POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Vérification de code pour le modèle k-ϵ standard de OpenFOAM

SAMUEL GAUDET

Département de génie mécanique

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*

Génie mécanique

Décembre 2019

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Ce mémoire intitulé :

Vérification de code pour le modèle k-ϵ standard de OpenFOAM

présenté par **Samuel GAUDET**

en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*

a été dûment accepté par le jury d'examen constitué de :

Jean-Yves TRÉPANIER, président

Dominique PELLETIER, membre et directeur de recherche

Stéphane ÉTIENNE, membre

DÉDICACE

À ma mère et à mon père

REMERCIEMENTS

Tout d’abord, je tiens à remercier mon directeur de recherche, Dominique Pelletier, pour sa patience, son temps et ses nombreux conseils. Les discussions que nous avons partagées sont d’une valeur inestimable à mes yeux, tant pour mon apprentissage que pour la passion de la CFD qu’il a su me transmettre.

Je souhaite aussi remercier René-Jean Lavallée et Vincent Restieri de Valtech Énergie. Leur confiance et l’intérêt qu’ils portent à ce projet ont été une source de motivation pour moi tout au long de ce travail.

Je remercie Michel et Catherine pour leur présence et leur soutien indéfectible dans les moments les plus difficiles.

Je remercie également ma famille, et surtout mes parents pour toutes ces années à soutenir et investir dans ma réussite.

Finalement, je remercie ma bien-aimée, Shekina, qui a été une complice d’études formidable tout au long de ce travail.

RÉSUMÉ

Dans un contexte industriel toujours plus compétitif, les industries sont constamment comparées les unes aux autres. Elles ont donc intérêt à se moderniser et investir dans des outils de pointe afin d'améliorer la qualité de leur procédé et de diminuer leurs coûts d'opération. Les outils de CFD répondent précisément à ce besoin, ce qui explique la demande grandissante pour ces outils dans l'industrie. Toutefois, la crédibilité des résultats obtenus reste un obstacle majeur pour l'application plus répandue des outils de CFD. Nous avons donc comme but de contribuer à la crédibilité des résultats en confirmant que les méthodes numériques, telles qu'implémentées dans le code, se comportent conformément à la théorie.

Dans le cadre de ce travail, nous avons choisi d'employer le code OpenFOAM, un logiciel à code source libre fréquemment employé dans l'industrie. Il offre une impressionnante collection d'outils entièrement gratuits, répondant en grande partie aux besoins industriels actuels. Cependant, ce logiciel est nettement plus complexe à employer qu'un logiciel de CFD commercial. De plus, on retrouve peu d'informations détaillant la vérification rigoureuse du code. Il nous semble donc approprié de porter plus d'attention à la vérification du code OpenFOAM pour les multiples solveurs, modèles et conditions limites disponibles.

Dans ce mémoire nous effectuerons la vérification du code OpenFOAM pour un écoulement turbulent et incompressible. En effet, ce type d'écoulement est représentatif de la majorité des situations rencontrées dans un contexte industriel. On modélise la turbulence avec le modèle k-ϵ standard à cause de sa simplicité d'utilisation et son faible coût de calcul. Ce modèle est un compromis intéressant pour sa précision et son coût de calcul.

Nous commencerons par vérifier le code pour un écoulement très simple, soit celui de Poiseuille. Nous appliquerons ensuite la méthode de solutions manufacturées imitant un écoulement laminaire 1D de type couche limite, un écoulement libre 2D laminaire, un écoulement libre 2D turbulent, et finalement un écoulement de type couche cisaillée. Afin de mieux comprendre les écarts entre le taux de convergence théorique et le taux de convergence observé, nous poursuivrons la vérification en y ajoutant l'étape de vérification de simulation à l'aide de l'extrapolation de Richardson. Ces cas nous permettront de confirmer si le taux de convergence théorique des schémas numériques est respecté ou non.

ABSTRACT

In an ever more competitive industrial context, factories are regularly compared with each other. Hence, they have every reason to modernize themselves and invest in state-of-the-art tools to improve the quality of their process and reduce their operating costs. CFD tools are tailored for this need. This explains the growing demand for such services in this industry. However, the credibility of predictions remains a major obstacle of a more widespread use of CFD tools. We therefore aim to contribute to the credibility of the results by confirming that numerical methods, as implemented in the code, behave as predicted by theory.

As part of this work, we chose to use the OpenFOAM code, an open-source software widely used in the industry. It offers an impressive collection of tools that meets most of the industry's needs. However, this software is significantly more complex to use than commercial CFD software. Also, there is little information detailing the rigorous verification of its code. Therefore, it seems appropriate for us to pay more attention to the verification of OpenFOAM's code for the numerous solvers, models and boundary conditions that are available.

In this thesis we will perform code verification of a turbulent incompressible flow solver. This flow is quite representative of most flows encountered in the industry. We use the standard k-ϵ model since it is easy to use and for its cheap computational cost. These two qualities make this model the ideal compromise between accuracy and computational costs.

We will start by verifying the code for a very simple Poiseuille flow. We will then apply the method of manufactured solutions to mimic a boundary layer 1D laminar flow, a laminar 2D free flow, a turbulent 2D free flow, and then finally a sheared layered flow. In order to better understand the disagreements between the theoretical and observed convergence rates, we will add a step of verification of simulations to this study, using Richardson's extrapolation. These cases will allow us to confirm whether the theoretical convergence rate of the numerical schemes is achieved or not.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SIGLES ET ABRÉVIATIONS

BNR	Bas nombre de Reynolds
CFD	Computational Fluid Dynamics
CHP	Calcul à haute performance
DNS	Direct Numerical Simulation
ÉDP	Équations aux dérivées partielles
GCI	Grid Convergence Index
GNU	License publique générale
HNR	Haut nombre de Reynolds
HOT	Higher Order Terms
LES	Large Eddy Simulation
MMS	Méthode des solutions manufacturées (Method of Manufactured Solutions)
PISO	Pressure-Implicit Splitting of Operators
RANS	Reynolds Averaged Navier-Stokes
RSM	Reynolds Stress Model
SIMPLE	Semi-Implicit Method for Pressure Linked Equations
SM	Solution manufacturée
V&V	Vérification et validation

Notation générale

$\nabla(\phi)$	Gradient du champ ϕ
$\nabla^2(\phi)$	Laplacien du champ ϕ
$\nabla \cdot \alpha$	Divergence du vecteur α

$O(h^i)$	Taux de convergence *i* pour une grandeur d'élément *h*
$\overrightarrow{AB}$	Vecteur allant du point *A* vers le point *B*
$\bar{x}$	Variable moyenne
x_m	Variable manufacturée

Caractères latins

E	Erreur de la solution
h	Taille caractéristique du maillage
I	Matrice identité
k	Énergie cinétique de la turbulence
l	Longueur de mélange
p	Pression
$\hat{p}$	Taux de convergence
r	Facteur de raffinement de maillage
R	Ratio de raffinement
Re	Nombre de Reynolds
s_{ij}	Tenseur des contraintes
u	Vitesse dans la direction de l'axe des x
u'	Composante fluctuante de la vitesse
U	Composante moyenne de la vitesse
v	Vitesse dans la direction de l'axe des y
y+	Distance adimensionnelle de la paroi

Caractères grecs

δ_{ij}	Symbole de Kronecker
ϵ	Taux de dissipation de l'énergie cinétique turbulente
μ	Viscosité dynamique
v	Viscosité cinématique
v_{eff}	Viscosité efficace
v_T	Viscosité cinématique turbulente
ρ	Masse volumique
τ_{ij}	Tenseur de Reynolds
ω	Taux de dissipation spécifique de la turbulence
U_∞	Vitesse loin de la paroi

LISTE DES ANNEXES

CHAPITRE 1 INTRODUCTION

Les industriels sont confrontés plus que jamais au défi grandissant de réduire leurs coûts énergétiques et d'améliorer leurs procédés. Ce défi est accentué par la compétition toujours grandissante des pays en développement qui ont accès à une main-d'œuvre moins dispendieuse et où les normes environnementales et sociales sont moins sévères. C'est pourquoi l'industrie se tourne de plus en plus vers des outils de pointe, permettant d'optimiser leurs systèmes et leurs procédés afin de rester compétitifs à l'échelle internationale.

Un des outils de plus en plus employés à cette fin est la dynamique des fluides numérique (*Computational Fluid Dynamics*), aussi connu sous l'acronyme CFD. Cet outil décisionnel a la capacité de simuler des écoulements très complexes avec une précision de calcul contrôlée. Les industriels ont donc accès à de l'information qui, auparavant, était impossible ou trop complexe à obtenir par les méthodes de mesures classiques. Cette information est cruciale pour les industriels, car elle permet, entre autres, d'optimiser des systèmes existants, d'améliorer la qualité des procédés, et même d'identifier et de réduire l'occurrence de bris ou d'endommagement des équipements.

L'engouement qui en résulte est non seulement dû à la variété des informations nouvellement disponibles, mais aussi dû à l'accessibilité de ces outils numériques. La puissance informatique requise pour les simulations, comme celles présentées dans ce mémoire, ne requiert pas un investissement exorbitant comme ce fut le cas par le passé. Or, il devient beaucoup plus courant pour des firmes de génie-conseil de se doter de tels équipements et outils de simulation afin de mieux répondre aux besoins de leurs clients. Cette démocratisation de la CFD entraine une réduction de coûts et rend donc cette gamme d'outils accessible, même aux plus petits joueurs dans le domaine.

Toutefois, il est encore difficile pour bon nombre de personnes de faire confiance aux résultats issus de la CFD. Alors que nous pouvons certainement attribuer une partie de cette méfiance au manque de connaissances sur le sujet, il reste que ces logiciels peuvent souvent être mal utilisés dus à la confiance aveugle des utilisateurs envers les logiciels. Nous en avons donc fait le but intrinsèque de cette recherche d'informer les utilisateurs sur la nécessité de vérifier les éléments du logiciel pouvant nuire à la précision des résultats obtenus.

Ce chapitre d'introduction pose les bases de ce travail. Tout d'abord, nous présentons brièvement les principes de base de la modélisation en CFD, avec une emphase sur les écoulements turbulents. Ensuite, nous introduisons OpenFOAM, le logiciel employé pour cette étude.

1.1 Concepts théoriques

La théorie couverte dans le présent chapitre jette les bases requises pour comprendre l'implémentation des concepts mathématiques dans le logiciel OpenFOAM. En effet, une bonne compréhension de l'aspect mathématique de la turbulence ainsi que des méthodes numériques est requise, car cela nous permet de configurer correctement les écoulements à être résolus. En plus, ces connaissances sont particulièrement pertinentes dans le cadre de ce travail, où nous effectuons la vérification de code via la méthode des solutions manufacturées. Nous exposons au lecteur les concepts clés qui permettront de critiquer, mais aussi d'apprécier la complexité des éléments présentés dans les chapitres subséquents. Nous commençons par développer la théorie de base de la turbulence, suivi des principes de sa modélisation avec le modèle de turbulence k-ϵ standard. Le chapitre se termine avec la théorie de la méthode des volumes finis. Cette méthode nous permet de passer d'un système d'ÉDP non linéaire vers un système d'équations linéaires.

Afin de décrire adéquatement le phénomène de la turbulence, nous faisons premièrement appel à la notion des tourbillons présents dans un écoulement. Un écoulement turbulent est composé de plusieurs échelles de tourbillons, comme illustré à la Figure 1.1. On remarque que les tourbillons sont plus petits près de la paroi. En effet, étant donné que l'épaisseur de la couche limite diminue avec l'augmentation du nombre de Reynolds, la dimension maximale des tourbillons près des surfaces diminue à son tour, rendant l'écoulement plus laborieux à résoudre.

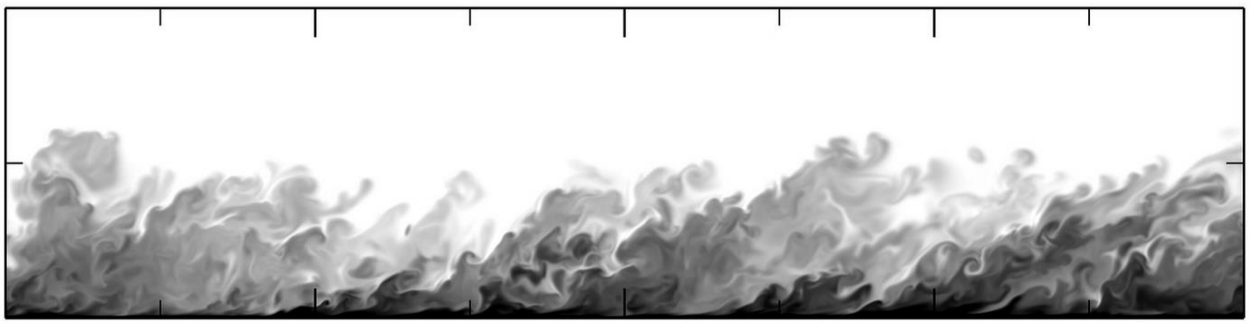

Figure 1.1: Poggie, J. (2016). Direct Numerical Simulation of Compressible, Turbulent Flow [Photographie]. Consulté le 23 juillet 2019, tiré de https://www.flickr.com/photos/argonne/24855410663.

L'énergie cinétique des plus grandes échelles de tourbillons est alors transmise aux échelles inférieures, en cascade, jusqu'à ce que les plus petites échelles dissipent l'énergie sous forme de chaleur (Wilcox, 1998). Nous pouvons donc en déduire que la quantité d'énergie dissipée est dépendante de l'énergie transmise par les plus grands tourbillons en amont de la cascade. Notons que ce phénomène est ardu à résoudre d'un point de vue mathématique. C'est pourquoi nous nous attardons plutôt à modéliser ce phénomène à partir des équations régissant la turbulence.

Les prochaines sous-sections passent en revue les équations décrivant la physique d'un écoulement turbulent ainsi que les modèles qui en découlent. En prenant comme point de départ les équations de Navier-Stokes, il faut invoquer plusieurs approximations et hypothèses pour en arriver à un modèle numérique exploitable pour la CFD. Le développement des équations de Navier-Stokes nous mènera vers la construction du modèle k-ϵ standard. De plus, nous introduirons brièvement les principes de la méthode par volume finis, d'usage courant dans les logiciels de CFD pour la résolution des équations différentielles. Finalement, nous présenterons la manière dont les équations sont mises en place dans OpenFOAM afin de familiariser le lecteur avec les avantages et défis liés à ce logiciel. Il est à noter que la base du développement des équations qui suivent est tirée des livres de Pope (2000) et de Wilcox (1998) alors que des notes et explications personnelles ont été ajoutées.

1.1.1 Équations de la turbulence

Nous commençons la description d'un écoulement turbulent en faisant appel aux équations de Navier-Stokes, soit celles de la continuité et de la quantité de mouvement pour un fluide newtonien et incompressible. Ces équations s'écrivent :

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{1.1}$$

$$\rho \frac{\partial u_i}{\partial t} + \rho u_j \frac{\partial u_i}{\partial x_j} = -\frac{\partial p}{\partial x_i} + 2\mu \frac{\partial s_{ij}}{\partial x_j} \tag{1.2}$$

où ρ est la masse volumique, u le vecteur vitesse, p la pression. De plus, le tenseur du taux de déformation s_{ij} est défini comme :

$$s_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) \tag{1.3}$$

L'application du théorème de la dérivée d'un produit conduit à :

$$\frac{\partial}{\partial x_j}(u_j u_i) = u_i \underbrace{\frac{\partial u_j}{\partial x_j}}_{=0\ (éq.1.1)} + u_j \frac{\partial u_i}{\partial x_j} \tag{1.4}$$

Substituant l'équation 1.4 dans l'équation 1.2, nous obtenons une nouvelle expression de l'équation de la quantité de mouvement qui se prête mieux aux opérations algébriques qui suivent.

$$\rho \frac{\partial u_i}{\partial t} + \rho \frac{\partial}{\partial x_j}(u_j u_i) = -\frac{\partial p}{\partial x_i} + 2\mu \frac{\partial s_{ij}}{\partial x_j} \tag{1.5}$$

En effet, le produit des vitesses instantanées facilite l'expression des équations de Navier-Stokes en fonction des vitesses moyennes. C'est d'ailleurs l'hypothèse de base de la méthode *Reynolds-Averaged Navier-Stokes* (RANS) que la turbulence peut être approximée par une vitesse moyenne, obtenue à partir de méthodes statistiques. L'expression d'un écoulement turbulent passe donc par la décomposition de la vitesse par ses composantes moyenne et fluctuante.

$$u_i(x,t) = \underbrace{U_i(x)}_{Moyenne} + \underbrace{u'_i(x,t)}_{Fluctuante} \tag{1.6}$$

Par l'hypothèse des modèles RANS, nous effectuons alors la moyenne du produit $u_j u_i$ de l'équation 1.5. La moyenne d'un produit de valeurs ayant une composante moyenne et fluctuante s'exprime comme :

$$\begin{aligned}\overline{u_j u_\iota} &= \overline{\left(U_j + u'_j\right)\left(U_\iota + u'_\iota\right)} = \overline{U_j U_\iota + \underbrace{U_j u'_\iota}_{=0} + \underbrace{U_\iota u'_j}_{=0} + u'_j u'_\iota} \\ &= U_j U_i + \overline{u'_j u'_\iota}\end{aligned} \tag{1.7}$$

Notons ici que la moyenne d'un produit d'une quantité moyenne avec une quantité fluctuante est nulle. Conséquemment, nous substituons les composantes de la vitesse des équations 1.6 et 1.7 dans les équations 1.1 et 1.5 afin d'obtenir le système d'équations qui suit.

$$\frac{\partial U_i}{\partial x_i} = 0 \tag{1.8}$$

$$\rho \frac{\partial U_i}{\partial t} + \rho \frac{\partial}{\partial x_j}\left(U_j U_i + \overline{u'_j u'_\iota}\right) = -\frac{\partial P}{\partial x_i} + 2\mu \frac{\partial S_{ij}}{\partial x_j} \tag{1.9}$$

Nous pouvons réécrire l'équation 1.9 pour faire apparaitre le produit des quantités fluctuantes du membre de droite de l'équation.

$$\rho \frac{\partial U_i}{\partial t} + \rho U_j \frac{\partial U_i}{\partial x_j} = -\frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j}\left(2\mu S_{ij} - \rho \overline{u'_j u'_\iota}\right) \tag{1.10}$$

Ce faisant, on introduit le tenseur de Reynolds :

$$\tau_{ij} = -\rho \overline{u'_\iota u'_j} \tag{1.11}$$

Or, ce tenseur cause des difficultés dans sa résolution, car il introduit six inconnues supplémentaires au système d'équations. Le problème n'est donc pas fermé, car il y a plus d'inconnues que d'équations. Afin d'ajouter des équations au système, nous devons trouver des relations à partir de principes mathématiques plutôt qu'à partir de relations physiques. Il en résulte donc une représentation simplifiée de la turbulence et une perte d'information non négligeable.

À ce point du développement, nous avons les inconnues suivantes :

- 3 pour la vitesse U_i;
- 1 pour la pression P;
- 6 pour le tenseur de Reynolds τ_{ij}.

Du côté des équations différentielles, nous avons :

- 1 pour la conservation de la masse;
- 3 pour la conservation de la quantité de mouvement.

Afin de fermer ce système, Wilcox (1998) présente le développement d'ÉDP pour le tenseur de Reynolds. Ces ÉDP nous permettent d'obtenir des corrélations d'ordre supérieur. Cependant, nous introduisons encore plus d'inconnues dans le système, sans jamais arriver à un nombre égal d'équations. Le résultat est donné à l'équation 1.12, où l'on remarque que des termes d'ordre supérieur se sont ajoutés, complexifiant davantage l'équation à résoudre.

$$\frac{\partial \tau_{ij}}{\partial t} + U_k \frac{\partial \tau_{ij}}{\partial x_k} = -\tau_{ik} \frac{\partial U_j}{\partial x_k} - \tau_{jk} \frac{\partial U_i}{\partial x_k} + \epsilon_{ij} - \Pi_{ij} + \frac{\partial}{\partial x_k}\left[\upsilon \frac{\partial \tau_{ij}}{\partial x_k} + C_{ijk}\right] \qquad (1.12)$$

Il est donc nécessaire de procéder autrement pour fermer le système d'équations. Une façon courante d'y parvenir pour les modèles RANS est d'introduire la viscosité turbulente de Boussinesq afin d'approximer la tension de Reynolds.

$$\overline{u'_i u'_j} = \frac{2}{3} k \delta_{ij} - \nu_T \left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) \qquad (1.13)$$

L'emploi de cette relation est souvent contesté dans la littérature. En effet, la viscosité turbulente de Boussinesq est à l'origine de plusieurs faiblesses qu'on retrouve dans les modèles de turbulence de type RANS. Le Chapitre 2 discute d'une manière plus approfondie des limites de validité de la viscosité turbulente de Boussinesq.

Jusqu'à présent, nous avons effectué plusieurs approximations et simplifications propres aux modèles RANS. Toutefois, elles ne sont pas suffisantes pour fermer le système d'équations de Navier-Stokes. Il est alors nécessaire de modéliser l'écoulement turbulent moyenné à l'aide d'équations supplémentaires qui, à leur tour, approximent ce phénomène complexe. Compte tenu

de ces nombreuses approximations, aucun modèle n'est en mesure de représenter adéquatement toute la complexité d'un écoulement turbulent. Nous avons donc choisi de nous attarder au modèle k-ϵ standard, car il s'agit d'un modèle turbulence très rependu dans l'industrie et bien maitrisé au niveau de ses forces et ses faiblesses.

1.1.2 Le modèle k-ϵ standard

Le modèle k-ϵ standard, comme son nom l'indique, fait usage de deux équations de transport pour modéliser les quantités turbulentes, soit k et ϵ. La première équation décrit l'énergie cinétique turbulente, utilisée dans la majorité des modèles à une et deux équations. Nous commençons à développer cette équation de transport en effectuant la trace de l'équation 1.12.

$$\underbrace{\frac{\partial k}{\partial t} + U_j \frac{\partial k}{\partial x_j}}_{\frac{Dk}{Dx}} = \underbrace{\tau_{ij} \frac{\partial U_i}{\partial x_j}}_{Production} - \epsilon + \frac{\partial}{\partial x_j}\left[\underbrace{\upsilon \frac{\partial k}{\partial x_j}}_{\substack{Diffusion \\ moléculaire}} - \frac{1}{2} \underbrace{\overline{u_i'^2 u_J'}}_{Transport} - \frac{1}{\rho} \underbrace{\overline{p' u_J'}}_{\substack{Diffusion \\ pression}} \right] \tag{1.14}$$

Le terme de dissipation s'exprimant comme suit :

$$\epsilon = \upsilon \frac{\partial u_i'^2}{\partial^2 x_k} \tag{1.15}$$

Notons que les termes τ_{ij}, de dissipation, de transport de turbulence ainsi que de la diffusion de la pression, sont toujours inconnus. Le nombre d'inconnues restantes dans l'équation 1.14 doit donc être réduit davantage en modélisant les termes de transport et de diffusion de la pression. Ainsi, il est courant de poser la relation suivante, comprenant les termes de transport de turbulence et de diffusion de pression.

$$\frac{1}{2}\overline{u_i'^2 u_J'} - \frac{1}{\rho}\overline{p' u_J'} = \frac{\upsilon_T}{\sigma_K}\frac{\partial k}{\partial x_j} \tag{1.16}$$

Ici, σ_K est un coefficient de fermeture.

Notons que cette relation n'est valide que pour un écoulement simple. Or, il est possible de générer des erreurs non négligeables pour une gamme importante d'écoulements turbulents en employant cette simplification. Malgré cela, son usage est très utile dans ce contexte, car il permet de simplifier grandement l'équation de transport. Il est donc possible de substituer l'équation 1.16 dans l'équation 1.14 afin d'obtenir l'équation de l'énergie cinétique de turbulence.

$$\frac{\partial k}{\partial t} + U_j \frac{\partial k}{\partial x_j} = \tau_{ij} \frac{\partial U_i}{\partial x_j} - \epsilon + \frac{\partial}{\partial x_j}\left[\left(\upsilon + \frac{\upsilon_T}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] \tag{1.17}$$

À cette étape du développement, il reste toujours à préciser les termes k et ϵ. La méthode utilisée pour résoudre ϵ varie en fonction du modèle de turbulence RANS employé. Pour le modèle k-ϵ standard, nous modélisons le taux de dissipation à l'aide d'une deuxième équation de transport (Launder & Sharma, 1974).

$$\frac{\partial \varepsilon}{\partial t} + U_j \frac{\partial \varepsilon}{\partial x_j} = C_{\varepsilon 1} \frac{\varepsilon}{k} \tau_{ij} \frac{\partial U_i}{\partial x_j} - C_{\varepsilon 2} \frac{\varepsilon^2}{k} + \frac{\partial}{\partial x_j}\left[\left(\upsilon + \frac{\upsilon_T}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right] \tag{1.18}$$

C'est alors que nous introduisons la viscosité cinématique turbulente pour un écoulement à haut nombre de Reynolds, donnée par Jones, W. P. et Launder (1972).

$$\upsilon_T = C_\mu \frac{k^2}{\varepsilon} \tag{1.19}$$

Nous remarquons que les équations de transport de k et ϵ dépendent de plusieurs constantes obtenues empiriquement. Ces constantes, toujours utilisées aujourd'hui, sont issues de la dernière mise à jour par Launder et Sharma (1974). Elles ont été définies de manière à représenter le mieux possible une majorité d'écoulements, mais peuvent nécessiter des ajustements pour certains écoulements particuliers.

$$C_{\varepsilon 1} = 1.44, \qquad C_{\varepsilon 2} = 1.92, \qquad C_\mu = 0.09, \qquad \sigma_k = 1.0, \qquad \sigma_\varepsilon = 1.3 \tag{1.20}$$

Jusqu'ici nous avons élaboré les équations différentielles décrivant un écoulement turbulent incompressible. Tout au long de la démonstration, plusieurs simplifications ont dû être effectuées, car la nature des équations de Navier-Stokes est trop complexe pour permettre sa résolution directe. Néanmoins, ces simplifications nous permettent d'obtenir un modèle suffisamment précis et

simple, pouvant être intégré dans un logiciel de CFD. Avec ces équations comme point de départ, il nous reste donc à expliquer comment elles sont traduites dans un logiciel de calcul numérique.

1.1.3 La méthode des volumes finis

Nous remarquons que les équations de la turbulence définies au Chapitre 1 sont des ÉDP qui se trouvent dans le domaine continu. En calcul numérique, nous devons plutôt résoudre des équations se trouvant dans le domaine discret sous la forme $Ax = b$. La Figure 1.2 illustre les différentes étapes nous permettant de passer des mesures d'un écoulement réel vers une solution numérique. À l'étape « Méthode numérique », les ÉDP sont discrétisés en un ensemble de volumes nommé maillage. Le logiciel de CFD peut alors facilement résoudre le système d'équations linéaire qui résulte de cette discrétisation.

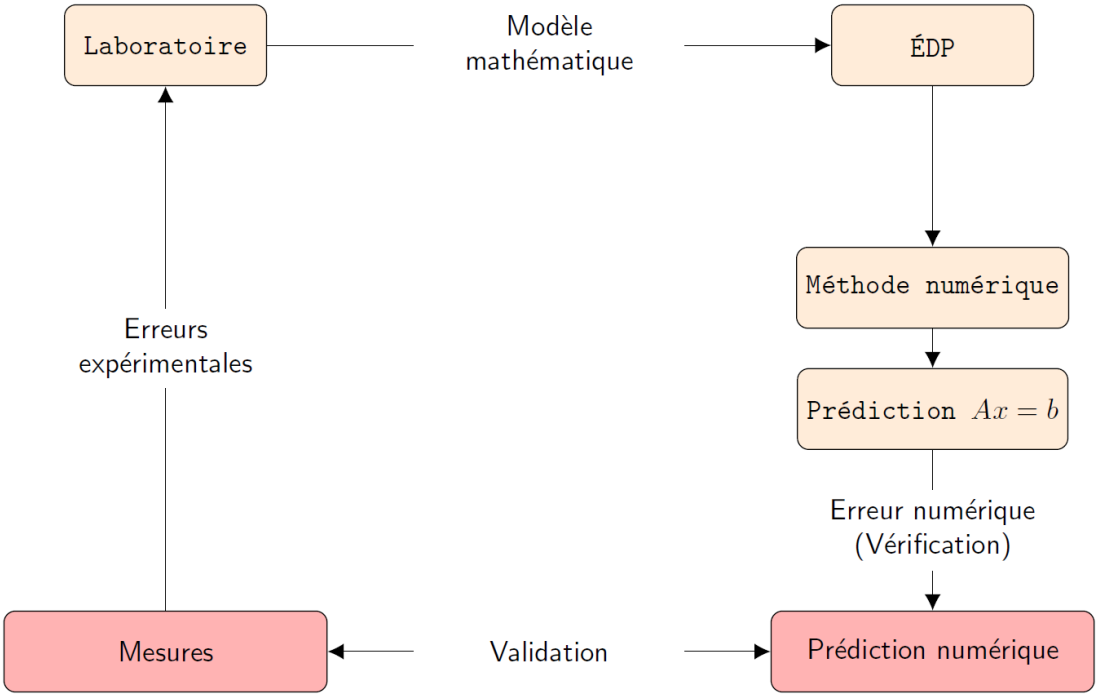

Figure 1.2: Logigramme de la discrétisation du domaine continue en CFD

Nous remarquons plusieurs sources d'erreurs à la Figure 1.2 lorsqu'on passe des mesures d'un écoulement réel à la résolution numérique de ce même écoulement. En effet, des erreurs sont présentes lors de la prise de mesures, lors de la modélisation des ÉDP et lors de la résolution numérique du problème. Pour ce travail, nous ne considérons que les sources d'erreurs numériques.

En d'autres mots, nous cherchons à nous assurer que le taux de convergence théorique est respecté lors de raffinements subséquents de maillages.

Alors qu'un maillage est raffiné, nous nous attendons à ce que la solution numérique tende vers la solution des ÉDP. L'erreur numérique e_h diminue donc proportionnellement au raffinement de maillage, tel que montré schématiquement à la Figure 1.3.

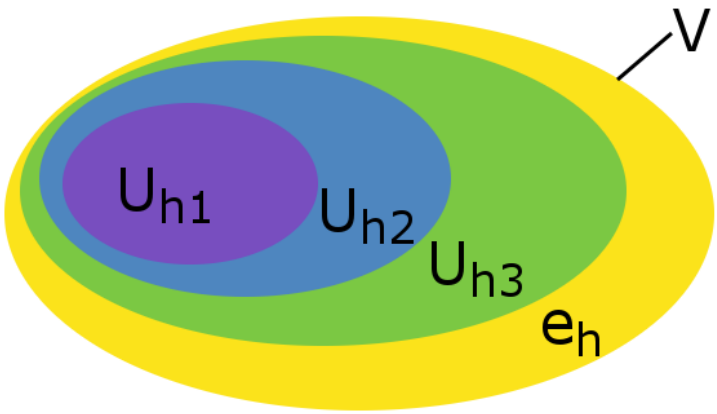

Figure 1.3: Progression de la solution U dans un espace V lors du raffinement du maillage

Ici, U_{hi} représente la solution numérique d'un écoulement U pour des éléments de maillage de dimension h_i. L'indice i représente le niveau de raffinement de maillage, tel que $h3 < h2 < h1$. Nous remarquons qu'au fur et à mesure que le maillage est raffiné, U_{hi} tend vers la solution V et l'erreur e_h diminue proportionnellement.

Il existe trois méthodes nous permettant de parvenir à cette discrétisation, soit : la méthode des différences finies, celle des volumes finis et celle des éléments finis. En ce qui concerne la CFD, on emploie surtout les volumes finis et les éléments finis pour résoudre les équations présentées au Chapitre 1. Dans le cas du présent mémoire, la méthode des volumes finis a été choisie, car celle-ci est employée dans la plupart des logiciels de CFD, dont OpenFOAM, le sujet de la présente étude.

Ce qui distingue la méthode des volumes finis par rapport celle des éléments finis, c'est le fait qu'elle est conservative, assurant ainsi naturellement l'équilibre entre les flux entrants et les flux sortants des éléments de maillages. De plus, la méthode des volumes finis peut s'appliquer à une grande variété de types de maillages, s'adaptant ainsi à une diversité de problèmes de CFD ayant divers niveaux de complexité. Ces qualités font des volumes finis la méthode de discrétisation idéale pour la résolution de problèmes en CFD.

Ayant sélectionné la méthode des volumes finis, nous cherchons à passer des équations continues (calculs d'intégrales) vers des équations discrètes (calculs algébriques). La mise en application de cette méthode est constituée de deux étapes principales (Moukalled, Mangani, & Darwish, 2016) :

1. les ÉDP sont intégrées sur les éléments de maillage, afin d'obtenir des équations algébriques;
2. les variables, internes aux volumes, sont projetées sur les frontières afin de trouver la variation de quantités à l'intérieur des éléments de maillage.

Il est à noter que cette méthode est théoriquement d'ordre 2 en précision lorsque les schémas de discrétisation sont évalués aux centres des volumes de maillage. Par conséquent, en divisant la dimension des éléments de maillage par deux, l'erreur de la solution obtenue devrait diminuer par un facteur quatre.

En ce qui concerne l'implémentation, les ÉDP d'un écoulement en régime permanent peuvent être représentées par l'équation de conservation présentée à l'équation 1.21. Notons que ϕ est un scalaire représentant une propriété de l'écoulement et que Γ est le coefficient de diffusion.

$$\underbrace{\nabla \cdot (\rho v \phi)}_{\substack{flux \\ convectif}} = \underbrace{\nabla \cdot \left(\Gamma^{\phi} \nabla \phi\right)}_{\substack{flux\ de \\ diffusion}} + \underbrace{Q^{\phi}}_{\substack{terme \\ source}} \tag{1.21}$$

À partir de l'équation 1.21, il est possible de développer les étapes de fonctionnement de la méthode des volumes finis. Cette démarche n'est pas répliquée dans ce texte, mais peut être consultée dans le livre de Moukalled et al. (2016).

Ayant exposé les principes de base des volumes finis, nous nous attardons maintenant à la manière dont ces éléments sont programmés dans le logiciel OpenFOAM. En effet, chaque logiciel peut intégrer la même physique à sa propre manière, pouvant générer certaines différences au niveau de la solution finale. Avec OpenFOAM, nous avons l'avantage de pouvoir consulter le code source sans difficulté et ainsi comparer l'implémentation des équations de turbulence avec la théorie.

1.2 Introduction à OpenFOAM

OpenFOAM (*Open Field Operation and Manipulation*) est le plus important logiciel de CFD complètement gratuit et à code source ouvert. Il consiste en un regroupement d'une vaste librairie

d'applications liées entre elles, servant à résoudre une grande variété de phénomènes. Ce logiciel comprend aussi des outils de génération et diagnostic de maillage et il est distribué de concert avec l'outil de visualisation Paraview. OpenFOAM est distribué sous une licence publique générale GNU, signifiant qu'il est possible pour toute personne de modifier son contenu ainsi que d'en faire la distribution. N'importe quel utilisateur peut donc adapter les fonctionnalités du code source en fonction de ses besoins.

Cependant, OpenFOAM comporte plusieurs obstacles pour l'utilisateur novice. Nous avons constaté qu'il y a très peu de documentation faisant office de mode d'emploi. De plus, aucune interface graphique n'est fournie avec le logiciel. En effet, la configuration du modèle doit être faite en modifiant les fichiers se trouvant dans une arborescence de dossiers, tels que ceux de l'exemple se trouvant à la Figure 1.4. Il peut alors être ardu pour l'utilisateur d'introduire de nouvelles fonctionnalités qui ne font pas partie du cas de base (termes sources, fluides multiphases, transfert de chaleur conjugué, etc.) tout en les incorporant avec les autres fichiers existants.

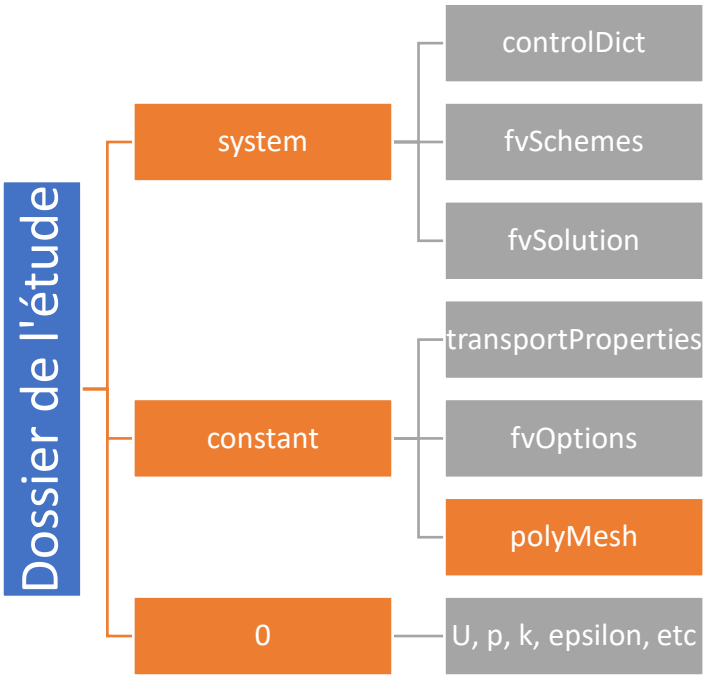

Figure 1.4: Arborescence typique des fichiers de OpenFOAM

Toutefois, il est possible de simplifier et d'accélérer la configuration de modèles similaires à des études existantes, car la majorité des fichiers peuvent être récupérés d'une étude à l'autre. De plus, la complexité liée à l'élaboration d'une étude CFD peut être surmontée par des utilisateurs expérimentés, ayant suffisamment maitrisé l'implémentation de OpenFOAM pour comprendre sa

logique sous-jacente. Notons toutefois que beaucoup de temps est nécessaire pour qu'un utilisateur puisse atteindre une telle maîtrise du code.

Dans un autre ordre d'idées, notons que dans le langage de OpenFOAM, la vitesse *U* comprend les variables *Ux*, *Uy* et *Uz*. En outre, les conditions limites sont appliquées simultanément aux trois composantes de *U* en tant que vecteur. Cela a comme effet de compliquer l'application des conditions de Neumann et Dirichlet sur la même frontière pour la vitesse. Pour ce cas particulier, une condition limite spécifiquement conçue pour de telles applications doit alors être spécifiée dans OpenFOAM.

Au niveau du code, le langage de base de OpenFOAM est le C++, un langage orienté objet. Nous avons donc l'avantage d'avoir un code plus compréhensible pour l'utilisateur et une formulation des équations plus près de leur représentation mathématique. Malgré cette formulation, l'importante quantité de classes employées dans le code complexifie la lecture du code source.

Le code définit les *classes* comme des objets qui servent à définir les constituants d'une formulation mathématique. Par exemple, la vitesse *U*, étant un champ de vecteurs, est un objet de la classe « vectorField ». La classe « vectorField » est elle-même créée à partir de deux autres classes, soit la classe « vector » (vecteur) et la classe « field » (champ). La représentation des équations à résoudre est simplifiée en utilisant cette nomenclature. En effet, il est prévu que le code soit le plus près possible de sa formulation mathématique afin d'en faciliter la compréhension et l'ajout de code personnalisé.

Dans les lignes qui suivent nous présentons l'implémentation de l'énergie cinétique turbulente dans le code OpenFOAM pour le solveur k-ϵ standard. Nous commençons avec l'équation de l'énergie cinétique de la turbulence présentée à l'équation 1.17.

$$\rho \frac{\partial k}{\partial t} + \rho U_j \frac{\partial k}{\partial x_j} = \rho \frac{\partial U_i}{\partial x_j} \tau_{ij} - \rho \epsilon + \rho \frac{\partial}{\partial x_j} \left[\left(\upsilon + \frac{\upsilon_T}{\sigma_k} \right) \frac{\partial k}{\partial x_j} \right] \tag{1.22}$$

L'équation 1.22 est implémentée de la manière suivante dans OpenFOAM :

$$\alpha\rho\frac{\partial k}{\partial t}+\alpha\rho U_j\frac{\partial k}{\partial x_j}-\alpha\rho\frac{\partial}{\partial x_j}\left[\left(\upsilon+\frac{\upsilon_T}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] = \alpha\rho\frac{\partial U_i}{\partial x_j}\tau_{ij}-\alpha\rho\varepsilon+k_s+fvOptions \tag{1.23}$$

Nous remarquons que certains termes ont été ajoutés à l'équation 1.23. Le terme α est associé à la diffusivité thermique pour un écoulement compressible. Il est donc égal à 1 pour ce travail. D'autre part, le terme k_s est égal à zéro, mais peut servir lors de la résolution d'écoulements de convection naturelle. Finalement, le terme *fvOptions* sert à ajouter un terme source défini par l'utilisateur, tel que dans le cas de l'implémentation d'une solution manufacturée. Nous réécrivons donc l'équation 1.23 en conservant le terme source et en éliminant α et k_s.

$$\rho\frac{\partial k}{\partial t}+\underbrace{\rho U_j\frac{\partial k}{\partial x_j}}_{Advection}-\underbrace{\rho\frac{\partial}{\partial x_j}\left[\left(\upsilon+\frac{\upsilon_T}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right]}_{Transport} = \underbrace{\rho\frac{\partial U_i}{\partial x_j}\tau_{ij}}_{Production}-\underbrace{\rho\varepsilon}_{Diffusion}+\underbrace{fvOptions}_{Terme\ source} \tag{1.24}$$

Pour des fins de simplification, le code stocke certains termes de l'équation 1.24 à l'intérieur de variables. Par exemple, le terme de production de l'équation 1.24 est donné par :

$$\rho\frac{\partial U_i}{\partial x_j}\tau_{ij}=\rho\underbrace{\upsilon_T\frac{\partial U_i}{\partial x_j}\left(\frac{\partial U_i}{\partial x_j}+\frac{\partial U_j}{\partial x_i}\right)}_{=G}-\frac{2}{3}\rho U_j\frac{\partial k}{\partial x_j} \tag{1.25}$$

Notons que le terme identifié par *G* à l'équation 1.25 n'est pas calculé comme tel dans OpenFOAM. En effet, les programmeurs ont plutôt représenté *G* par une équation équivalente qui se prête mieux aux opérations numériques effectuées par le solveur. La variable *G* est donc représentée dans le code de la manière suivante :

$$G = \upsilon_T[dev(\nabla U + \nabla U^T) : \nabla U]$$
$$= \upsilon_T \left[\left(\frac{\partial U_i}{\partial x_i}\right)^2 + \left(\frac{\partial U_i}{\partial x_j}\right)^2 + \left(\frac{\partial U_j}{\partial x_i}\right)^2 + \left(\frac{\partial U_j}{\partial x_j}\right)^2 - 2\frac{\partial U_i}{\partial x_i}\frac{\partial U_j}{\partial x_j} + 2\frac{\partial U_i}{\partial x_j}\frac{\partial U_j}{\partial x_i} \right] \quad (1.26)$$

Étant donné que le fluide étudié est incompressible (équation 1.1), l'équation 1.26 se simplifie de manière à ce que *G* soit égal au terme qu'il substitue dans l'équation 1.25.

$$G = \upsilon_T \frac{\partial U_i}{\partial x_j}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) \quad (1.27)$$

L'équation 1.26 est donc implémentée de la manière suivante dans le code OpenFOAM :

Code de l'initialisation de G :

```
    (
        this->GName(),
        nut.v()*(dev(twoSymm(tgradU().v())) && tgradU().v())
    );
```

La fonction *dev* est le déviateur d'une matrice et l'opération *twoSymm* correspond à $\nabla U + \nabla U^T$. Nous constatons que la syntaxe du code est très similaire à celle de l'équation 1.26. Nous pouvons donc facilement confirmer qu'il s'agit de la même équation, sans nécessiter des connaissances en C++.

D'autre part, nous introduisons une seconde variable afin de calculer les effets visqueux du terme de transport de l'équation 1.24.

$$Dkeff = \frac{\upsilon_T}{\sigma_k} + \upsilon \quad (1.28)$$

L'équation 1.28 est implémentée dans le code comme suit :

Code de l'initialisation de DkEff :

```
    (
        "DkEff",
        (this->nut_/sigmak_ + this->nu())
    )
```

En substituant les équations 1.25, 1.27 et 1.28 dans l'équation 1.24, nous obtenons l'équation que OpenFOAM résout pour l'énergie cinétique de la turbulence.

$$\rho\frac{\partial k}{\partial t}+\rho U_j\frac{\partial k}{\partial x_j}-\rho\frac{\partial}{\partial x_j}\left[(dkEff)\frac{\partial k}{\partial x_j}\right] = \rho G-\frac{2}{3}\rho U_j\frac{\partial k}{\partial x_j}-\rho\varepsilon+fvOptions \tag{1.29}$$

Avant de présenter l'implémentation de l'équation 1.29, il est nécessaire de couvrir certaines fonctions employées dans le code OpenFOAM. Les explications qui suivent se trouvent dans le guide du programmeur du logiciel (Greenshields, 2015).

La fonction « fvm » discrétise les dérivés pour représenter les coefficients sous forme matricielle. Donc pour une ÉDP représentée sous la forme discrète Ax=b, la fonction « fvm » renvoie les coefficients de la matrice A. Cette fonction est suivie des termes (dérivés) à évaluer pour l'ÉDP. Par exemple, la fonction « fvm ::ddt(ϕ) » évalue la dérivée temporelle de la variable ϕ. Les termes suivants sont aussi employés dans l'extrait de code de l'énergie cinétique turbulente : la convection « div », le laplacien « laplacian », et les termes sources « SuSp » et « Sp ».

D'autre part, OpemFOAM emploie la variable *phi* afin de représenter le flux massique passant au travers des faces des volumes. Nous pouvons donc représenter *phi* de la manière suivante :

$$\phi=(\rho\vec{u})\cdot\vec{n}S \tag{1.30}$$

L'implémentation de l'équation de transport pour l'énergie cinétique turbulente est donc effectuée de la manière suivante :

Code de l'énergie cinétique turbulente :

```
    (
        fvm::ddt(alpha, rho, k_)
      + fvm::div(alphaRhoPhi, k_)
      - fvm::laplacian(alpha*rho*DkEff(), k_)
     ==
    alpha()*rho()*G
      - fvm::SuSp((2.0/3.0)*alpha()*rho()*divU, k_)
      - fvm::Sp(alpha()*rho()*epsilon_()/k_(), k_)
      + kSource()
      + fvOptions(alpha, rho, k_)
    );
```

Nous remarquons qu'un effort considérable a été mis de l'avant afin de simplifier la syntaxe du code OpenFOAM. L'utilisateur familier avec cette syntaxe peut facilement repérer les équations pour en faire la modification ou pour ajouter ses propres lignes de code. Cependant, il demeure complexe pour l'utilisateur novice de comprendre l'ensemble des fonctions utilisées. En effet, il existe une quantité considérable de fonctions disponibles et la documentation du logiciel ne fournit que très peu d'informations sur leur utilité. En outre, comprendre le code consomme beaucoup de temps et l'utilisateur désirant apporter des modifications au code devra dépendre de la méthode essai-erreur.

D'autres exemples semblables concernant l'interprétation du code source peuvent être consultés dans le travail de Furbo (2010). Ceux-ci comportent des explications supplémentaires, bien qu'ils aient été élaborés pour une version antérieure du logiciel.

Ayant fait la démonstration de la lecture du code source de OpenFOAM, il est relativement simple de transposer cette méthode de façon à aussi comprendre l'implémentation de la vitesse, la pression, ou toute autre variable d'intérêt. Un solveur est ensuite utilisé afin de résoudre ce système d'équations suivant une séquence spécifique en fonction du type d'écoulement étudié.

1.2.1 Présentation du solveur simpleFoam

Le solveur employé dans le cadre de ce travail a été sélectionné en fonction d'un écoulement en régime permanent, turbulent et incompressible. Ceci étant, la documentation de OpenFOAM (Greenshields, 2018) précise que le solveur idéal pour ce type d'écoulement est le solveur *simpleFoam*, une version adaptée pour OpenFOAM de l'algorithme SIMPLE (*Semi-Implicit Method for Pressure Linked Equations*). L'algorithme SIMPLE a été créé par Caretto, Gosman, Patankar et Spalding (1973) comme une méthode qui approxime les valeurs recherchées et les corrige à l'aide de l'équation de la quantité de mouvement. Plusieurs calculs de pression et de vitesse se succèdent afin d'arriver à une solution comportant une erreur sous un seuil défini par l'utilisateur. Cette séquence logique, telle qu'implémentée dans OpenFOAM, est illustrée à la Figure 1.5. Quoique ce solveur soit principalement utilisé pour des écoulements turbulents, nous pouvons aussi en faire usage pour un écoulement laminaire. En effet, le solveur simpleFoam ne fera qu'omettre de calculer les termes de turbulence, soit k et ϵ.

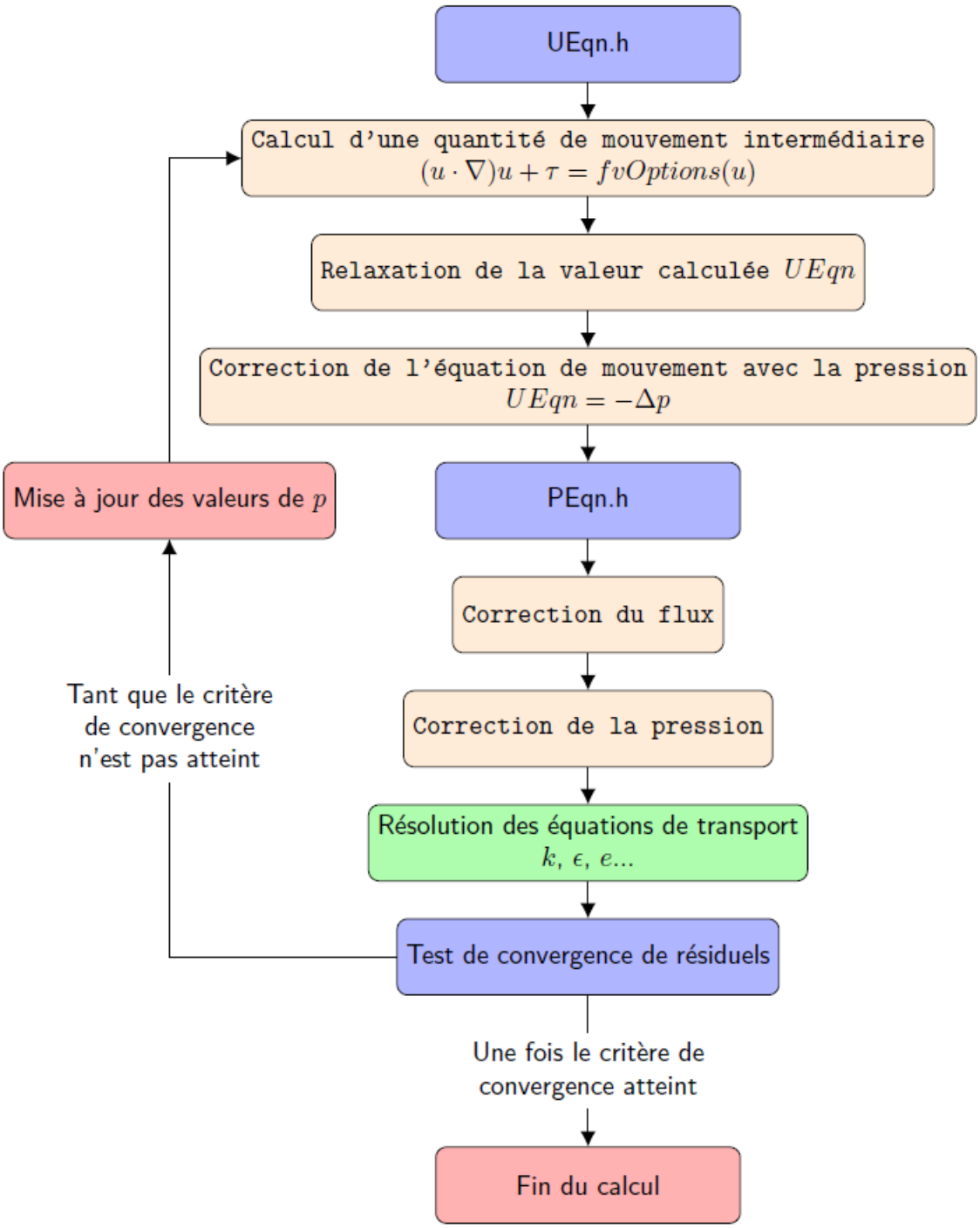

Figure 1.5: Logigramme du solveur simpleFoam

Sans faire la démonstration mathématique complète du solveur, nous présentons tout de même la portion initiale du calcul de *UEqn*, étant relativement complexe par rapport au restant de la démarche. Toutes les équations sous forme mathématique présentées ci-dessous peuvent être consultées à même le fichier du code source *UEqn.H*. Le restant du code de simpleFoam se trouve dans les fichiers *pEqn.H* pour le calcul de la pression et dans *simpleFoam.C*, là où se trouve l'appel des fonctions.

Afin de simplifier la démonstration, nous développons les équations pour un écoulement laminaire en ignorant les termes de turbulence. Dans un premier temps, le solveur résout l'équation suivante :

$$(u \cdot \nabla)u + \nabla \cdot \tau = 0 \tag{1.31}$$

où

$$\nabla \cdot \tau = \nabla \cdot \sigma^{dev} = \nabla \cdot \left(\nu_{eff}(\nabla u)\right) + \nabla \cdot \left(\nu_{eff} * dev((\nabla u)^T)\right) \tag{1.32}$$

À l'équation 1.32 nous retrouvons la composante déviatorique du gradient de la vitesse transposé. Selon le manuel du programmeur de OpenFOAM (Greenshields, 2015), ce calcul est effectué comme suit :

$$dev((\nabla u)^T) = (\nabla u)^T - \frac{1}{3} tr((\nabla u)^T)\mathrm{I} \tag{1.33}$$

En combinant les équations 1.32 et 1.33, nous obtenons l'expression calculée telle quelle par OpenFOAM lors de la résolution de *UEqn*.

$$\nabla \cdot \tau = \nabla \cdot \left(\nu_{eff}(\nabla u)\right) + \nabla \cdot \left(\nu_{eff}\left[(\nabla u)^T - \frac{1}{3} tr((\nabla u)^T)\mathrm{I}\right]\right) \tag{1.34}$$

Afin de comparer l'équation 1.34 avec la théorie, nous en faisons une réécriture qui facilitera les simplifications à venir.

$$\nabla \cdot \tau = \nabla \cdot \left(\nu_{eff}(\nabla u) + \nu_{eff}\left[(\nabla u)^T - \frac{1}{3} tr((\nabla u)^T)\mathrm{I}\right]\right) \tag{1.35}$$

À ce point du développement, Holzmann (2018) propose d'employer une relation permettant d'échanger la trace pour un opérateur de divergence.

$$tr\left(\frac{1}{2}[\nabla u + (\nabla u)^T]\right) = \nabla \cdot u \tag{1.36}$$

Nous substituons alors l'équation 1.36 dans l'équation 1.35 afin de faire apparaitre la divergence du vecteur vitesse. Par le principe de la continuité, la divergence venant d'apparaitre devient donc nulle.

$$\nabla \cdot \tau = \nabla \cdot \left(\nu_{eff}(\nabla u) + \nu_{eff}\left[(\nabla u)^T - \frac{1}{3}\underbrace{(\nabla \cdot u)}_{=0} \mathrm{I}\right]\right) \tag{1.37}$$

Nous obtenons alors une équation grandement simplifiée par rapport au calcul initial de simpleFoam.

$$\nabla \cdot \tau = \nabla \cdot \left(\nu_{eff}[\nabla u + (\nabla u)^T]\right) \tag{1.38}$$

Finalement, il nous est possible d'isoler le tenseur du taux de contrainte de l'équation 1.38. Rappelons que ce tenseur a été défini à l'équation 1.3 au début du chapitre en tant que S_{ij}.

$$\nabla \cdot \tau = \nabla \cdot \left(2\nu_{eff}\left[\underbrace{\frac{1}{2}[\nabla u + (\nabla u)^T]}_{s_{ij}}\right]\right) \tag{1.39}$$

En substituant l'équation 1.39 dans l'équation 1.31, nous retrouvons l'expression de la quantité de mouvement définie au début du chapitre, moins la pression. En effet, l'équation de la quantité de mouvement est corrigée avec la pression à une étape ultérieure de l'algorithme SIMPLE.

1.3 Plan du mémoire

Le mémoire est structuré de la manière suivante. Une revue de la littérature scientifique succède au présent chapitre, détaillant les avancées sur les modèles de turbulence, la vérification de code, et plus précisément, la vérification du code OpenFOAM. Une emphase est d'ailleurs mise sur ce dernier afin de détailler les lacunes se trouvant dans la littérature. Par la suite, nous abordons au Chapitre 3 la méthodologie employée afin de vérifier le code OpenFOAM. Nous y détaillons les

schémas numériques disponibles dans le logiciel, la méthode des solutions manufacturées, ainsi que les cas tests employés dans le cadre de ce mémoire. Enfin, au Chapitre 4 nous exposons les résultats obtenus à la suite de l'application des cas tests, présentés préalablement au Chapitre 3. Ces résultats sont accompagnés d'une analyse sur les éléments pouvant dégrader le taux de convergence observé du code. Le présent mémoire clôt avec une synthèse des résultats obtenus ainsi que des limitations qui ont été observées lors de la vérification du logiciel.

CHAPITRE 2 REVUE DE LITTÉRATURE

L'accroissement soutenu de la puissance et de la capacité de stockage des ordinateurs a permis le développement d'une variété de modèles numériques applicables à la CFD. Cependant, à ce jour il n'existe pas de modèle numérique en mesure de représenter entièrement la complexité d'un écoulement turbulent. C'est pourquoi de nombreux chercheurs dans le domaine continuent de développer de nouveaux modèles de turbulence, rendant possible la prédiction de différentes caractéristiques d'écoulements. Alors qu'il existe une panoplie de modèles de turbulence, on compte aussi plusieurs logiciels de CFD (Fluent, Star-CCM+, CFX, OpenFOAM, etc) qui implémentent ces modèles d'une façon propre à chacun. Afin que l'industrie puisse avoir confiance en ces outils, il devient alors nécessaire de quantifier rigoureusement la précision des modèles en fonction de l'usage qu'on prévoit en faire.

Dans le présent mémoire, nous avons choisi d'étudier le logiciel de CFD OpenFOAM, car il est bien répandu dans l'industrie, tout en offrant l'avantage d'être de code source libre. La librairie de solveurs qui y sont implémentés contient une variété de fonctions permettant de calculer de nombreux écoulements complexes, des réactions chimiques ainsi que des champs magnétiques en électromagnétique (OpenCFD, 2018). Alors que cet outil peut être librement utilisé et modifié par tous selon leurs besoins, il ne comporte aucune garantie sur l'implémentation adéquate de ses modèles. Il en revient donc à l'usager de déterminer si le modèle utilisé dans OpenFOAM se comporte comme prévu et s'il correspond à ses besoins pour un problème donné.

La revue des connaissances qui suit traite principalement de deux sujets, soit de la modélisation en CFD et de la vérification et validation. Tout d'abord, nous mettons en relief plusieurs modèles de turbulence tout en exposant certaines de leurs forces et faiblesses. Par la suite, nombre d'études sur la vérification et la validation sont présentées, servant à mettre en lumière la carence de vérification de code pour OpenFOAM. Ce chapitre se termine avec une présentation du but et des objectifs de ce mémoire.

2.1 Modèles de turbulence

2.1.1 Méthodes de modélisation

Trois principales approches regroupent les modèles de turbulence, ces dernières variant en précision et en temps de calcul. Ces approches, présentées dans les paragraphes qui suivent, sont la DNS (*Direct Numerical Simulation*), la LES (*Large Eddy Simulation*) et la RANS (*Reynolds Averaged Navier-Stokes*).

L'approche DNS consiste en un calcul direct des équations de Navier-Stokes et la résolution de toutes les échelles de turbulence. C'est-à-dire que la DNS n'est pas tributaire de modèles qui approximent le comportement d'un écoulement turbulent. Ceci offre l'avantage d'une grande précision dans les résultats obtenus, pouvant même servir comme substitut aux données expérimentales lors de comparaisons avec d'autres modèles de turbulences. Cependant, la densité du maillage requis, et conséquemment le coût de calcul occasionné, est beaucoup plus élevée pour la DNS que pour les méthodes classiques. De même, les ressources informatiques requises augmentent davantage pour des géométries complexes et pour des nombres de Reynolds élevés (Argyropoulos & Markatos, 2015). À titre d'exemple, Pope (2000) estime que, pour un écoulement à haut nombre de Reynolds tridimensionnel, le nombre de modes de fourrier (ou nœuds) requis varie en fonction de :

$$N^3 \approx 4.4 Re_L^{9/4} \tag{2.1}$$

où Re_L est le nombre de Reynolds turbulent. Comme cette variable influence la puissance de calcul requise pour résoudre un écoulement, on en déduit que les ressources informatiques nécessaires peuvent rapidement devenir exorbitantes.

À cela se rajoute le fait que la puissance de calcul requise s'accentue pour des écoulements réels et complexes (Moin & Mahesh, 1998). L'industrie n'ayant habituellement pas accès aux ressources informatiques que ces études requièrent, cette approche n'est généralement employable que pour des applications en recherche.

Les exigences au niveau du calcul ne sont toutefois pas un frein au nombre croissant de publications employant la DNS. De ceux-ci, une première étude pour un écoulement confiné, limitée par la puissance informatique de l'époque, fut publiée en 1987 par Kim, Moin et Moser (1987) lors de

leur recherche sur la couche limite. Plus récemment, on trouve aussi plusieurs exemples d'études de validation par la DNS, dont les travaux de Bricteux, Zeoli et Bourgeois (2017) concernant la validation des performances de OpenFOAM en employant le calcul à haute performance (CHP). Chaoqun, Yonghua et Yong (2014) ont également utilisé la DNS afin d'identifier les limitations de l'approximation de Boussinesq, utilisé dans plusieurs modèles RANS. Bien entendu, ces exemples s'inscrivent dans un plus vaste répertoire d'études employant la DNS à des fins de recherche.

Étant significativement moins couteuse à résoudre, la RANS est une approche statistique où l'on cherche à modéliser la turbulence comme un phénomène ayant une composante moyenne et une autre fluctuante. La moyenne des vitesses est prise sur une période de temps, sur le domaine ou sur un ensemble d'expérimentations (Wilcox, 1998), ce qui engendre des termes non-linéaires additionnels dans les équations de Navier-Stokes. Ces termes posent un grand défi dans la résolution des ÉDP, et doivent être modélisés par l'hypothèse de viscosité turbulente de Boussinesq (voir chapitre 1.1.1). C'est d'ailleurs dû à l'hypothèse de Boussinesq que les modèles RANS ont typiquement de la difficulté à représenter les écoulements tourbillonnants et les écoulements séparés (Chaoqun et al., 2014). Or, depuis plusieurs années certains chercheurs comme Schmitt (2007) reprochent à l'hypothèse de Boussinesq de ne pas être valide dans la majorité des écoulements rencontrés. Cette approche est beaucoup moins couteuse à employer au niveau des ressources informatiques que la DNS et la LES. C'est principalement pour cette raison que la RANS a donné lieu aux premiers modèles de turbulence, dont certains sont toujours employés en industrie aujourd'hui. En effet, ces modèles profitent d'une importante littérature, résultat du grand nombre de recherches qui ont été publiées ces dernières décennies (Albets-Chico, Pérez-Segarra, Oliva, & Bredberg, 2008; Chaoqun et al., 2014; Ilinca, Florin & Pelletier, 1998; Zhao, Zhang, Jiang, & Bo, 2017). Les lacunes des modèles RANS ont l'atout d'être connues et bien maitrisées, faisant de cette approche l'outil idéal pour la modélisation d'écoulements turbulents retrouvés en industrie. Étant la méthode employée dans ce mémoire, la RANS sera développée davantage à la section 2.1.2.

Finalement, l'approche LES offre un compromis entre la DNS et la RANS. Cette méthode résout numériquement les plus grandes échelles de turbulence alors que la modélisation est réservée aux plus petites échelles (plus dispendieuses à résoudre). À cette fin, un filtre appliqué aux ÉDP doit être paramétré par l'utilisateur dans le but de séparer les plus grandes des plus petites échelles de turbulence (Pope, 2000). Il en résulte que la LES offre une solution se rapprochant de celle de la

DNS, mais sans avoir à assumer le coût de calcul élevé des plus petites échelles de turbulence. Concernant la littérature sur le sujet, une première publication a été effectuée Smagorinsky (1963), appliqué au domaine de la météorologie. Ce n'est que quelques années plus tard qu'un modèle LES a été appliqué à un écoulement turbulent confiné (Deardorff, 1970), démontrant au passage l'utilité de cette approche pour des nombres de Reynolds élevés. Depuis, la recherche sur la LES s'est accentuée afin de permettre sa mise en œuvre pour des écoulements complexes et similaires aux problèmes rencontrés en industrie (Geurts, 2009; Piomelli, 2001). Parmi ceux-ci, certaines des applications comprennent le domaine de l'énergie, tel que la modélisation d'une flamme dans une chambre à combustion par Gicquel, Staffelbach et Poinsot (2012). De même, Foroutan et al. (2018) se sont servis du ELES, une des nombreuses variantes du LES, afin d'identifier les concentrations de polluants à proximité d'un bâtiment, et ainsi valider leurs résultats avec ceux d'une soufflerie.

Pour le reste, plusieurs variantes de modèles de turbulence ne sont pas développées dans le présent ouvrage. Ces variantes incluent de façon non limitative les MILES (*Monotone integrated LES*), ILES (*Implicit LES*), VLES (*Very LES*) et DES (*Detached-Eddy Simulation*). La LES et ses variantes ont l'avantage sur les RANS de résoudre des écoulements plus complexes, telle la gamme des écoulements instables. En revanche ils comportent toujours certaines faiblesses qui entravent leur application à des écoulements couramment rencontrés en industrie. Tout d'abord, les résultats obtenus par la LES varient en fonction du maillage utilisé (Pope, 2000). Il est alors nécessaire de connaître les caractéristiques de l'écoulement avant sa résolution, ou d'employer un maillage adaptatif afin de s'assurer de la validité de la solution. D'autre part, il s'avère nécessaire de modéliser l'écoulement près de la paroi pour les écoulements confinés, sans quoi le coût de calcul peut devenir comparable à celui de la DNS. Cette façon de faire introduit des erreurs supplémentaires lors du couplage entre solveur LES et la méthode employée pour modéliser l'écoulement près de la paroi. Ces erreurs se rajoutent à celles déjà présentes dues aux hypothèses d'approximation de l'écoulement près de la paroi (Kawai & Larsson, 2012). La démonstration fut notamment effectuée par Nikitin, Nicoud, Wasistho, Squires et Spalart (2000) qui ont trouvé une erreur de l'ordre de 15% sur le coefficient de friction d'une paroi, largement attribuable à cette erreur de couplage. Quoique très puissante, nous constatons qu'il est encore laborieux d'employer la LES pour résoudre les écoulements couramment rencontrés en industrie. La puissance de calcul requise est certainement moindre que celle de la DNS, mais demeure importante par rapport aux ressources disponibles dans la majorité des firmes d'ingénierie.

Ayant couvert sommairement les approches de turbulence, nous justifions le choix d'un modèle RANS par le fait que ceux-ci sont bien maitrisés pour les applications industrielles. En effet, leur relativement faible coût de calcul par rapport à la précision obtenue offre un compromis acceptable pour la résolution de nombreux écoulements.

2.1.2 Méthode RANS

Les modèles RANS se caractérisent par les relations mathématiques ainsi que les coefficients employés pour fermer le système d'équations de Navier-Stokes. Dans la littérature, ces méthodes sont souvent classifiées comme étant les modèles algébriques, à une demi-équation, à une équation, à deux équations, ainsi que les modèles aux tensions de Reynolds (RSM). Les modèles issus de cette dernière catégorie, contrairement aux autres modèles RANS, ne nécessitent pas l'hypothèse de viscosité turbulente de Boussinesq pour fermer le système d'équations de Navier-Stokes. Étant donnée la variété de méthodes disponibles et du nombre important de modèles qui les composent, nous avons choisi de ne présenter, dans ce qui suit, qu'une courte revue des méthodes les plus répandues. Le lecteur est invité à consulter la revue détaillée des modèles disponibles à ce jour, effectuée par Argyropoulos et Markatos (2015).

Les modèles algébriques (zéro équation) sont parmi les premiers modèles développés et utilisés pour la modélisation d'écoulements turbulents. Ceux-ci se différencient des autres RANS par le fait qu'ils n'emploient que des relations algébriques plutôt que d'avoir recours à des équations différentielles pour calculer les quantités turbulentes. À titre d'exemple, le modèle bien connu de la longueur de mélange de Prandtl (1925) nécessite que la viscosité turbulente soit définie par l'utilisateur plutôt que calculée.

$$\nu_T = l^2 \left| \frac{\partial U_i}{\partial x_j} \right| \tag{2.2}$$

La longueur de mélange l dans l'équation 2.2 est une propriété de l'écoulement qui doit être déterminée empiriquement et non pas une propriété du fluide. Cet exemple montre que ce modèle est tributaire de valeurs empiriques qui ne sont pas disponibles a priori pour des écoulements complexes. Cette caractéristique offre tout de même l'avantage de pouvoir prédire certaines propriétés des écoulements, telle que la séparation, sans avoir recours aux modèles à deux équations (Gatski, Rumsey, Launder, & Sandham, 2002). Cependant, les modèles algébriques sont

aujourd'hui considérés comme étant obsolètes dus aux progrès réalisés sur les autres modèles de turbulence. Actuellement, les modèles algébriques sont surtout employés pour l'initialisation des variables d'un écoulement plutôt que pour la résolution de ce dernier (Hanjalic, 2004).

Contrairement aux modèles algébriques, les modèles à une équation nécessitent une équation de transport afin de définir la quantité turbulente d'un écoulement. Dans la majorité des cas, il s'agit de l'équation de l'énergie cinétique turbulente *k* (Argyropoulos & Markatos, 2015). Tout comme pour les modèles algébriques, ces modèles sont dits « ouverts », car il est toujours nécessaire de définir des relations propres aux caractéristiques physiques de l'écoulement sans qu'elles soient connues a priori. Un des modèles les plus utilisés appartenant à cette catégorie est celui de Spalart, P. et Allmaras (1992). Il est d'ailleurs toujours couramment employé aujourd'hui, notamment dans le domaine de l'aéronautique grâce à sa représentation conforme des écoulements externes (Hanjalic, 2004). Or, les modèles à une équation demeurent inadéquats pour la prédiction des écoulements internes complexes. Pour de tels écoulements, la communauté scientifique s'est plutôt tournée vers les modèles à deux équations, soit des modèles dits « complets ».

Les modèles à deux équations sont à ce jour ceux auxquels l'industrie a le plus recours puisqu'ils offrent un bon compromis entre la rapidité d'exécution et la précision obtenue. Tout comme pour les modèles algébriques et les modèles à une équation, ils nécessitent généralement l'utilisation de l'hypothèse de Boussinesq afin de modéliser le tenseur de Reynolds. En outre, ces modèles sont dits « complets », car une deuxième équation de transport décrivant une propriété turbulente de l'écoulement vient fermer le système d'équations de Navier-Stokes (Wilcox, 1998). Les modèles les mieux documentés et les plus utilisés de cette catégorie sont les modèles k-ϵ et k-ω où l'on calcule respectivement le taux de dissipation de l'énergie cinétique turbulente ϵ et le taux de dissipation turbulente spécifique ω. Selon ce qui a été démontré, le modèle k-ω donne de bons résultats près des parois, et ce même pour des écoulements à forts gradients adverses (Wilcox, 1998). Au contraire, le modèle k-ϵ standard, décrit comme étant à haut nombre de Reynolds (HNR), surestime la viscosité (équation 1.19) près des parois (Pope, 2000).

2.1.3 Modèles à haut et bas nombres de Reynolds

Il a été possible de pallier certaines faiblesses des modèles à HNR dès 1974 en introduisant des lois de parois qui modélisent le comportement physique du fluide dans la couche limite (Launder & Spalding, 1974). Malgré l'amélioration de ces fonctions au fil du temps, elles comportent toujours

certaines lacunes, notamment leurs faibles performances pour la séparation d'écoulements et pour le calcul de transfert de chaleur (Albets-Chico et al., 2008). On reproche aussi aux lois de parois de nécessiter un maillage où les premiers éléments ont une hauteur y+ entre 30 et 300, alors qu'il n'est possible de confirmer le respect de cette plage qu'après la résolution de l'écoulement (Lacasse, Turgeon, & Pelletier, 2004). Comme alternative aux lois de parois, certains chercheurs ont développé en parallèle des modèles à bas nombre de Reynolds (BNR), dont une première version par Jones, W. P. et Launder (1972). Les modèles à BNR consistent à résoudre l'écoulement jusqu'à la surface solide, évitant ainsi les approximations effectuées par les lois de parois. En revanche, il revient tout de même à l'usager de définir la transition où l'écoulement passe de laminaire à turbulent dans la couche limite. D'autre part, les éléments près des parois sont de faibles dimensions, de telle sorte que des erreurs de calcul sont causées par le fort élancement qui en résulte (Mohammadi & Puigt, 2006). Bien que conséquente, cette source d'erreur peut bien être insuffisante pour contrebalancer les gains en précision des modèles à BNR. Pourtant, cette augmentation en précision est accompagnée d'un coût de calcul accru, car les modèles à BNR nécessitent des maillages beaucoup plus fins que les modèles à HNR. Or, récemment ce fait fût minimisé par Spalart, P. R. (2015) dans sa publication sur les sophismes en modélisation de la turbulence. En effet, il ne considère pas l'utilisation des lois de parois comme une économie de calcul à ce point prépondérante. Son argument réside dans le fait que seuls les éléments se trouvant entre $1 < y+ < 50$ n'ont pas à être maillés lors de l'utilisation d'une loi de parois. En supposant les premiers éléments de maillage placés à $y+ = 50$ et un facteur d'accroissement de 1.25, l'auteur avance alors une économie d'une douzaine d'éléments de maillage seulement. Nous sommes toutefois d'avis que l'économie peut être substantielle pour les écoulements 3D, internes et complexes comprenant d'importantes surfaces. De plus, récemment Eça, Pereira et Vaz (2018) ont remis en question la précision des modèles à BNR en comparant plusieurs solveurs RANS sans les lois de parois pour un écoulement sur une plaque plane. Ils ont démontré que même la condition $y+<1$ n'a pas été suffisante pour obtenir une solution précise pour le modèle k-ω SST. En effet, un y+ dix fois plus petit a été requis, afin d'obtenir la même précision que les autres modèles RANS étudiés.

Par conséquent, nous en concluons que malgré leurs lacunes et contraintes, les lois de parois sont préférables aux modèles à BNR pour des applications industrielles. À ce jour, les avantages à

mailler l'écoulement jusqu'à la paroi ne semblent pas suffisamment importants pour justifier le coût de calcul accru des modèles à BNR.

2.2 Vérification et validation

Bien que synonymes dans la vie courante, les termes vérification et validation (V&V) prennent des significations différentes dans le jargon technique du génie par la simulation. La vérification relève des méthodes numériques et pose la question : avons-nous bien résolu les ÉDP utilisées? D'autre part, la validation relève de la physique. Elle pose la question : avons-nous choisi la bonne physique pour simuler l'écoulement qui nous intéresse? C'est dans cette optique que nous cherchons à nous assurer que les équations se comportent comme elles se doivent par la vérification, et que les bonnes équations sont employées par la validation (Tremblay, Etienne, & Pelletier, 2006).

Pour des raisons historiques, la vérification se décline en deux catégories : la vérification de code et la vérification de simulation d'un écoulement réel. La vérification de code sert à établir que l'implémentation du schéma numérique se comporte conformément à la théorie. La vérification procède en évaluant l'erreur $e = u_{ex} - u_n$ où u_{ex} est la solution exacte et u_n est la solution numérique. Pour ce faire, nous avons besoin de la solution exacte d'un écoulement avec laquelle comparer la solution numérique. Cependant, il existe très peu de solutions exactes aux équations de Navier-Stokes, et dans la plupart des cas, elles sont trop simples pour être en mesure de vérifier l'ensemble des termes désirés. Nous privilégions donc la méthode des solutions manufacturées (MMS). La MMS permet de générer une solution exacte, choisie afin d'activer les termes des équations de Navier-Stokes. Le Chapitre 3 présente la méthode des solutions manufacturées en détail.

D'autre part, la vérification de solution de cas réels nous force à utiliser des estimateurs d'erreurs, car il n'y a pas de solution exacte pour ces cas. La vérification de solution sert à estimer (et borner) l'erreur numérique de la solution attribuable à la discrétisation du modèle mathématique. C'est-à-dire que l'on cherche à déterminer si le taux de convergence, préalablement vérifié, se dégrade lorsqu'on applique le code à un écoulement réel. Ce test est effectué à l'aide du GCI (*grid convergence index*), obtenu en comparant plusieurs solutions provenant du raffinement (ou grossissement) des éléments de maillage. Pour plus d'information sur le sujet, le lecteur est invité à consulter le livre de Roache (1998) concernant la théorie du GCI ainsi que des exemples.

Plusieurs articles en matière de V&V ont été publiés ces dernières années, entre autres, pour l'élaboration de fonctions qui améliorent le traitement numérique de parois (Ahsan, 2014; Albets-Chico et al., 2008; Lacombe, 2017) et pour l'implémentation de modèles mathématiques dans un code CFD (Eça, Hoekstra, Hay, & Pelletier, 2007; Nilsson, 2006). Dans le domaine des méthodes de maillages adaptatifs, Lacasse, Turgeon et Pelletier (2001) ont employé la V&V pour faire la démonstration des gains en précision pour des écoulements dans un « turnaround duct ». Il a d'ailleurs été démontré que le modèle k-ϵ, couplé avec une loi de paroi, était en mesure de prédire la séparation d'un écoulement après 4 cycles de raffinement de maillage. Outre ces publications, il existe de nombreux exemples illustrant des résultats crédibles, appuyés par l'application rigoureuse des concepts de V&V.

2.2.1 Applications au modèle k-ϵ standard

Le modèle de turbulence k-ϵ standard est très répandu en industrie et il a été caractérisé à l'aide de nombreuses études par le passé. On compte parmi ces études la vérification de code par la méthode des solutions manufacturées (Eça et al., 2007; Eça, Hoekstra, Hay, & Pelletier, 2010), la vérification de solution par le GCI (Sakri, Ali, & Salim, 2016), ainsi que des analyses de sensibilité sur les solutions (Colin, Etienne, Pelletier, & Borggaard, 2005). La littérature contient aussi de nombreuses études de validation, mettant les performances du modèle k-ϵ standard en comparaison avec des résultats expérimentaux ou avec des prédictions obtenues à l'aide d'autres modèles de turbulence. Parmi ces études, celle de Chaoqun et al. (2014) a servi à démontrer l'invalidité de l'approximation de Boussinesq pour un écoulement passant dans un générateur de vortex. De ce fait, les auteurs quantifient l'erreur qu'engendre l'approximation de Boussinesq lors du calcul d'écoulements de séparation et de rotation pour différents modèles RANS.

Malgré la recherche exhaustive sur le sujet, Lacasse et al. (2004) remarquent d'importants écarts entre les résultats présentés dans la littérature pour des écoulements similaires. Ce manque de fidélité des résultats s'explique en partie par l'utilisation de codes comportant divers types d'erreurs, dont certaines pourraient être liées à des erreurs d'implémentation. Nous constatons alors l'importance de l'application rigoureuse et standardisée des méthodes de V&V pour l'obtention de résultats crédibles et dont la marge d'erreur est connue.

2.2.2 Applications au code OpenFOAM

La vérification et la validation des modèles implémentés dans le logiciel OpenFOAM ont fait l'objet de récentes recherches. En effet, il y a un grand intérêt pour appliquer des méthodes de V&V à ce logiciel libre qui n'offre aucune garantie de performance. La responsabilité revient donc entièrement à l'utilisateur d'être bien au fait de la littérature et de comprendre les limitations impliquées lors de la modélisation d'un écoulement réel.

On trouve plusieurs études de validation dans la littérature où des modèles de turbulence de OpenFOAM sont comparés avec des données expérimentales (Andreini, Cerutti, Facchini, & Mangani, 2008; Fureby et al., 2016), des études DNS (Bohorquez & Parras, 2011) ou avec d'autres logiciels de CFD commerciaux comme ANSYS Fluent (Jones, D. A., Chapuis, Liefvendahl, Norrison, & Widjaja, 2016; Sebastian Muntean, 2009). Très peu de ces ouvrages proposent une démarche rigoureuse de vérification comme celle décrite par Roache (1998). Toutefois, nous avons trouvé une étude de V&V pour des modèles RANS implémentés dans OpenFOAM qui fait exception à cette règle (Robertson, Choudhury, Bhushan, & Walters, 2015). Dans ce cas, la vérification de solution par test de convergence de maillage a été effectuée avant la validation, mais sans vérification préalable du code. Malgré tout, la V&V s'est avéré utile pour identifier les configurations optimales de OpenFOAM pour chaque modèle RANS étudié.

Nous avons trouvé peu d'exemples qui emploient la MMS pour la vérification de code. On cite les travaux de Constant, Favier, Meldi, Meliga et Serre (2017) et de Fisch, R., Franke, Wüchner et Bletzinger (2014) pour des écoulements compressibles et pour des problèmes d'interactions fluide-structure respectivement.

En outre, certains modèles RANS de OpenFOAM ont été validés pour l'étude des profils de sous-marins (Johansson, 2012) où on a noté la bonne performance des modèles k-ϵ, k-ω, k-ω SST et k-ϵ RNG. Un rapport du Defence Science and Technology Group (Jones, D. A. et al., 2016) présente des résultats obtenus avec l'algorithme SIMPLE et le modèle k-ϵ standard, soulignant les bonnes performances observées. L'objectif était de valider les coefficients de friction et de pression comparant les résultats à des mesures expérimentales et aux prédictions obtenues à l'aide de ANSYS Fluent. Les auteurs ont d'ailleurs noté un écart maximal de l'ordre de 4.9% sur le coefficient de friction entre les résultats de OpenFOAM et ceux de Fluent (Jones, D. A. et al., 2016). Nous remarquons toutefois qu'aucune étude de vérification n'a été effectuée au préalable

dans ces travaux. De même, Röhrig, Jakirlić et Tropea (2015) ont comparé les prédictions d'un modèle LES avec celles d'un modèle RANS pour un coude à 90° afin de démontrer la performance des simulations LES. Malheureusement, l'article ne mentionne aucune V&V pour ces modèles, réduisant ainsi la crédibilité des résultats présentés.

D'autre part, la possibilité d'implémenter des codes personnalisés dans OpenFOAM a permis la création de codes spécifiques pour une variété d'applications, comme les turbomachines (Mangani, 2008), les vaisseaux navals (Shan et al., 2011), et les lois de paroi (Liu, S., 2016). Certaines de ces publications traitent aussi des modifications du code source OpenFOAM, pour mieux prédire certains écoulements, telles les instabilités hydrodynamiques (Liu, Q., Gómez, Pérez, & Theofilis, 2016). Ces études seraient d'excellents candidats pour effectuer des études de V&V en amont de leur utilisation pour des applications industrielles. Une procédure a d'ailleurs été mise en place par Persson (2017) afin de vérifier et valider les modèles implémentés dans OpenFOAM. La MMS a été appliquée à plusieurs schémas numériques afin de vérifier des solveurs pour des écoulements compressibles et incompressibles.

Plus récemment, Noriega, Guibault, Reggio et Magnan (2018a) ont effectué la vérification de l'opérateur de diffusion du code OpenFOAM. Ils ont identifié la source d'une perte de convergence d'un schéma théoriquement d'ordre deux. Une solution manufacturée appliquée à l'équation de Poisson pour différents types de maillages a démontré que la non-orthogonalité aux frontières d'un maillage compromet le taux de convergence. Dans un second ouvrage, les auteurs ont proposé et vérifié des solutions restaurant le taux de convergence théorique du schéma étudié (Noriega, Guibault, Reggio, & Magnan, 2018b). Ces deux études démontrent l'intérêt d'effectuer la vérification de code à ce logiciel libre.

Cette revue de littérature nous a permis de recenser plusieurs études sur la V&V de OpenFOAM et le modèle k-ϵ standard. Toutefois, nous n'avons pas trouvé d'exemples appliquant la MMS spécifiquement au code OpenFOAM pour un écoulement turbulent, calculé avec le modèle k-ϵ standard. Considérant l'utilisation très repandue qu'on fait de ce modèle dans l'industrie, il est important de vérifier que le taux convergence observé concorde bien avec le taux théorique.

2.3 Éléments de la problématique

Nous remarquons que l'étape de vérification est souvent négligée dans le processus de la V&V. Ce fait est d'ailleurs fortement critiqué par Roache (1998), qui souligne l'importance de faire la vérification avant la validation et de vérifier le code avant de vérifier la solution. À ce sujet, il existe très peu d'articles détaillant une V&V rigoureuse des modèles trouvés dans OpenFOAM. Pourtant, ce dernier est couramment utilisé dans l'industrie et en recherche, principalement dû à la simplicité de modifier le code source et pour sa capacité de résoudre une vaste gamme d'écoulements complexes. Il est donc étonnant de constater que malgré l'importante librairie de fonctions que comporte OpenFOAM, il existe très peu d'études vérifiant leur implémentation. En effet, Ashton et Skaperdas (2019) notent même que la plus grande faiblesse actuelle de OpenFOAM est le manque de V&V pour son code. Parmi les études existantes, nous constatons que le taux de convergence n'est pas nécessairement respecté lors de l'utilisation du solveur SIMPLE pour des maillages non orthogonaux. Pourtant, il ne fait aucun doute que ce type de maillage est courant pour les applications industrielles, où les géométries sont généralement complexes. Il devient donc pertinent d'étendre la vérification du code OpenFOAM afin d'assurer le respect du taux de convergence théorique de ses schémas pour une gamme variée d'écoulements.

Enfin, notre étude bibliographique révèle que souvent, l'opération de vérification ne précède pas celle de validation comme cela devrait être le cas. Ceci rend impossible de tirer quelque conclusion que ce soit sur les résultats obtenus.

2.4 But et objectifs

Le but de ce mémoire est de faire la vérification de code OpenFOAM, appliquée à un écoulement turbulent, 2D et incompressible. Plutôt que faire la vérification de code directement pour un écoulement turbulent, nous privilégions de procéder étape par étape afin de mieux être en mesure d'identifier les sources d'erreurs dans le code. Pour atteindre notre but, nous avons identifié les objectifs suivants :

1. Appliquer la MMS pour vérifier OpenFOAM sur des écoulements laminaires
2. Appliquer la MMS pour vérifier OpenFOAM sur des écoulements turbulents
3. Appliquer l'extrapolation de Richardson pour les écoulements turbulents

L’atteinte de ces objectifs permet donc de vérifier le code OpenFOAM en employant le modèle de turbulence k-ϵ standard. Le code est vérifié en considérant les conditions limites ainsi que les schémas numériques utilisés lors des simulations. Ce faisant, plusieurs concepts mathématiques se doivent d’être maitrisés et appliqués en amont de la conception d’une solution manufacturée. Tous ces points sont couverts par la méthodologie de vérification présentée au Chapitre 3 avant leur application, présentée au Chapitre 4.

CHAPITRE 3 MÉTHODOLOGIE

Ce chapitre passe en revue les méthodes utilisées dans notre recherche sur la vérification de code et la vérification de simulation pour un écoulement turbulent. Dans un premier temps, nous couvrons les schémas numériques que nous avons retenus parmi ceux qui sont implémentés par défaut dans OpenFOAM. Ensuite, deux méthodes de vérification sont introduites : la méthode des solutions manufacturées pour la vérification de code et l'extrapolation de Richardson pour la vérification de simulation. Finalement, les écoulements utilisés pour faire la vérification de code, ainsi que leurs solutions manufacturées sont présentées en détail dans les sections qui suivent. Les éléments constituant la méthodologie nous permettent de mieux expliquer les écarts observés entre l'aspect mathématique des schémas employés et les résultats numériques calculés par le logiciel de CFD.

Il est à noter que pour ce travail, nous employons la sixième version de OpenFOAM paramétré avec le solveur simpleFoam (basé sur l'algorithme SIMPLE) et le modèle k-ϵ standard. Le taux de convergence observé est comparé à ceux de schémas numériques théoriquement du deuxième ordre pour la vitesse (u et v), la pression (p), l'énergie cinétique de la turbulence (k) et le taux de dissipation de l'énergie cinétique turbulente (ϵ).

3.1 Schémas numériques

Nous présentons une courte description des schémas numériques utilisés dans le cadre de ce travail parmi le registre disponible dans OpenFOAM. Le lecteur pourra ainsi constater que le choix des schémas correspond à nos attentes d'une convergence de deuxième ordre. Alors que seuls quelques schémas sont présentés, nous souhaitons rappeler qu'une liste exhaustive des schémas numériques disponibles est présentée dans le guide officiel de OpenFOAM (Greenshields, 2018). C'est notamment à partir de ce document de référence que nous avons basé les explications qui suivent.

Les schémas numériques décrits dans cette section sont définis dans le fichier *fvSchemes*, qui se trouve lui-même dans le répertoire « system » présenté à la Figure 1.4. Nous pouvons appliquer ces schémas à une variété de termes, comprenant le calcul des dérivées et l'interpolation de valeurs discrètes. Les paragraphes qui suivent sont d'ailleurs divisés selon les catégories qui regroupent ces schémas numériques en fonction des termes qui y sont discrétisés.

3.1.1 Discrétisation de la dérivée temporelle

Il est requis de spécifier un schéma de discrétisation pour la dérivée temporelle pour toutes les études dans OpenFOAM, qu'elles soient transitoires ou en régime permanent. Toutefois, le schéma « steadyState » est le seul schéma disponible pour les écoulements en régime permanent. D'autre part, il existe une variété de schémas pour les études transitoires, servant à résoudre les dérivées $\frac{\partial}{\partial t}$ se trouvant dans les ÉDP.

Le sujet de l'étude est en régime permanent, donc nous ne discuterons pas davantage de la dérivée temporelle. Toutefois, le lecteur intéressé est invité à consulter le guide utilisateur (Greenshields, 2018) pour plus d'informations.

3.1.2 Discrétisation du gradient

L'opérateur du gradient est utilisé dans le calcul des termes dérivés des équations de transport. Il consiste à faire l'interpolation des valeurs partant du centre des éléments de maillage vers ses faces, tel qu'illustré à la Figure 3.1 pour un maillage 2D. Dans cet exemple, la vitesse calculée au point *C* est interpolée sur les faces adjacentes, soient les frontières séparant *OC*, *NC*, *CE* et *CS*. Ces valeurs sont ensuite interpolées sur les centres des éléments *O*, *N*, *E* et *S* respectivement.

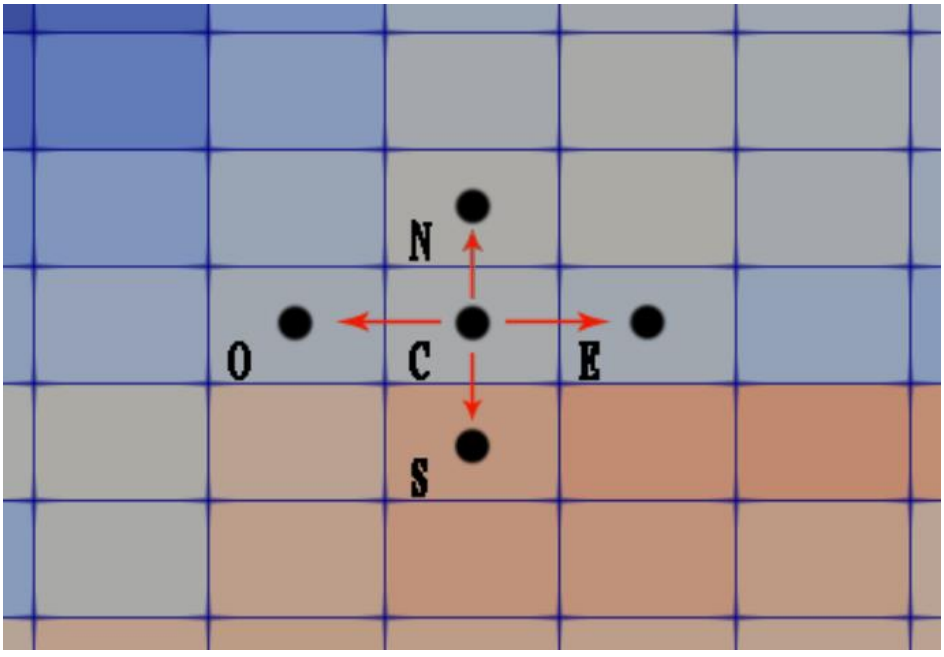

Figure 3.1: Calcul du gradient dans un espace 2D

Pour les maillages composés de quadrilatères (ou d'hexaèdres pour les maillages 3D), on suggère d'employer un schéma du deuxième ordre basé sur le théorème de Gauss, soit le schéma « Gauss linear ». Lorsque les maillages sont composés de triangles (ou tétraèdres) ou lorsque les éléments de maillage sont significativement obliques, on recommande plutôt d'employer la méthode des moindres carrées, soit en employant le schéma « leastSquares ». Pour ce travail, nous employons le schéma Gauss linéaire étant donné que le maillage est composé de quadrilatères orthogonaux.

De plus, des limiteurs sont disponibles afin d'empêcher qu'une valeur extrapolée ne devienne non physique. Les limiteurs servent donc à borner la solution en fonction des valeurs dans les éléments de maillage adjacents. Deux types de limiteurs sont disponibles dans OpenFOAM, soit les limiteurs dits « cellLimited » et ceux dits « faceLimited ».

Quoique ces derniers soient couramment utilisés en pratique, ils ne sont pas employés dans le cadre de ce travail afin de ne pas affecter la précision de la solution. Nous ne détaillons donc pas les limiteurs davantage dans le but de concentrer nos efforts sur les sections du code pouvant affecter la convergence de la solution.

3.1.3 Discrétisation de la dérivée normale à la surface

La composante de la dérivée dans la direction normale à la surface est évaluée à partir des informations aux centres des deux mailles partageant une face donnée. Il est à noter que cette dérivée est nommée « gradient normal à la surface » dans les termes de OpenFOAM. Il est important de noter que les schémas de gradient normal à la surface ne discrétisent que la correction apportée au calcul pour les maillages non orthogonaux. Bien entendu, plus le maillage est non orthogonal, plus la correction sera importante afin de maintenir un calcul théoriquement du deuxième ordre.

La Figure 3.2 illustre la démarche de cette correction. Dans cet exemple, il y a un angle α entre le vecteur joignant les centres des maillages P et N et le vecteur normal à la surface. Un vecteur de correction est donc nécessaire afin de projeter le vecteur du gradient sur $\overrightarrow{PN}$.

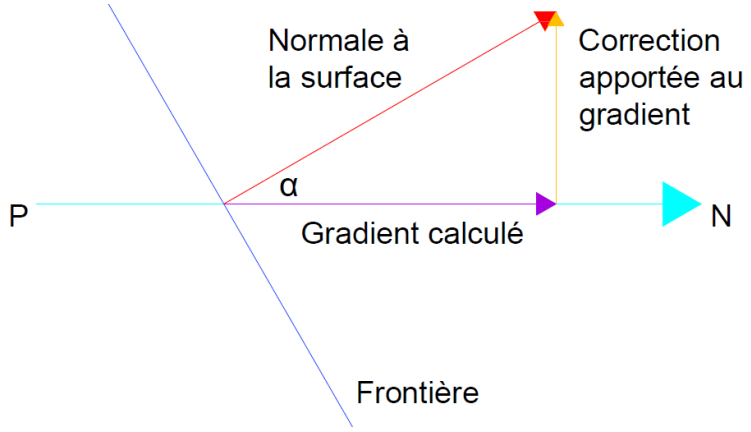

Figure 3.2: Traitement de non-orthogonalité dans OpenFOAM

On rappelle qu'au Chapitre 2 on a montré que la non-orthogonalité du maillage dégrade le taux de convergence du code, même si l'on utilise une correction conçue spécifiquement pour cela (Noriega et al., 2018a). Conséquemment, dans ce travail nous employons un maillage orthogonal ainsi qu'un schéma n'offrant aucune correction afin d'éviter de reproduire des résultats ayant un taux de convergence de 1.

3.1.4 Discrétisation de la divergence

Les schémas servant à la discrétisation de la divergence sont accompagnés d'un schéma d'interpolation. Par exemple, il est commun d'employer le schéma « Gauss linear » afin de calculer la divergence de la vitesse. Dans cet exemple, un schéma de discrétisation basé sur le théorème de Gausse est employé et une interpolation linéaire est effectuée sur les valeurs aux centres des éléments de maillages vers leurs faces.

Par défaut, tous les schémas de discrétisation disponibles dans OpenFOAM sont basés sur le théorème de Gauss. Lors de la configuration du modèle, l'utilisateur n'a qu'à spécifier le schéma d'interpolation à employer. Ce schéma d'interpolation n'affecte pas seulement l'ordre de convergence du calcul, mais aussi les oscillations présentes dans la solution numérique.

Il y a plusieurs schémas d'interpolation de disponibles dans OpenFOAM. Pour ce travail nous nous concentrons sur des schémas du deuxième ordre, car ceux-ci offrent une précision de calcul suffisante pour un coût de calcul raisonnable. En outre, la documentation du logiciel prône l'utilisation d'un schéma linéaire, ou « linear » en anglais, pour les termes non advectifs des équations de Navier Stokes. Le schéma « linearUpwind » est aussi couramment utilisé dans les

exemples fournis avec le logiciel. Selon le code source, ce schéma est dérivé du schéma « upwind » (1er ordre), mais il est combiné avec une correction explicite sur le gradient, permettant au schéma d'être du deuxième ordre.

Cela dit, nous avons employé le schéma « linearUpwind » pour le calcul de la quantité de mouvement dans ce travail. Après comparaison avec le schéma « linear », nous avons remarqué que la convergence résiduelle est plus rapide avec le schéma « linearUpwind », quoique ces deux schémas produisent des résultats similaires.

3.1.5 Discrétisation du laplacien

Ces schémas sont utilisés pour le calcul des termes de diffusion dans les équations de Navier-Stokes. Comme pour les schémas de convection, seul le schéma de Gauss est implémenté dans OpenFOAM et il doit être utilisé de concert avec un schéma d'interpolation et un coefficient de diffusion.

Dans la majorité des cas, le schéma d'interpolation linéaire est employé afin d'avoir un taux de convergence d'ordre deux. Il doit cependant être combiné avec des limiteurs en fonction de la non-orthogonalité du maillage. Cela dit, nous employons le schéma « Gauss linear orthogonal » pour ce travail.

3.1.6 Schémas d'interpolation

Ces schémas servent à évaluer la solution aux faces des volumes à partir des valeurs aux centroïdes des cellules. Ils ont la fonction d'assurer la conservation du flux de vitesses entrant et sortant d'un élément de maillage.

Parmi les schémas disponibles, nous utilisons systématiquement le schéma d'interpolation linéaire, sauf pour certains cas d'exception dont nous ne discuterons pas ici.

Quelques-uns des nombreux schémas disponibles dans OpenFOAM ont été décrits afin de justifier les choix qui ont été faits dans ce travail. Ce choix est simplifié par l'utilisation d'un maillage orthogonal, car aucune étape de correction n'est nécessaire. D'autre part, les schémas de dérivée temporelle, du laplacien et d'interpolation ont une valeur par défaut compte tenu de la nature de l'écoulement étudié. Dans ce travail, nous n'avons eu qu'à spécifier le schéma de convection « linearUpwind ». Nous rappelons toutefois que l'usager peut ajouter des schémas au code existant.

Ces nouveaux schémas pourront ensuite être vérifiés par la même procédure que celle décrite à la section suivante. Dans ce travail, nous ne vérifions que les aspects du code existants à l'aide de la méthode des solutions manufacturées.

3.2 Méthode des solutions manufacturées

La méthode des solutions manufacturées (MMS) est un puissant outil permettant de déterminer si le taux de convergence observé d'un schéma numérique est égal à son taux de convergence théorique, sans avoir recours à des solutions analytiques. Il existe d'ailleurs très peu de solutions analytiques aux équations de Navier-Stokes dans la littérature et elles n'activent pas tous les termes des ÉDP comme le ferait un écoulement complexe. À l'inverse, la MMS est une méthode très simple où l'utilisateur à la flexibilité de choisir les termes des ÉDP qu'il souhaite vérifier. Afin d'en arriver à appliquer cette méthode, nous présentons les concepts et exigences relatives à l'implémentation d'une solution manufacturées dans ce qui suit.

Tout d'abord, nous définissons les équations de Navier-Stokes de manière symbolique à l'aide de la fonction suivante :

$$L(U) = 0 \tag{3.1}$$

où U est une solution inconnue aux équations de Navier-Stokes.

Dans le cas de la MMS, nous choisissons plutôt une solution manufacturée U_m, indépendamment de l'ÉDP. En effet, nous pouvons choisir U_m sans égard aux ÉDP à résoudre ou même sans savoir quels ÉDP nous avons à résoudre. Cette solution manufacturée peut donc servir à vérifier plusieurs solveurs différents (Euler, Navier-Stokes, RANS, etc.).

La solution U_m ne satisfait pas les ÉDP, de sorte que $L(U_m) \neq 0$. À vrai dire, cette équation sera plutôt égale à un terme source Q_m. Nous calculons ce terme source en fonction de la solution manufacturée choisie.

$$Q_m = L(U_m) \tag{3.2}$$

Compte tenu de la complexité des ÉDP, il est recommandé d'employer un logiciel de calcul symbolique comme *Matlab* ou *SageMath* afin de résoudre l'équation 3.2. En effet, il est long et complexe d'effectuer les multiples dérivés et calculs à la main.

Nous ajoutons ensuite le terme source Q_m au code et nous cherchons à obtenir la solution numérique au problème $\tilde{L}$, tel que

$$\tilde{L}(U) = Q_m \tag{3.3}$$

La solution numérique à ce problème est évidemment U_m, la solution manufacturée que nous avons choisie. Il est à noter que les conditions limites que nous employons pour cette solution manufacturée sont, elles aussi, fonction de U_m.

Pour ce travail, nous appliquons une solution manufacturée à l'équation de mouvement avec la méthode SIMPLE (voir section 1.2.1). Il en résulte que l'équation à résoudre s'exprime de la manière suivante :

$$(U_m \cdot \nabla)U_m + \nabla \cdot \tau = fvOptions(U_m) \tag{3.4}$$

où *fvOptions* est un fichier dans OpenFOAM contenant le terme source Q_m.

Or, avant de choisir une solution manufacturée pour la vérification de code, U_m doit tout d'abord respecter certaines exigences. En effet, il importe de tenir compte des conditions suivantes pour éviter d'avoir un taux de convergence observé erroné.

1. La solution manufacturée doit donner une solution non triviale aux ÉDP
2. La solution manufacturée doit être suffisamment régulière (différentiable)

Il n'est toutefois pas exigé que U_m imite un écoulement réel, car dans le cadre d'une vérification de code, nous nous intéressons uniquement à l'aspect mathématique du problème et non à son sens physique. D'autre part, la solution manufacturée n'a pas à respecter l'équation de continuité, mais faire ainsi évite la nécessité d'ajouter un terme source supplémentaire pour compenser l'inéquation qui en résulte.

Une fois la solution manufacturée sélectionnée, nous obtenons la solution numérique de l'écoulement afin d'en extraire l'erreur globale. Pour ce faire, nous employons des normes d'erreur L_2 et L_∞. Celles-ci peuvent s'appliquer aux valeurs discrètes se trouvant aux centres des éléments de maillages.

La norme L_2 de l'erreur tient compte de l'ensemble des nœuds du maillage. Elle est donc calculée comme suit :

$$E_{L_2} = \sqrt{\frac{1}{n}\sum_{k=1}^{n} |\phi_{MMS} - \phi_{Calculé}|^2} \tag{3.5}$$

où *n* est le nombre d'éléments de maillage et ϕ est la solution d'une variable stockée se trouvant au centre d'un volume.

La norme L_∞ permet d'identifier les sources d'erreurs localisées. Cette norme ne tient compte que de l'élément de maillage où l'erreur est la plus importante.

$$E_{L_\infty} = MAX|x_{MMS} - x_{Calculé}| \tag{3.6}$$

La norme L_2 offre l'avantage de prendre en compte tout le domaine de l'écoulement. Toutefois, la norme de l'erreur L_∞ est généralement plus conservatrice, comme nous pourrons le constater au Chapitre 4.

À l'aide de l'erreur globale de la solution, il est maintenant possible de trouver le taux de convergence observé ($\hat{p}$) du code. En effet, ce dernier apparait lorsqu'on effectue le développement en série de l'erreur due à la discrétisation (Roache, 1998). Toutefois, cette équation n'est vraie que si le code vérifié est « consistant », soit qu'il tend vers une solution unique alors que la dimension des éléments de maillage diminue.

$$E = f(\Delta) - f^{exacte} - C\Delta^{\hat{p}} + HOT \tag{3.7}$$

Ici, Δ est la dimension d'un élément de maillage, *C* est un coefficient inconnu et *HOT* (*Higher order terms*) sont les termes d'ordre supérieur. Ces derniers sont ensuite négligés dans le calcul, car nous supposons que l'erreur tend asymptotiquement vers 0.

La constante *C* peut ensuite être éliminé de l'équation 3.7 en effectuant le calcul sur deux maillages de taille différente. Après simplification, nous pouvons isoler le taux de convergence tel que :

$$\hat{p}_i = \frac{ln\left(\frac{E_{i+1}}{E_i}\right)}{ln(r)} \tag{3.8}$$

Où r est le facteur de raffinement de maillage et E_i est la norme de l'erreur (L_∞ ou L_2) pour le maillage i. Nous obtenons le taux de convergence observé $\tilde{p}$ du code pour une solution passant du maillage i vers le maillage raffiné i+1. En calculant plusieurs valeurs de $\tilde{p}_i$, il est possible de trouver la zone asymptotique du taux de convergence observé du code.

Ces étapes que nous avons décrites pour l'application de la MMS peuvent être résumées à l'aide de la procédure élaborée par Roy (2005).

1. choisir la forme des équations principales;
2. choisir la forme de la solution manufacturée;
3. dériver les équations principales modifiées;
4. résoudre la forme discrète des équations principales sur plusieurs maillages;
5. évaluer l'erreur de discrétisation globale dans la solution numérique;
6. évaluer le taux de convergence et le comparer avec le taux de convergence théorique du schéma de discrétisation employé.

À l'aide de la MMS, nous sommes donc en mesure d'évaluer le taux de convergence du code avec une grande précision, compte tenu de l'importante sensibilité de la méthode. Cependant, nous remarquons que l'application de la MMS implique la modification du code source du logiciel à moins que les programmeurs aient prévu des dispositifs à cet effet. Alors que ce n'est assurément pas le cas de tous les codes de CFD, cette possibilité s'offre à nous pour le code OpenFOAM.

3.3 L'extrapolation de Richardson

L'extrapolation de Richardson est le deuxième outil de vérification présenté dans ce travail. Il sert à estimer l'erreur de discrétisation d'une solution réelle pour laquelle nous n'avons pas de solution exacte. Contrairement à la MMS, cette méthode ne requiert aucune modification du code source du logiciel. On peut donc l'appliquer à n'importe quel code sans avoir à manipuler des termes sources et des conditions limites variables. Ces principes de fonctionnement sont présentés dans ce qui suit.

Tout d'abord, l'extrapolation de Richardson nécessite le respect de certains critères pour sa mise en application. Par exemple, l'erreur doit tendre vers 0 asymptotiquement et l'écoulement ne peut

contenir des discontinuités comme des chocs (il doit être différentiable). Comme ces critères sont aussi valables pour l'application de la MMS, nous les respectons sans difficulté.

Le point de départ de l'extrapolation de Richardson est similaire à celui de la MMS. À partir de l'équation 3.7, l'erreur est estimée en négligeant les termes d'ordre supérieur.

$$\tilde{E} = C\Delta^{\hat{p}_{th}} \tag{3.9}$$

Ici $\hat{p}_{th}$ est le taux de convergence théorique.

Tout comme pour la MMS, nous éliminons la constante *C* de l'équation 3.9 en utilisant des raffinements de maillages *r*, tel que :

$$r = \frac{h_2}{h_1} \tag{3.10}$$

Le raffinement *r* peut être constant ou non. Toutefois, un minimum de deux maillages est requis pour estimer l'erreur, car celle-ci est inconnue. Nous estimons donc l'erreur avec :

$$\tilde{E} = \frac{x_{r2} - x_{r1}}{r^{\hat{p}_{th}} - 1} \tag{3.11}$$

où x_{r2} est une valeur sur un maillage raffiné par rapport à celui de la valeur x_{r1}. L'erreur est donc estimée pour la valeur x_{r2} afin d'en extrapoler la solution exacte.

L'extrapolation de Richardson nous permet d'obtenir une solution équivalente à celle d'un maillage plus raffiné que ceux employés dans calcul des solutions x_{r1} et x_{r2}. La solution exacte estimée par cette méthode s'exprime donc comme :

$$x_{h=0} \cong x_1 + \frac{x_{r2} - x_{r1}}{r^{\hat{p}_{th}} - 1} \tag{3.12}$$

Notons que dans le cadre de ce travail, nous ne calculons pas la solution extrapolée comme montrée à l'équation 3.12. En effet, nous cherchons plutôt à obtenir l'estimation de l'erreur telle que présentée à l'équation 3.11.

À l'aide de cette erreur estimée, il nous est possible de calculer le taux de convergence observé, comme pour la MMS à l'équation 3.8. Les valeurs estimées de l'erreur et du taux de convergence

peuvent ensuite être comparées à celles obtenues par la MMS afin de déceler les erreurs d'implémentation dans le logiciel.

En appliquant l'extrapolation de Richardson, nous pouvons aussi évaluer les caractéristiques de convergence de l'erreur afin de nous assurer que le maillage se trouve dans la zone asymptotique. Pour ce faire, on définit le ratio de convergence comme étant :

$$R = \frac{x_{r3} - x_{r2}}{x_{r2} - x_{r1}} \tag{3.13}$$

La valeur de *R* est ensuite utilisée avec le Tableau 3.1 pour identifier le type de convergence de l'erreur estimé. Pour les écoulements étudiés, nous nous attendons à une convergence monotone de l'erreur pour toute la zone asymptotique.

Tableau 3.1: Types de convergences

Valeur de R	**Type de convergence observé**
0 < R < 1	Convergence monotone
1 < R	Divergence monotone
-1 < R < 0	Convergence oscillatoire
R < -1	Divergence oscillatoire

En plus de sa simplicité d'implémentation, l'extrapolation de Richardson est aussi très versatile. En effet, elle peut s'appliquer à n'importe quel code pour estimer l'erreur de solution. Nous n'avons qu'à choisir une quantité d'intérêt de l'écoulement pour y évaluer l'évolution de l'erreur. Ces caractéristiques peuvent être, par exemple, la perte de charge sur une paroi, ou le déplacement d'une composante solide pour un écoulement d'interaction fluide-structure. Lorsque la MMS est utilisée, nous avons l'occasion de nous servir de l'extrapolation de Richardson dans le but de contrevérifier le calcul de l'erreur et l'ordre de convergence observé.

3.4 Cas tests pour la vérification de code

La vérification du code OpenFOAM a été effectuée pour des écoulements de complexité croissante afin d'identifier toutes les sources d'erreurs potentielles dans les fonctions ou conditions limites utilisées. Tout d'abord, nous effectuons la résolution d'un écoulement de Poiseuille, où nous disposons d'une solution analytique pour un écoulement plan 2D. Ensuite, des solutions manufacturées sont ajoutées au code pour un écoulement de type couche limite 1D et un écoulement libre 2D laminaire. Ce après quoi nous vérifions le code pour un écoulement turbulent

en réutilisant la solution manufacturée de l'écoulement libre 2D. Finalement, nous ajoutons une dernière SM qui imite une couche cisaillée. Chacune de ces solutions manufacturées est présentée dans les sous-sections qui suivent avec les conditions limites employées lors de leur résolution.

3.4.1 Écoulement de Poiseuille

Un premier test est effectué pour l'écoulement de Poiseuille, soit un écoulement très simple dont la solution analytique se trouve facilement dans la littérature. Ce test a comme objectif de vérifier les conditions limites employées afin de détecter toute source d'erreur pouvant influencer le taux de convergence observé. Étant donné que nous étudions les écoulements dans l'espace 2D, nous prenons comme solution l'équation décrivant l'écoulement théorique entre deux plaques planes.

$$u = -\frac{1}{2\mu}\frac{\partial p}{\partial x} y(h-y) \qquad v = 0 \tag{3.14}$$

L'équation 3.14 s'applique pour un domaine où y varie de 0 à h. Afin de compléter l'équation, les valeurs des paramètres suivantes ont été retenues :

$$\rho = 1 \qquad \mu = 1 \qquad \frac{\partial p}{\partial x} = -1 \qquad h = 1 \tag{3.15}$$

Il en résulte un écoulement unidirectionnel laminaire où les isolignes de pression sont verticales et varient entre 1 et 0. La vitesse et la pression exactes sont représentées aux Figures 3.3 et 3.4 respectivement.

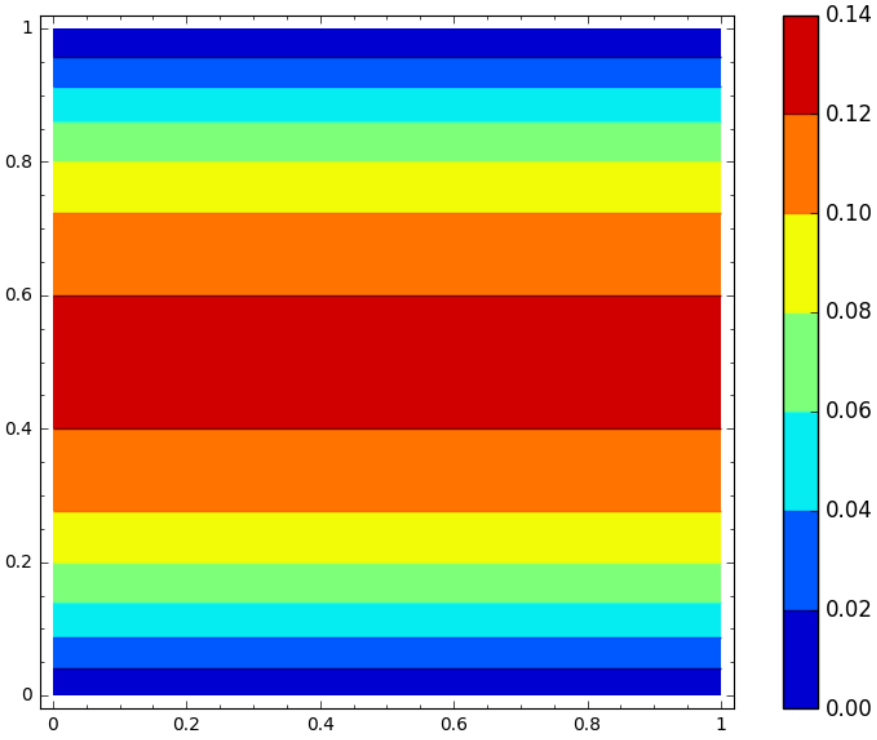

Figure 3.3: Profil de vitesse *u* pour un écoulement de Poiseuille

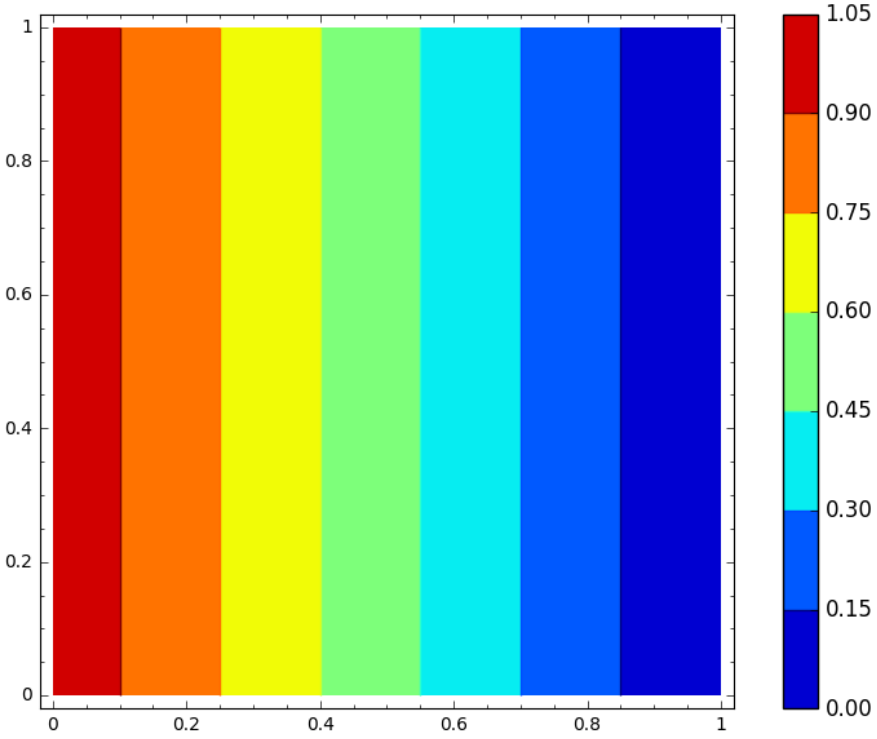

Figure 3.4: Profil de la pression *p* pour un écoulement de Poiseuille

Nous avons prévu deux tests pour cet écoulement. Pour le premier test, la pression est nulle sur toute la frontière en sortie (p_1). Pour le deuxième test, nous imposons une pression nulle au point [1 0] et une condition de Neumann sur la pression (p_2) à la frontière en sortie.

La condition de Neumann est imposée en utilisant la condition limite *fixedGradient*. Cette condition limite fixe la valeur de la dérivée de ϕ tel que :

$$fixedGradient = \frac{\partial \phi}{\partial n} \tag{3.16}$$

Nous avons donc les conditions limites suivantes de la pression pour l'écoulement de Poiseuille :

$$\left.\frac{\partial p}{\partial n}\right|_{x=0} = 1$$
$$\left.\frac{\partial p}{\partial n}\right|_{x=1} = -1 \tag{3.17}$$

Pour la deuxième configuration, nous changeons la condition limite de Neumann pour la vitesse (U_1) à une condition de Dirichlet (U_2). Les conditions limites employées pour cet écoulement sont présentées au Tableau 3.2.

Tableau 3.2: Conditions limites pour l'écoulement de Poiseuille

Surface	**Variable**	**Condition limite**	**Valeur**
Entrée	U	Dirichlet	SM
	p	Neumann	1.0
Sortie	U_1	Neumann	zeroGradient
	U_2	Dirichlet	SM
	p_1	Dirichlet	0
	p_2	Neumann	-1.0
Paroi du bas	U	Dirichlet	u & v = 0
	p	Neumann	zeroGradient
Paroi du haut	U	Dirichlet	u & v = 0
	p	Neumann	zeroGradient

Il est à noter que nous n'obtiendrons pas la solution exacte pour cet écoulement comme ce serait le cas pour la méthode des éléments finis. En effet, la méthode des volumes finis discrétise les flux entrants et sortants des volumes de contrôles ce qui ne permet pas à l'algorithme de résoudre exactement l'équation analytique.

Quoique ce cas est très simple par rapport aux prochains tests, il est tout de même pertinent de vérifier que le taux de convergence théorique est atteint avant l'utilisation d'une solution manufacturée. Il nous est aussi possible de tester certaines variantes de conditions limites sur la pression avant d'en faire l'application aux autres cas étudiés. Après le succès de ce test, il convient alors de vérifier le code à l'aide d'une première solution manufacturée.

3.4.2 Écoulement de type couche limite

Pour ce test, nous effectuons à nouveau la vérification du code pour un écoulement 1D, mais cette fois à l'aide d'une solution manufacturée. Dans ce cas, notre objectif est de confirmer l'implémentation adéquate du terme source résultant de la MMS.

Il est important de remarquer que des connaissances minimales en programmation sont requises afin d'ajouter un terme source au code OpenFOAM. Deux options s'offrent à l'utilisateur :

- modifier le code source du solveur *simpleFoam* et y ajouter les termes souhaités;
- ajouter les termes sources au fichier nommé *fvOptions*.

Nous avons constaté que les deux options proposées requirent une bonne compréhension du fonctionnement de OpenFOAM ainsi que des connaissances minimales en programmation. Toutefois, l'option utilisant *fvOptions* est plus simple, car elle évite à l'utilisateur de modifier et recompiler le code source. Le code du fichier *fvOptions* servant pour l'écoulement de type couche limite a d'ailleurs été ajouté à l'Annexe A comme exemple.

La solution manufacturée utilisée imite une couche limite où la vitesse est nulle à la paroi et augmente rapidement vers une vitesse U_∞=1. Comme pour l'écoulement de Poiseuille, nous définissons une pression variant dans la direction des x uniquement. Cette solution manufacturée s'écrit :

$$u_m = \frac{e^{-\alpha y} - 1}{e^{-\alpha} - 1}$$
$$v_m = 0 \qquad (3.18)$$
$$p_m = 0.2 - x$$

Les dérivés de la vitesse et de la pression servant aux conditions limites de Neumann sont calculés de la manière suivante :

$$\left.\frac{\partial p}{\partial n}\right|_{x=0} = 1$$

$$\frac{\partial p}{\partial y} = 0 \qquad (3.19)$$

$$\frac{\partial u}{\partial x} = 0$$

De plus, nous définissons les propriétés suivantes pour cet écoulement :

$$\rho = 1 \qquad \mu = 1 \qquad \alpha = 10 \qquad (3.20)$$

Notons que la variable α influence l'épaisseur de la couche limite, plus la valeur de α est élevée, plus la variation de la vitesse sera prononcée.

Ce cas simple respecte la conservation de la masse de l'écoulement tout en modélisant une couche limite suffisamment mince pour montrer une variation prononcée de la vitesse dans le domaine. Le profil de la vitesse *u* est illustré à la Figure 3.5 alors que la pression *p* se trouve à la Figure 3.6.

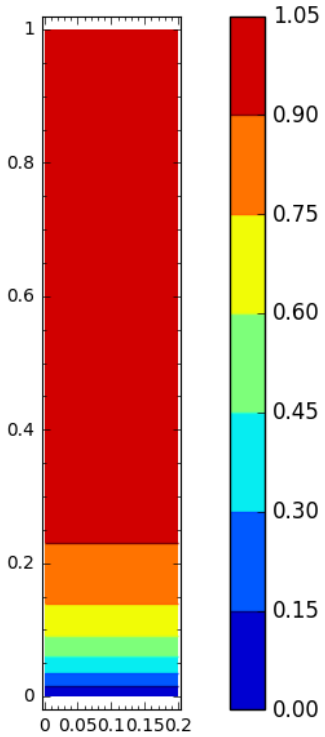

Figure 3.5: Profil de la vitesse u_m pour un écoulement de type couche limite

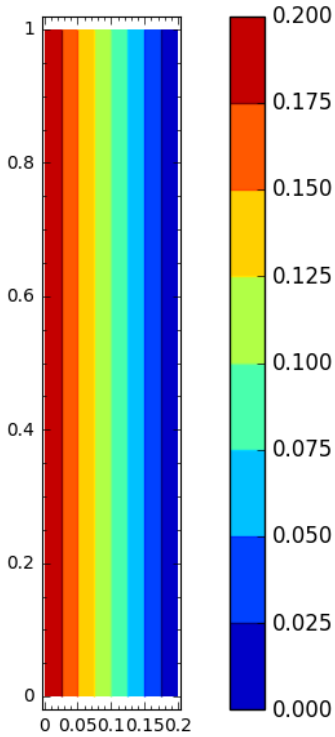

Figure 3.6: Profil de la pression p_m pour un écoulement de type couche limite

Les conditions limites pour cet écoulement, présentées au Tableau 3.3, sont très similaires à celles de l'écoulement de Poiseuille. Toutefois, une condition de Dirichlet variable est imposée en entrée du domaine afin de satisfaire la solution manufacturée.

Tableau 3.3: Conditions limites employées pour l'écoulement de type couche limite

Surface	**Variable**	**Condition limite**	**Valeur**
Entrée	*U*	Dirichlet	SM
	p	Neumann	1.0
Sortie	*U*	Neumann	zeroGradient
	p	Dirichlet	0
Paroi du bas	*U*	Dirichlet	u & v = 0
	p	Neumann	zeroGradient
Frontière du haut	*U*	Dirichlet	u & v = 1
	p	Neumann	zeroGradient

Ce test complète la vérification des écoulements 1D laminaires. Nous nous attendons à ce que le taux de convergence $O(h^2)$ soit respecté pour ces cas, car la vérification de code a déjà été effectuée

pour des écoulements laminaires incompressibles. Par la suite, nous passons à l'implémentation d'une solution manufacturée 2D.

3.4.3 Écoulement libre 2D laminaire

Nous vérifions maintenant le code à l'aide d'une solution manufacturée dont la vitesse a une composante en *x* et en *y*. Le but de ce test est de vérifier le code pour un écoulement laminaire dont la SM pourra être reprise pour la vérification d'un écoulement turbulent.

La solution manufacturée de l'écoulement libre 2D laminaire est présentée à l'équation 3.21. La SM de la pression est inspirée du travail de Furbo (2010) alors que nous avons développé celles des vitesses en fonction de nos besoins.

$$u_m = \frac{\pi x + sin(\pi x) + sin(\pi y)}{4}$$

$$v_m = -\frac{\pi y cos(\pi x) - \pi y}{4} \tag{3.21}$$

$$p_m = 50\big((x-1)x(y-1)y\big)^2 cos(0.7\pi y) + 2$$

Les propriétés de l'écoulement suivantes ont été employées pour ce cas test :

$$\rho = 1 \qquad \mu = 0.1 \tag{3.22}$$

Le choix des solutions manufacturées de *u*, *v* et *p* dépendent de la possibilité d'imposer des conditions limites à gradients nuls (*zeroGradient*) aux frontières requérant des conditions limites de Neumann. Ce choix a été fait, car des irrégularités dans la solution ont été relevées lorsque des gradients non nuls ont été appliqués aux frontières. Ce constat fait écho aux expériences similaires de Elsworth (2014) et Fisch, R. F., Jörg; Wüchner, Roland; Bletzinger, Kai-Uwe (2012). En effet, ceux-ci ont remarqué la présence d'erreurs dans la solution en lien avec les conditions limites de Neumann non nul. Nous avons donc fait le choix d'éviter ces erreurs en élaborant des solutions manufacturées où il est permis d'imposer des gradients nuls sur les frontières.

Concernant les MS définies pour ce test, elles sont présentées aux Figures 3.7, 3.8 et 3.9 pour *u*, *v* et *p* respectivement.

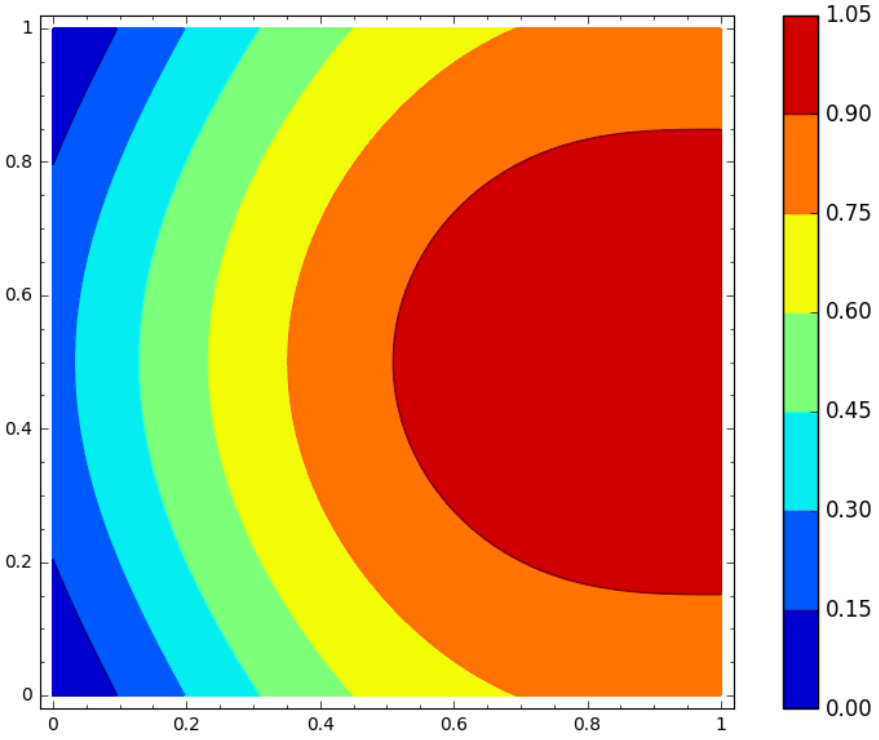

Figure 3.7: Profil de la vitesse u_m pour un écoulement libre 2D

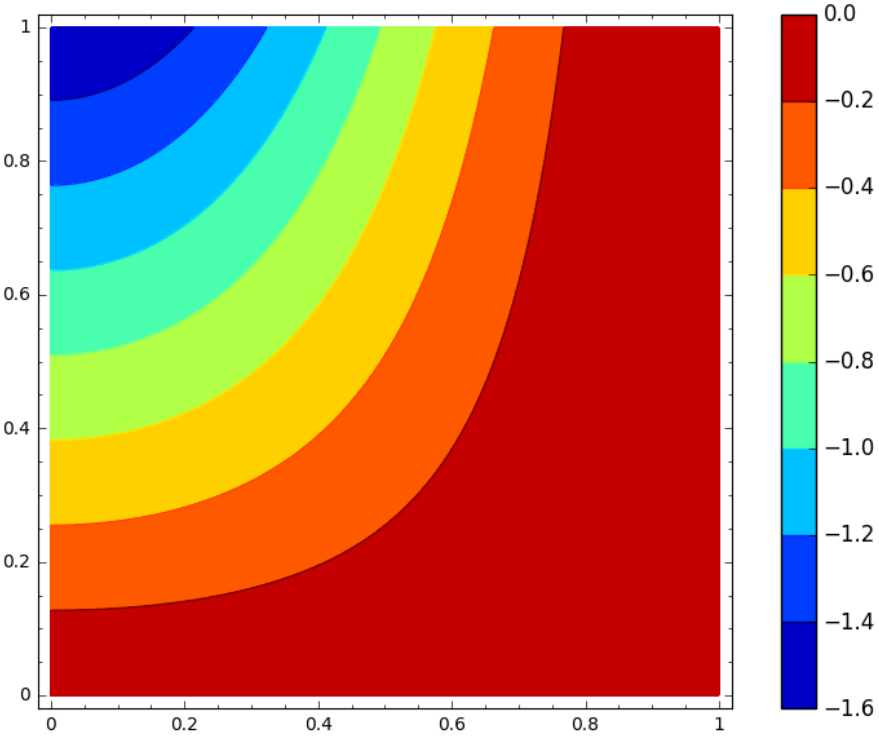

Figure 3.8: Profil de la vitesse v_m pour un écoulement libre 2D

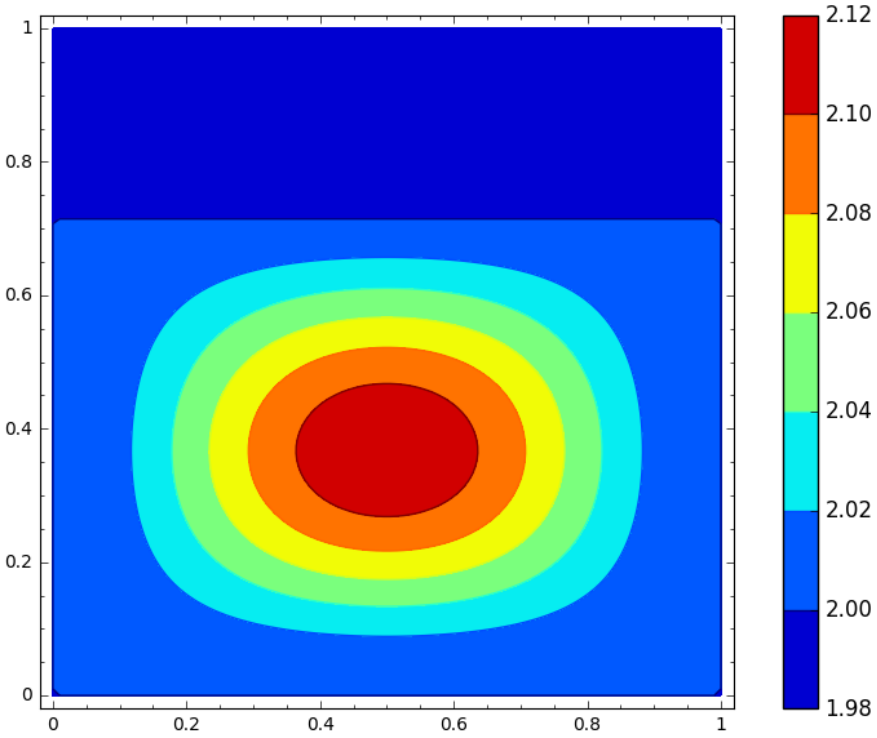

Figure 3.9: Profil de la pression p_m pour un écoulement libre 2D

Les dérivés de la vitesse et de la pression de l'équation 3.21, employés comme conditions limites de Neumann, sont les suivantes :

$$\frac{\partial u}{\partial x} = \frac{\pi + (\pi cos(\pi x))}{4}$$

$$\frac{\partial v}{\partial x} = \frac{y\pi^2 sin(\pi x)}{4}$$

$$\begin{aligned}\frac{\partial p}{\partial x} &= 50x^2y^2cos\left(\frac{7\pi y}{10}\right)(2x-2)(y-1)^2 \\ &\quad + 100xy^2cos\left(\frac{7\pi y}{10}\right)(x-1)^2(y-1)^2\end{aligned} \tag{3.23}$$

$$\begin{aligned}\frac{\partial p}{\partial y} &= 50x^2y^2cos\left(\frac{7\pi y}{10}\right)(2y-2)(x-1)^2 \\ &\quad + 100x^2ycos\left(\frac{7\pi y}{10}\right)(x-1)^2(y-1)^2 \\ &\quad - 35x^2y^2\pi sin\left(\frac{7\pi y}{10}\right)(x-1)^2(y-1)^2\end{aligned}$$

Nous substituons ensuite la position en x et y des frontières de l'écoulement pour en déduire les conditions limites.

$$\left.\frac{\partial u}{\partial x}\right|_{x=1} = 0$$
$$\left.\frac{\partial v}{\partial x}\right|_{x=1} = 0$$
$$\left.\frac{\partial p}{\partial x}\right|_{x=0} = 0 \qquad (3.24)$$
$$\left.\frac{\partial p}{\partial y}\right|_{y=0} = 0$$
$$\left.\frac{\partial p}{\partial y}\right|_{y=1} = 0$$

Le résumé des conditions limites employées pour ce cas test est présenté au Tableau 3.4. Comme prévu lors de la conception de la solution manufacturée, chaque condition de Neumann est un gradient nul (*zeroGradient*).

Tableau 3.4: Conditions limites employées pour l'écoulement libre 2D

Surface	Variable	Condition limite	Valeur
Frontière gauche	U	Dirichlet	SM
	p	Neumann	zeroGradient
Frontière droite	U	Neumann	zeroGradient
	p	Dirichlet	SM
Frontière du bas	U	Dirichlet	SM
	p	Neumann	zeroGradient
Frontière du haut	U	Dirichlet	SM
	p	Neumann	zeroGradient

Ayant évité les sources potentielles de détérioration du taux de convergence, nous nous attendons à atteindre l'ordre théorique des schémas numériques. Une fois le code OpenFOAM vérifié pour cette SM, nous procédons ensuite à la vérification du code pour un écoulement turbulent.

3.4.4 Écoulement libre 2D turbulent

Ce test a comme objectif de vérifier le code OpenFOAM pour un écoulement turbulent 2D. Nous utilisons les solutions manufacturées de *u*, *v* et *p* présentées à l'équation 3.21 pour un écoulement

laminaire. En effet, nous souhaitons modifier les paramètres de l'écoulement le moins possible lorsque nous passons de laminaire à turbulent afin de pouvoir détecter plus facilement les sources d'erreurs présentes dans le code.

Nous n'avons qu'à incorporer les solutions manufacturées des variables du modèle k-ϵ standard afin de compléter la modélisation de la turbulence.

$$k_m = \frac{1}{2} e^{\left(\left(\frac{1}{3}y^3 - \frac{1}{2}y^2\right)\left(\frac{1}{2}x^2 - x\right)\right)}$$
$$\epsilon_m = e^{\left(\left(\frac{1}{3}y^3 - \frac{1}{2}y^2\right)\left(\frac{1}{2}x^2 - x\right)\right)} \tag{3.25}$$

Comme pour les vitesses et la pression, les solutions de k et ϵ sont définies de sorte à avoir un gradient nul sur les frontières. Ceci n'est fait que par principe de précaution étant donné que nous n'avons pas trouvé d'articles sur le sujet détaillant une possible source erreur.

La viscosité et la masse volumique définies pour cet écoulement sont les suivantes :

$$\rho = 1 \qquad \mu = 10^{-6} \tag{3.26}$$

S'ajoutant ainsi aux profils illustrés dans le test précédent, les profils des variables k et ϵ sont illustrées aux Figures 3.10 et 3.11.

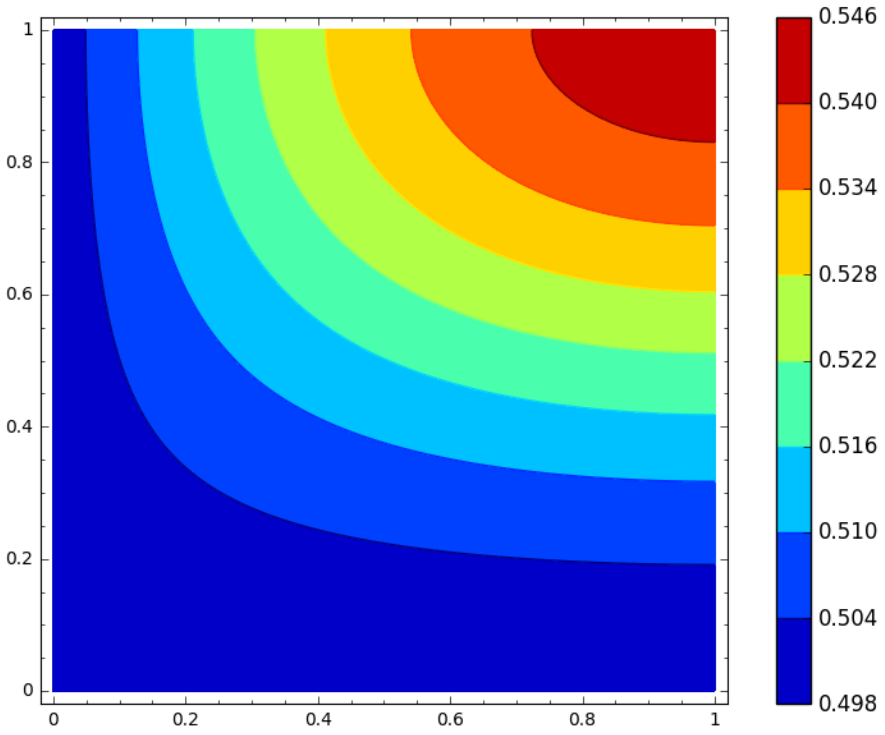

Figure 3.10: Profil de l'énergie cinétique turbulente k_m pour un écoulement libre 2D turbulent

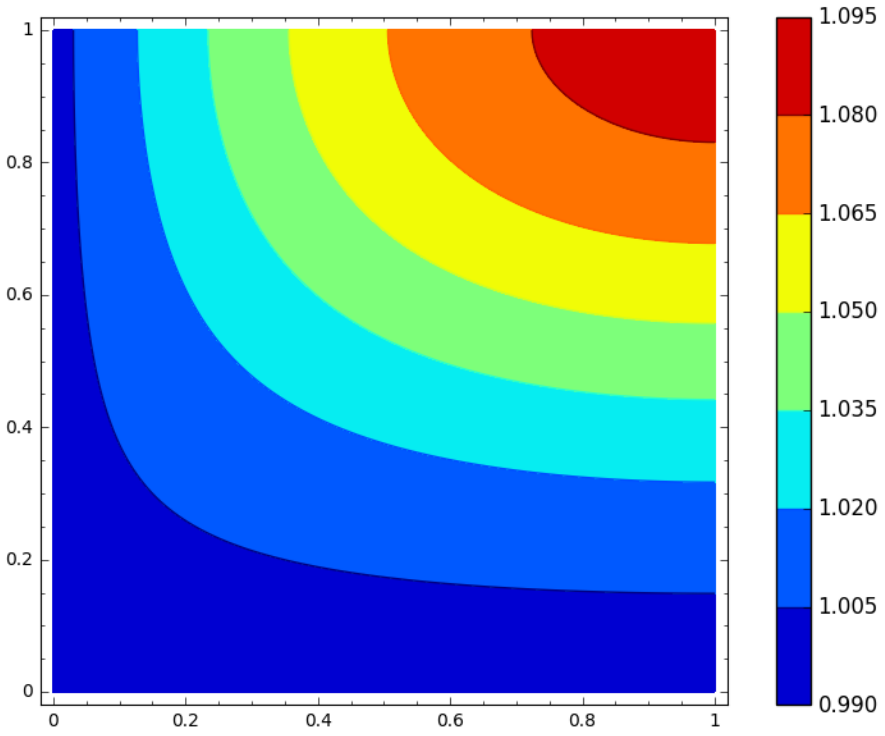

Figure 3.11: Profil du taux de dissipation de l'énergie cinétique turbulente ϵ_m pour un écoulement libre 2D turbulent

Pour ce qui est des conditions limites, celles de *u*, *v* et *p* peuvent être consultés au Tableau 3.4, car elles sont identiques à celles utilisées pour l'écoulement libre 2D laminaire. Dans le cas présent, nous n'avons qu'à ajouter des conditions limites de Dirichlet pour k et ϵ sur l'ensemble des frontières.

Cette MS nous permet de vérifier le code pour un écoulement turbulent alors que toutes les conditions de Neumann sont de type *zeroGradient*. Selon notre expérience, cette condition est l'une des plus fréquemment employées dans les exemples de OpenFOAM. C'est pourquoi nous jugeons nécessaire d'en faire la vérification dans un premier temps. Après avoir mis le code à l'épreuve avec cette solution manufacturée, nous introduisons une MS qui, cette fois, incorpore des conditions de Neumann variables.

3.4.5 Écoulement turbulent de type couche cisaillé

Pour ce dernier test d'un écoulement turbulent, nous employons une solution manufacturée qui imite un écoulement cisaillé (Ilinca, F., Pelletier, & Garon, 1994). Nous disposons alors d'un écoulement manufacturé plus réaliste, mais aussi plus complexe à résoudre, car il fait intervenir la fonction d'erreur (*erf*). De plus, les conditions limites de Neumann pour les vitesses ne peuvent être nulles contrairement au cas précédent. Cette SM nous permet donc de tester le code pour de nouvelles conditions limites tout en permettant d'identifier la résolution nécessaire pour que le code capte la forte variation de la fonction d'erreur. La solution manufacturée est la suivante :

$$
\begin{aligned}
u_m &= U_1\left[\left(\frac{1+r}{2}\right)-\left(\frac{1-r}{2}\right)erf\left(\frac{\sigma y}{x}\right)\right] \\
v_m &= U_1\left[\left(\frac{1-r}{2}\right)\left(\frac{1}{\sigma\sqrt{\pi}}\right)e^{-\left(\frac{\sigma y}{x}\right)^2}\right] \\
p_m &= 0 \\
k_m &= k_0\left[C_k + e^{-\left(\frac{\sigma y}{x}\right)^2}\right] \\
\epsilon_m &= \frac{\epsilon_0}{x}\left[C_k + e^{-\left(\frac{\sigma y}{x}\right)^2}\right]
\end{aligned}
\tag{3.27}
$$

avec

$$k_0 = \frac{343}{75000} U_1 (1 - r) \left(\frac{\sigma}{\sqrt{\pi}} \right)$$
$$\epsilon_0 = \frac{343}{225000} C_\mu {U_1}^3 (1 - r)^2 \left(\frac{\sigma^2}{\pi} \right) \quad (3.28)$$

Les coefficients de cette SM sont présentés au Tableau 3.5. Certaines des constantes sont spécifiques au modèle de turbulence utilisé, mais elles sont tout de même reproduites ici afin de faciliter la compréhension de la SM.

Tableau 3.5: Coefficients pour la solution manufacturée de l'écoulement cisaillé

Coefficients	**Valeur**
U_1	1
C_μ	0.09
C_k	0.1
r	0
σ	13.5

La viscosité et la masse volumique définies pour cet écoulement sont définies à l'équation 3.29.

$$\rho = 1 \qquad \mu = 10^{-4} \quad (3.29)$$

Les profils de vitesses et de pression résultant des variables décrites à l'équation 3.27 sont illustrés aux figures suivantes. Nous constatons que, quoique non requis, cet écoulement comporte un certain réalisme physique. Bien entendu cela n'offre aucun avantage sur le plan mathématique, mais permet d'assurer que les termes activés dans l'équation de Navier-Stokes le sont d'une manière proportionnelle à la physique.

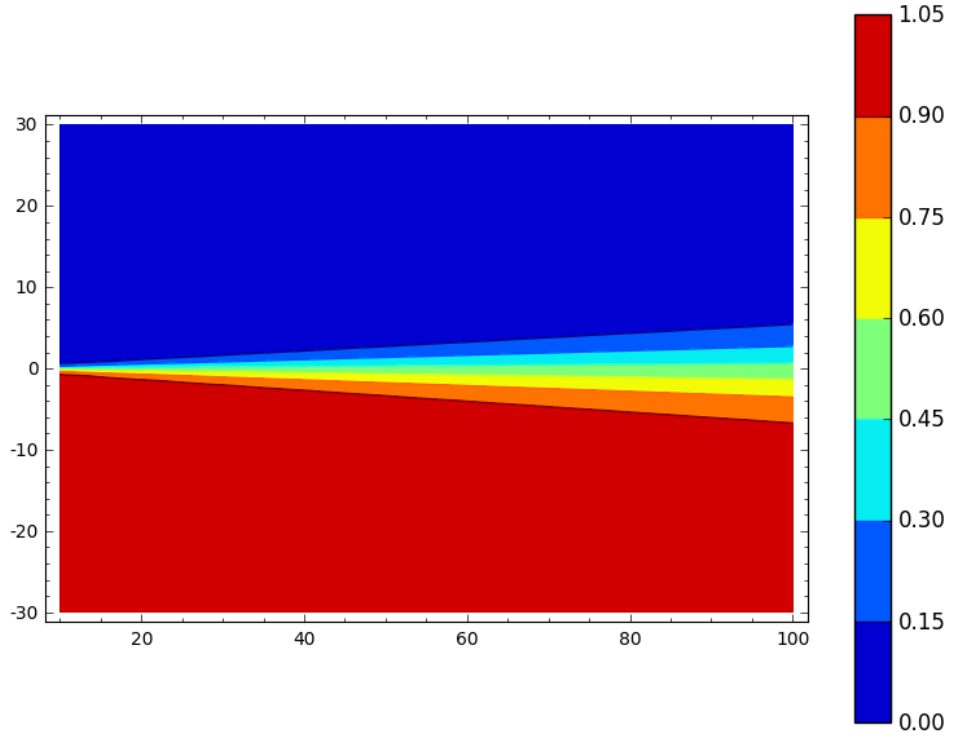

Figure 3.12: Profil de la vitesse u_m pour un écoulement cisaillé

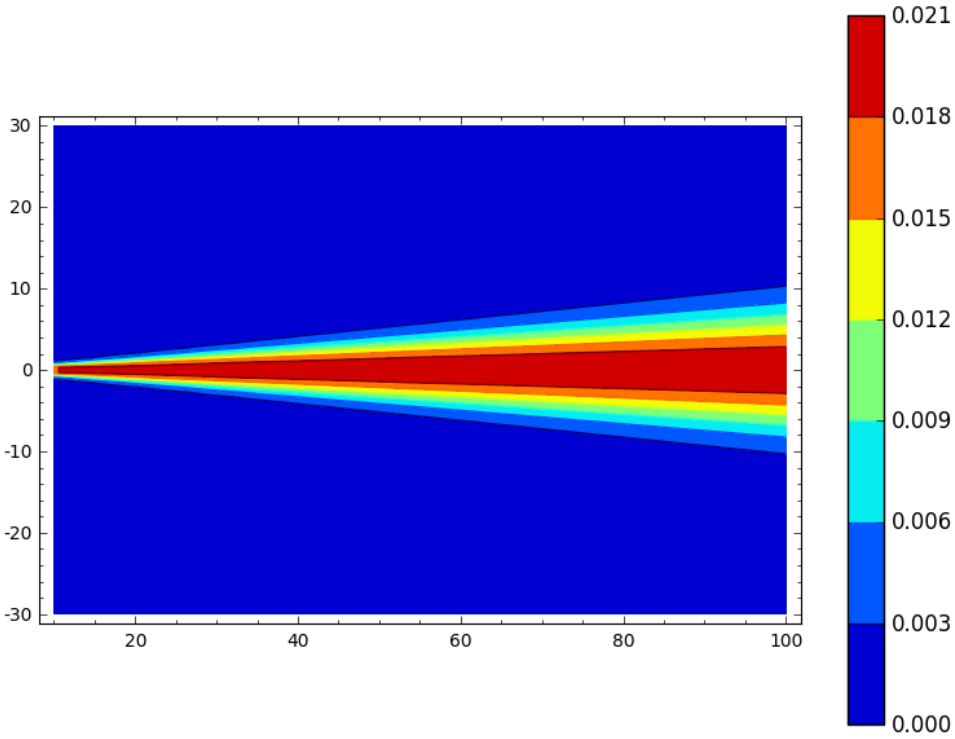

Figure 3.13: Profil de la vitesse v_m pour un écoulement cisaillé

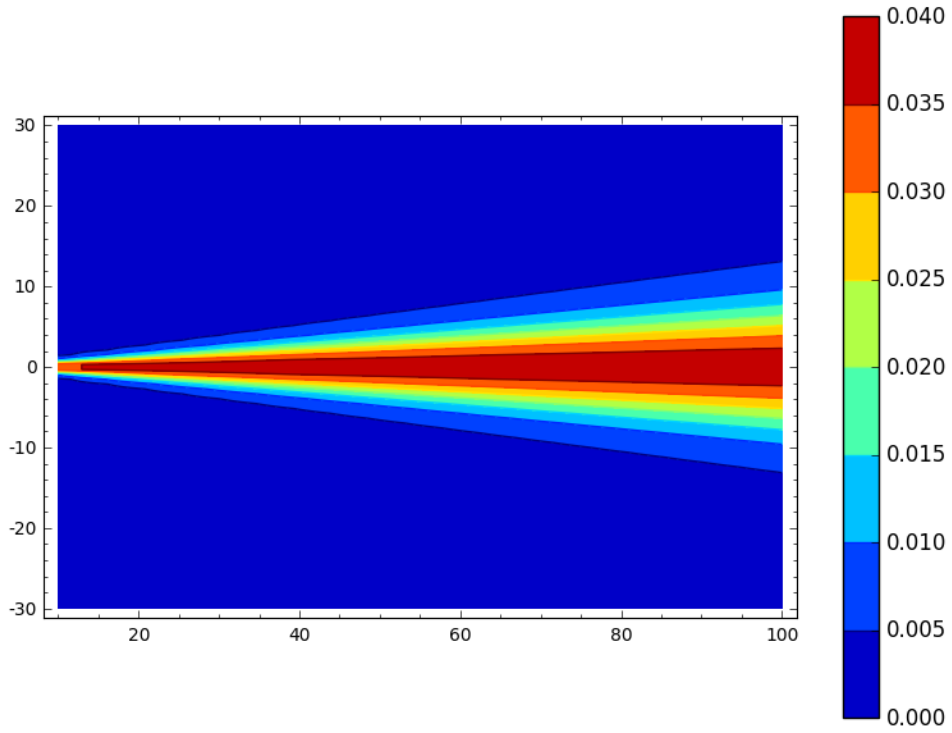

Figure 3.14: Profil de l'énergie cinétique de turbulence k_m pour un écoulement cisaillé

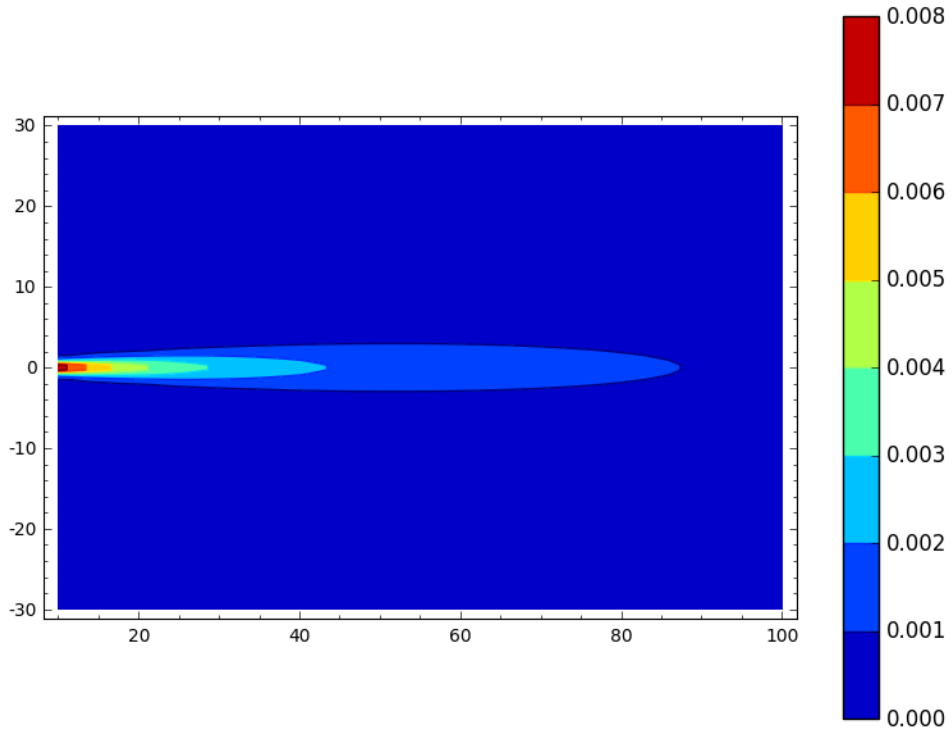

Figure 3.15: Profil du taux de dissipation de l'énergie cinétique turbulente ϵ_m pour un écoulement cisaillé

Les dérivés de la vitesse, employés pour les conditions limites de Neumann de cet écoulement, sont les suivantes :

$$\frac{\partial u}{\partial x} = \frac{\sigma y e^{-\frac{\sigma^2 y^2}{x^2}}}{x^2\sqrt{\pi}}$$
$$\frac{\partial v}{\partial x} = \frac{\sigma y^2 e^{-\frac{\sigma^2 y^2}{x^2}}}{x^3\sqrt{\pi}} \tag{3.30}$$

La complexité accrue de la SM employée ne permet pas d'appliquer la condition limite *zeroGradient* à *u*, *v*, *k* et ϵ. Conséquemment, l'élaboration de conditions limites variables est requise afin d'imposer des valeurs de Neumann variant selon l'axe des *x* et *y*. Les conditions limites employées pour ce test sont présentées au Tableau 3.6.

Tableau 3.6: Conditions limites employées pour l'écoulement cisaillé

Surface	Variable	Condition limite	Valeur
Entrée	*u*, *v*, *k*, ϵ	Dirichlet	SM
	p	Neumann	zeroGradient
Sortie	*u*, *v*, *k*, ϵ	Neumann	SM
	p	Dirichlet	0
Frontière du bas	*u*, *v*, *k*, ϵ	Dirichlet	SM
	p	Neumann	zeroGradient
Frontière du haut	*u*, *v*, *k*, ϵ	Dirichlet	SM
	p	Neumann	zeroGradient

À l'aide de cette SM, nous vérifions le code pour un écoulement comportant un certain réalisme et ayant des conditions de Neumann aux frontières variables. Toutefois, ces conditions limites ont été vérifiées en prenant une pression nulle sur tout le domaine. Comme mentionné précédemment, une étude approfondie serait donc à effectuer pour vérifier la combinaison de conditions de Neumann variables pour la pression et la vitesse alors qu'un terme source est ajouté au champ de vitesse.

Cette solution manufacturée complète donc la gamme d'écoulements dont nous nous servirons dans le cadre de ce travail. Nous avons développé les SM nécessaires à la vérification de code par la MMS afin de déterminer si le code respecte le taux de convergence théorique des schémas numériques employés. En effet, les schémas que nous employons avec ces écoulements devraient nous permettre d'atteindre une convergence du deuxième ordre. Le lecteur a donc les informations nécessaires pour l'analyse des résultats présentés au prochain chapitre.

CHAPITRE 4 RÉSULTATS ET DISCUSSION

Dans ce chapitre, nous utiliserons la suite de tests élaborée au chapitre précédent pour effectuer la vérification de code pour divers écoulements laminaires et turbulents. Il est donc possible d'atteindre le but de cette recherche, soit d'effectuer la vérification de code pour des écoulements turbulents, modélisés à l'aide du modèle de turbulence k-ϵ standard.

Le code a été vérifié en employant la gamme d'écoulements présentés à la section 3.4 afin de nous permettre d'identifier les sources d'erreurs pouvant nuire aux résultats des solutions. Les résultats obtenus à la suite de l'application de la méthodologie sont présentés et analysés dans cette section. Tout d'abord, nous présentons les résultats issus de la vérification de code par la méthode des solutions manufacturées. Ensuite, nous poursuivons la vérification de code à l'aide de l'extrapolation de Richardson pour deux solutions turbulentes. Nous mettons en comparaison deux écoulements comportant peu de différences, mais ayant d'importants écarts au niveau des résultats obtenus. Ce faisant, cette étape nous permet d'identifier les possibles causes de l'erreur d'implémentation identifiée.

4.1 Application de la méthode des solutions manufacturées

Dans cette section nous présentons les résultats issus de la vérification de code par la MMS. Les équations ainsi que les conditions limites que nous avons détaillées précédemment servent donc de base pour les explications fournies dans la présente section. Ainsi, nous mettons tout d'abord en relief les résultats pour les écoulements laminaires : un écoulement de Poiseuille, un écoulement de type couche limite et un écoulement libre 2D laminaire. Les résultats obtenus pour les écoulements turbulents suivent pour l'écoulement libre 2D turbulent et l'écoulement de type couche cisaillé.

Lors de la vérification, nous cherchons à confirmer que le taux de convergence observé atteint le taux de convergence théorique de 2 ($O(h^2)$). En effet, nous nous attendons à ce qu'une division par 2 de la hauteur h des éléments de maillage réduise l'erreur globale par un facteur de 4. Il est à préciser que dans le cadre de ce travail, les raffinements de maillages sont toujours effectués en divisant la dimension des éléments par deux ($r=2$).

Nous présentons le taux de convergence observé versus sa valeur théorique pour chaque cas qui suit. Nous illustrons aussi les sources d'erreurs présentes dans le domaine afin d'identifier les éléments pouvant nuire au taux de convergence observé. Nous complétons ensuite ces sections avec une discussion sur les sources d'erreurs pouvant affecter le taux de convergence.

4.1.1 Écoulement de Poiseuille

L'écoulement de Poiseuille est le seul calcul que nous avons réalisé sans utiliser de solution manufacturée. En effet, il admet une solution analytique nous permettant de vérifier le code sans avoir recours à l'implémentation d'un terme source dans le fichier *fvOptions*. Toutefois, cet écoulement est trop simple pour la vérification de tous les termes des équations de Navier-Stokes. Il constitue donc un point de départ pour la vérification plus approfondie du code.

Pour un maillage orthogonal, nous nous attendons à ce que le taux de convergence théorique soit atteint sans difficulté, car le code a été préalablement vérifié pour une équation de Poisson (Noriega et al., 2018a). Le taux de convergence observé par rapport à l'ordre théorique est d'ailleurs illustré à la Figure 4.1. Pour ce cas, nous avons commencé avec un maillage de 20 éléments dans la direction des x et 5 éléments dans la direction des y. À chaque raffinement de maillage, nous avons doublé le nombre d'éléments de maillage dans la direction des y. Nous constatons, comme prévu, que l'erreur converge au deuxième ordre pour les normes L_2 et L_∞. Cependant, il est intéressant de remarquer qu'un maillage relativement dense est requis afin d'atteindre la zone asymptotique. En outre, un maillage uniforme composé d'environ 40 éléments dans la direction des *y* est nécessaire pour atteindre cette zone.

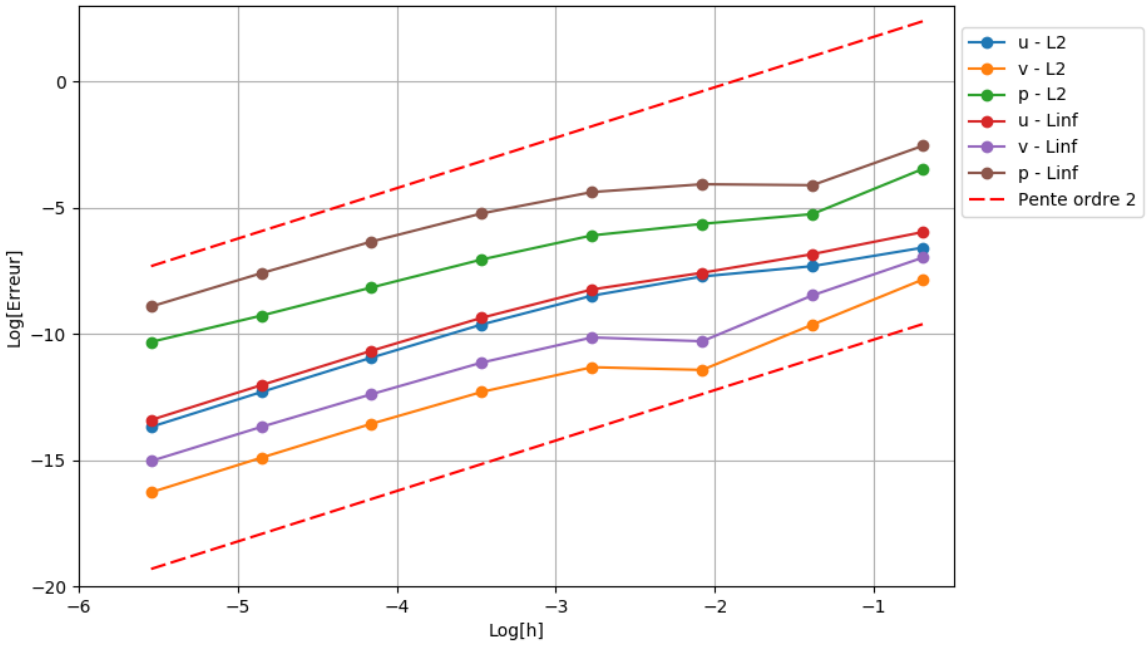

Figure 4.1: Erreur observée en fonction de la dimension des éléments de maillage pour le calcul de l'écoulement de Poiseuille

Un point supplémentaire qui a retenu notre attention est l'erreur maximale de la pression ($|u_{mms} - u_{calculé}|$), illustrée à la Figure 4.2. Comme nous pouvons le constater, l'erreur maximale est située ponctuellement aux centroïdes des éléments de maillage aux coins supérieur et inférieur gauches du domaine. Notons que cette anomalie coïncide avec l'intersection de deux frontières ayant des conditions de Dirichlet sur la vitesse, alors que deux conditions de Neumann se trouvent aussi sur ces mêmes frontières pour la pression. Il semble donc que les conditions limites soient responsables de l'erreur située sur les éléments de maillage communs à ces deux frontières.

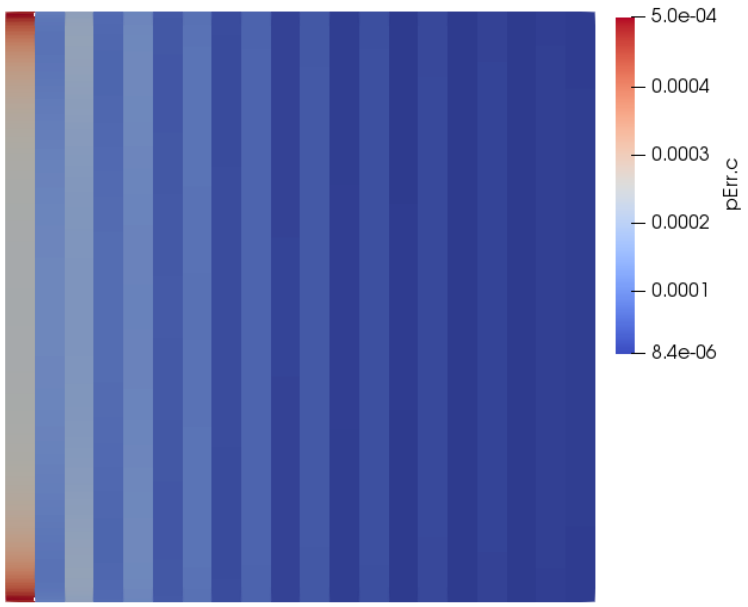

Figure 4.2: Erreur sur la pression pour l'écoulement de Poiseuille

Les profils de la pression à x=0 pour les maillages employés dans ce test sont illustrés à la Figure 4.3. Nous remarquons que la pression tend vers 1 au fur et à mesure que la dimension des éléments de maillages diminue. Cependant, la pression près des bords du domaine (y=0 et y=1) tend vers la solution moins rapidement que la pression au centre du domaine.

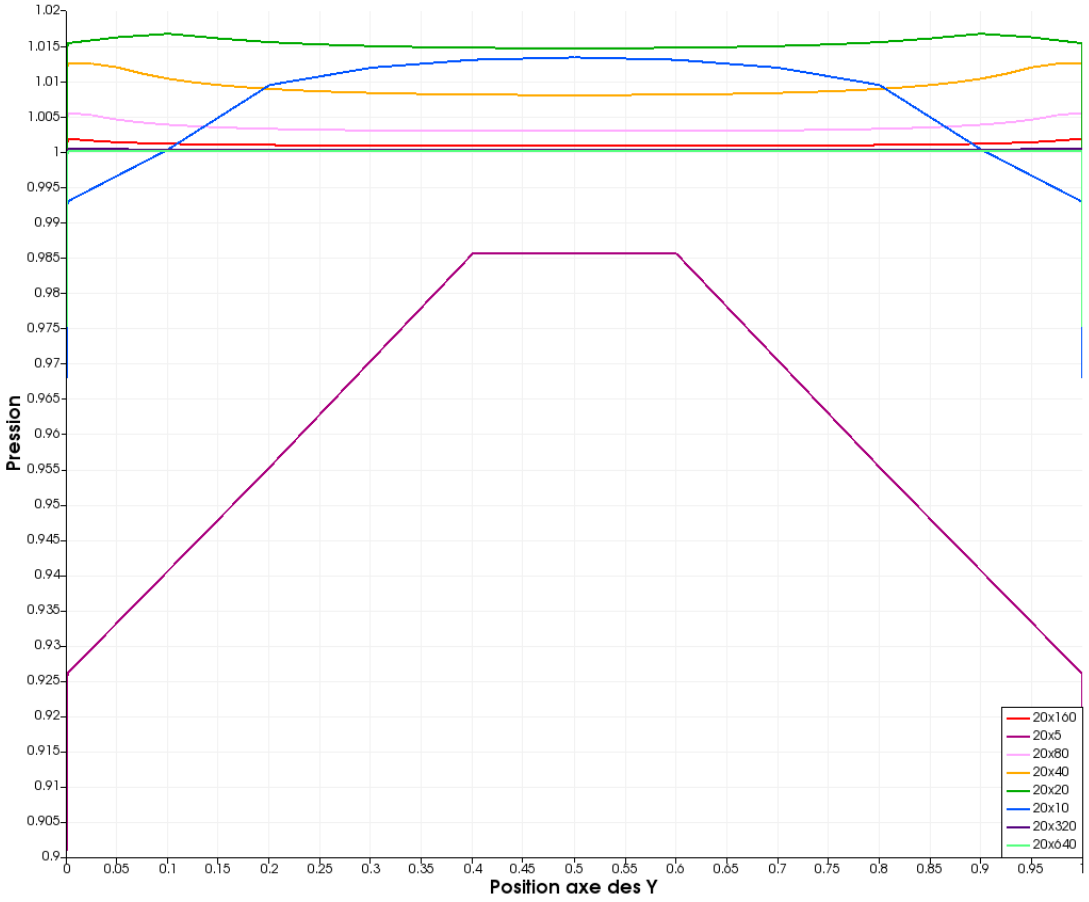

Figure 4.3: Profils de la pression à x=0 pour maillages utilisés avec l'écoulement de Poiseuille

Ce phénomène peut aussi être observé pour les profils de vitesse illustrés à la Figure 4.4. En effet, nous remarquons que la vitesse près des parois n'est pas nulle pour les maillages les moins raffinés. Ceci s'explique par le fait que les valeurs de *u* et *v* sont stockés aux centres des éléments de maillages plutôt que sur les nœuds de la frontière. Les valeurs de vitesses doivent donc être extrapolées aux frontières pour obtenir cette représentation graphique.

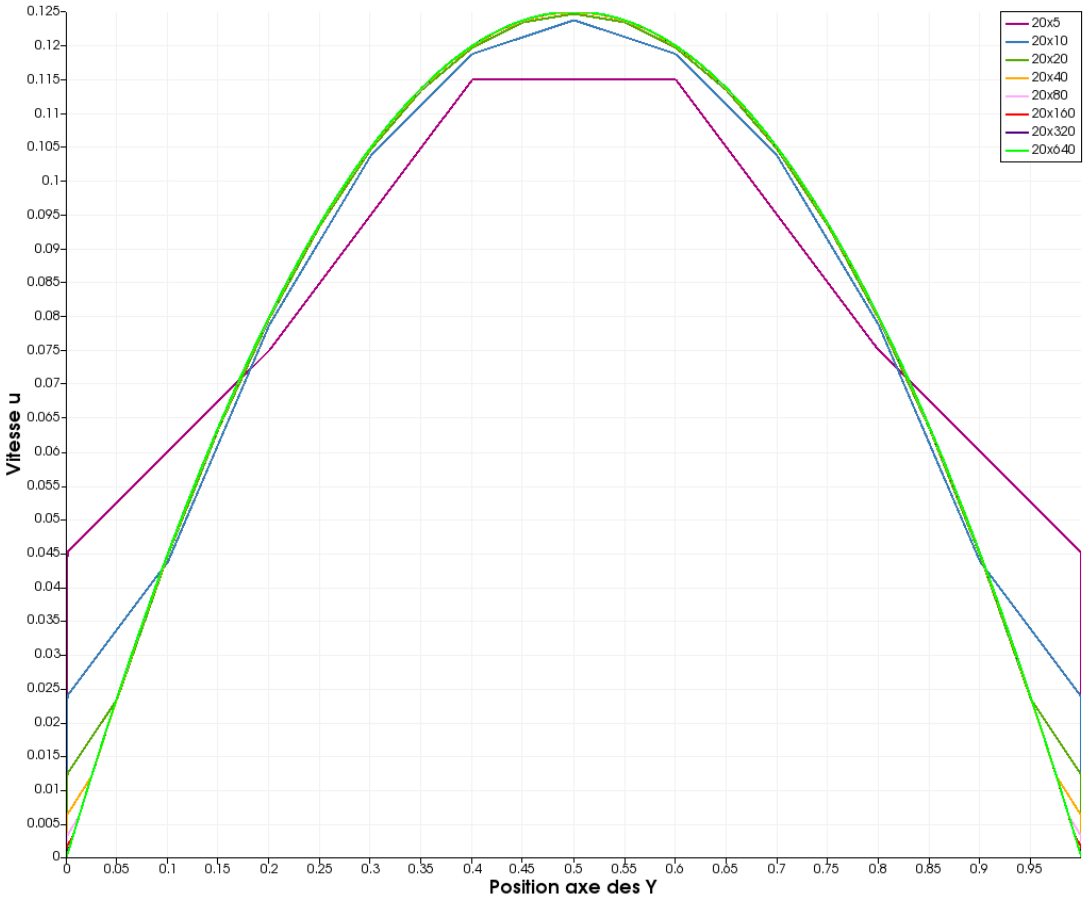

Figure 4.4: Profils de la vitesse u à x=0 pour les maillages utilisés avec l'écoulement de Poiseuille

Nous avons ensuite poursuivi le test en appliquant une pression nulle au point [1, 0] au lieu d'appliquer *p=0* sur la sortie de l'écoulement. Ce faisant, nous souhaitons analyser davantage l'effet qu'ont les conditions limites sur le taux de convergence de la pression.

Les résultats de ce deuxième test sont présentés à la Figure 4.5. Dans ce cas, le taux de convergence de la pression est réduit à zéro pour les deux normes d'erreur calculées. Malgré cela, les autres variables conservent une convergence de $O(h^2)$ comme prédit par la théorie. En comparant les courbes de ces variables, nous remarquons qu'elles sont identiques à la solution obtenue pour la condition limite *p=0* sur la sortie.

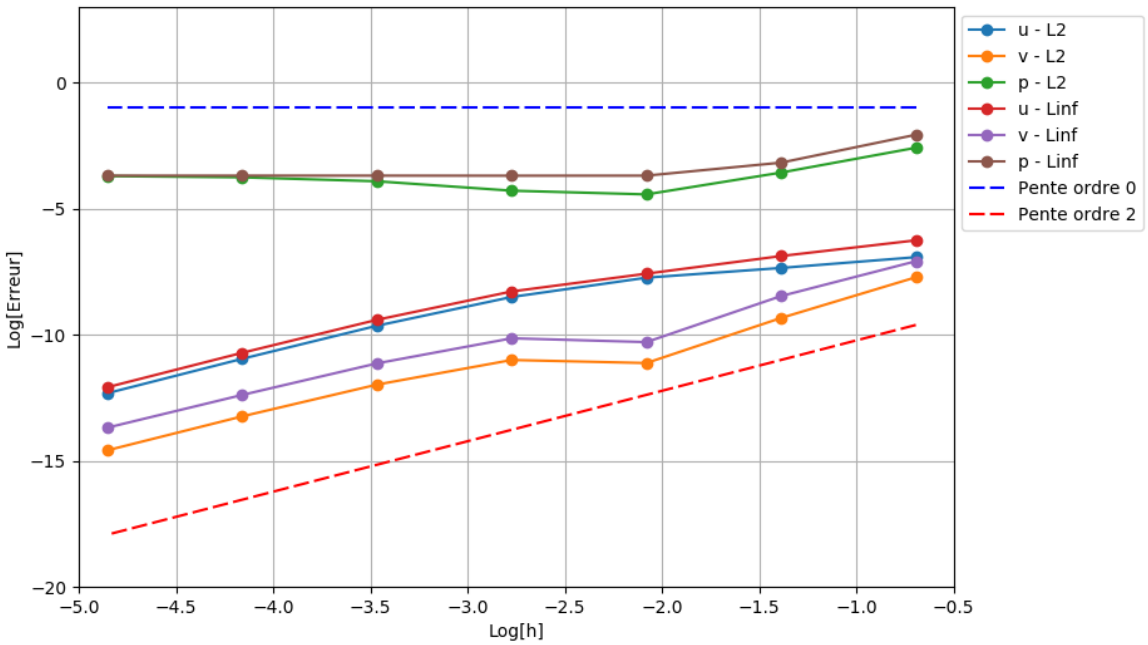

Figure 4.5: Erreur observée en fonction de la dimension des éléments de maillage pour le calcul de l'écoulement de Poiseuille lorsque $p=0$ en un point

La nouvelle erreur de pression est présentée à la Figure 4.6. Nous remarquons que les extremums se trouvent toujours aux coins du domaine, mais cette fois il n'y a que très peu de différence entre la valeur minimale et la valeur maximale de l'erreur. Il semble donc que la solution de l'écoulement a convergé vers une solution alternative de la pression, tout en respectant quand même l'équation de Poiseuille. La condition de $p=0$ en un point n'est donc pas une condition suffisamment contraignante pour obtenir la solution désirée de la pression, même si les autres variables ne semblent pas affectées par cette dernière.

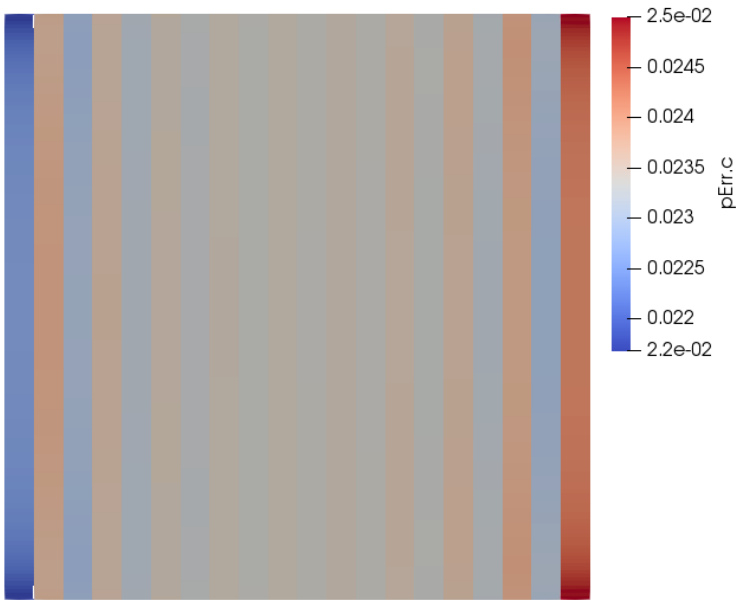

Figure 4.6: Erreur sur la pression pour l'écoulement de Poiseuille lorsque la pression est imposée en un point

Il en résulte que, parmi les deux cas étudiés, le taux de convergence théorique n'est atteint que lorsqu'un Dirichlet est imposé sur la sortie de l'écoulement pour la pression. D'autre part, il semble qu'imposer $p=0$ en un point implique que la solution de la pression pourrait converger vers une solution alternative à celle cherchée. Il nous reste donc toujours à identifier la source de l'erreur située aux coins du domaine pour ensuite évaluer si cette dernière peut affecter le taux de convergence observé.

Sachant cela, nous remarquons que le taux de convergence théorique est atteint pour les variables étudiées, malgré les sources d'erreurs présentes. Nous estimons toutefois qu'il est mieux de faire usage de précaution lorsqu'on effectue un calcul requérant des valeurs sur un élément de maillage situé à un coin du domaine.

4.1.2 Écoulement de type couche limite

En poursuivant la vérification de code pour des écoulements 1D, nous implémentons maintenant une première solution manufacturée pour un écoulement laminaire. Comme pour le cas précédent, nous pouvons nous attendre à un taux de convergence de $O(h^2)$ pour u, v et p, car une condition de Dirichlet est imposée à la pression sur toute la frontière de la sortie.

La Figure 4.7 confirme que toutes les variables atteignent le taux de convergence théorique. Le maillage de départ est de 20 éléments dans la direction des x et 5 éléments dans la direction des y. Nous multiplions par deux le nombre d'éléments de maillage dans la direction des y à chaque raffinement de maillage. De manière similaire à l'écoulement de Poiseuille, plusieurs raffinements de maillages sont requis afin de parvenir à la zone asymptotique. En effet, cette zone n'est atteinte que pour un maillage de 160 éléments dans la direction des y. Cela s'explique par la forte variation de la vitesse près de la paroi qui a été définie à la section 3.4.2. Bien entendu, plus la variation de vitesse est prononcée, plus un maillage raffiné sera requis pour capturer ce phénomène, et ainsi atteindre la zone asymptotique du taux de convergence.

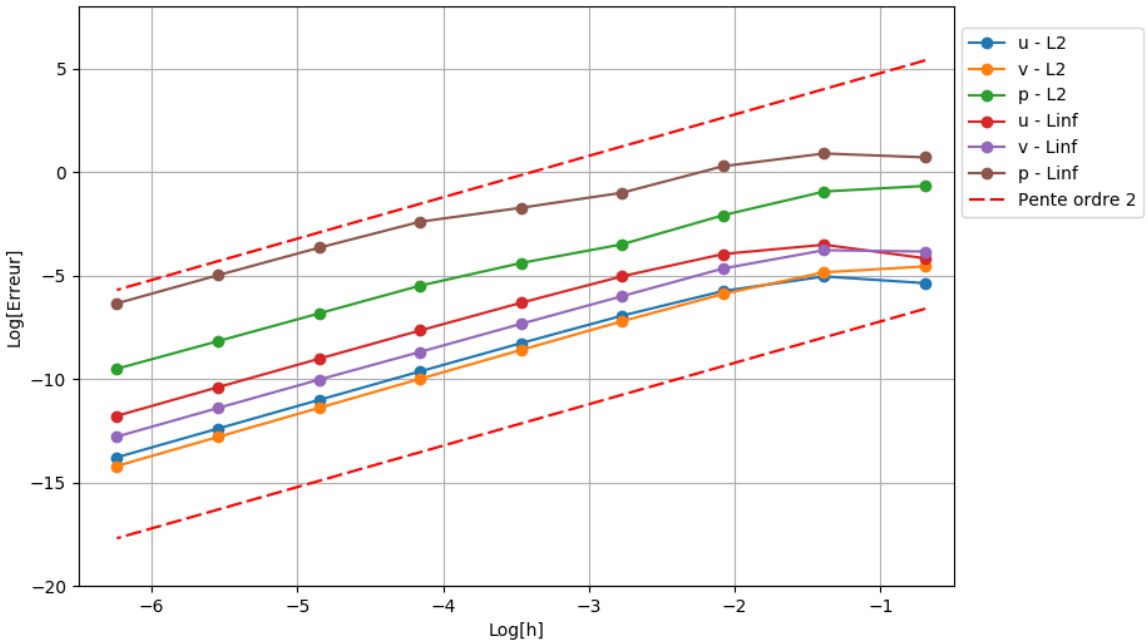

Figure 4.7: Erreur observée en fonction de la dimension des éléments de maillage pour le calcul de l'écoulement de couche limite

D'autre part, nous étudions l'erreur située aux coins du domaine à la Figure 4.8 afin de mieux comprendre leur origine. Nous remarquons que l'erreur maximale de la pression se trouve toujours sur l'élément de maillage où coïncident deux conditions de Dirichlet. Cependant, il est intéressant de noter que cette erreur est moins significative à la frontière supérieure du domaine où la vitesse est fixée à U_∞=1. Ce phénomène s'explique par le fait que la vitesse varie davantage près de la paroi du bas que de celle du haut. Pour un maillage uniforme comme celui employé dans le cas

présent, la résolution n'est donc pas suffisante près de la paroi pour obtenir une erreur du même ordre de grandeur que celle trouvée pour le reste de l'écoulement.

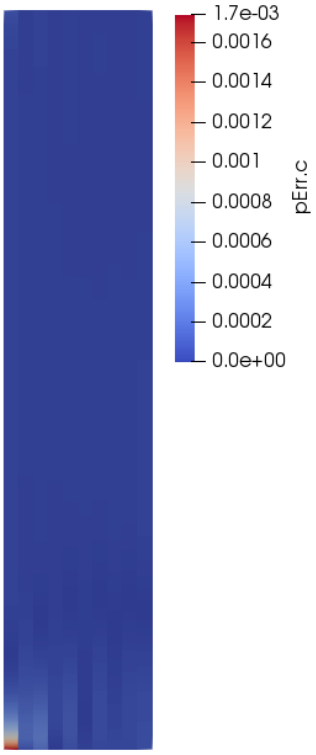

Figure 4.8: Erreur sur la pression pour l'écoulement de type couche limite

Ce phénomène peut aussi être observé à la Figure 4.11 où les profils de pression en entrée du domaine sont illustrés pour les différents maillages employés pour ce test. Nous remarquons que la pression tend vers 0.2, mais que la pression près de y=0 a un écart significatif avec le restant de la solution. Toutefois, cela n'empêche pas la pression de converger vers la solution manufacturée au deuxième ordre comme prévu par la théorie.

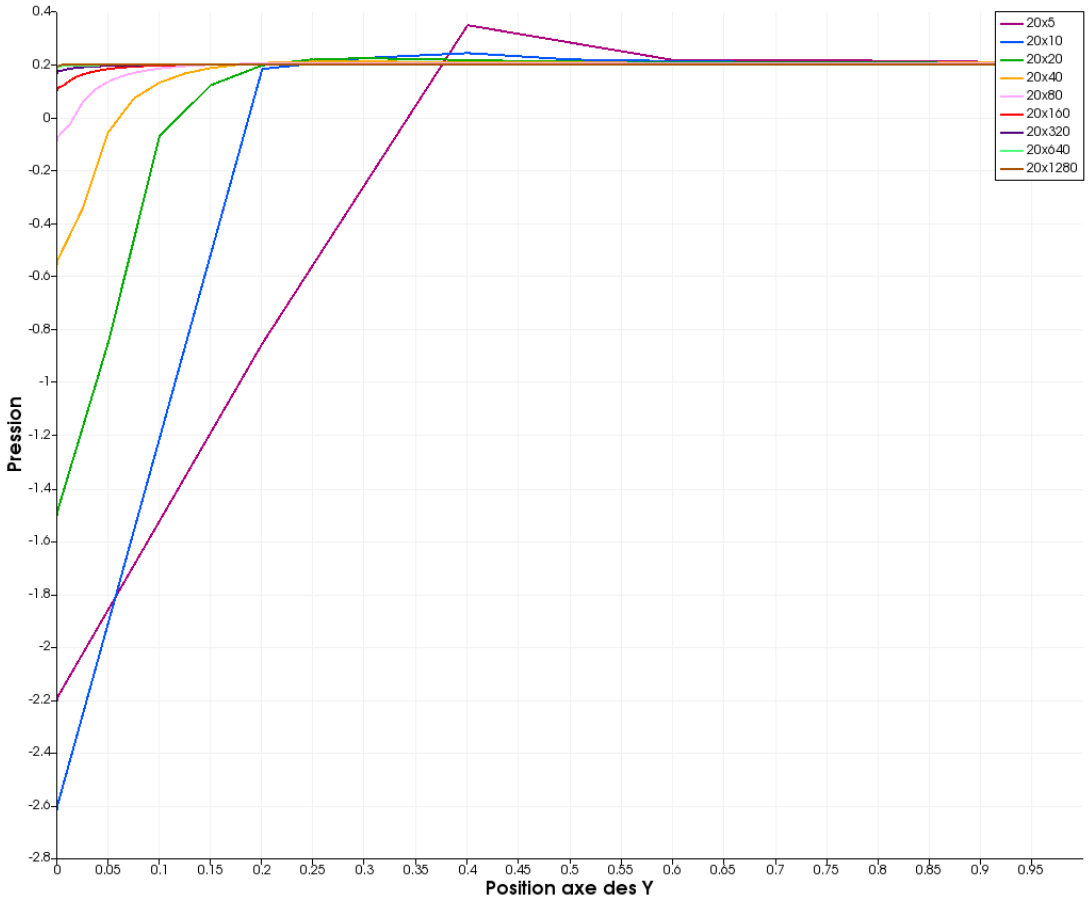

Figure 4.9: Profils de la pression à x=0 pour les maillages utilisés avec l'écoulement de type couche limite

Le code se comporte conformément à la théorie pour un écoulement laminaire unidirectionnel, lorsque nous imposons des conditions limites de *fixedValue*, *fixedGradient* et *zeroGradient*. Au passage, nous avons pu observer que l'erreur maximale sur la pression serait due à la forte variation de la vitesse de l'écoulement près de la paroi. Malgré cette source d'erreur localisée, celle-ci diminue selon le même taux de convergence que le reste de l'écoulement.

De manière accessoire, il nous a aussi été possible de confirmer que le code employé pour implémenter le terme source dans le fichier *fvOptions* fonctionne comme prévu. Nous pouvons donc poursuivre l'analyse pour les écoulements 2D en n'effectuant que des modifications mineures au code.

4.1.3 Écoulement libre 2D laminaire

L'écoulement laminaire 2D étudié dans cette section est utilisé comme gabarit pour la vérification de code pour un écoulement turbulent. Avant d'être combinée avec des termes de k et ϵ, nous cherchons tout d'abord à démontrer que le taux de convergence est atteint pour un écoulement 2D laminaire.

Le maillage de départ a 10x10 éléments dans les directions x et y. Nous multiplions par deux le nombre d'éléments de maillages dans les deux directions à chaque raffinement de maillage. Comme pour les deux cas précédents, le taux de convergence théorique est respecté pour la norme L_2 de l'erreur, tel qu'illustré à la Figure 4.10. Cependant, le taux de convergence de la pression est réduit à $O(h)$ lorsqu'on prend la norme L_∞ de l'erreur. Comme ce phénomène n'a pas été observé pour les cas 1D, une analyse plus détaillée s'impose afin de déceler la source de cette détérioration.

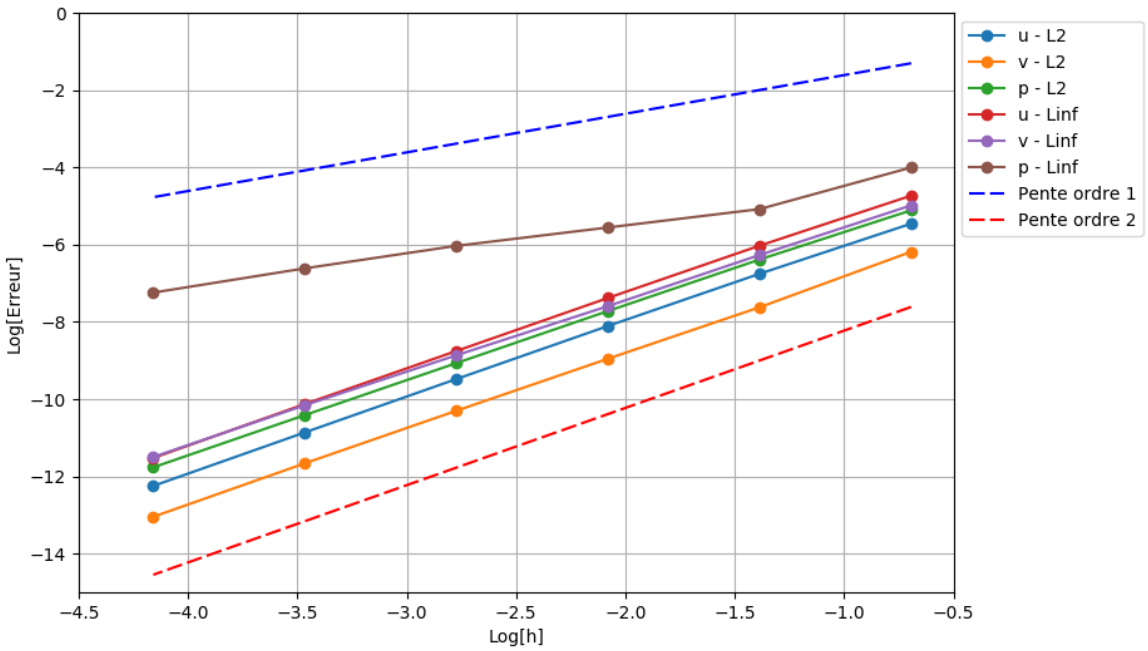

Figure 4.10: Erreur observée en fonction de la dimension des éléments de maillage pour le calcul de l'écoulement libre 2D laminaire

Afin d'expliquer cette détérioration du taux de convergence, nous illustrons l'erreur de la pression à la Figure 4.11. Comme nous pouvons le constater, les erreurs maximales se trouvent une fois de plus au coin du domaine, mais sans que celles-ci diminuent en fonction de $O(h^2)$. De plus, ces

erreurs ne sont pas engendrées par la forte variation de la vitesse comme pour les cas précédents, car aucune paroi n'est présente dans cet écoulement.

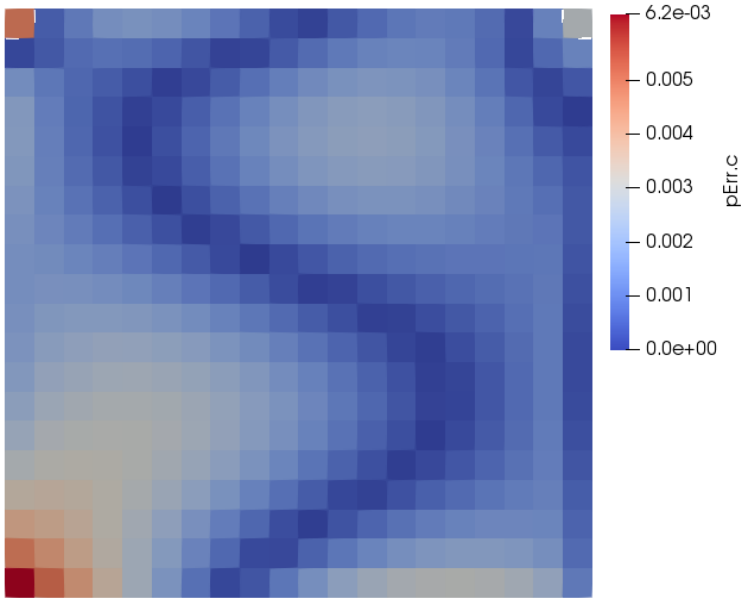

Figure 4.11: Erreur sur la pression pour l'écoulement libre 2D laminaire

Les profils de la pression à x=0 sont illustrés à la Figure 4.12. Nous pouvons observer le comportement de la pression près des frontières du haut et du bas du domaine où l'on retrouve les plus importants écarts avec la pression exacte (p_m=2).

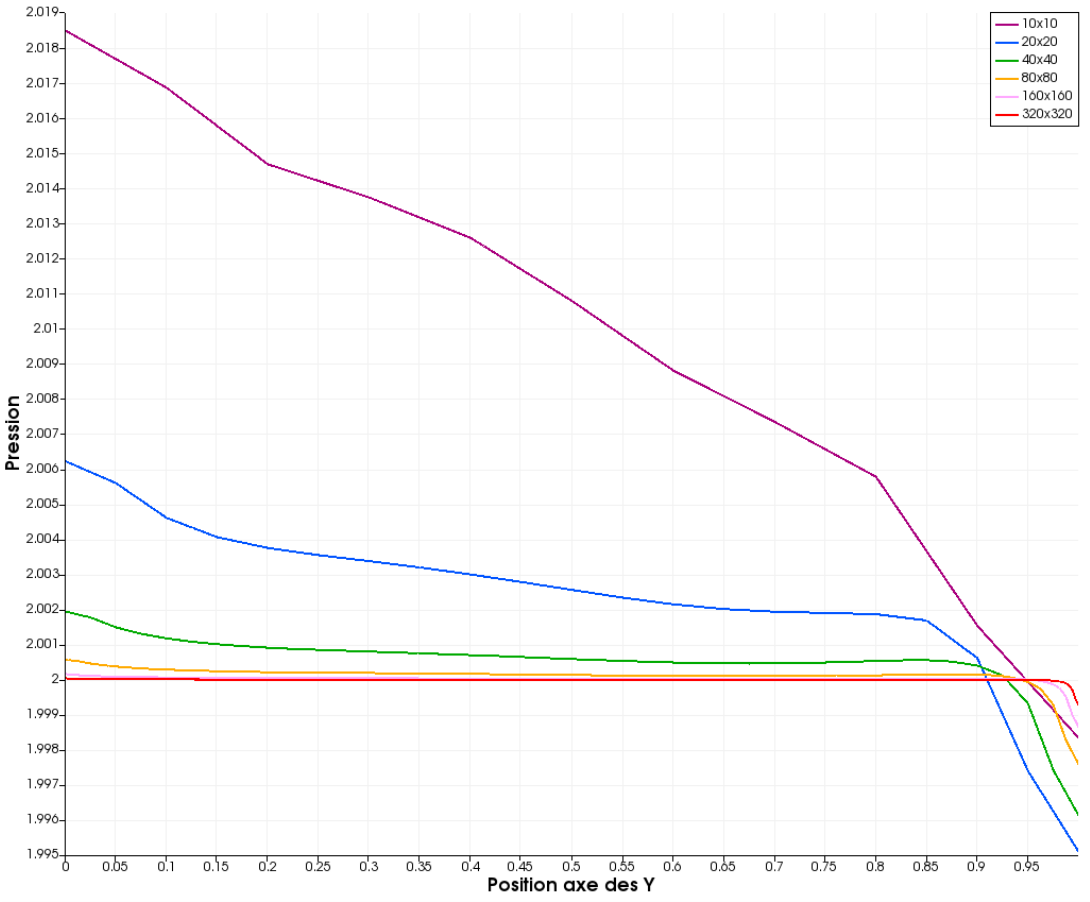

Figure 4.12: Profils de la pression à x=0 pour les maillages utilisés avec l'écoulement livre 2D laminaire

Cette détérioration du taux de convergence est susceptible d'être causée par l'imposition de conditions limites variables sur la vitesse pour *u* et *v*, ce qui n'était pas le cas des écoulements précédents. Il est donc probable que la combinaison de ces conditions limites sur les frontières d'un élément de maillage entraine une erreur additionnelle aux coins du domaine. Cette erreur serait vraisemblablement due à l'implémentation des conditions limites dans OpenFOAM. Il nous reste toutefois à identifier la source exacte de cette erreur afin de confirmer cette hypothèse.

Malgré la perte du taux de convergence de la pression, nous constatons que celle-ci ne semble pas influencer le taux de convergence des autres variables étudiées (*u*, *v*, *k* et ϵ). Nous sommes donc rassurés dans le fait que le code semble implémenté adéquatement malgré ses faibles performances pour la norme L_∞ de l'erreur de la pression.

Nous avons démontré avec succès que le code fonctionne comme prévu dans la mesure où on résout un écoulement laminaire avec un maillage orthogonal. À la prochaine étape, nous intégrons les équations de transport de k et ϵ ainsi que leur schéma de discrétisation. Nous pouvons nous attendre à ce que toutes les erreurs, autres que celle issue de la norme L_∞ pour la pression, convergent vers la solution exacte au deuxième ordre.

4.1.4 Écoulement libre 2D turbulent

Pour ce cas, les équations de transport de k et ϵ requièrent deux termes sources supplémentaires, soit une pour k et l'autre pour ϵ. De même, nous spécifions au solveur *simpleFoam* de résoudre les nouvelles équations de transport, comme illustré à la Figure 1.5.

L'évolution de la norme L_2 de l'erreur est présentée à la Figure 4.13 pour un premier écoulement turbulent. Comme c'est le cas pour les écoulements laminaires, le taux de convergence des variables tend vers l'ordre théorique sans difficulté pour la norme L_2 de l'erreur. Le maillage le moins dense est composé de 5 éléments de maillage dans chaque direction. Un maillage relativement peu dense est d'ailleurs requis pour atteindre la zone asymptotique, soit environ 400 (20 x 20) éléments de maillage.

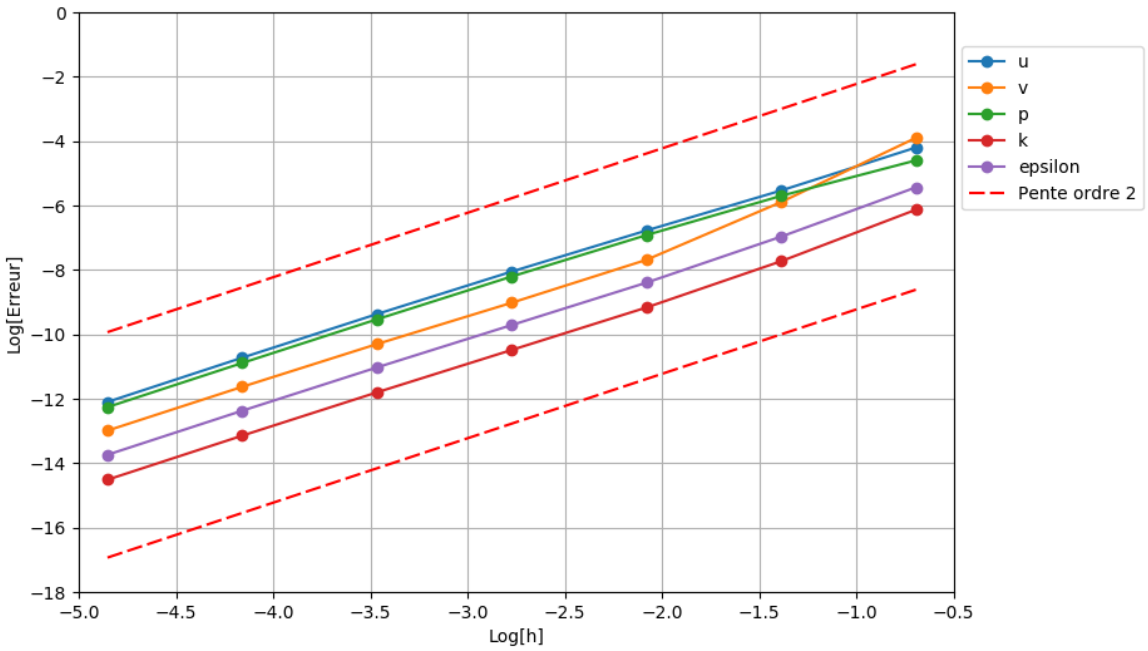

Figure 4.13: Norme L_2 l'erreur observée en fonction de la dimension des éléments de maillage pour le calcul de l'écoulement libre 2D turbulent

La norme L_∞ de l'erreur illustrée à la Figure 4.14 présente une détérioration du taux de convergence de la pression comme anticipé. Pourtant, le taux de convergence de la pression pour les premiers maillages est du deuxième ordre avant d'être réduit au premier ordre pour les maillages subséquents. Il est donc possible que la source d'erreur polluant la solution ne devienne significative qu'après un certain nombre de raffinements. Afin de mieux comprendre ce phénomène, nous illustrons l'erreur de la pression à la Figure 4.15.

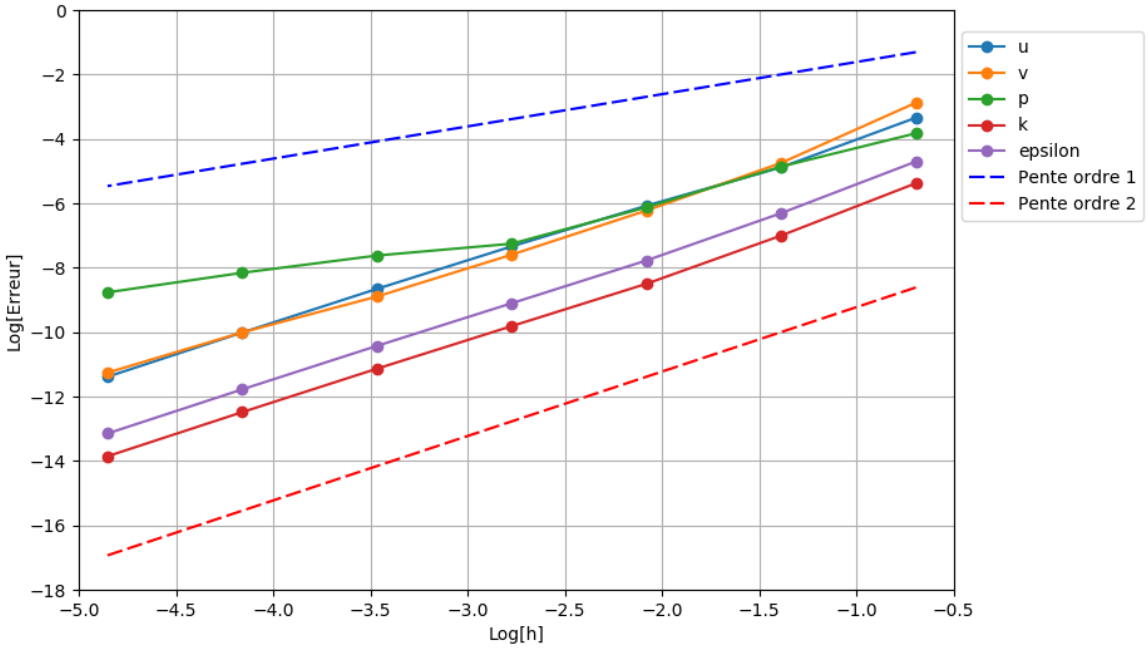

Figure 4.14: Erreur maximale observée en fonction de la dimension des éléments de maillage pour le calcul de l'écoulement libre 2D turbulent

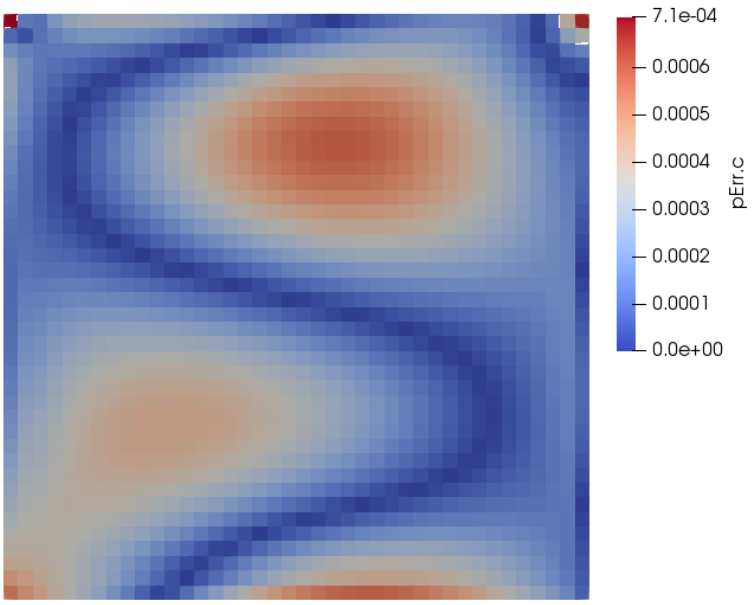

Figure 4.15: Erreur de la pression pour l'écoulement libre 2D turbulent

Contrairement aux cas précédents, cette fois l'erreur maximale se trouve aux coins supérieurs du domaine. Ceux-ci ne coïncident donc pas avec une configuration particulière de conditions limites. Toutefois, nous constatons que le taux de convergence passe de $O(h^2)$ à $O(h)$, ce qui peut signifier qu'il existe une source d'erreur qui pollue le taux de convergence. Cette source d'erreur pourrait ainsi devenir significative par rapport à l'erreur globale de la solution à la suite de l'obtention d'un maillage suffisamment raffiné.

Afin d'approfondir le sujet, nous effectuons une deuxième vérification pour un écoulement turbulent. Nous souhaitons ainsi tester d'autres conditions limites sur la vitesse afin d'identifier les sources potentielles de cette perte du taux de convergence observé.

4.1.5 Écoulement de type couche cisaillé

Pour ce dernier test, nous employons une SM turbulente comportant des conditions limites différentes de celles du cas précédent. Entre autres, cette SM est plus complexe, car elle varie en entrée suivant la fonction d'erreur (*erf*). Cependant, l'équation de la pression y est simplifiée, car celle-ci est nulle sur tout le domaine.

Le maillage employé est composé de 10 éléments dans les directions x et y. Nous multiplions par deux le nombre d'éléments dans les deux directions à chaque raffinement de maillage.

Nous constatons alors que la forte variation de vitesse attribuable à la fonction d'erreur rend l'atteinte de la zone asymptotique plus ardue, telle que montrée à la Figure 4.16. En effet, un maillage d'environ 1600 éléments est requis pour atteindre cette zone pour u, v, k et ϵ alors qu'on ne parvient jamais à la zone asymptotique de la pression.

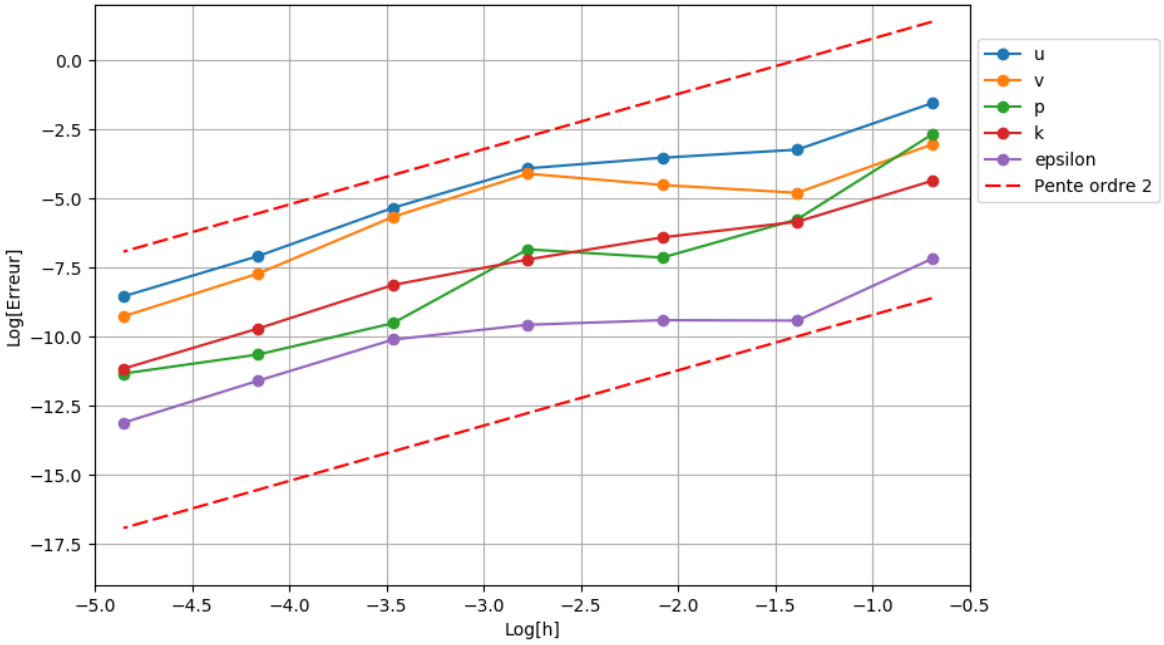

Figure 4.16: Norme L_2 de l'erreur observée en fonction de la dimension des éléments de maillage observé pour le calcul de l'écoulement turbulent cisaillé

De manière similaire, en prenant la norme L_∞ nous obtenons le taux de convergence présenté à la Figure 4.17. Comme pour la norme L_2, le taux de convergence de toutes les variables, à l'exception de la pression, tend vers $O(h^2)$ comme prévu. La norme L_∞ de l'erreur de la pression, quant à elle, semble plutôt converger à un taux nul. Nous obtenons donc une stagnation graduelle de la norme L_∞ de l'erreur de la pression malgré des raffinements de maillages supérieurs.

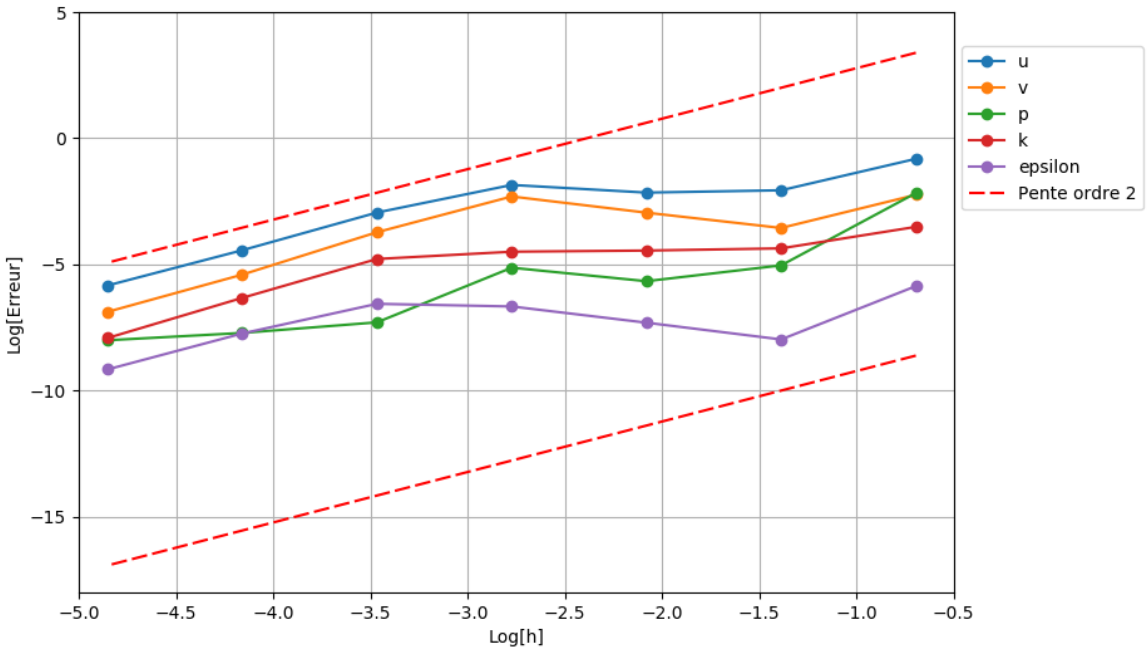

Figure 4.17: Norme L_∞ de l'erreur observée en fonction de la dimension des éléments de maillage pour le calcul de l'écoulement turbulent cisaillé

Notons que les conditions limites pour ce cas sont très similaires à celles de l'écoulement libre turbulent, à l'exception près que les conditions *zeroGradient* sur la vitesse sont maintenant des conditions de Neumann différentes de zéro. De plus, l'erreur de la pression présentée à la Figure 4.18 montre que l'erreur maximale se situe en entrée et en sortie de l'écoulement plutôt qu'aux coins du domaine, tel que pour les cas précédents. Ce sont donc ces nouvelles sources d'erreurs qui sont responsables de la détérioration du taux de convergence de la pression.

Figure 4.18: Erreur sur la pression pour l'écoulement turbulent de type couche cisaillé

Il est probable que l'erreur en sortie de l'écoulement soit attribuable aux conditions limites de Neumann variables employées pour cet écoulement. En effet, ce cas est similaire à ceux de Fisch, R. F., Jörg; Wüchner, Roland; Bletzinger, Kai-Uwe (2012) qui ont décelé un taux de convergence du premier ordre lorsqu'ils ont employé de telles conditions limites. Il est donc plausible que, pour des raffinements de maillages supérieurs, le taux de convergence de la pression tende vers $O(h)$ pour la norme L_2 de l'erreur.

En entrée du domaine, l'erreur est maximale là où la résolution du maillage est insuffisante pour capturer la forte variation de la fonction d'erreur (*erf*) dans l'équation de la vitesse. Ce fait est aussi observé à l'aide des courbes de pression en entrée du domaine pour les maillages employés (Figure 4.12).

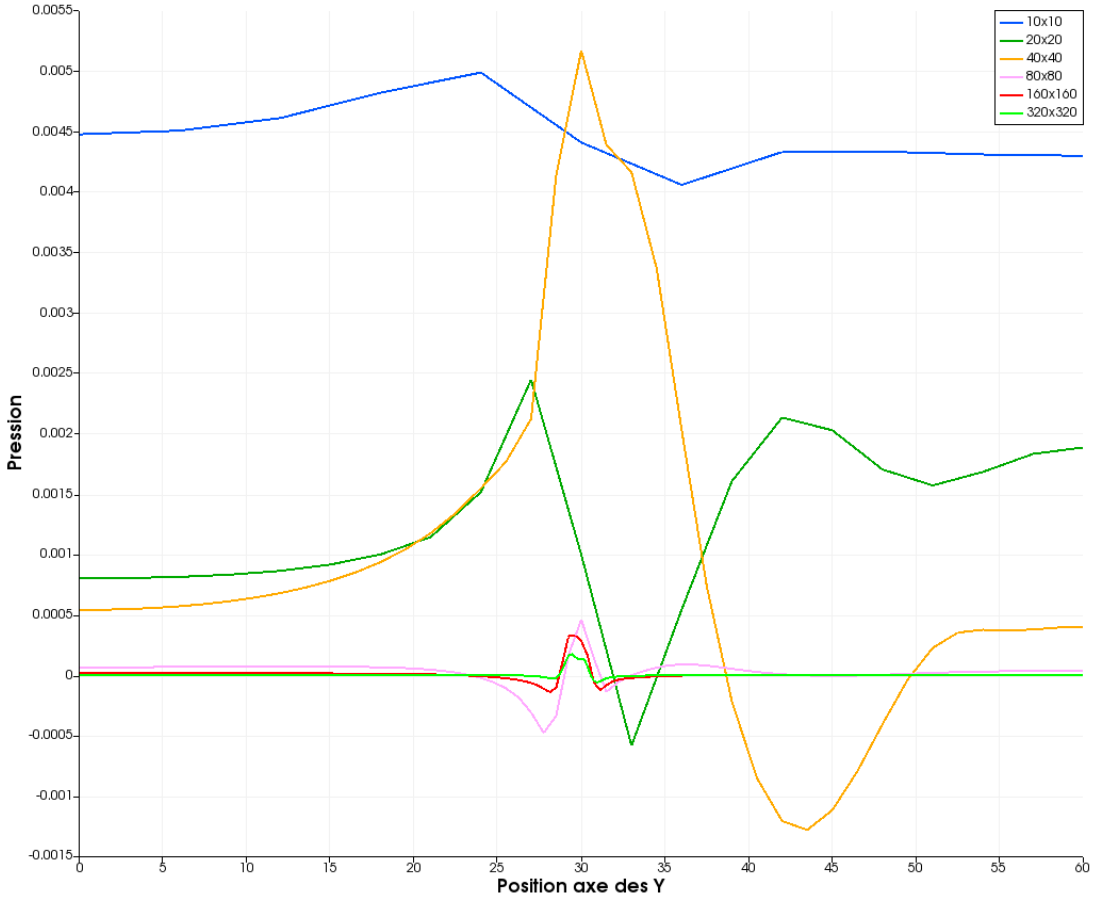

Figure 4.19: Profils de la pression à x=30 pour les maillages utilisés avec l'écoulement de type couche cisaillé

Ayant analysé les résultats des différents écoulements employés pour la vérification de code, nous émettons l'hypothèse que le taux de convergence théorique est maintenu lorsque nous employons la condition limite *zeroGradient*. Les erreurs sur les éléments de maillages aux coins du domaine nous ont permis de déceler une possible lacune dans l'implémentation des conditions limites dans le code. Toutefois, pour l'écoulement de Poiseuille et l'écoulement de type couche limite, cette lacune ne semble pas avoir d'impact sur le taux de convergence observé. Notons que l'erreur située aux coins du domaine peut être causée par l'interpolation de « Rhie and Chow » (Kärrholm, 2006) qui est connue pour entrainer des difficultés dans l'implémentation des conditions limites.

Dans le cas particulier de l'écoulement de type couche cisaillé, une source d'erreur supplémentaire est introduite en sortie du domaine, nuisant sévèrement au taux de convergence observé pour la pression. Il est possible que cette lacune soit due aux conditions limites de Neumann variables sur la vitesse en sortie du domaine. Alors que ces conditions limites sont aussi présentes en haut et en

bas du domaine, nous n'avons pas observé d'impact sur le taux de convergence. Ajoutant à ces observations, nous confirmons que les variables autres que la pression convergent toutes selon $O(h^2)$ comme prédit par la théorie.

Quoique le code soit vérifié pour le modèle k-ϵ standard, certaines questions persistent sur la cause de la détérioration du taux de convergence de la pression pour l'écoulement de type couche cisaillé. Nous poursuivons donc la vérification de code dans le but d'en apprendre davantage sur le comportement de l'erreur de la pression pour les écoulements turbulents.

4.2 Application de l'extrapolation de Richardson

Dans cette section, nous estimons l'erreur de la simulation en appliquant la méthode d'extrapolation de Richardson aux résultats obtenus par la MMS de la section précédente. L'extrapolation de Richardson est utilisée pour cette vérification, car elle permet son application à plusieurs simulations ayant différents raffinements de maillages, sans requérir de modification supplémentaire aux solutions. À l'aide de l'erreur estimée par cette méthode, nous avons calculé à nouveau le taux de convergence observé afin d'en faire la comparaison avec les résultats issus de la MMS. Ce faisant, nous cherchons à confirmer l'estimation de l'erreur de la solution tout en apportant quelques clarifications supplémentaires au sujet des difficultés rencontrées à la section précédente.

L'erreur estimée par l'extrapolation de Richardson est comparée avec la norme L_2 de l'erreur obtenue par la MMS. Nous cherchons ainsi à confirmer si les erreurs ponctuelles (L_∞) près des frontières polluent la solution se trouvant aux frontières.

Concernant la variable d'intérêt choisie pour l'extrapolation de Richardson, nous avons employé la différence de pression moyenne entre deux frontières opposées du domaine. Les variables sont donc :

$$\phi_{gd} = \overline{p_{gauche}} - \overline{\underbrace{p_{droite}}_{cste=0}}$$
$$\phi_{hb} = \overline{p_{haut}} - \overline{p_{bas}} \qquad (4.1)$$

où $\overline{p_f}$ est la pression moyenne à la frontière *f*. À noter que ces variables devraient toutes tendre vers zéro lors de raffinements subséquents du maillage.

Ce choix est basé sur l'importance de vérifier la pression pour les écoulements étudiés. En effet, c'est surtout la pression qui semble éprouver des difficultés à atteindre le taux de convergence théorique. Elle est effectivement plus sensible aux erreurs d'implémentation qui pourraient avoir lieu dans le code.

Notons qu'à l'équation 4.1, la pression moyenne du côté droit du domaine est toujours une constante. Il en résulte que seule l'erreur sur la pression moyenne de gauche (en entrée du domaine) est évaluée pour cette variable d'intérêt. Cette valeur est d'ailleurs particulièrement pertinente à caractériser dans notre cas, car on trouve une forte variation de la vitesse à cette frontière pour l'écoulement de type couche cisaillé.

Nous appliquons l'extrapolation de Richardson premièrement à l'écoulement libre 2D turbulent. Nous y quantifions l'estimation de l'erreur, ce qui nous permet de confirmer si l'ordre $O(h)$ de la pression affecte le taux de convergence aux frontières. En second lieu, l'extrapolation de Richardson est appliquée à l'écoulement de type couche cisaillé. En effet, nous n'avons pas obtenu un taux de convergence fixe pour la pression lorsque nous avons appliqué la MMS à cet écoulement. Nous cherchons donc à bonifier les explications relatives à cet écart en employant la vérification de simulation.

4.2.1 Extrapolation de Richardson pour l'écoulement libre 2D turbulent

Lors de la vérification de code pour cet écoulement à la section 4.1, nous avons remarqué que seule la norme L_∞ de l'erreur de la pression ne diminuait pas selon $O(h^2)$. Étant donné que la norme L_∞ de l'erreur se trouve aux coins du domaine, il est pertinent d'évaluer si elle affecte aussi le taux de convergence de la pression aux frontières. Nous pourrons ainsi confirmer si ces valeurs ont un impact sur les calculs employant la pression, comme celui du calcul de la perte de charge.

Concernant les résultats obtenus pour cet écoulement, nous considérons que la zone asymptotique est atteinte lorsque le maillage comporte 80x80 (1600) éléments. En effet, ce maillage a lieu une fois que la vitesse v atteint un taux de convergence $O(h^2)$, après le 3^e raffinement (4^e solution). Ce fait peut d'ailleurs être observé à la Figure 4.14 de la section précédente. C'est pourquoi les résultats issus des maillages moins denses sont grisés dans les tableaux qui suivent.

Les résultats de l'application de l'extrapolation de Richardson sont présentés au Tableau 4.1. Nous constatons que l'erreur estimée par l'extrapolation de Richardson est très similaire à celle calculée

par la MMS. Conséquemment, les taux de convergence observés sont très similaires une fois que la zone asymptotique est atteinte. Nous remarquons aussi que le taux de convergence de la pression ne s'est pas détérioré à *O(h)* comme c'est le cas pour la norme L_∞ de l'erreur. Même si la norme L_∞ de l'erreur est située aux coins du domaine, elle n'influence pas le taux de convergence observé sur les frontières.

Tableau 4.1: Comparaison de l'erreur et du taux de convergence obtenus par l'extrapolation de Richardson avec la norme L_2 de l'erreur de la MMS pour l'écoulement libre 2D turbulent

Nb élém.	E_{gd}	E_{hb}	E_{mms}	p_{hb}	p_{gd}	p_{mms}
5x5	-	-	1.01E-02	-	-	-
10x10	2.74E-03	3.27E-03	3.35E-03	-	-	1.59
20x20	9.93E-04	6.48E-04	9.89E-04	2.34	1.46	1.76
40x40	2.79E-04	1.42E-04	2.73E-04	2.19	1.83	1.85
80x80	7.20E-05	3.17E-05	7.24E-05	2.16	1.95	1.92
160x160	1.78E-05	7.09E-06	1.87E-05	2.16	2.01	1.96
320x320	4.28E-06	1.69E-06	4.74E-06	2.06	2.06	1.98

Ainsi, il est évident que la pression aux frontières se comporte comme la norme L_2 de l'erreur. La norme L_∞ de l'erreur sur la pression semble être ponctuelle sur les éléments de maillages aux coins du domaine, sans avoir d'influence significative sur le reste de la solution.

4.2.2 Extrapolation de Richardson pour l'écoulement cisaillé

Contrairement à l'écoulement libre 2D, la norme L_2 de l'erreur de la pression n'atteint pas le taux de convergence théorique pour l'écoulement de type couche cisaillé. Nous cherchons donc à comprendre l'étendue de cette lacune en étudiant l'évolution de l'erreur de la pression aux frontières du domaine. Ce faisant, nous cherchons aussi à confirmer l'hypothèse que les conditions limites de Neumann variables sur la vitesse sont responsables de ce phénomène. Pour ce faire, nous employons l'extrapolation de Richardson pour vérifier la solution de l'écoulement cisaillé et ainsi comparer les résultats obtenus avec ceux de la MMS.

En effet, bien que le taux de convergence théorique ne soit pas atteint pour ce test, la pression converge malgré tout vers la solution de manière monotone, telle qu'illustrée au Tableau 4.2. En effet, la valeur de R doit se trouver entre 0 et 1 pour avoir une convergence monotone. Il est à noter qu'il y a une exception à cette convergence vers 6400 éléments de maillages où nous remarquons une augmentation disproportionnée du ratio de convergence. Cependant, cette prédiction de

l'erreur n'est pas valable, car elle est basée sur les valeurs se trouvant en amont de la zone asymptotique des variables u, v, k et ϵ (zone grisée).

Tableau 4.2: Ratio de convergence de la pression pour l'écoulement cisaillé

Nb élém.	R
20x20	0.03
40x40	0.10
80x80	2.88
160x160	0.02
320x320	0.18
640x640	0.82

En ce qui concerne l'estimation de l'erreur, nous remarquons au Tableau 4.3 qu'il y a un décalage d'ordre de grandeur entre celle estimée et celle calculée par la MMS. En effet, le taux de convergence ne semble pas avoir atteint la zone asymptotique pour la pression, malgré un maillage dense de 409600 éléments.

Tableau 4.3: Comparaison de l'erreur et du taux de convergence obtenus par Richardson des frontières gauche/droite avec ceux de la MMS

Nb élém.	$E_{gauche/droite}$	E_{mms}	$p_{gauche/droite}$	p_{mms}
5x5	-	6.74E-02	-	-
10x10	3.11E-02	3.14E-03	-	4.42
20x20	1.07E-03	7.87E-04	4.86	2.00
40x40	1.06E-04	1.06E-03	3.34	-0.43
80x80	3.05E-04	7.34E-05	-1.53	3.85
160x160	6.42E-06	2.36E-05	5.57	1.63
320x320	1.15E-06	1.19E-05	2.49	0.99
640x640	9.38E-07	6.54E-06	0.29	0.86

La variable d'intérêt est aussi calculée pour les frontières du haut et du bas du domaine. Une comparaison entre les erreurs extrapolées et les taux de convergences calculés se trouvent au Tableau 4.4. Alors que nous savons que l'erreur sur la pression est plus importante en entrée et en sortie du domaine, il est intéressant de constater qu'on n'obtient pas nécessairement un meilleur taux de convergence observé pour les frontières du haut et du bas. Il semblerait alors que le taux de convergence observé de la pression fluctue sur l'ensemble du domaine.

Tableau 4.4: Comparaison de l'erreur et du taux de convergence obtenus par Richardson des frontières gauche/droite avec les frontières haut/bas

Nb élém.	$E_{gauche/droite}$	$E_{haut/bas}$	$p_{gauche/droite}$	$p_{haut/bas}$
100	3.11E-02	3.26E-02	-	-
400	1.07E-03	9.26E-05	4.86	8.46
1600	1.06E-04	8.72E-05	3.34	0.09
6400	3.05E-04	5.18E-05	-1.53	0.75
25600	6.42E-06	4.00E-06	5.57	3.69
102400	1.15E-06	2.34E-07	2.49	4.10
409600	9.38E-07	1.34E-07	0.29	0.81

Le calcul de la solution pour l'écoulement de type couche cisaillé est donc pollué par une source d'erreur qui n'est pas présente dans l'écoulement turbulent libre. À ce sujet, nous suspectons les conditions limites de Neumann variables sur la vitesse d'être à l'origine de cette erreur, car elles distinguent cet écoulement des autres cas étudiés précédemment.

Ce sachant, OpenFOAM atteint tout de même le taux de convergence théorique pour la norme L_2 de l'erreur pour les autres tests effectués, démontrant que le code est implémenté adéquatement lorsque des conditions limites *zeroGradient* sont employées. Alors que la norme L_∞ de l'erreur de la pression se détériore pour les écoulements 2D, nous avons constaté qu'elle était le résultat d'un phénomène localisé qui n'affecte pas le taux de convergence observé aux frontières.

Nous avons toutefois relevé une lacune dans l'implémentation du code qui semble être causée par une condition limite de Neumann variable sur la vitesse. Cette lacune engendre un bruit qui détériore le taux de convergence observé pour la pression pour les deux normes d'erreurs employées. Cependant, il reste toujours à proposer une correction à l'implémentation de cette condition limite afin de confirmer qu'elle est bien responsable de la perte du taux de convergence.

CHAPITRE 5 CONCLUSION ET RECOMMANDATIONS

5.1 Synthèse

Nous avons atteint le but de ce projet, soit de faire la vérification du code OpenFOAM pour un écoulement turbulent, 2D et incompressible. Nous avons modélisé la turbulence à l'aide du modèle k-ϵ standard. Malgré ses lacunes, ce modèle répond bien aux besoins actuels de l'industrie. En effet, c'est un modèle de turbulence offrant un bon compromis entre l'exactitude des résultats obtenus et le coût de calcul requis. Nous avons toutefois remarqué qu'il existe déjà une littérature exhaustive sur le sujet, mais qu'aucune de ces études ne fait état de son implémentation dans le code OpenFOAM. Ainsi, dans ce travail de recherche nous avons cherché à combler ce manque en employant des méthodes rigoureuses de vérification du code.

Notre premier objectif était d'appliquer la méthode des solutions manufacturées à des écoulements laminaires. Nous avons montré que le taux de convergence théorique était atteint sans difficulté pour les écoulements laminaires 1D. Pour les écoulements 2D, nous avons constaté que la norme L∞ de l'erreur de la pression diminue selon $O(h)$ plutôt que $O(h^2)$, sans pourtant affecter la norme L_2 de l'erreur.

L'analyse a ensuite porté sur notre deuxième objectif, soit la vérification de code pour le modèle de turbulence k-ϵ standard. Cette étape nous a permis de constater une détérioration du taux de convergence observé pour la pression. En effet, la solution manufacturée de l'écoulement de type couche cisaillée comporte des conditions limites de Neumann variables qui pourrait être à l'origine d'un bruit dans la solution de la pression.

Notre troisième objectif a été d'employer l'extrapolation de Richardson afin de confirmer le taux de convergence observé pour les deux écoulements turbulents étudiés. Nous avons démontré que l'erreur maximale localisée des écoulements 2D n'affecte pas le taux de convergence aux frontières de l'écoulement. Leur présence n'a donc qu'un impact minime sur les calculs usuels en ingénierie, tel le calcul de la perte de charge. Cependant, nous suggérons tout de même d'user de précaution lorsque des calculs sont effectués près des éléments de maillage situés à l'intersection de deux frontières comportant des conditions de Dirichlet.

Dans le cas de l'écoulement de type couche cisaillée, nous avons montré que la solution de la pression converge vers une autre solution que celle de la solution manufacturée. Il est possible que ce phénomène soit dû au bruit causé par une faute dans l'implémentation des conditions limites de Neumann variables dans OpenFOAM. Cependant, nous n'avons pas développé de solution à cette implémentation dans ce travail afin de confirmer notre hypothèse.

5.2 Limitations de l'étude

La portée de cette étude est principalement limitée par les hypothèses de l'écoulement employé ainsi que par les maillages utilisés. En effet, nous avons vérifié le code uniquement pour des écoulements 1D et 2D. La grande complexité des écoulements réels limite fortement la possibilité de simplifier leurs géométries de manière à en faire des modèles 2D. La vérification de code pour un écoulement turbulent 3D serait donc nécessaire dans une étude future afin de mieux représenter les écoulements réels.

Pour la présente étude, nous avons employé des maillages composés de quadrilatères alors qu'en pratique il est courant d'employer des maillages composés de triangles pour des géométries complexes. De plus, nous n'avons pas considéré les maillages non orthogonaux, car nous savions au préalable que le taux de convergence théorique ne serait pas atteint. Nous notons que l'utilisateur est tout de même informé que le taux de convergence observé tend vers $U(h^2)$ pour un maillage quasi orthogonal (minimalement non orthogonal). Il est alors de la responsabilité de l'utilisateur de limiter la non-orthogonalité de son maillage le plus possible afin de conserver un taux de convergence acceptable.

De plus, aucune solution n'a été proposée dans cette étude afin de corriger l'implémentation de la condition de Neumann variable. Nous suspectons cette condition limite d'être la cause de la détérioration du taux de convergence, mais il ne nous a pas été possible de confirmer cette hypothèse.

5.3 Améliorations proposées

Dans le cadre de ce travail, nous avons identifié une lacune dans l'implémentation du code qui détériore le taux de convergence observé de la pression. Notre hypothèse est que cette détérioration du taux de convergence est causée par la condition limite *fixedGradient*. Cependant, il reste

toujours à proposer une correction à l'implémentation actuelle afin de confirmer notre hypothèse. Cette correction faciliterait grandement la tâche aux prochains usagers désirant vérifier le code par la méthode des solutions manufacturées. En effet, plusieurs solutions manufacturées retrouvées dans la littérature nécessitent des conditions limites de Neumann variables sur la vitesse et la pression.

Une fois cette erreur d'implémentation corrigée, cela simplifierait l'élaboration d'une solution manufacturée pour la vérification de code d'un écoulement turbulent 3D. Il serait alors important de confirmer qu'il n'y a pas d'erreur non détectée jusqu'à présent qui influence le taux de convergence lorsqu'on passe à un écoulement de dimension supérieure, comme cela fût le cas dans cette étude en passant d'un écoulement 1D à un écoulement 2D.

Finalement, il reste toujours un grand nombre de solveurs et conditions limites disponibles dans OpenFOAM n'ayant pas encore été vérifiés. Notre étude porte sur un écoulement incompressible en régime permanent, mais OpenFOAM offre aussi une panoplie d'autres solveurs permettant de résoudre divers types d'écoulements. Nous avons vérifié des conditions limites de Dirichlet et Neumann, mais l'utilisateur a à sa portée plusieurs conditions limites dérivées de ceux-ci. Les utilisateurs emploient certaines conditions limites selon le type d'écoulement à simuler et nous devons nous assurer que celles-ci ne détériorent pas le taux de convergence observé.

RÉFÉRENCES

Ahsan, M. (2014). Numerical analysis of friction factor for a fully developed turbulent flow using k–ε turbulence model with enhanced wall treatment. *Beni-Suef University Journal of Basic and Applied Sciences, 3*(4), 269-277. doi:10.1016/j.bjbas.2014.12.001

Albets-Chico, X., Pérez-Segarra, C. D., Oliva, A., & Bredberg, J. (2008). Analysis of wall-function approaches using two-equation turbulence models. *International Journal of Heat and Mass Transfer, 51*(19-20), 4940-4957. doi:10.1016/j.ijheatmasstransfer.2008.03.002

Andreini, A., Cerutti, M., Facchini, B., & Mangani, L. (2008). *Modeling of turbulent combustion and radiative heat transfer in a object-oriented CFD code for gas turbine application.* Communication présentée à ASME Turbo Expo 2008: Power for Land, Sea, and Air, Berlin, Allemagne (p. 809-822). doi:10.1115/gt2008-51117

Argyropoulos, C. D., & Markatos, N. C. (2015). Recent advances on the numerical modelling of turbulent flows. *Applied Mathematical Modelling, 39*(2), 693-732. doi:10.1016/j.apm.2014.07.001

Ashton, N., & Skaperdas, V. (2019). Verification and validation of OpenFOAM for high-lift aircraft flows. *Journal of Aircraft*, 1-17. doi:10.2514/1.C034918

Bohorquez, P., & Parras, L. (2011). Three-dimensional numerical simulation of the wake flow of an afterbody at subsonic speeds. *Theoretical and Computational Fluid Dynamics, 27*(1-2), 201-218. doi:10.1007/s00162-011-0251-9

Bricteux, L., Zeoli, S., & Bourgeois, N. (2017). Validation and scalability of an open source parallel flow solver. *Concurrency and Computation: Practice and Experience, 29*(21), e4330. doi:10.1002/cpe.4330

Caretto, L. S., Gosman, A. D., Patankar, S. V., & Spalding, D. B. (1973). *Two calculation procedures for steady, three-dimensional flows with recirculation.* Communication présentée à Proceedings of the Third International Conference on Numerical Methods in Fluid Mechanics (p. 60-68). doi:10.1007/BFb0112677

Chaoqun, L., Yonghua, Y., & Yong, Y. (2014). *DNS study on eddy viscosity turbulence model* (Rapport n^{o} 2014-09). University of Texas at Arlington. Tiré de http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.640.8464&rep=rep1&type=pdf

Colin, E., Etienne, S., Pelletier, D., & Borggaard, J. (2005). Application of a sensitivity equation method to turbulent flows with heat transfer. *International Journal of Thermal Sciences, 44*(11), 1024-1038. doi:10.1016/j.ijthermalsci.2005.04.002

Constant, E., Favier, J., Meldi, M., Meliga, P., & Serre, E. (2017). An immersed boundary method in OpenFOAM : verification and validation. *Computers & Fluids, 157*, 55-72. doi:10.1016/j.compfluid.2017.08.001

Deardorff, J. W. (1970). A numerical study of three-dimensional turbulent channel flow at large Reynolds numbers. *Journal of Fluid Mechanics, 41*(02), 453-480. doi:10.1017/s0022112070000691

Eça, L., Hoekstra, M., Hay, A., & Pelletier, D. (2007). Verification of RANS solvers with manufactured solutions. *Engineering with Computers, 23*(4), 253-270. doi:10.1007/s00366-007-0067-9

Eça, L., Hoekstra, M., Hay, A., & Pelletier, D. (2010). A manufactured solution for a two-dimensional steady wall-bounded incompressible turbulent flow. *International Journal of Computational Fluid Dynamics, 21*(3-4), 175-188. doi:10.1080/10618560701553436

Eça, L., Pereira, F. S., & Vaz, G. (2018). Viscous flow simulations at high Reynolds numbers without wall functions: Is y+≃1 enough for the near-wall cells? *Computers & Fluids, 170*(2018), 157-175. doi:10.1016/j.compfluid.2018.04.035

Elsworth, C. W. (2014). *Verification of an overset-grid enabled fluid-structure interaction solver.* (Mémoire de maîtrise, Pennsylvania State University, State College, États-Unis). Tiré de https://etda.libraries.psu.edu/catalog/22622

Fisch, R., Franke, J., Wüchner, R., & Bletzinger, K.-U. (2014). *Code verification of a partitioned FSI environment for wind engineering applications using the method of manufactured solutions.* Communication présentée à World Congr. Comput. Mech., WCCM, Eur. Conf. Comput. Mech., ECCM Eur. Conf. Comput. Fluid Dyn., ECFD, Barcelone, Espagne (p. 2186-2197). Tiré de http://congress.cimne.com/iacm-eccomas2014/admin/files/fileabstract/a2011.pdf

Fisch, R. F., Jörg; Wüchner, Roland; Bletzinger, Kai-Uwe. (2012). *Code verification of OpenFOAM® solvers using the method of manufactured solutions.* Communication présentée à 7th OpenFOAM Workshop Darmstadt, Darmstadt, Allemagne. Tiré de

https://sourceforge.net/projects/openfoam-extend/files/OpenFOAM_Workshops/OFW7_2012_Darmstadt/Workshop-Documents/Presentations-Talks/FischRupert/

Foroutan, H., Tang, W., Heist, D. K., Perry, S. G., Brouwer, L. H., & Monbureau, E. M. (2018). Numerical analysis of pollutant dispersion around elongated buildings: An embedded large eddy simulation approach. *Atmos Environ (1994), 187*(2018), 117-130. doi:10.1016/j.atmosenv.2018.05.053

Furbo, E. (2010). *Evaluation of RANS turbulence models for flow problems with significant impact of boundary layers.* (Mémoire de maîtrise, Uppsala University, Uppsala, Suède). Tiré de http://www.diva-portal.org/smash/record.jsf?pid=diva2%3A379743&dswid=4506

Fureby, C., Anderson, B., Clarke, D., Erm, L., Henbest, S., Giacobello, M., . . . Zhu, S. (2016). Experimental and numerical study of a generic conventional submarine at 10° yaw. *Ocean Engineering, 116*(2016), 1-20. doi:10.1016/j.oceaneng.2016.01.001

Gatski, T. B., Rumsey, C. L., Launder, B. E., & Sandham, N. D. (2002). Linear and nonlinear eddy viscosity models. Dans *Closure strategies for turbulent and transitional flows* (p. 9-46). doi:10.1017/cbo9780511755385.003

Geurts, B. J. (2009). Analysis of errors occurring in Large Eddy Simulation. *Philosophical Transactions of the Royal Society, 367*(1899), 2873-2883. doi:10.1098/rsta.2009.0001

Gicquel, L. Y. M., Staffelbach, G., & Poinsot, T. (2012). Large Eddy Simulations of gaseous flames in gas turbine combustion chambers. *Progress in Energy and Combustion Science, 38*(6), 782-817. doi:10.1016/j.pecs.2012.04.004

Greenshields, C. J. (2015). *OpenFOAM - Programmer's guide*. Tiré de foam.sourceforge.net/docs/Guides-a4/ProgrammersGuide.pdf

Greenshields, C. J. (2018). *OpenFOAM User guide* (6e éd.). Tiré de http://foam.sourceforge.net/docs/Guides-a4/OpenFOAMUserGuide-A4.pdf

Hanjalic, K. (2004). *Closure models for incompressible turbulent flows.* (Delft University of Technology, Delft, Pays-Bas). Tiré de https://www.researchgate.net/publication/292444554_Closure_models_for_incompressible_turbulent_flows

Holzmann, T. (2018). Mathematics, numerics, derivations and OpenFOAM(R). doi:10.13140/RG.2.2.27193.36960

Ilinca, F., & Pelletier, D. (1998). Positivity preservation and adaptive solution for the k-epsilon model of turbulence. *AIAA Journal, 36*(1), 44-50. doi:10.2514/2.350

Ilinca, F., Pelletier, D., & Garon, A. (1994). *An adaptive finite element method for a two-equation turbulence model in wail bounded flows.* Communication présentée à AIAA Fluid Dynamics Conference, Colorado Springs, CO. doi:10.2514/6.1994-2390

Johansson, M. (2012). *Hydrodynamic investigation of an axisymmetric streamlined body with respect to the velocity distribution in the stern region.* (Mémoire de maîtrise, KTH School of Engineering Sciences (SCI), Stockholm, Suède). Tiré de http://www.diva-portal.org/smash/record.jsf?pid=diva2%3A565743&dswid=5947

Jones, D. A., Chapuis, M., Liefvendahl, M., Norrison, D., & Widjaja, R. (2016). *RANS simulations using OpenFOAM software* (Rapport n^{o} AD1002391). Fishermans Bend, Australie: Defence Science and Technology Group. Tiré de http://www.dtic.mil/docs/citations/AD1002391

Jones, W. P., & Launder, B. E. (1972). The prediction of laminarization with a two-equation model of turbulence. *International Journal of Heat and Mass Transfer, 15*(2), 301-314. doi:10.1016/0017-9310(72)90076-2

Kärrholm, F. P. (2006). Rhie-Chow interpolation in OpenFOAM. Tiré de http://www.tfd.chalmers.se/~hani/kurser/OS_CFD_2007/rhiechow.pdf

Kawai, S., & Larsson, J. (2012). Wall-modeling in Large Eddy Simulation: length scales, grid resolution, and accuracy. *Physics of Fluids, 24*(1), 015105. doi:10.1063/1.3678331

Kim, J., Moin, P., & Moser, R. (1987). Turbulence statistics in fully developed channel flow at low Reynolds number. *Journal of Fluid Mechanics, 177*(1987), 133-166. doi:10.1017/S002211208700089

Lacasse, D., Turgeon, É., & Pelletier, D. (2001). Prediction of turbulent separated flow in a turnaround duct using wall functions and adaptivity. *International Journal of Computational Fluid Dynamics, 15*(3), 209-225. doi:10.1080/10618560108970030

Lacasse, D., Turgeon, É., & Pelletier, D. (2004). On the judicious use of the k–ε model, wall functions and adaptivity. *International Journal of Thermal Sciences, 43*(10), 925-938. doi:10.1016/j.ijthermalsci.2004.03.004

Lacombe, F. (2017). *Vérification et validation d'une loi de paroi consistante du modèle de turbulence k−omega SST.* (Mémoire de maîtrise, École Polytechnique de Montréal, Montréal, QC). Tiré de https://publications.polymtl.ca/2557/

Launder, B. E., & Sharma, B. I. (1974). Application of the energy-dissipation model of turbulence to the calculation of flow near a spinning disc. *Letters in Heat and Mass Transfer, 1*(2), 131-137. doi:10.1016/0094-4548(74)90150-7

Launder, B. E., & Spalding, D. B. (1974). The numerical computation of turbulent flows. *Computer Methods in Applied Mechanics and Engineering, 3*(2), 269-289. doi:10.1016/0045-7825(74)90029-2

Liu, Q., Gómez, F., Pérez, J. M., & Theofilis, V. (2016). Instability and sensitivity analysis of flows using OpenFOAM®. *Chinese Journal of Aeronautics, 29*(2), 316-325. doi:10.1016/j.cja.2016.02.012

Liu, S. (2016). *Implementation of a complete wall function for the standar k—ϵ turbulence model in OpenFOAM 4.0*, Chalmers University of Technology, Göteborg, Suède. Tiré de http://www.tfd.chalmers.se/~hani/kurser/OS_CFD_2016

Mangani, L. (2008). *Development and validation of an object oriented CFD solver for heat transfer and combustion modeling in turbomachinery applications.* (Thèse de doctorat, Universita degli Studi di Firenze, Florence, Italie). Tiré de https://foam-extend.fsb.hr/wp-content/uploads/2017/01/Mangani_PhD_2008.pdf

Mohammadi, B., & Puigt, G. (2006). Wall functions in computational fluid mechanics. *Computers & Fluids, 35*(10), 1108-1115. doi:10.1016/j.compfluid.2005.02.009

Moin, P., & Mahesh, K. (1998). Direct numerical simulation: A tool in turbulence research. *Annual Review of Fluid Mechanics, 30*(1), 539-578. doi:10.1146/annurev.fluid.30.1.539

Moukalled, F., Mangani, L., & Darwish, M. (2016). *The finite volume method in computational fluid dynamics: An advanced introduction with OpenFOAM® and Matlab®* Fluid Mechanics and its Applications, (1e éd.). doi:10.1007/978-3-319-16874-6

Nikitin, N. V., Nicoud, F., Wasistho, B., Squires, K. D., & Spalart, P. R. (2000). An approach to wall modeling in Large-Eddy Simulations. *Physics of Fluids, 12*(7), 1629-1632. doi:10.1063/1.870414

Nilsson, H. (2006). *Evaluation of OpenFOAM for CFD of turbulent flow in water turbines.* Communication présentée à 23rd IAHR Symposium, Yokohama, Japon (p. 1-9). Tiré de https://pdfs.semanticscholar.org/75c6/bc59c0b9c4e219e0a52ad994bf89dca579f9.pdf

Noriega, H., Guibault, F., Reggio, M., & Magnan, R. (2018a). A case-study in open-source CFD code verification, Part I: Convergence rate loss diagnosis. *Mathematics and Computers in Simulation, 147*(2018), 152-171. doi:10.1016/j.matcom.2017.12.002

Noriega, H., Guibault, F., Reggio, M., & Magnan, R. (2018b). A case-study in open-source CFD code verification. Part II: Boundary condition non-orthogonal correction. *Mathematics and Computers in Simulation, 147*(2018), 172-193. doi:10.1016/j.matcom.2017.12.001

OpenCFD. (2018). OpenFOAM® - Official home of the open source computational fluid dynamics (CFD) toolbox. Tiré de https://www.openfoam.com/

Persson, S. (2017). *Development of a test suite for verification & validation of OpenFOAM.* (Mémoire de maîtrise, Chalmers University of Technology, Göteborg, Suède). Tiré de http://studentarbeten.chalmers.se/publication/250253-development-of-a-test-suite-for-verification-validation-of-openfoam

Piomelli, U. (2001). *Large-eddy and direct simulation of turbulent flows*. Communication présentée à CFD2001 – 9e conférence annuelle de la société Canadienne de CFD, Kitchener, Canada. Tiré de http://rsta.royalsocietypublishing.org/content/367/1899/2873

Pope, S. B. (2000). *Turbulent flows* (1e éd.). New York, États-Unis: Cambridge University Press.

Prandtl, L. (1925). Bericht über untersuchungen zur ausgebildeten turbulenz (N. A. C. f. Aeronautics, trad.) (vol. 5, No. 2, p. 136-139). Washington, États-Unis: National Advisory Comittee for Aeronautics.

Roache, P. J. (1998). *Verification and validation in computational science and engineering*. Albuquerque, New Mexico: Hermosa.

Robertson, E., Choudhury, V., Bhushan, S., & Walters, D. K. (2015). Validation of OpenFOAM numerical methods and turbulence models for incompressible bluff body flows. *Computers & Fluids, 123*(2015), 122-145. doi:10.1016/j.compfluid.2015.09.010

Röhrig, R., Jakirlić, S., & Tropea, C. (2015). Comparative computational study of turbulent flow in a 90° pipe elbow. *International Journal of Heat and Fluid Flow, 55*(2015), 120-131. doi:10.1016/j.ijheatfluidflow.2015.07.011

Roy, C. J. (2005). Review of code and solution verification procedures for computational simulation. *Journal of Computational Physics, 205*(1), 131-156. doi:10.1016/j.jcp.2004.10.036

Sakri, F. M., Ali, M. S. M., & Salim, S. A. Z. S. (2016). Computational investigations and grid refinement study of 3D transient flow in a cylindrical tank using OpenFOAM. *IOP Conference Series: Materials Science and Engineering, 152*(2016), 012058. doi:10.1088/1757-899x/152/1/012058

Schmitt, F. G. (2007). About Boussinesq's turbulent viscosity hypothesis: Historical remarks and a direct evaluation of its validity. *Comptes Rendus Mécanique, 335*(9-10), 617-627. doi:10.1016/j.crme.2007.08.004

Sebastian Muntean, H. N., Romeo Resiga. (2009). *3D numerical analysis of the unsteady turbulent swirling flow in a conical diffuser using Fluent and Openfoam.* Communication présentée à 3rd IAHR International Meeting of the Workgroup on Cavitation and Dynamic Problems in Hydraulic Machinery and Systems, Brno, République Tchèque (p. 155-164). Tiré de https://www.researchgate.net/publication/250612504_3D_NUMERICAL_ANALYSIS_OF_THE_UNSTEADY_TURBULENT_SWIRLING_FLOW_IN_A_CONICAL_DIFFUSER_USING_FLUENT_AND_OPENFOAM

Shan, H., Delaney, K., Kim, S.-E., Rhee, B., Gorski, J., & Ebert, M. (2011). *Guide to NavyFOAM V1.0* (Rapport n^{o} NSWCCD-50-TR-2011/025). West Bethesda, États-Unis: Naval Surface Warfare Center Carderock Division. Tiré de http://www.dtic.mil/docs/citations/ADA542846

Smagorinsky, J. (1963). General circulation experiments with the primitive equations. *Monthly Weather Review, 91*(3), 99-164. doi:10.1175/1520-0493(1963)091<0099:Gcewtp>2.3.Co;2

Spalart, P., & Allmaras, S. (1992). *A one-equation turbulence model for aerodynamic flows*. Communication présentée à 30th Aerospace Sciences Meeting and Exhibit, Reno, États-Unis.

Spalart, P. R. (2015). Philosophies and fallacies in turbulence modeling. *Progress in Aerospace Sciences, 74*(2015), 1-15. doi:10.1016/j.paerosci.2014.12.004

Tremblay, D., Etienne, S., & Pelletier, D. (2006). *Code verification and the method of manufactured solutions for fluid-structure interaction problems.* Communication présentée à 36th AIAA Fluid Dynamics Conference and Exhibit, San Francisco, États-Unis. doi:https://doi.org/10.2514/6.2006-3218

Wilcox, D. C. (1998). *Turbulence modeling for CFD* (2e éd.). La Canada, États-Unis: DCW industries, Inc.

Zhao, C.-R., Zhang, Z., Jiang, P.-X., & Bo, H.-L. (2017). Influence of various aspects of low Reynolds number k - ε turbulence models on predicting in-tube buoyancy affected heat transfer to supercritical pressure fluids. *Nuclear Engineering and Design, 313*(2017), 401-413. doi:10.1016/j.nucengdes.2016.12.033

ANNEXE A – CODE DU FICHIER FVOPTIONS POUR L'ÉCOULEMENT DE TYPE COUCHE LIMITE

```
/*--------------------------------*- C++ -*----------------------------------*\
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     | Website:  https://openfoam.org
    \\  /    A nd           | Version:  6
     \\/     M anipulation  |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "constant";
    object      fvOptions;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

    momentumSource
    {
        type            vectorCodedSource;
        active          true;
        name            velocitySource;

        vectorCodedSourceCoeffs
        {
            selectionMode   all;
            fields          (U);

            codeInclude
            #{

            #};

            codeCorrect
            #{

            #};

            codeAddSup
            #{
                const vectorField& C = mesh_.C();
                const scalarField& V = mesh_.V();
                vectorField& Usource = eqn.source();
                forAll(C, i)
                {
                    const scalar y = C[i].component(1);
                    const scalar vol = V[i];
                    const scalar fx = (-100.0*exp(-10.0*y)/(exp(-10.0)-1.0)-1.0)*vol;

                    Usource[i] -= vector(fx, 0, 0);
                }
                Pout << "***codeAddSup***" << endl;
            #};
```

```
        codeSetValue
        #{

        #};

        code
        #{
            $codeInclude
            $codeCorrect
            $codeAddSup
            $codeSetValue
        #};
    }

    velocitySourceCoeffs
    {
        $vectorCodedSourceCoeffs;
    }
}
```