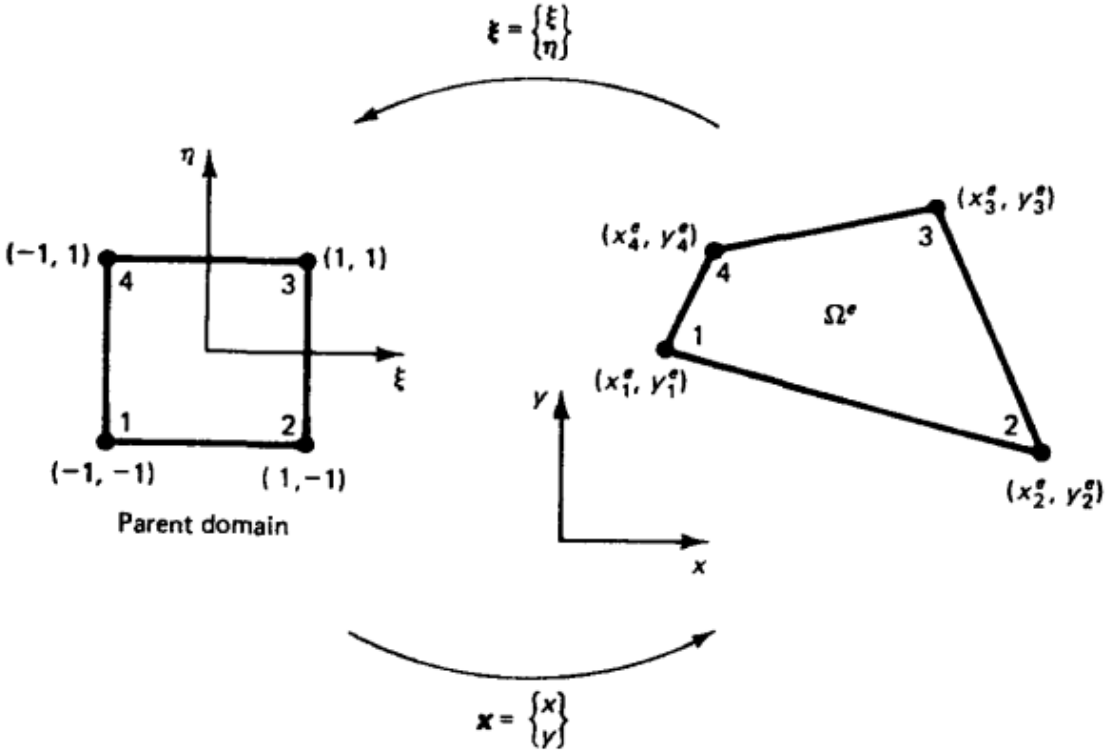

FIGURE 5 – Changement d'espace pour un quadrilatère et numérotation locale de nœuds. Figure tirée de Hugues [9]

4.2 Calcul des éléments isoparamétriques

La transformation isoparamétrique est définie via des fonctions de forme, écrites sous forme de polynômes. Dans le cas d'un quadrangle en deux dimensions, on dénombre quatre fonctions de forme N_i définies par la relation :

$$\begin{cases} x(\xi,\eta) = \displaystyle\sum_{i=1}^{4} N_i(\xi,\eta) x_i^e \\ y(\xi,\eta) = \displaystyle\sum_{i=1}^{4} N_i(\xi,\eta) y_i^e \end{cases} \tag{20}$$

avec x_i^e et y_i^e les valeurs aux nœuds, comme montré fig. 5. La bilinéarité des éléments permet d'écrire :

$$\begin{cases} x(\xi,\eta) = a_0 + a_1\xi + a_2\eta + a_3\xi\eta \quad \text{avec} \quad (a_0,a_1,a_2,a_3) \in \mathbb{R}^4 \\ y(\xi,\eta) = b_0 + b_1\xi + b_2\eta + b_3\xi\eta \quad \text{avec} \quad (b_0,b_1,b_2,b_3) \in \mathbb{R}^4 \end{cases} \tag{21}$$

Il n'y a pas de méthode explicite pour passer de l'espace physique à l'espace isoparamétrique, on ne peut pas explicitement inverser le problème. Dans notre cas on utilisera un algorithme de Newton présenté en 5.1.2. Pour trouver la valeur de ces nouvelles

inconnues, on utilise la valeur aux nœuds $\boldsymbol{x}_i^e$.

$$\begin{cases} x(\xi_i, \eta_i) = x_i^e \\ y(\xi_i, \eta_i) = y_i^e \end{cases} \tag{22}$$

Dans cet exemple et dans la plupart des utilisations des transformations isoparamétrique concernant un quadrangle, on utilise le carré $[-1, 1]^2$, c'est-à-dire que ξ_i et η_i valent -1 ou 1, mais JAGUAR utilise le carré $[0, 1]^2$. D'après les équations (20) et (22), les fonctions de forme doivent vérifier la relation suivante :

$$N_i(\xi_j, \eta_j) = \delta_{ij} \tag{23}$$

avec δ_{ij} le symbole de Kronecker

$$\delta_{ij} = \begin{cases} 1 \,\text{si}\, i = j \\ 0 \,\text{sinon} \end{cases} \tag{24}$$

Par exemple dans le cas du carré $[-1, 1]^2$ on obtiendra le système :

$$\big(x_1, x_2, x_3, x_4\big) = \begin{pmatrix} 1 & \xi_1 & \eta_1 & \xi_1\eta_1 \\ 1 & \xi_2 & \eta_2 & \xi_2\eta_2 \\ 1 & \xi_3 & \eta_3 & \xi_3\eta_3 \\ 1 & \xi_4 & \eta_4 & \xi_4\eta_4 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \tag{25}$$

$$\big(y_1, y_2, y_3, y_4\big) = \begin{pmatrix} 1 & \xi_1 & \eta_1 & \xi_1\eta_1 \\ 1 & \xi_2 & \eta_2 & \xi_2\eta_2 \\ 1 & \xi_3 & \eta_3 & \xi_3\eta_3 \\ 1 & \xi_4 & \eta_4 & \xi_4\eta_4 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} \tag{26}$$

Après résolution du système, on obtiendra les fonctions de forme suivantes :

$$N_i(\xi, \eta) = \frac{1}{4}(1 + \xi_i\xi)(1 + \eta_i\eta) \quad \text{avec} \quad (\xi_i, \eta_i) \in \{-1, 1\}^2 \tag{27}$$

On pourrait se contenter de résoudre le système matriciel précédent. Cependant connaître la valeur des N_i peut être utile. En effet les fonctions de forme permettent aussi d'interpoler une fonction scalaire connaissant ses valeurs d_i^e aux nœuds.

$$u(\boldsymbol{\xi}) = \sum_i N_i(\boldsymbol{\xi}) d_i^e \tag{28}$$

Le lecteur est renvoyé à Hugues [9] pour plus de détails.

Les fonctions de forme permettent donc d'effectuer un changement de variable de l'espace isoparamétrique (ξ, η) vers l'espace physique (x, y), c'est-à-dire que connaissant les coordonnées (ξ_i, η_i) d'un point d'une cellule, on peut immédiatement trouver les coordonnées physiques correspondantes en vertu de la relation (20). Cependant, cette relation n'est pas inversible analytiquement. Ainsi il faut recourir à une méthode itérative pour trouver les coordonnées (ξ_i, η_i) d'un point d'une cellule connaissant (x_i, y_i).

L'idée dans JAGUAR est de considérer ces transformations comme des changements de variable en définissant une jacobienne $\mathcal{J}$. $\mathcal{J}$ joue un rôle crucial dans le code actuel et est de plus très utile pour la partie localisation comme on le verra plus tard.
Pour exemple en 2D on peut appliquer la règle de la chaîne à $\mathcal{F}$:

$$\frac{\partial \mathcal{F}}{\partial \boldsymbol{x}} = \frac{\partial \mathcal{F}}{\partial \boldsymbol{\xi}} \frac{\partial \boldsymbol{\xi}}{\partial \boldsymbol{x}} \quad = \quad \frac{\partial \xi}{\partial x}\frac{\partial F}{\partial \xi} + \frac{\partial \eta}{\partial x}\frac{\partial F}{\partial \eta} + \frac{\partial \xi}{\partial x}\frac{\partial G}{\partial \xi} + \frac{\partial \eta}{\partial x}\frac{\partial G}{\partial \eta} \tag{29}$$

On peut alors écrire :

$$\frac{\partial \mathcal{F}}{\partial \boldsymbol{x}} = \mathcal{J}^{-1} \frac{\partial \mathcal{F}}{\partial \boldsymbol{\xi}} \tag{30}$$

avec

$$\mathcal{J} = \begin{pmatrix} \dfrac{\partial X}{\partial \xi} & \dfrac{\partial X}{\partial \eta} \\ \dfrac{\partial Y}{\partial \xi} & \dfrac{\partial Y}{\partial \eta} \end{pmatrix} \tag{31}$$

avec (X, Y) les fonctions telles que $\begin{cases} X(\xi, \eta, \psi) = x \\ Y(\xi, \eta, \psi) = y \end{cases}$.
Pour exemple,

$$\frac{\partial X}{\partial \eta} = (1 - \psi)(X_1 - X_2) + \psi(X_4 - X_3) \tag{32}$$

avec $X_{i \in [\![1,8]\!]}$ les sommets de l'élément dans l'espace physique. On retrouve le même type d'expression qu'avec le calcul des N_i.

D'ailleurs, on a vu que :

$$\mathcal{J}_{ij} = \sum_{k=1}^{4} \frac{\partial X_{ik}}{\partial \xi_j} \tag{33}$$

en combinant avec l'équation (20), on peut écrire :

$$\mathcal{J}_{ij} = \sum_{k=1}^{4} \frac{\partial \hat{N}_k}{\partial \xi_j} x^e_{ik} \tag{34}$$

$\hat{N}_k$, comme N_k, est une interpolation. Le qualificatif d'**isoparamétrique** est employé quand, comme dans notre cas, $\hat{N}_k = N_k$, c'est-à-dire quand l'élément est basé sur des interpolations identiques pour sa géométrie et ses inconnues (voir [10] pour plus de détails)

Remarque: Pour des éléments d'ordre plus élevés, on utilisera les polynômes de Lagrange. Cela sera notamment utile pour les éléments courbes, qui nécessitent plus de points par élément. Le cas bilinéaire est aussi basé sur les polynômes de Lagrange parce qu'ils satisfont par construction la relation (23).

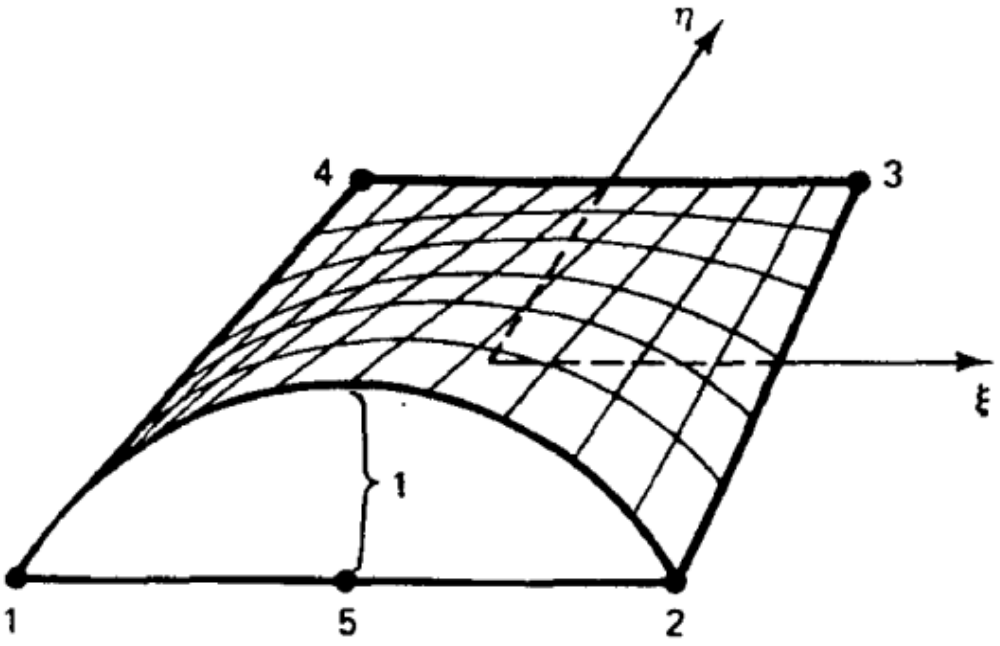

FIGURE 6 – Influence d'un nœud supplémentaire sur la courbure d'un élément. Figure tirée de l'ouvrage [9]

5 Localisation

5.1 Algorithmes

Contrairement aux sections précédentes, ces idées et résultats proviennent majoritairement de mon travail personnel sur JAGUAR.

5.1.1 Algorithme d'initialisation

A l'initialisation, il est nécessaire de calculer les coordonnées isoparamétriques de la particule à partir de ses coordonnées physiques. Ceci requiert cependant la connaissance de la cellule contenant la particule car les transformations isoparamétriques sont locales. De plus le passage des coordonnées physiques aux coordonnées isoparamétriques demande l'évaluation de la matrice jacobienne. Cependant cette dernière n'est définie explicitement que dans l'espace isoparamétrique, cf eq. (20), eq. (31) et eq. (32). L'algorithme implémenté est alors le suivant : sur chaque cellule, on réalise le test d'Haselbacher présenté 3.3.2, c'est-à-dire qu'on calcule

$$\langle \boldsymbol{n}_i, \boldsymbol{X} - \boldsymbol{G} \rangle \tag{35}$$

avec $i \in [\![1, 6]\!]$ pour les six faces de l'hexaèdre et

$$\boldsymbol{G} = \frac{P_1 + P_2 + P_3 + P_4}{4} \tag{36}$$

avec $P_i, i \in [\![1, 4]\!]$ les coordonnées des sommets d'une face. Dans JAGUAR les normales ne sont pas stockées dans la version initiale, elles sont donc recalculées via un produit vectoriel sur les arêtes. Cet algorithme ne présente pas de difficulté à être adapté à des tétraèdres.

L'algorithme d'initialisation est assez coûteux, ($O(n_p n_c)$) avec n_p le nombre de particules et n_c le nombre de cellules, mais il est appelé une seule fois, à l'initialisation. De plus il pourra être amélioré en utilisant un système d'octrees, c'est-à-dire des "boîtes" dans lesquelles on peut facilement prélocaliser la particule. On pourra ensuite stocker les cellules appartenant à chaque "boîte" et n'effectuer la recherche que sur ces dernières.

5.1.2 Algorithme de localisation

Nous présentons l'algorithme global, très proche de l'algorithme 2. La réalisation des différents tests et étapes est explicitée plus bas.
L'algorithme consiste, en partant d'une cellule, à calculer un vecteur déplacement $\Delta\boldsymbol{\xi}$ dans l'espace isoparamétrique puis à trouver la face potentielle de sortie de la particule, comme montré figure 9 et 10. Par un calcul d'intersection illustré fig. 11 on peut trouver l'unique face de sortie. A partir des coordonnées de l'intersection on peut réitérer l'algorithme.

Algorithm 3: Algorithme implémenté dans JAGUAR pour la localisation de particules

Input: $\boldsymbol{X}(t)$ la position de la particule à t, $\boldsymbol{X}(t+dt)$ la position de la particule à t+dt, CurrCell l'élément contenant $\boldsymbol{X}(t)$
Output: cellule contenant $\boldsymbol{X}(t+dt)$

```
NewCell=CurrCell
ΔX = X(t+dt) − X(t)
while la particule n'est pas localisée do
    Calcul du vecteur déplacement dans l'espace isoparamétrique Δξ = J^{-1} ΔX
    for chaque face do
        Effectuer un test de sortie de cellule
        if la particule quitte la face then
            Calculer l'intersection P_intersection entre Δξ et le plan contenant la face
            if l'intersection se fait sur la face then
                #La particule est passée par la face i
                Mettre à jour NewCell
                X(t) = COORDS_PHYSIQUE(P_intersection)
                ΔX = X(t+dt) − X(t)
            end
        end
    end
end
```

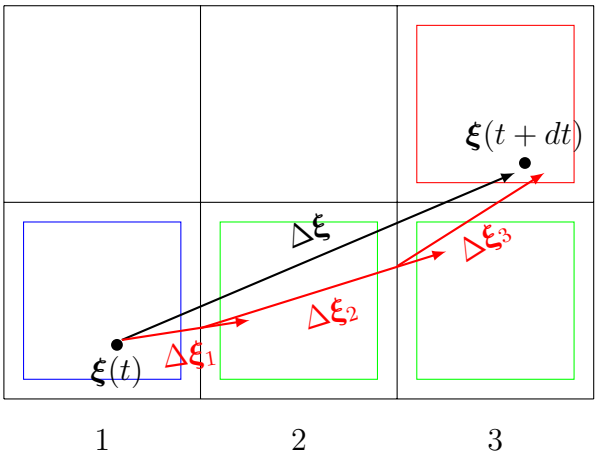

FIGURE 7 – Exemple de localisation 2D, □ indique les cellules traversées, □ la cellule de départ, □ la cellule d'arrivée. ⟶ représente un cas où la jacobienne $\mathcal{J}$ est la même quelle que soit la cellule, c'est-à-dire que les cellules sont identiques dans le maillage d'origine, comme dans un maillage cartésien par exemple. Les flèches rouges ⟶ représentent une localisation en 3 étapes dans un cas quelconque.

On rappelle que les transformations isoparamétriques sont locales, c'est-à-dire spécifiques à chaque élément. Ainsi l'algorithme de repèrage basé sur ces transformations devra nécessairement réévaluer les jacobiennes dans chaque élément. Par exemple dans la figure 7 les déplacements $\Delta\boldsymbol{\xi}_1$, $\Delta\boldsymbol{\xi}_2$ et $\Delta\boldsymbol{\xi}_3$ correspondent respectivement à des jacobiennes calculées dans les éléments 1, 2 et 3.

Passage en coordonnées isoparamétriques et méthode de Newton
Pour pouvoir effectuer le calcul $\Delta\boldsymbol{\xi} = \mathcal{J}^{-1}\Delta\boldsymbol{X}$, il faut connaître $\mathcal{J}$ qui dépend de l'élément et de la position et du point d'application du vecteur déplacement. A l'initialisation il faut donc connaître $\boldsymbol{\xi}$ (les coordonnées isoparamétriques de la particule), ne connaissant que la cellule contenant la particule et ses coordonnées dans l'espace physique. Pour cela on va localiser le point solution (cf Annexes C et D) le plus proche de la particule. On se place alors dans l'espace isoparamétrique car on connaît les coordonnées isoparamétriques du point solution le plus proche et on converge vers les coordonnées isoparamétriques de la particule à l'aide d'un algorithme de Newton.

Algorithm 4: Algorithme de passage en coordonnées isoparamétriques

Input: $\boldsymbol{\xi_s}$ les coordonnées isoparamétriques du point solution, $\boldsymbol{X_s}$ les coordonnées cartésiennes du point solution le plus proche de la particule, $\boldsymbol{X_p}$ les coordonnées cartésiennes de la particule

Output: $\boldsymbol{\xi_p}$ les coordonnées isoparamétriques de la particule

while $\Delta\boldsymbol{\xi}_{err} < \epsilon$ **do**
  $\Delta\boldsymbol{X}_{err} = \boldsymbol{X}_s - \boldsymbol{X}_p$
  $\Delta\boldsymbol{\xi}_{err} = \mathcal{J}^{-1}(\boldsymbol{\xi}_s)\Delta\boldsymbol{X}_{err}$
  $\boldsymbol{\xi}_p = \boldsymbol{\xi}_s + \Delta\boldsymbol{\xi}_{err}$
  $\boldsymbol{X}_p = COORDS\_PHYSIQUE(\boldsymbol{\xi}_p)$
end
$\boldsymbol{\xi}_s = \boldsymbol{\xi}_p$

Cela donne l'idée principale de l'algorithme. Dans notre implémentation il se peut que le déplacement dans la direction de la particule soit trop important et que l'erreur sur $\boldsymbol{\xi}$ augmente, c'est-à-dire que l'algorithme de Newton ne converge pas. Pour cela après chaque itération on calcule un nouveau $\Delta \boldsymbol{X_{err}}$ et on le compare à la valeur précédente pour voir si on s'est rapproché des réelles coordonnées de la particule. Si ce n'est pas le cas, on recommence le calcul avec $a\, d\boldsymbol{X_{err}}, a \in]0,1[$, $a = 0.75$ semble assez efficace mais aucune étude détaillée n'a été effectuée à ce sujet.

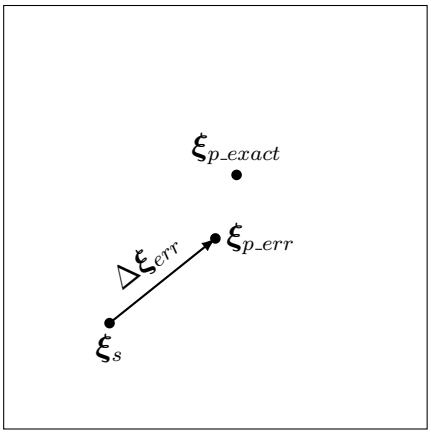

FIGURE 8 – Une étape du schéma de Newton. On obtient ξ_{p_err}, plus proche de ξ_{p_exact} que ξ_s mais il y a toujours une erreur. Plus on itère et plus cette erreur diminue.

Détection de sortie de la particule Pour sortir d'une cellule par une certaine face, une particule doit satisfaire deux conditions, nécessaires mais non suffisantes sur son déplacement durant dt :

— La 1ère condition consiste à tester le produit scalaire

$$\langle \boldsymbol{n}_i, \Delta\boldsymbol{\xi} \rangle > 0 \tag{37}$$

$\boldsymbol{n}_i$ étant le vecteur sortant de la face i comme indiqué figure 9. La normale et le vecteur déplacement doivent avoir le même sens (plus formellement les demi-droites engendrées par leur direction doivent être contenues dans le même demi-plan).

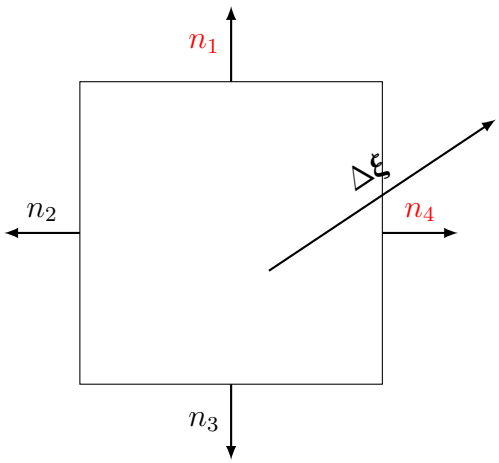

FIGURE 9 – Pour cette figure, la particule peut sortir par les faces 1 et 4 en vertu du test sur l'orientation du vecteur déplacement par rapport aux vecteurs normaux aux faces.

— La 2e condition s'écrit

$$|\langle \boldsymbol{n}_i, 2(\boldsymbol{\xi} + d\boldsymbol{\xi}) - \mathbb{1}\rangle| > 1, \mathbb{1} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \tag{38}$$

On vérifie si $\boldsymbol{\xi}$ déplacé de $\Delta\boldsymbol{\xi}$ sort de la cellule. On réécrit juste l'inégalité

$$0 \leq \langle \boldsymbol{\xi} + \Delta\boldsymbol{\xi}, \boldsymbol{n}_i \rangle \leq 1 \tag{39}$$

qui est une condition nécessaire et suffisante (CNS) pour que la particule reste dans l'élément, en $-1 < \langle 2(\boldsymbol{\xi} + \Delta\boldsymbol{\xi}) - \mathbb{1}, n_i \rangle \leq 1$. Pour cela on multiplie (39) par 2 et on retranche 1 en prenant en compte que les normales n'ont qu'une seule composante non nulle.
On notera que les normales utilisées ici, contrairement à l'algorithme d'initialisation, sont les normales de l'espace isoparamétrique. Elles sont donc connues et simples à écrire. En suivant les conventions utilisées dans JAGUAR, on a :

$$n_1 = \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix} \; n_2 = \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix} \; n_3 = \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix} \; n_4 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \; n_5 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \; n_6 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

On en déduit la CNS pour que la particule sorte de l'élément.

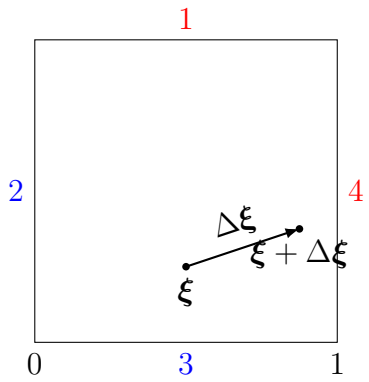

FIGURE 10 – Exemple de test de détection pour la vitesse de déplacement, il y avait une possibilité de sortie par les faces 1 et 4, mais le déplacement n'est pas assez grand, donc la particule reste dans la cellule

Détection de la face de sortie par calcul d'intersection Les conditions précédentes, illustrées figure 9 et 10, ont permis de déterminer au maximum 3 faces candidates à la traversée de la particule dans le cas 3D, 2, dans le cas 2D (dans le cas 3D la condition $\langle \boldsymbol{n}_i, d\boldsymbol{\xi} \rangle > 0$ va déterminer 3 faces voisines, ou moins dans des cas particuliers). Pour connaître l'unique face de sortie, on calcule l'intersection entre le plan contenant une face et la droite contenant $\Delta\boldsymbol{\xi}$. Si cette intersection est contenue dans la face testée, alors cette face est la face de sortie.
On rappelle que l'équation paramétrique d'une droite peut se mettre sous la forme :

$$\boldsymbol{X} = \boldsymbol{X_0} + t\boldsymbol{A} \tag{40}$$

$\boldsymbol{X_0} \in \mathbb{R}^3$ est un point de la droite, $\boldsymbol{A} \in \mathbb{R}^3$ la direction de la droite, $t \in \mathbb{R}$ le paramètre. Appliqué dans notre cas au point d'intersection, cela donne :

$$\boldsymbol{P} = \boldsymbol{\xi} + \Delta\boldsymbol{\xi} t_I \tag{41}$$

avec $\boldsymbol{P}$ le point d'intersection. Il reste à trouver t_I. Considérons l'équation cartésienne du plan :

$$ax + by + cz + d = 0 \quad \text{avec} \quad [a, b, c, d] \in \mathbb{R}^4 \tag{42}$$

que l'on peut écrire :

$$\langle \boldsymbol{B}, \boldsymbol{X} \rangle + d = 0 \tag{43}$$

avec $\boldsymbol{B} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$. $\boldsymbol{P}$ est dans ce plan donc :

$$\langle \boldsymbol{B}, \boldsymbol{P} \rangle + d = 0 \tag{44}$$

d'où :

$$\langle \boldsymbol{B}, \boldsymbol{\xi} + t_I \boldsymbol{\Delta\xi} \rangle + d = 0 \tag{45}$$

et finalement :

$$t_I = -\frac{d + \langle \boldsymbol{B}, \boldsymbol{\xi} \rangle}{\langle \boldsymbol{B}, \Delta\boldsymbol{\xi} \rangle} \tag{46}$$

Dans notre cas les équations sont relativement simples. En effet les plans contenant les faces des cubes dans l'espace isoparamétrique sont d'équation $x = 0$ ou $x = 1$ ou $y = 0$ ou $y = 1$ ou $z = 0$ ou $z = 1$ ce qui permet d'avoir simplement :

$$t_I = -\frac{d + \xi_j}{d\xi_j} \tag{47}$$

avec d pouvant valoir 0 ou -1 et $\xi_j, j \in [\![1, 3]\!]$ une des trois composantes de $\boldsymbol{\xi}$.

Par exemple, pour la face 1, de normale sortante $n_1 = \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix}$, l'équation du plan contenant la face est $z = 0$, ce qui correspond à $d = 0$ et donc $t_I = \frac{-\xi_3}{d\xi_3}$.

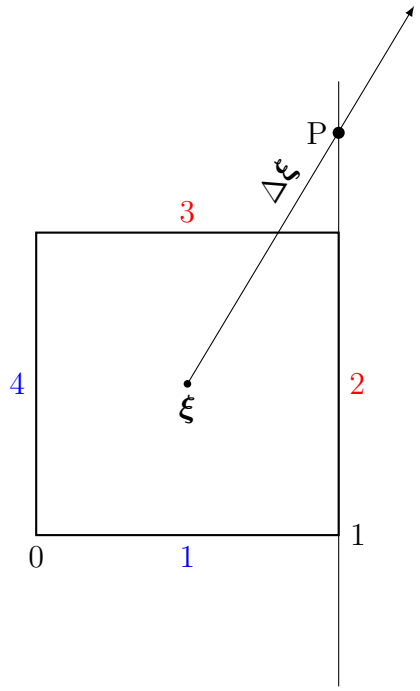

FIGURE 11 – La particule peut sortir par les faces 2 et 3. Cependant sa trajectoire n'intersecte pas la face 2, uniquement la face 3. La particule sort donc par la face 3.

5.1.3 Mise à jour des coordonnées isoparamétriques lors du changement d'élément

Une difficulté qui n'avait pas été prise en compte au tout début du stage est que les systèmes de coordonnées isoparamétriques de cellules accolées dans l'espace physique n'ont pas nécessairement la même orientation. Ceci est lié au fait que cette orentation découle de façon implicite de la numérotation des fonctions de forme, cf eq. (20). En 3D, il y a 8 sommets et 3 possibilités d'orientation par sommet (le trièdre doit être direct), ce qui fait 24 possibilités. Il n'est pas envisageable de tester tous les cas de figure.

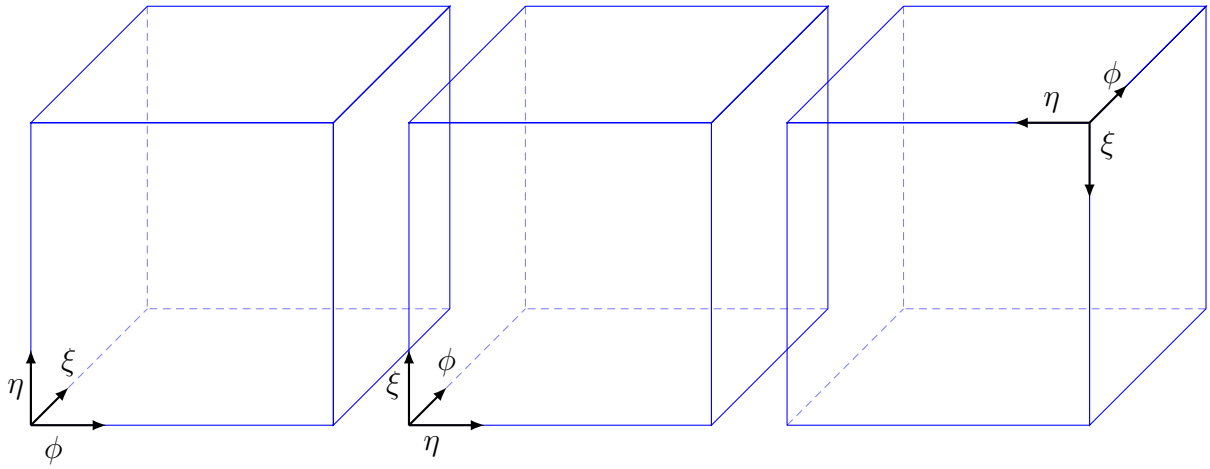

FIGURE 12 – Exemples d'orientation du trièdre sans changement d'origine entre la gauche et le milieu, avec changement d'origine entre le milieu et la droite

On commencera par étudier les transformations du cas 2D, plus simple mais analogue au cas 3D.

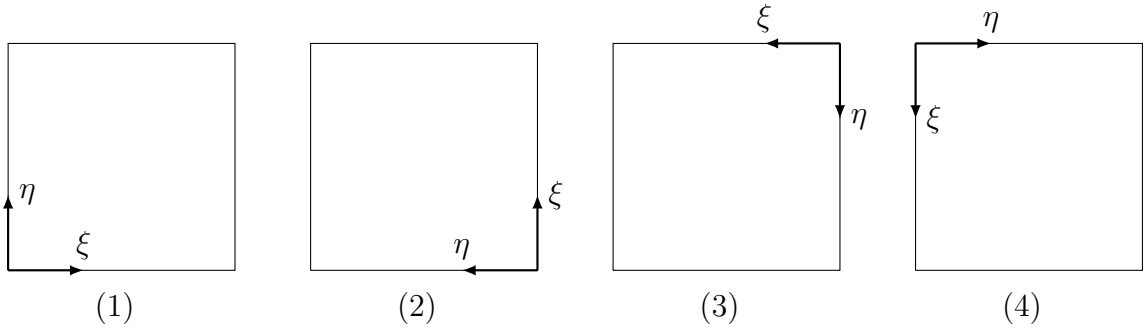

FIGURE 13 – Les 4 orientations possibles en 2D.

En 2D, il n'y a que 4 orientations possibles illustrées figure 13. Pour passer du cas 1 au cas i ($i \in [\![1, 4]\!]$) on effectue une transformation affine :

$$\boldsymbol{\xi_i} = \boldsymbol{P}_i^1 \boldsymbol{\xi_1} + \boldsymbol{A} \tag{48}$$

$\boldsymbol{A}$ est la translation correspondant au changement d'origine, et $\boldsymbol{P}_i^1$ est la matrice de passage de i à 1, c'est-à-dire la matrice dont les colonnes sont les coordonnées des vecteurs de 1, exprimés dans la base i.
On note $\boldsymbol{\xi}_i, \boldsymbol{\eta}_i$ les vecteurs de la base i. Pour calculer P_2^1, le raisonnement est le suivant :

$$\begin{cases} \boldsymbol{\xi}_1 = -\boldsymbol{\eta}_2 \\ \boldsymbol{\eta}_1 = \boldsymbol{\xi}_2 \end{cases} \text{ donc } \boldsymbol{P}_2^1 = \underbrace{\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}}_{\xi_1 \quad \eta_1}$$

$\boldsymbol{P}_2^1$ étant une matrice orthogonale, on peut écrire $\boldsymbol{P}_1^2 = (\boldsymbol{P}_2^1)^{-1} = (\boldsymbol{P}_2^1)^T$. On peut donc aussi voir la matrice sous la forme $\begin{matrix} \xi_2 \\ \eta_2 \end{matrix} \begin{cases} \\ \\ \end{cases} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$

On a donc les 4 matrices correspondant aux 4 transformations possibles :

$$\boldsymbol{P}_1^1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \boldsymbol{P}_2^1 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \boldsymbol{P}_3^1 = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} \boldsymbol{P}_4^1 = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

On remarquera que $\det(P_i^1) = 1$, ce qui correspond au caractère direct du trièdre.
En 3D les matrices de transformations sont donc les matrices de $\mathcal{M}_{3,3}(\mathbb{N})$ telles que :

- Les coefficients sont 0, 1 ou -1
- Il y a un seul coefficient non nul par ligne
- Il y a un seul coefficient non nul par colonne
- le déterminant de la matrice est 1

Avec cette définition on retrouve les 24 transformations évoquées pour le cas 3D (3 positions et 2 signes possibles pour la 1re ligne, 2 positions et 2 signes possibles pour la 2e ligne, 1 position et 1 seul signe possible pour la 3e ligne pour avoir un déterminant valant 1 pour la 3e ligne).

Il faut ensuite déterminer A. Pour cela on utilise le fait qu'on travaille localement dans le cube $[0,1]^3$ (carré $[0,1]^2$ en 2D). Ainsi un changement d'orientation des deux systèmes de coordonnées implique nécessairement un décalage d'origine entre les deux repères. De manière plus pratique, on ajoute 1 à une i-ème coordonnée après transformation quand il y a un -1 dans la ligne i.
On peut alors écrire :

$$A_i = \begin{cases} 1 \text{ si } \sum_j P_{ij} = -1 \\ 0 \text{ sinon} \end{cases} \tag{49}$$

ou bien :

$$A_i = \frac{1}{2}(1 - \sum_j P_{ij}) \tag{50}$$

Il est alors temps de soulever une subtilité de notre cas, qui est que la particule change de cellule à l'interface. En se référant à la figure 14, on souhaite garder la même origine quand on passe de la cellule L à R. On réalise cette étape préliminaire pour simplifier les étapes suivantes. On pourrait en effet calculer des nouvelles coordonnées de la forme :

$$\boldsymbol{\xi_i} = \boldsymbol{P}_i^1 \boldsymbol{B} + \boldsymbol{C} \tag{51}$$

Cependant $\boldsymbol{C}$ n'est en pratique pas évident à calculer. Sur l'algorithme global du changement d'orientation cela reviendrait à prendre le complément (changer un 0 en 1 et en 1 en 0) de la coordonnée de la particule à l'interface si et seulement si il y a un changement de sens du vecteur correspondant. On préférera introduire le vecteur $\boldsymbol{B}$ tel que $\boldsymbol{C} = \boldsymbol{P}_i^1 \boldsymbol{B} + \boldsymbol{A}$ ce qui donnera :

$$\boldsymbol{\xi_i} = \boldsymbol{P}_i^1 (\boldsymbol{\xi_1} + \boldsymbol{B}) + \boldsymbol{A} \tag{52}$$

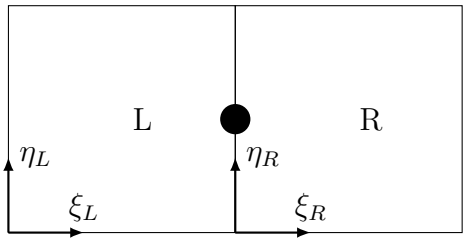

FIGURE 14 – La particule à l'interface avec la cellule Left et la cellule Right. Même si l'orientation reste la même, la coordonnée en ξ changera selon la cellule.

$$B_j = \begin{cases} 1 \text{ si j correspond à la coordonnée à l'interface et } \boldsymbol{\xi} = 0 \\ -1 \text{ si j correspond à la coordonnée à l'interface et } \boldsymbol{\xi} = 1 \\ 0 \text{ sinon} \end{cases}$$

En notant ξ_{inter} la coordonnée à l'interface :

$$B_j = \begin{cases} 1 - \xi_{inter} \text{ si j correspond à la coordonnée à l'interface} \\ 0 \text{ sinon} \end{cases}$$

On peut résumer la transformation en 3 étapes :

— Changer la coordonnée à l'interface en son complément (0 en 1, 1 en 0)
— Multiplier ξ_1 par la matrice constituée des vecteurs de la base 1 dans la base i
— Ajouter 1 aux coordonnées si il y a un changement de sens pour un vecteur (-1 dans une ligne de la matrice de passage)

Nous allons illustrer ces étapes par un exemple concret en 3D, cf fig. 15 :

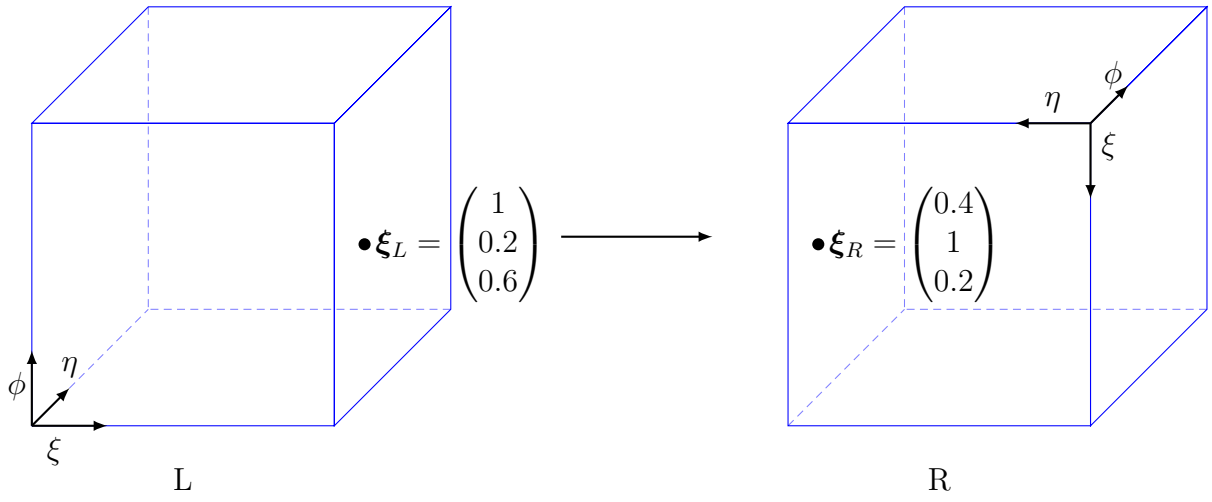

FIGURE 15 – Exemple où la particule passe de la cellule L à la cellule R.

La particule se trouve dans la cellule L et elle instersecte une face de cette cellule en $\begin{pmatrix} 1 \\ 0.2 \\ 0.6 \end{pmatrix}$.

— La particule se trouve dans le plan $\xi = 1$, on la place donc dans le plan $\xi = 0$ (cf figure 14)

$$\underbrace{\begin{pmatrix} 1 \\ 0.2 \\ 0.6 \end{pmatrix}}_{\boldsymbol{\xi}_L} + \underbrace{\begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix}}_{\boldsymbol{B}} = \begin{pmatrix} 0 \\ 0.2 \\ 0.6 \end{pmatrix}$$

— On multiplie par la matrice de passage

$$\underbrace{\begin{pmatrix} 0 & 0 & -1 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}}_{\boldsymbol{P}_R^L} \begin{pmatrix} 0 \\ 0.2 \\ 0.6 \end{pmatrix} = \begin{pmatrix} -0.6 \\ 0 \\ 0.2 \end{pmatrix}$$

— De la matrice de passage peut se déduire $\boldsymbol{A}$

$$\begin{pmatrix} -0.6 \\ 0 \\ 0.2 \end{pmatrix} + \underbrace{\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}}_{\boldsymbol{A}} = \underbrace{\begin{pmatrix} 0.4 \\ 1 \\ 0.2 \end{pmatrix}}_{\boldsymbol{\xi}_R}$$

5.2 Cas des éléments courbes

5.2.1 Initialisation dans l'espace physique

Comme expliqué dans 5.1.1, il est nécessaire de déterminer la cellule contenant une particule donnée à l'initialisation afin de pouvoir déterminer ses coordonnées isoparamétriques par une approche itérative. Déterminer si un point contenu dans un volume donné est plus difficile en présence de faces courbes.

Cas des éléments quadratiques 2D

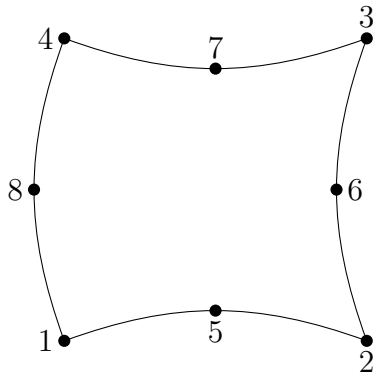

FIGURE 16 – Un élément courbe quadratique 2D et sa numérotation selon la convention gmsh (et donc JAGUAR)

Nous allons procéder en trois étapes :

— Définir l'équation de la courbe (2D) ou de la surface (3D) contenant la face par interpolation
— Réaliser la projection orthogonale de la particule sur chacune des faces, notée $\boldsymbol{p}_i$
— Dans la lignée de 5.1.1, tester le signe d'un produit scalaire : $\langle \boldsymbol{p}_i - \boldsymbol{X}, \boldsymbol{n}_i \rangle$

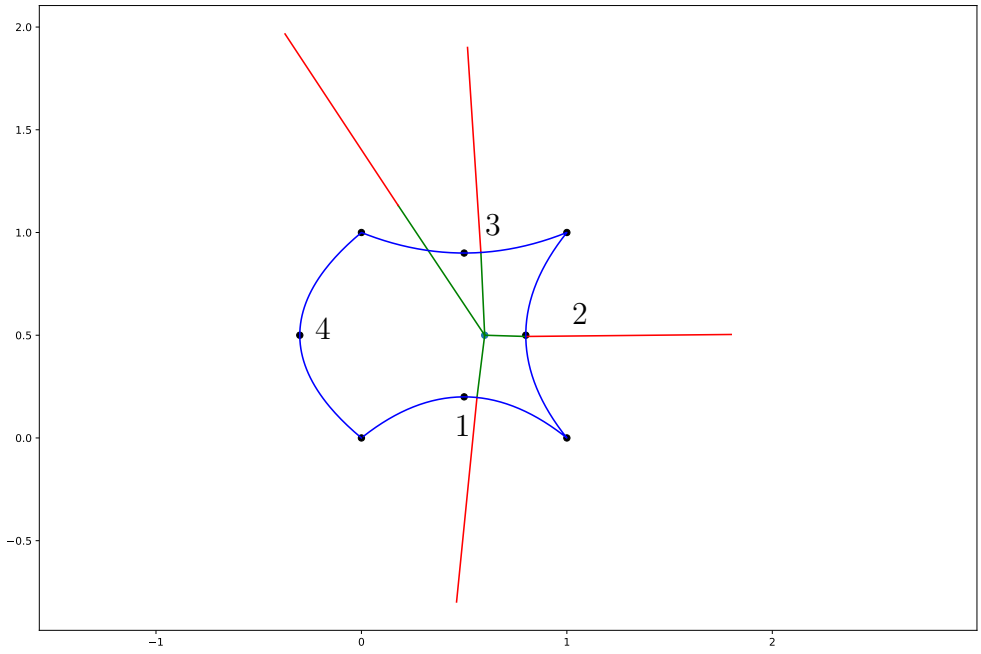

FIGURE 17 – Algorithme (codé en Python) de test de présence dans un élément courbe. Il y a projection même si l'intersection se situe en dehors de la face, par exemple sur la face 4.

Interpolation
Les polynômes de Lagrange seront utilisés, on rappelle la forme générale :

$$L(t) = \sum_{j=0}^{n} X_j(t) l_j(t) \tag{53}$$

$$l_j(t) = \prod_{\substack{i=1 \\ j \neq i}}^{n} \frac{t - t_i}{t_j - t_i} \tag{54}$$

Cette méthode suppose de connaître les t_j Cela signifie que le paramètre t peut prendre trois valeurs aux nœuds : t=0.0, t=0.5 ou t=1.0.
On a donc les équations :

$$\begin{cases} l_1(t) = 2(t - 0.5)(t - 1.0) \\ l_2(t) = -4t(t - 1.0) \\ l_3(t) = 2t(t - 0.5) \end{cases}$$

Projection L'idée est de minimiser le produit scalaire entre la projection et la tangente, car par définition $\langle (f(P) - p, \nabla f(P) \rangle = 0$, comme illustré fig. 5.2.1.

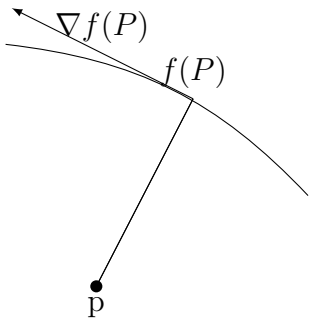

FIGURE 18 – Par définition de la projection orthogonale, la tangente portée par $\nabla f(P)$ doit être orthogonale à $f(P) - p$

Algorithm 5: Algorithme de projection orthogonale

Output: $t_0 \in [0,1]$, $\boldsymbol{X}_p$ la position de la particule, f la fonction interpolée, $0 < \alpha < 1$

Input: $\boldsymbol{P}$ le projeté orthogonal

```
while |d| > ε do
    d = ⟨f(t_0) − X_p, ∇f(t_0)⟩
    t_0 = t_0 − αd
end
P = f(t_0)
```

Cas des éléments quadratiques 3D L'algorithme précédent peut être adapté au cas 3D, en utilisant les surfaces paramétriques. Cependant actuellement, la plupart des logiciels de visualisation comme Ensight ne proposent pas encore de pouvoir afficher des éléments courbes, ce qui rend la validation délicate.

5.2.2 Utilisation de l'espace isoparamétrique appliqué aux éléments courbes

L'intérêt de la méthode des transformées isoparamétriques est que seul le calcul des jacobiennes doit être modifié, l'algorithme principal reste le même, on se ramène à des cubes. Cependant comme expliqué dans la sous-partie 5.2.1, il est difficile de vérifier l'efficacité de l'algorithme.

6 Etude sur l'erreur des coordonnées isoparamétriques de la particule et corrections

6.1 Etude théorique de l'erreur

On rappelle l'équation (19) :

$$\frac{d\boldsymbol{x}(t)}{dt} = \boldsymbol{u_p}(\boldsymbol{x}(t)) \tag{55}$$

En introduisant les déplacements infinitésimaux isoparamétriques :

$$\underbrace{\frac{d\boldsymbol{x}}{d\boldsymbol{\xi}}(\boldsymbol{\xi}(t))}_{\mathcal{J}}\frac{d\boldsymbol{\xi}}{dt}(t) = \boldsymbol{u_p}(\boldsymbol{x}(t)) \tag{56}$$

$$\frac{d\boldsymbol{\xi}(t)}{dt} = \mathcal{J}^{-1}(\boldsymbol{x}(\boldsymbol{\xi}(t)))\boldsymbol{u}_p(\boldsymbol{x}(t)) \tag{57}$$

On peut alors écrire un déplacement isoparamétrique $\Delta\boldsymbol{\xi}$ entre deux instants t_0 et t_1 sous la forme :

$$\Delta\boldsymbol{\xi} = \int_{\boldsymbol{\xi}(t_0)}^{\boldsymbol{\xi}(t_1)} \mathrm{d}\boldsymbol{\xi}(t) = \int_{t_0}^{t_1} \mathcal{J}^{-1}(\boldsymbol{x}(\boldsymbol{\xi}(t)))\boldsymbol{u}_p(\boldsymbol{x}(t))\,\mathrm{d}t \tag{58}$$

La jacobienne de l'équation (58) est difficile à calculer car on ne traite pas des champs continus mais des particules discrètes. Ainsi on utilise des schémas d'intégration supposant une vitesse constante pour la durée d'intégration Δt. La jacobienne étant fonction de l'espace, la supposer constante induit une erreur que l'on cherchera à quantifier.

En posant $\boldsymbol{\xi}_0 = \boldsymbol{\xi}(t_0)$, $\boldsymbol{x}_0 = \boldsymbol{x}(\boldsymbol{\xi}_0)$ et $\Delta\boldsymbol{x} = \boldsymbol{x}_0 - \boldsymbol{x}$ et à l'aide de la formule de Taylor-Young à l'ordre 1 on peut réaliser les approximations suivantes :

$$\boldsymbol{u}_p(\boldsymbol{x}(t)) = \boldsymbol{u}_p(\boldsymbol{x}_0) + \underbrace{\frac{\partial \boldsymbol{u}_p}{\partial \boldsymbol{x}}}_{\mathcal{J}_{u_p}}(\boldsymbol{x}_0)\Delta\boldsymbol{x} + o_{\boldsymbol{x}_0}(\Delta\boldsymbol{x}) \approx \boldsymbol{u}_p(\boldsymbol{x}_0) \tag{59}$$

$$\mathcal{J}^{-1}(\boldsymbol{x}(\boldsymbol{\xi}(t)) = \mathcal{J}^{-1}(\boldsymbol{x}_0) + \frac{\partial \mathcal{J}^{-1}}{\partial \boldsymbol{x}}(\boldsymbol{x}_0)\Delta\boldsymbol{x} + o_{\boldsymbol{x}_0}(\Delta\boldsymbol{x}) \approx \mathcal{J}^{-1}(\boldsymbol{x}_0) = (\mathcal{J}(\boldsymbol{\xi}_0))^{-1} \tag{60}$$

avec le produit tensoriel :

$$\frac{\partial \mathcal{J}^{-1}}{\partial \boldsymbol{x}}\Delta\boldsymbol{x} = (\mathcal{H}_x\Delta\boldsymbol{x}, \mathcal{H}_y\Delta\boldsymbol{x}, \mathcal{H}_z\Delta\boldsymbol{x}) \tag{61}$$

x, y et z sont ici les fonctions définies en (20). $\mathcal{H}_x$ est la hessienne définie par :

$$\mathcal{H}_x = \begin{pmatrix} \dfrac{\partial^2 x}{\partial \xi^2} & \dfrac{\partial^2 x}{\partial \xi \partial \eta} & \dfrac{\partial^2 x}{\partial \xi \partial \psi} \\ \dfrac{\partial^2 x}{\partial \eta \partial \psi} & \dfrac{\partial^2 x}{\partial^2 \eta} & \dfrac{\partial^2 x}{\partial \eta \partial \psi} \\ \dfrac{\partial^2 x}{\partial \psi \partial \xi} & \dfrac{\partial^2 x}{\partial \psi \partial \eta} & \dfrac{\partial^2 x}{\partial \psi^2} \end{pmatrix} \tag{62}$$

En pratique, dans JAGUAR, on calculera $\mathcal{J}(\boldsymbol{\xi}_0)$ et on l'inversera, car on ne sait pas calculer directement son inverse (cf section 4.2).
Les deux erreurs, c'est-à-dire l'approximation par une jacobienne constante le long de la trajaectoire entre les instants t_0 et t_1 et l'approximation par une vitesse constante entre ces mêmes instants peuvent donc être quantifiés par un développement de Taylor. L'approximation sur la vitesse ne sera pas étudiée en détail ici. On se contentera de remarquer qu'on utilisant des schémas explicites de type Runge-Kutta, les ordres de précision temporelle souhaitée peuvent être atteints.
Nous considérerons alors que la mise à jour de la position particulaire se résume à :

$$\Delta\boldsymbol{\xi} \approx \mathcal{J}^{-1}(\boldsymbol{\xi}_0)\underbrace{\boldsymbol{u}_p(\boldsymbol{x}_0)\Delta t}_{\approx\Delta\boldsymbol{x}} \tag{63}$$

La différence entre ces deux termes représente l'erreur de l'approximation :

$$e_{\Delta\boldsymbol{\xi}} = \| \underbrace{\Delta\boldsymbol{\xi}}_{\boldsymbol{\xi}-\boldsymbol{\xi}_0} - \mathcal{J}^{-1}(\boldsymbol{x}_0)\Delta\boldsymbol{x}\| \tag{64}$$

On utilise une nouvelle fois une formule de Taylor-Young, à l'ordre 2 cette fois :

$$\boldsymbol{\xi}(\boldsymbol{x}(t)) = \boldsymbol{\xi}_0 + \mathcal{J}^{-1}(\boldsymbol{x}_0)\Delta\boldsymbol{x} + \frac{1}{2}\Delta\boldsymbol{x}^T\underline{\underline{\mathcal{H}}}^{-1}(\boldsymbol{\xi}_0)\Delta\boldsymbol{x} + o_{\boldsymbol{x}_0}(\|\Delta\boldsymbol{x}\|^2) \tag{65}$$

Avec $\underline{\underline{\mathcal{H}}}$ un tenseur d'ordre 3, tel que :

$$\underline{\underline{\mathcal{H}}} = (\mathcal{H}_x, \mathcal{H}_y, \mathcal{H}_z)$$

De (66) et de (63) on tire :

$$e_{\Delta\xi} = \frac{\|\Delta\boldsymbol{x}\|^2}{2}\|\mathcal{H}^{-1}(\boldsymbol{\xi}_0)\| + o_{\boldsymbol{x}_0}(\|\Delta\boldsymbol{x}\|^2)\| \tag{66}$$

$e_{\Delta\xi}$ dépend donc logiquement de la déformation du maillage par rapport à un maillage cartésien. Il n'y aura pas d'erreur sur un maillage cartésien alors qu'on pourra s'attendre à une erreur non négligeable sur des quadrangles d'angles éloignés de $\frac{\pi}{2}$.
On pourrait se servir directement du terme $\frac{1}{2}\Delta\boldsymbol{x}^T\underline{\underline{\mathcal{H}}}^{-1}(\boldsymbol{\xi}_0)\Delta\boldsymbol{x}$ pour gagner en précision mais le coût mémoire ($\underline{\underline{\mathcal{H}}}$ contient 27 coefficients) comme le coût temporel seraint trop importants. On peut penser également à approcher ce terme en différenciant les 3 jacobiennes $(\mathcal{J}_x, \mathcal{J}_y, \mathcal{J}_z)$. On se contentera donc de corriger cette erreur avec un algorithme de Newton, comme expliqué dans 6.2. Cela est également la raison pour laquelle l'étude n'a pas été plus poussée, car elle ne reflète pas l'erreur finale qui est juste celle d'un algorithme de Newton, et qui peut-être raffinée par le nombre d'itérations.

6.2 Erreurs dans la localisation et corrections

Les coordonnées utilisées dans le calcul de la vitesse des particules sont dans l'espace physique, elles ne sont pas affectées par le calcul de la localisation (sauf si la particule est localisée dans une mauvaise cellule). Un calcul erroné sur les coordonnées isoparamétriques risque d'avoir deux conséquences :

— risquer de trouver une mauvaise cellule d'arrivée, comme dans la figure 19

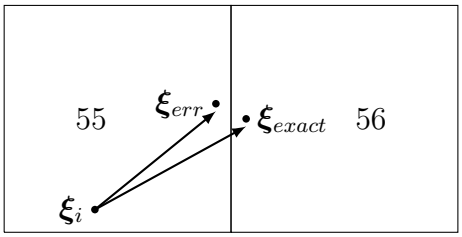

FIGURE 19 – La particule de coordonnées de départ $\boldsymbol{\xi}_i$ dont les coordonnées du point d'arrivée sont situées $\boldsymbol{\xi}_{exact}$ dans la cellule 56 risque d'avoir des coordonnées calculées $\boldsymbol{\xi}_{err}$ et d'être localisée dans la cellule 55

On détectera l'erreur en utilisant l'algorithme d'initialisation qui a été défini en 5.1.1. Si on constate que la particule n'est toujours pas dans la bonne cellule on effectuera une nouvelle itération de l'algorithme de localisation, comme dans la figure 20.

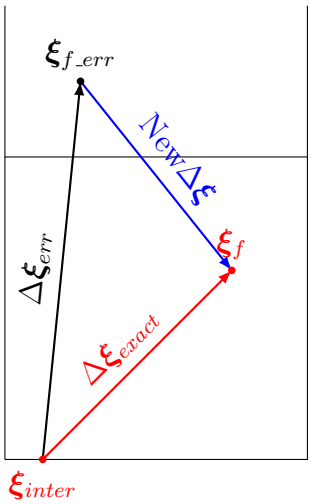

FIGURE 20 – Itération supplémentaire pour corriger l'erreur de localisation

— être dans la bonne cellule, mais risquer d'avoir une forte erreur pour les coordonnées isoparamétriques, ce qui pourrait entraîner des erreurs de localisation ou une divergence si les erreurs se cumulent.

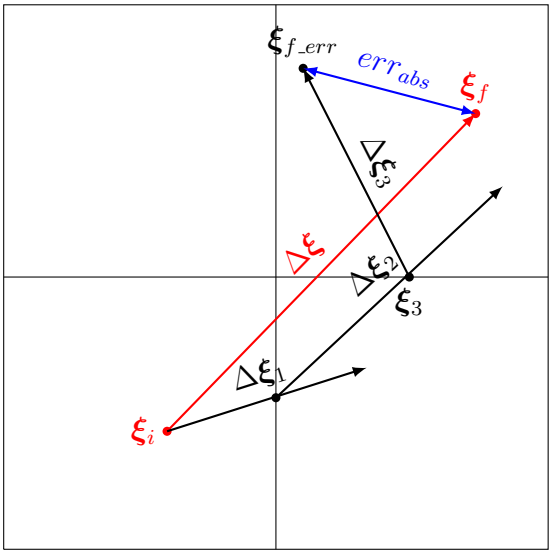

FIGURE 21 – Exemple de localisation en 3 étapes avec une erreur sur $\Delta\boldsymbol{\xi}_j$ volontairement très grande. On comprend que l'erreur qu'il est important de quantifier est celle sur $\Delta\boldsymbol{\xi}_3$, c'est elle qui impactera $\boldsymbol{\xi}_{f_{err}}$

.

Sur cette localisation figure 21 en 3 étapes, on s'aperçoit que seule la dernière itération est source d'erreur. On pourra la corriger en partie à l'aide d'un algorithme de Newton semblable à l'algorithme 4, si err_{abs} est trop grand.

7 Tests et résultats

7.1 Erreurs testées

Dans JAGUAR a été implémenté un mode de débug, permettant à chaque itération pour chaque particule de faire 3 vérifications :

— Vérifier que lors d'une sortie de cellule les coordonnées physiques dans la cellule L soient les mêmes que dans la cellule R (voir figure 14), c'est-à-dire si le changement d'orientation entre deux cellules a été correctement effectué.
— Vérifier qu'une face de sortie a été trouvée quand une condition de sortie a été détectée, et qu'on finisse bien par localiser (trouver la cellule correspondante) la particule.
— Calculer l'erreur entre $\boldsymbol{X}(t+dt)$ et les coordonnées physiques calculées à partir de $\boldsymbol{\xi}_f(t+dt)$, avant et après application d'un algorithme de Newton. Le seuil d'erreur est choisie selon la précision souhaitée.

7.2 Tests sur différents maillages

L'objectif étant principalement de valider la localisation on utilisera un vecteur déplacement constant. Dans le cas de maillages non structurés comme figure 27 il n'y a pas d'intérêt à imposer des déplacements plus compliqués.
Les premiers tests ont été effectué sur le maillage présenté figure 25. C'est un maillage cartésien basique, les tests ne permettront pas de vérifier les calculs de jacobiennes et les transformations isoparamétriques, car les éléments sont déjà des cubes, il y aura juste une homothétie. Néanmoins les tests dans ce maillage permettent de vérifier les tests de sortie et le calcul de l'intersection entre la face et la trajectoire de la particule (cf 5.1.2).
Pour une validation plus complète il est nécessaire de passer à des maillages non cartésiens. On utilisera le même volume, un pavé extrudé d'un cylindre, d'abord avec un maillage structuré (celui figure 26) puis non structuré (figure 27). Le maillage 26 est presque cartésien, la déformation des éléments dans l'espace isoparamétrique sera faible donc il y aura peu d'erreurs sur les jacobiennes. Les tests sur le maillage figure 27 seront donc les plus complets et les plus exigeants.

On générera aléatoirement les positions de centaines de particules. Une visualisation sera possible avec le logiciel Ensight comme figure 22. On affichera le maillage et les particules avec le numéro de chaque cellule et le numéro calculé de la cellule associée à chaque particule, ils doivent bien sûr coïncider.

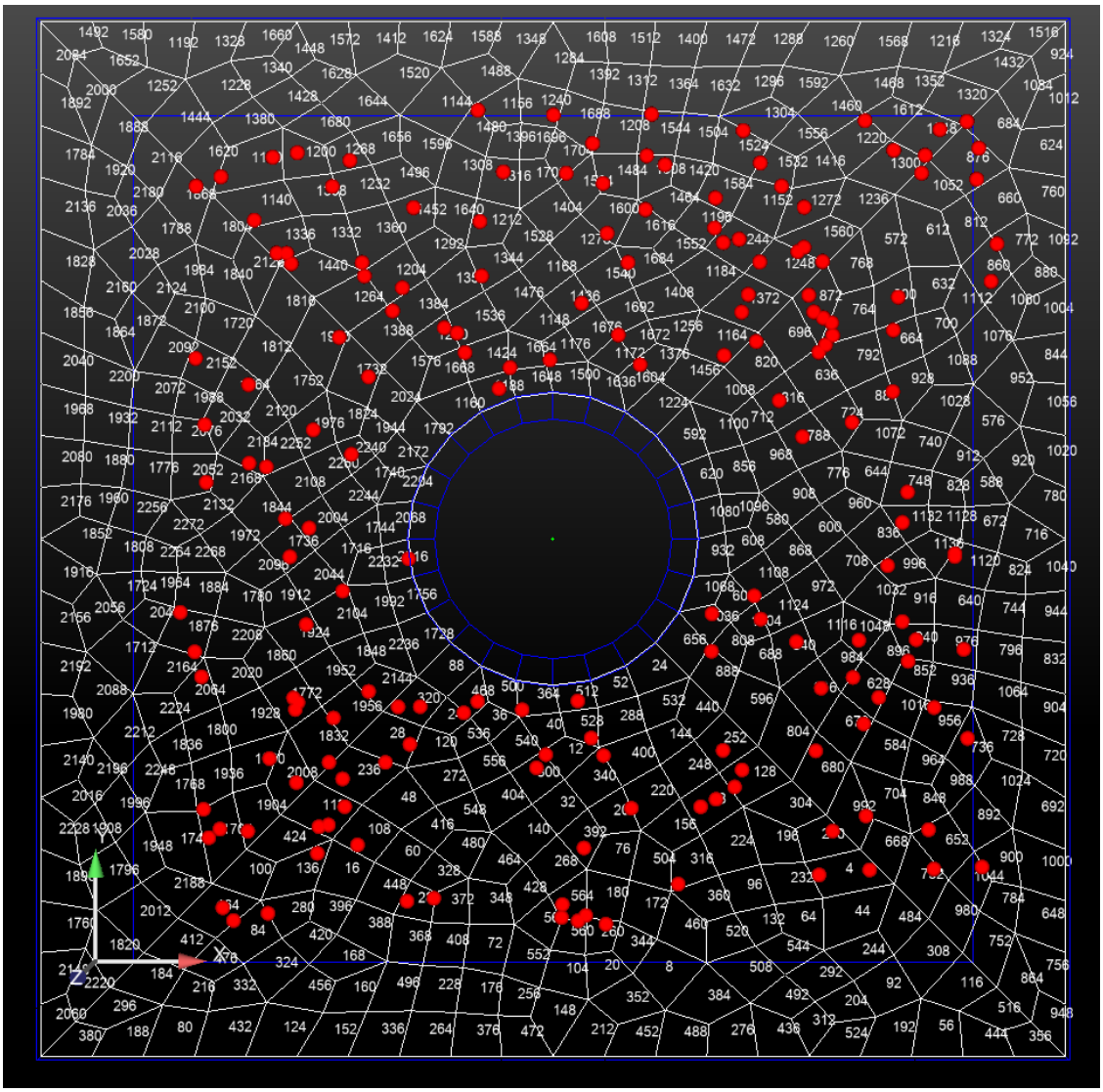

FIGURE 22 – Visualisation avec Ensight d'un maillage non structuré avec des particules aux positions générées aléatoirement.

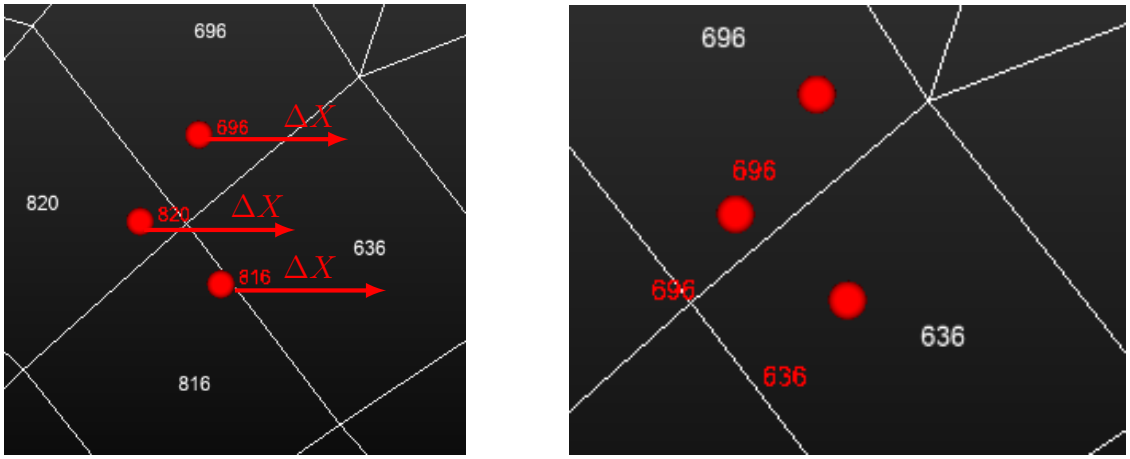

FIGURE 23 – Localisation de particules à t à gauche et $t + \Delta t$ à droite avec le même déplacement $\Delta \boldsymbol{X}$. Le numéro blanc est celui de la cellule, le numéro rouge donne le numéro de cellule associée à la particule. Ils doivent coïncider.

7.3 Tests spécifiques

Certaines situations peuvent mettre en défaut l'algorithme, par exemple à cause des erreurs machines. Certains tests spécifiques ont été éprouvés, tels que :

- longer une arête
- passer par un sommet d'une cellule : l'algorithme détectera une sortie, mais il est possible de ne pas trouver de face de sortie à cause de l'arithmétique flottante, c'est-à-dire que l'intersection ne sera dans aucune face, aucun carré $[0,1]^2$.
  Pour régler ce problème, on effectue le test d'appartenance à une face sur $[-\epsilon, 1+\epsilon]$
- se déplacer en une itération temporelle de 4 ou 5 cellules

Ces tests seront plus facilement implémentables sur des maillages cartésiens.

7.4 Performances

Pour avoir une idée de l'efficacité de l'algorithme, il aurait fallu le comparer avec un algorithme "classique", c'est-à-dire n'utilisant pas les éléments isoparamétriques , comme l'algorithme d'Haselbacher. Son implémentation dans JAGUAR aurait demandé un temps conséquent et n'a donc pas été réalisée. L'idée était de compenser le coût des changements d'espaces (calcul de jacobiennes...) par le gain apporté par des calculs plus simples dans l'espace isoparamétrique . On peut dénombrer trois critères principaux de performance :

- le coût temporel ou le nombre d'opérations, qui n'a donc pas été directement étudié
- la précision des coordonnées isoparamétriques , étudiée dans 6.2 : la correction par l'algorithme de Newton permet d'assurer dans une très grande majorité de cas la précision souhaitée, même si cela alourdit le calcul
- la fiabilité de la localisation : on peut vérifier la localisation avec un algorithme de type Haselbacher utilisé 3.3.2

Sur les maillages de test (cf 7.2), sur plusieurs jeux de 1000 particules aux positions générées aléatoirement, aucune erreur (cf 7.1) n'a été détectée en utilisant la version finale de l'algorithme.

8 Conclusion et perspectives

Le repérage particulaire consiste à déterminer la cellule d'arrivée d'une particule connaissant sa position et sa cellule initiale ainsi que son vecteur déplacement. L'approche de type différences spectrales nécessite la transformation des cellules du maillage de calcul vers un élément de référence dit isoparamétrique afin de pouvoir effectuer certaines opérations de calcul. Dans ce contexte, il semblait pertinent d'examiner la possibilité d'effectuer le repérage particulaire dans l'espace isoparamétrique et non dans l'espace physique. Un grand avantage d'une telle approche semble être sa capacité à gérer des maillages complexes, notamment avec des éléments à faces courbes, via la transformation du vecteur déplacement dans l'espace isoparamétrique. Il a été montré dans le cadre de ce stage qu'il était possible d'écrire un tel algorithme. Seule l'étape d'initialisation nécessite alors la détermination des coordonnées isoparamétriques d'une particule à partir de ses coordonnées physiques. Cette opération semble la plus coûteuse dans le contexte des transformations isoparamétriques puisque basée une méthode itérative. Lorsque le vecteur déplacement implique une sortie de la cellule contenant la particule donnée par une face donnée, il est nécessaire de s'appuyer sur des matrices de passage puisque les systèmes de coordonnées isoparamétriques sont définis implicitement par la numérotation du maillage. L'algorithme a pu être implémenté avec succès dans le code JAGUAR. Il a

également pu être validé sur des maillages non structurés constituées d'hexaèdres, le seul type d'élément que ce code puisse gérer à ce jour. Par ailleurs, des tests préliminaires semblent indiquer que l'algorithme proposé permet une gestion naturelle du repérage sur éléments hexaédriques d'ordre 2, c'est-à-dire avec des faces courbes représentées par des polynômes quadratiques. L'adaptation à d'autres types d'éléments ne semble pas poser de difficulté *a priori*, mais cet aspect devra être examiné plus en détail lorsque la gestion d'éléments tétraédriques sera effective dans le code JAGUAR.

Naturellement, des pistes d'améliorations ont pu être identifiées. L'algorithme d'initialisation permettant de déterminer la cellule contenant une particule et les coordonnées isoparamétriques de la particule dans cette cellule connaissant ses coordonnées physiques utilisé dans le cadre des présents travaux est peu optimal. En effet, son coût moyen est de $\mathcal{O}\left(\frac{n_p n_c}{2}\right)$, avec n_p et n_c respectivement le nombre de particules et le nombre de cellules du maillage. En effet, pour chaque particule on parcourt en moyenne la moitié des cellules avant de trouver celle contenant la particule. Cette optimisation était loin d'être prioritaire compte tenu de l'objectif du stage, mais elle le deviendra si des simulations réalistes doivent être effectuées un jour. Une idée pourrait consister à découper le maillage en octrees. La version développée est séquentielle, mais on pourrait la paralléliser, et pas seulement pour l'initialisation. Pour N_{Proc} processeurs, on peut aussi affecter un processeur pour $\lfloor \frac{n_p}{N_{Proc}} \rfloor$ particules.

L'approche proposée est entâchée d'une erreur géométrique supplémentaire liée à l'utilisation d'une jacobienne de transformation constante dans chaque cellule. L'erreur est proportionelle à la hessienne de la transformation et donc du caractère non-linéaire de la transformation.

De plus, le cas des éléments courbes a finalement été peu étudié puisqu'il ne semble pas que ces maillages puissent être visualisés par les logiciels communément utilisés en mécanique des fluides numériques.

Par ailleurs, la partie physique qui concerne le solveur lagrangien et la simulation d'écoulements diphasiques dispersés reste à finaliser. Pour que le code JAGUAR puisse réellement simuler un écoulement diphasique dispersé il faudrait implémenter l'interpolation des propriétés fluides à la position des particules, le calcul du bilan de force approché et la mise à jour de la position et de la vitesse particulaires. Par ailleurs, un algorithme purement séquentiel a été implémenté dans le cadre de ce travail et sa parallélisation devra être effectuée.

Finalement, une comparaison du nombre d'opérations flottantes entre le présent algorithme de repérage et un algorithme standard de type Haselbacher *et al.* [7] ou Chorda *et al.* [8] semble nécessaire afin de conclure sur la pertinence réelle de la méthode proposée. Cependant, il est important de préciser que ces algorithmes ne sont pas tout à fait comparables puisque la méthode proposée devrait permettre une gestion naturelle d'éléments d'ordre élevé, ce qui n'est pas le cas des type Haselbacher *et al.* [7] ou Chorda *et al.* [8].

Il semble également important d'insister sur le caractère novateur de ce travail, puisqu'à la connaissance de l'auteur, aucun algorithme décrivant la gestion du repérage particulaire dans l'espace isoparamétrique ne semble avoir été publié dans la littérature à ce jour.

D'un point de vue personnel, ce stage m'a permis de découvrir des notions mathématiques et physiques intéressantes, et m'a conforté dans l'idée de continuer en thèse. Ce sera d'ailleurs la prochaine aventure à partir d'octobre 2018, et ce stage a été un fabuleux

complément de formation dans cette optique.

Références

[1] Alfred Barnard Basset. *A treatise on hydrodynamics : with numerous examples*, volume 2. Deighton, Bell and Company, 1888.

[2] J Boussinesq. *Theorie analytique de la chaleur*. Gauthier-Villars, Paris, 1903.

[3] Carl Wilhelm Oseen. *Hydrodynamik*, volume 1. Akademische Verlagsgesellschaft, Leipzig, 1927.

[4] MR Maxey and JJ Riley. Equation of motion for a small rigid sphere in a nonuniform flow. *The Physics of Fluids*, 26(4) :883–889, 1983.

[5] Renée Gatignol. The faxén formulas for a rigid particle in an unsteady non-uniform stokes-flow. *Journal de Mécanique théorique et appliquée*, 2(2) :143–160, 1983.

[6] L. Schiller and A. Nauman. A drag coefficient correlation. *Zeitschrift des Vereins Deutscher Ingenieure*, 77 :318–320, 1935.

[7] A Haselbacher, FM Najjar, and JP Ferry. An efficient and robust particle-localization algorithm for unstructured grids. *Journal of Computational Physics*, 225(2) :2198–2213, 2007.

[8] R Chorda, JA Blasco, and N Fueyo. An efficient particle-locating algorithm for application in arbitrary 2d and 3d grids. *International Journal of Multiphase Flow*, 28(9) :1565–1580, 2002.

[9] TJR Hughes. *The finite element method : linear static and dynamic finite element analysis*. Courier Corporation, 2012.

[10] M Abbas. La méthode des éléments finis isoparamétriques. Technical report, 2013.

[11] Antony Jameson. A proof of the stability of the spectral difference method for all orders of accuracy. *Journal of Scientific Computing*, 45(1-3) :348–358, 2010.

Annexes

A Compléments sur les équations d'Euler

Dans les membres de gauche du système (1- 3), le premier terme décrit la variation temporelle de la quantité considérée, le second le transport de cette quantité par l'écoulement fluide ainsi que les contributions des forces de pression. Ce second terme représente les flux "Euler". Les membres de droite des équations (2) et (3) regroupent les effets visqueux. Ces termes étant seulement présents dans les équations de Navier-Stokes, ils sont communément appelés flux "Navier-Stokes". Ce système d'équations comporte plus d'inconnues que d'équations, il doit donc être fermé par des relations supplémentaires.

Tout d'abord une relation permettant d'exprimer le tenseur des contraintes visqueuses en fonction du vecteur vitesse est nécessaire. Pour un fluide dit newtonien, le tenseur des contraintes peut être exprimé en fonction du gradient du vecteur vitesse ainsi que de sa divergence :

$$\underline{\boldsymbol{\tau}} = \mu(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T) - \frac{2}{3}\mu \nabla \cdot (\boldsymbol{u} \cdot \underline{\boldsymbol{I}}) \tag{67}$$

avec μ la viscosité dynamique du fluide et $\underline{\boldsymbol{I}}$ la matrice identité. De plus, une loi d'état est nécessaire afin de relier les variables pression P, température T et densité ρ. La plus simple est la loi des gaz parfaits :

$$p = \frac{\mathcal{R}}{M}\rho T = R\rho T \tag{68}$$

avec $\mathcal{R} = 8.314$ JK^{-1}mol^{-1} la constante universelle des gaz parfaits, M la masse molaire du fluide et R la constante du fluide, qu'on suppose être constitué d'une seule espèce. Cette loi d'état permet notamment d'expliciter l'énergie totale du fluide E, définie comme la somme de l'énergie cinétique e_c et de l'énergie interne e_i :

$$E = E(T) = \underbrace{\frac{\|\boldsymbol{u}\|^2}{2}}_{e_c} + \underbrace{\int_{T_0}^{T} c_v(\theta)\,\mathrm{d}\theta - \frac{RT_0}{M}}_{e_i} \tag{69}$$

où c_v désigne la capacité calorifique à volume constant du fluide, qui ne dépend que de la température pour un gaz caloriquement parfait. L'indice "0" désigne un état thermodynamique caractérisé par une température T_0 et une pression p_0 de référence.

Finalement, le flux de chaleur peut être exprimé en fonction du gradient de température en fonction de la loi de Fourier :

$$\boldsymbol{q} = -\lambda \nabla T \tag{70}$$

avec T la température du fluide et λ la conductivité thermique.

B Maillages

Il y a différents types de maillages, de complexités diverses, et plusieurs sont évoqués dans ce rapport.
Nous les présentons de manière non exhaustive par ordre croissant de complexité.

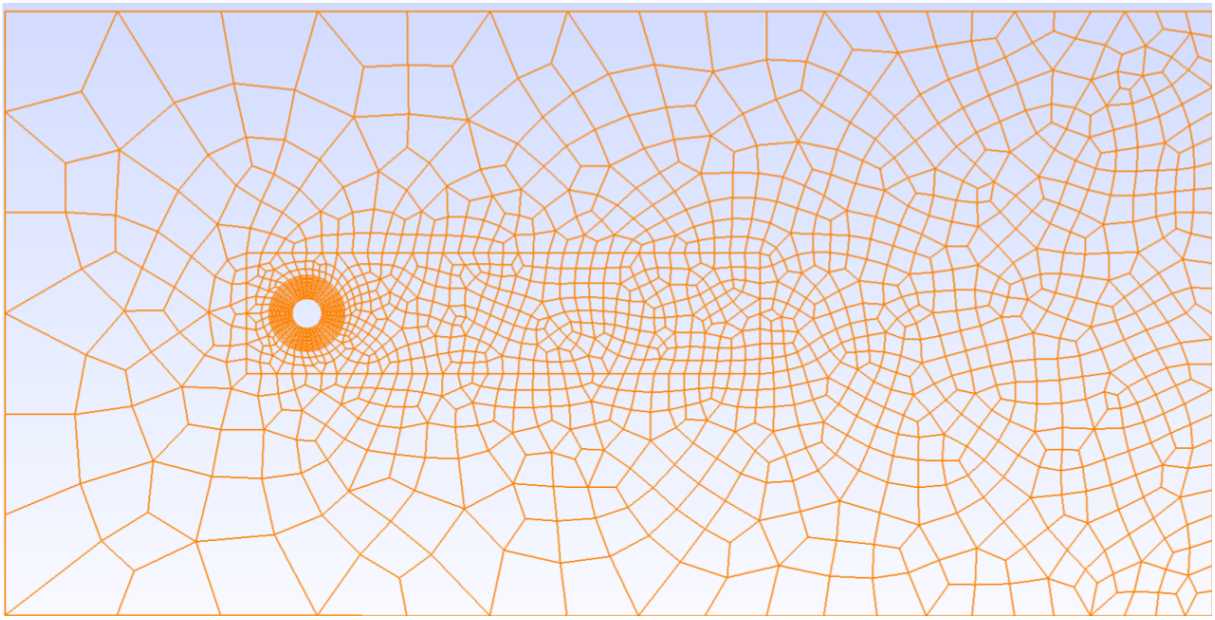

FIGURE 24 – Exemple de maillage pour un cylindre utilisé dans JAGUAR

B.1 Maillages cartésien et structuré

Le maillage cartésien est analogue à un repère orthogonal. Il peut être utilisé pour des calculs d'ordre élevé. Néanmoins, il est assez difficile de mailler des géométries en cartésien. Contrairement aux maillages non structurés, il n'existe à ce jour pas de d'outil numérique permettant de générer un maillage cartésien de façon automatique. Ainsi, la génération de maillages cartésiens pour des géométries complexes peut nécessiter des semaines de travail.

Un maillage structuré n'a pas de condition d'orthogonalité ou de parallélisme, contrairement au maillage cartésien. Néanmoins, dans la littérature, ces deux termes sont souvent confondus, l'opposition principale se trouvant entre maillages non structurés et structurés. Ces derniers possèdent une correspondance directe entre une coordonnée spatiale et la cellule du maillage associé. Ainsi, déterminer un numéro de cellule à partir d'une coordonnée physique donnée est direct, de même que l'accès aux cellules voisines d'une cellule donnée. Dans le cas d'un maillage non structuré, cet accès est indirect et fourni par une table de connectivité. Un exemple de maillage cartésien très simple est fourni fig. 25

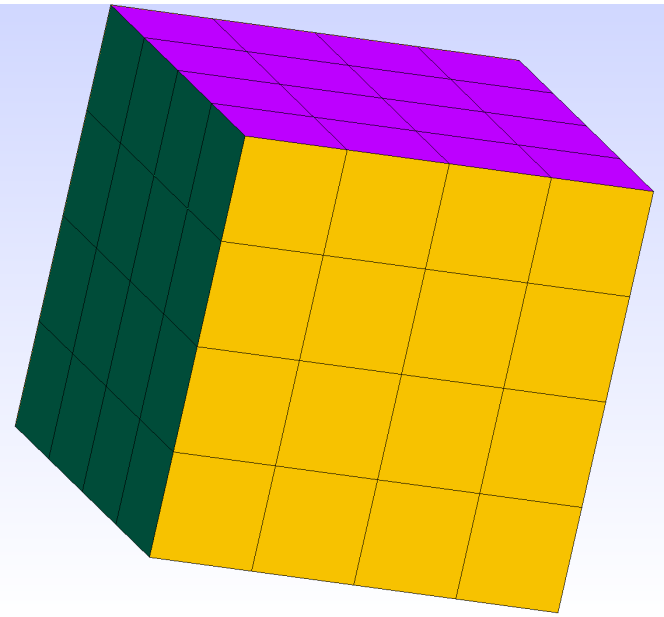

FIGURE 25 – Cube maillé par un maillage cartésien basique

Ce maillage sera utilisé pour les premiers tests de localisation. On fera ensuite les calculs sur le maillage figure 26

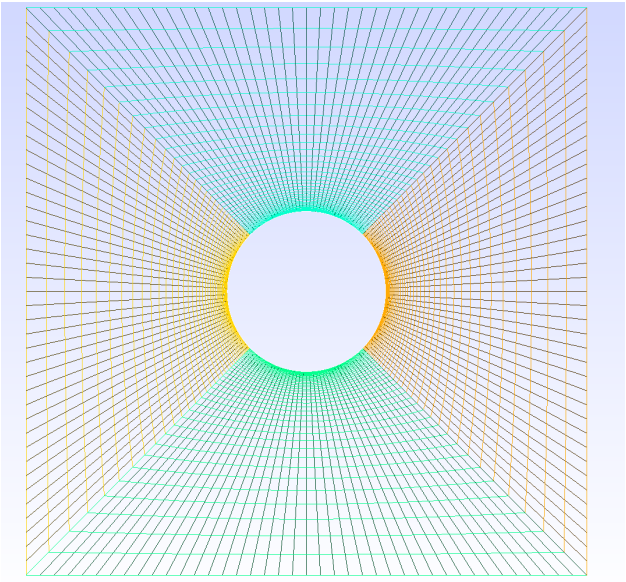

FIGURE 26 – Exemple de maillage structuré non cartésien

B.2 Maillages non structuré

Contrairement au maillage structuré, le maillage non structuré peut être constitué de divers types d'éléments, par exemple des tétraèdres, des prismes et des pyramides. Néanmoins à l'heure où le rapport est écrit JAGUAR ne gère que des hexaèdres. On perd alors en partie l'intérêt des maillages non structurés qui est la facilité à générer des maillages d'une géométrie complexe.

Comme cela a déjà été mentionné ci-dessus, un maillage non structuré doit s'accompagner d'une table de connectivité. De plus il est plus difficile d'y implémenter des schémas

spatiaux d'ordre supérieur à deux puisqu'on ne peut pas recourir à des schémas de types différences finies compacts. JAGUAR utilise donc une description polynomiale locale des variables d'intérêt associée à des transformations isoparamétriques. Ainsi, cette méthode reste locale, c'est à dire qu'elle ne requiert des échanges de données qu'avec les cellules du premier voisinage, ce qui la rend efficace du point de vue de la parallélisation.

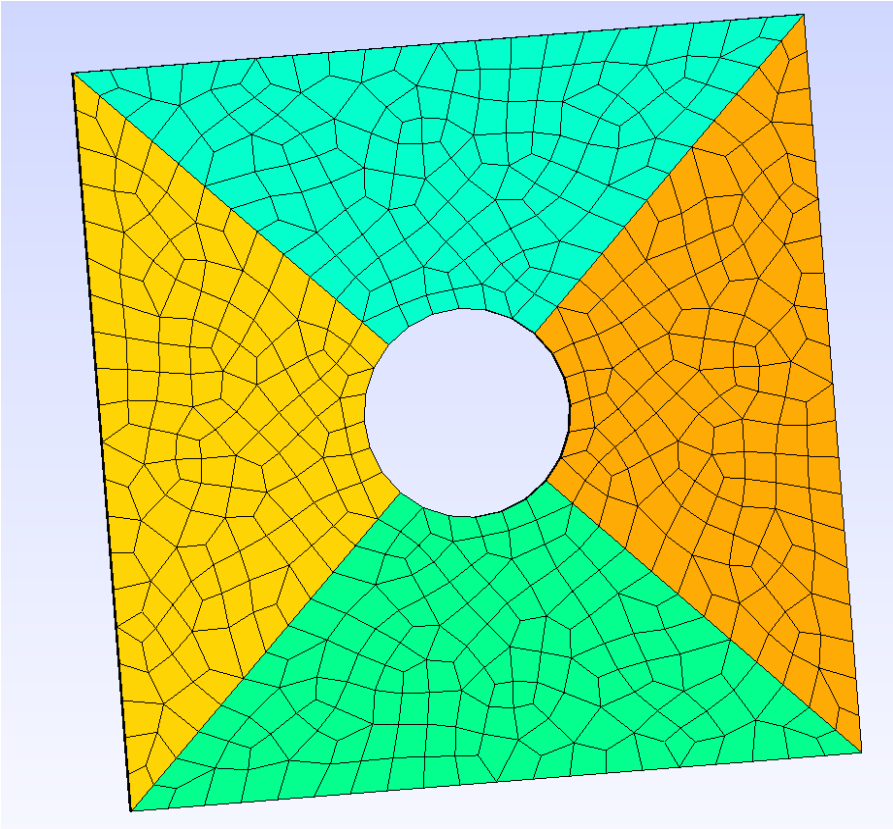

FIGURE 27 – Exemple de maillage non structuré constitué de quadrangles

La figure 27 fournit un exemple de maillage non structuré constitué de quadrangles. Ce maillage sera le plus utilisé pour les tests car il permet d'avoir de nombreuses configurations. C'est-à-dire qu'il permet de tester des nombreuses matrices jacobiennes et une bonne partie des changements d'orientations possibles.

B.3 Maillages multi-éléments

Le maillage multi-élément fait partie des maillages non structurés. Souvent les éléments près des parois sont des prismes afin de permettre une description précise de la couche limite. La plupart des codes CFD actuels utilisent ce type de maillage et JAGUAR est destiné à terme à les gérer. Ce sera alors l'occasion de tests plus complets pour la localisation.

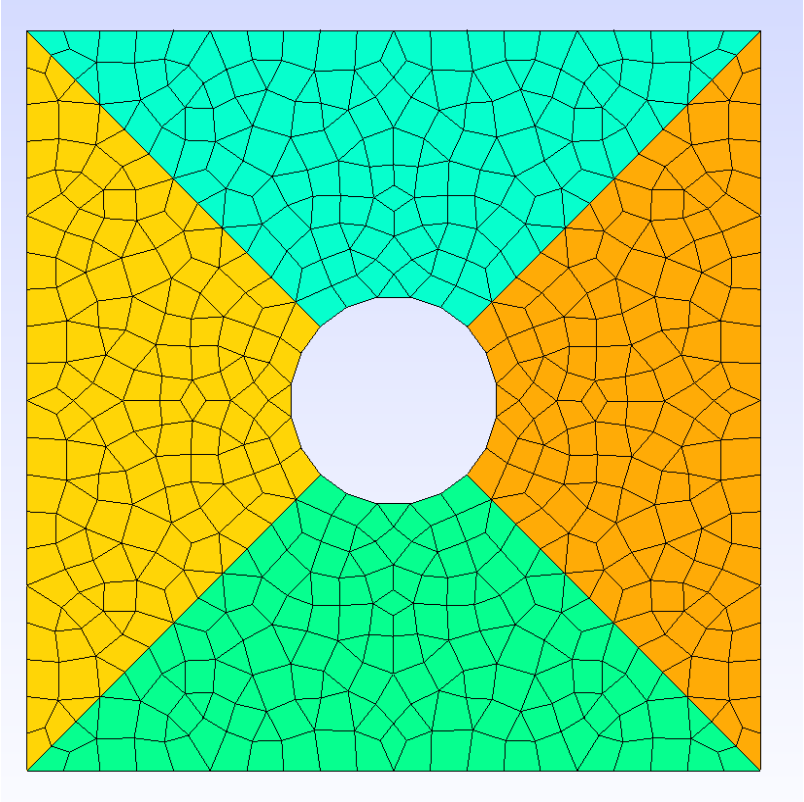

FIGURE 28 – Exemple de maillage multi-éléments

B.4 Maillage à éléments courbes

Ces maillages sont plus complexes mais ils permettent une description précise de la géométrie sous-jacente.

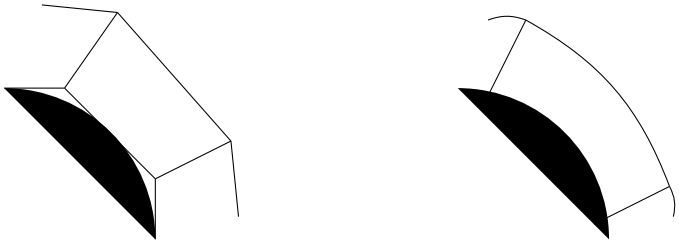

FIGURE 29 – Adaptation d'élements droits à gauche / d'élements courbes à droite à une frontière parabolique

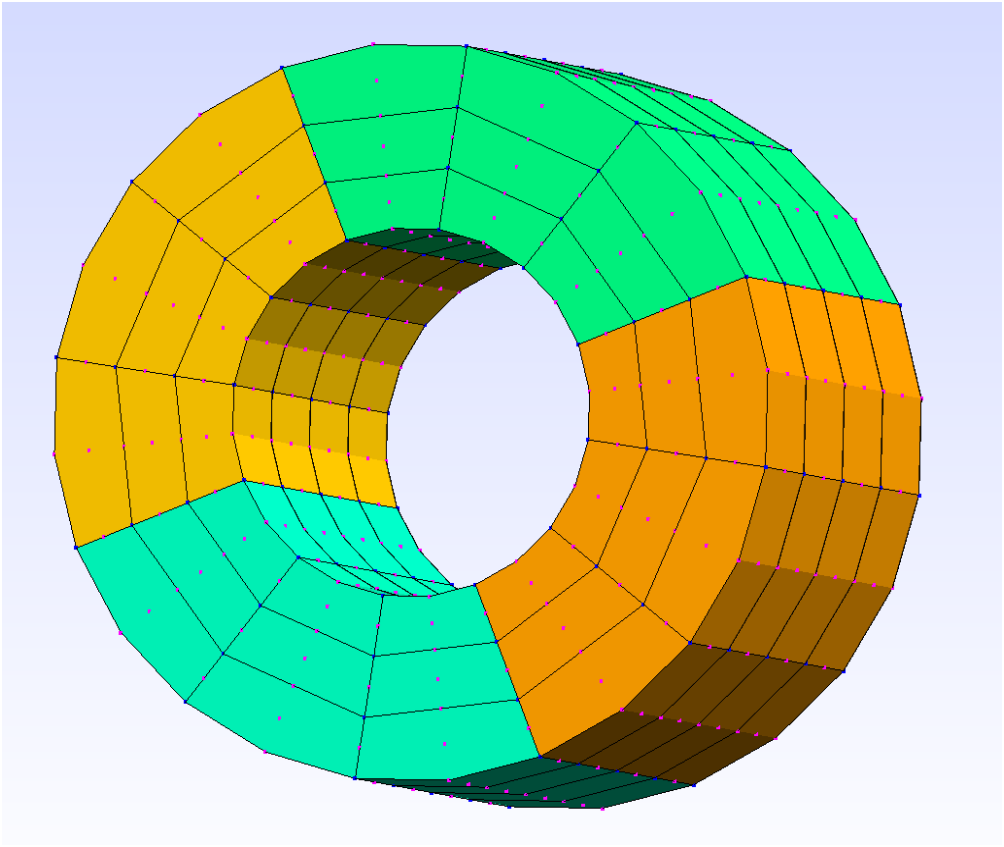

FIGURE 30 – Ce maillage a été utilisé comme maillage test. GMSH représente les maillages à éléments courbes de cette manière, c'est à dire avec neuf points par face et un point au centre de l'élément en 3D, ce qui donne 27 points par élément.

C Résolution numérique : méthode des différences spectrales

Les équations de Navier-Stokes sont non-linéaires et les seules solutions analytiques connues à ce jour de ces équations se limitent à des cas élémentaires. Ainsi, une résolution numérique de ces équations s'avère nécessaire dès qu'on considère des cas d'application pratiques. Les méthodes numériques de résolution des équations de Navier-Stokes sont multiples et ne seront pas détaillées dans le présent rapport. On se contentera de donner un aperçu de l'approche SD (spectral differences) utilisées dans le code JAGUAR.

Comme la plupart des méthodes numériques de résolution des équations de Navier-Stokes, l'approche SD consiste d'abord en une sous-division du domaine de calcul en volumes de contrôle, ce qui correspond en pratique à la génération d'un maillage de calcul.

À ce jour, seuls des volumes de contrôle de type quadrangles ou prismes peuvent être utilisés dans le code JAGUAR. Afin d'augmenter la souplesse de génération du maillage, une extension de la méthode permettant la gestion d'élements triangulaires / tétraédriques est actuellement en cours.

À l'intérieur de chaque volume de contrôle, une représentation polynomiale des variables est utilisée. Cette représentation implique nécessairement des polynômes de degrés différents. En effet, le système des équations de Navier-Stokes tel que synthétisé équation (4) indique que le vecteur des variables conservatives est dérivé par rapport au temps, alors

que les vecteurs flux sont dérivés par rapport à l'espace. Pour des raisons élémentaires de consistance, chaque terme doit être représenté par un polynôme du même ordre. Ainsi, les polynômes représentant les vecteurs flux doivent être d'un ordre supérieur aux polynômes représentant les variables conservatives. En pratique, cela conduit à un stockage colocalisé, c'est-à-dire que les vecteurs vitesse et flux ne sont pas stockés aux mêmes points du maillage. Par ailleurs, la description des flux par un polynôme de degré supérieur implique un point de stockage supplémentaire par direction par rapport aux points solution. Finalement, afin de garantir la stabilité de la méthode, le positionnement des points solution ne peut être choisi arbitrairement. Jameson [11] a prouvé la stabilité de la méthode pour la résolution d'une équation d'advection linéaire mono-dimensionnelle lorsque la position des points flux coïncidait avec les racines du polynôme de Legendre correspondant.

Une illustration des principales étapes de la méthode SD est fournie fig. 31, celles-ci seront détaillées par la suite pour le cas mono-dimensionnel. On considère l'équation :

$$\frac{\partial u}{\partial t} + \frac{\partial F}{\partial x} = 0 \tag{71}$$

Dans chaque cellule i on aura une solution approchée u_i de u sous la forme d'un polynôme de degré p. On peut distinguer 6 étapes illustrées fig. 31 :

- Interpolation des valeurs de u aux p+1 points solutions de coordonnés x_s ▼ : calcul de $u_s = u(x_s)$ ●
- Extrapolation des valeurs de u aux points flux de coordonnées x_f ▼ : on introduit les p+2 points flux et on calcule $u_f = u(x_f)$ ●
- Evaluation du flux aux points flux : on calcule $F(u_f)$ ■
- Utilisation d'un solveur de Riemann permettent d'obtenir un flux continu à l'interface entre deux cellules. On obtient un nouveau flux $\bar{F}$ aux interfaces entre cellules.
- Calcul du résidu : $R_s = \dfrac{\partial F}{\partial x} = \sum_f \bar{F}_f(X_f) l'_f(x_s)$
- Dans le cas de l'utilisation d'un schéma explicite de type Runge-Kutta, le calcul du résidu permet la mise à jour de la solution à la sous-itération suivante $u(t + \Delta t_n)$, avec n l'indice de sous-itération Runge-Kutta.

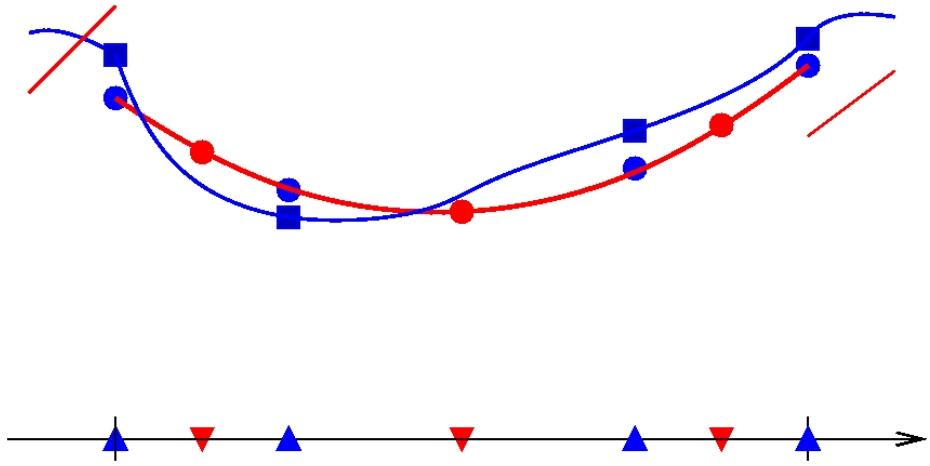

FIGURE 31 – Schéma de principe illustrant les principales étapes d'une approche de type SD. ▼ : point solution, ▲ : point flux, _ : polynôme d'interpolation aux points solution, _ : interpolation aux points flux après résolution de la continuité du flux par le solver de Riemann, ● valeur de la solution calculée aux points solutions, ● valeur de la solution calculée aux points flux, ■ valeur du flux aux points flux. Image tirée de la documentation JAGUAR

La répartition des points solution et flux pour le cas 2D est illustrée dans l'annexe D. Comme son nom l'indique, la méthode SD utilise la méthode des différences pour le calcul des dérivées spatiales. Cette approche s'oppose donc aux méthodes de types éléments finis ou volumes finis, où une intégration sur les volumes de contrôle est effectuée et où les intégrales volumiques des dérivées spatiales sont réexprimées en intégrales surfaciques, soit en vertu du théorème de Green-Gauss (volumes finis), soit en vertu d'une intégration par parties (éléments finis). Afin de pouvoir calculer des dérivées par l'approche des différences finies, mais également afin de permettre le positionnement des points solution et flux pour des quadrangles et prismes déformés, une transformation linéaire vers un élément de référence dit isoparamétrique est nécessaire. Cette transformation isoparamétrique étant un élément fondamental du présent travail, elle sera décrite plus en détail dans 4.

Finalement, la mise à jour temporelle des équations peut s'effectuer par des approches explicites ou des approches implicites. Aujourd'hui, seule une résolution explicite via des méthodes de type Runge-Kutta est disponible dans le code JAGUAR.

D Points flux et points solutions en deux dimensions

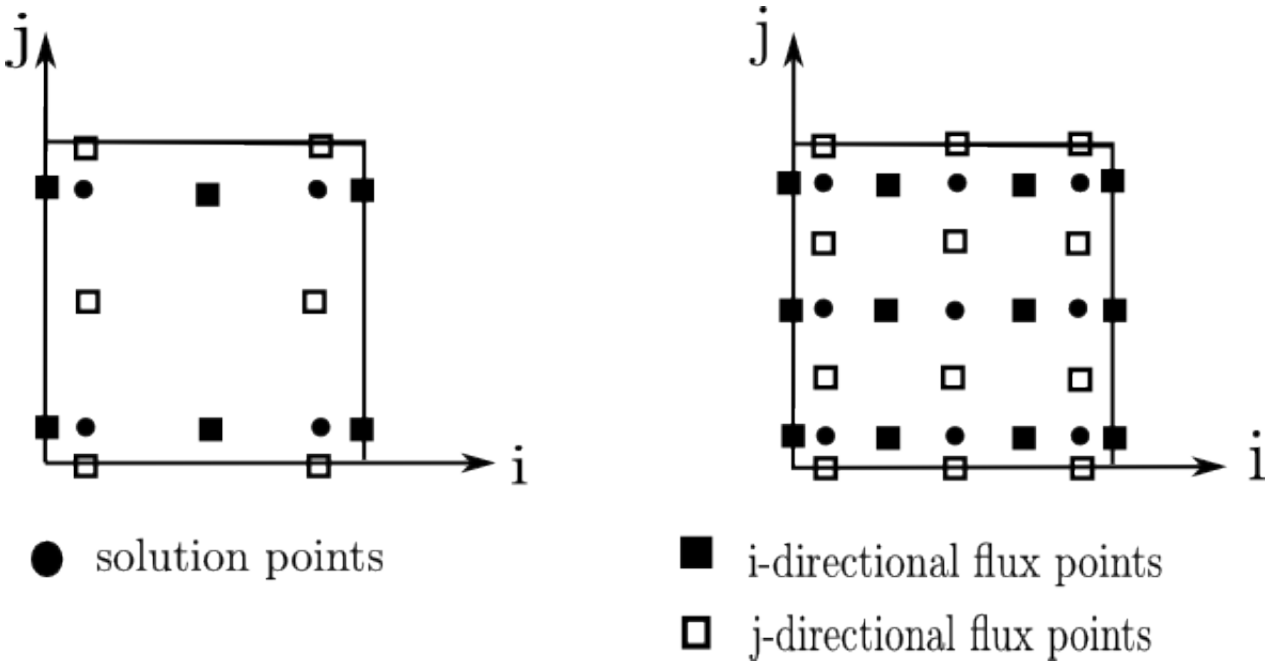

FIGURE 32 – Distribution des points flux et solution. Droite : p=1. Gauche : p=2. Image tirée de la documentation JAGUAR

En 2D, il y aura $(p+1)^2$ points solution et $(p+2)(p+1)$ points flux.
En 3D, il y aura $(p+1)^3$ points solution et $(p+2)(p+1)^2$ points flux.