جامعة تونس المنار
Université de Tunis El Manar

Département Génie Industriel

Projet de fin d'année II

Etude d'écoulements anisothermes par la méthode de la vorticité

Réalisé par :

Nabil ACHOUR

Classe :

2AMIndS

Encadré par :

Dr. Rodolphe HEYD

Année universitaire 2015/2016

"Il est bien des choses qui ne paraissent impossibles que tant qu'on ne les a pas tentées."

André Gide

Remerciements

J'adresse mes remerciements à mon encadreur de ce projet de fin d'année, Dr. Rodolphe HEYD. Je tiens à lui exprimer ma profonde reconnaissance pour ses précieux conseils et les innombrables heures qu'il m'a consacrées. La qualité de son encadrement, sa lucidité scientifiques, et son esprit de rigueur ont été primordiaux tout au long de projet.

Table des matières

Table des figures **vi**

Liste des tableaux **vii**

Nomenclature **viii**

Introduction générale **1**

1 Écoulement de poiseuille **2**
1.1 La méthode de vorticité 2
1.2 Résolution analytique et numérique 1D 3
1.2.1 Solution analytique 1D en RS 4
1.2.2 Solution numérique 1D 4
1.3 Résolution numérique 2D 9
1.3.1 La discrétisation des équations 9
1.3.2 Les conditions aux limites 10
1.3.3 Algorithme de résolution et visualisation des résultats 10
1.4 Étude des erreurs entre les courbes numériques et analytique de u_y 13
1.4.1 Erreur $\mid u_{ythe} - u_{ynume} \mid$ 13
1.4.2 Erreur $max(\mid u_{ythe} - u_{ynume} \mid)$ 15
1.4.3 Commentaires 15

2 La convection forcée **16**
2.1 Mise en équations 16
2.2 La discrétisation de l'équation de chaleur 16
2.3 Les conditions aux limites 17
2.4 Algorithme de résolution et visualisation de résultat 17
2.5 Influence de Re sur la couche limite thermique 18

3 La convection naturelle **21**
3.1 Mise en équations 21
3.1.1 Adimensionnement des équations 22
3.1.2 La méthode vortex 22
3.2 La discrétisation des équations 23
3.3 Algorithme de résolution 24
3.4 Les conditions aux limites et les résultats 25
3.4.1 Première configuration 25
3.4.2 Deuxième configuration 32

4 Conclusion **36**

Bibliographie **37**

A Codes Python accompagnant le rapport **38**
A.1 Écoulement de poiseuille 1D 38
A.2 Écoulement de poiseuille 2D 39
A.3 La Convection Forcée 42
A.4 La Convection Naturelle 43

Table des figures

1.1 Écoulement entre deux plans parallèles avec Lx << Ly. 2
1.2 Solution analytique 1D en RS de la composante de vitesse u_y pour Lx=1. . . . 4
1.3 Solution analytique 1D en RS de la composante de vitesse u_y pour Lx=1.8. . . 4
1.4 Résultats de la simulation 1D : la vorticité w, la composante de la vitesse u_y et la fonction de courant ψ. 8
1.5 Conditions aux limites de ω et ψ 2D. 10
1.6 La vorticité ω avec un nombre de reynolds Re=21.6. 11
1.7 La fonction de courant ψ, avec Re=21.6. 11
1.8 La composantes de la vitesse u_y numérique et analytique, pour Re=21.6. . . . 12
1.9 La composantes de la vitesse u_x, pour Re=21.6. 12
1.10 La divergence de la vitesse $\vec{u}$. 12
1.11 Les courbes d'erreur $\mid u_{ythe} - u_{ynume} \mid$ en fonction de x et pour un y fixé, pour des nombres de Reynolds varient de 10.8 et à 21.6. 13
1.12 Les courbes d'erreur $\mid u_{ythe} - u_{ynume} \mid$ en fonction de x et pour un y fixé, pour des nombres de Reynolds varient de 30.2 à 320.6. 14
1.13 Les courbes d'erreur $\mid u_{ythe} - u_{ynume} \mid$ en fonction de x et pour un y fixé, pour un nombre de Reynolds Re=540. 15
1.14 Erreur=$max(\mid u_{ythe} - u_{ynume} \mid)$ en fonction de Re. 15

2.1 Convection forcée : Conditions aux limites. 17
2.2 Résultat de la simulation de T lorsque le RS est atteint. 18
2.3 Le champ de Température pour Re de 12 à 280 avec x en abscisse et y en ordonné. 19
2.4 Le champ de Température pour Re=330 et Re=600, avec x est en abscisse et y en ordonné. 20

3.1 Conditions aux limites de la configuration 1. 25
3.2 La température dans l'air siège d'une CN pour des Ra allant de 300 à 4000. . . 27
3.3 La température dans l'eau siège d'une CN pour des Ra allant de 10 à 90 . . . 29
3.4 La température dans l'eau siège d'une CN pour des Ra allant de 100 à 300 . . 30
3.5 Le champ de la vitesse dans l'eau siège d'une CN pour des Ra allant de 100 à 300 lorsque le RS est atteint. 30
3.6 Le nombre de Nusselt moyen $\mathrm{N}u_{Ly}$ en fonction de Ra pour l'air. 31
3.7 Le nombre de Nusselt moyen $\mathrm{N}u_{Ly}$ en fonction de Ra pour l'eau. 32
3.8 Conditions aux limites de la configuration 2. 32
3.9 Convection libre pour Nx=20, Ny=40, Pr=1, Ra=20000, $T_c = 1$ et $T_f = 0$. 33
3.10 Convection libre pour Nx=20, Ny=80, Pr=1, Ra=20000, $T_c = 1$ et $T_f = 0$ 33
3.11 Les champs de la Température et de la vitesse pour Pr=1 et Ra=5000. 34
3.12 Les champs de la Température et de la vitesse pour Pr=1 et Ra=10^4. 34

3.13 Les champs de la Température et de la vitesse pour Pr=1 et Ra=2.10^4. 34
3.14 Les champs de la Température et de la vitesse pour Pr=1 et Ra=4.10^4 35
3.15 Les champs de la Température et de la vitesse pour Pr=1 et Ra=8.10^4 35
3.16 La période du motif en fonction de Ra. 35

Liste des tableaux

1.1 La discrétisation des dérivées partielles des équations de l'écoulement de poiseuille 1D. 4
1.2 La discrétisation des dérivées partielles des équations de l'écoulement de poiseuille 2D. 9
1.3 Erreur= max($| u_{yth} - u_{ynum} |$) en fonction de Re. 15

2.1 La discrétisation des dérivées partielles de l'équation de chaleur. 16

3.1 La discrétisation des dérivées partielles des équations pour la convection naturelle 23
3.2 La valeur numérique de $\mathrm{N}u_{Ly}$ en fonction de Ra pour l'air. 31
3.3 La valeur numérique de $\mathrm{N}u_{Ly}$ en fonction de Ra pour l'eau. 32

Nomenclature

T	La Température	$[K,^0C]$
ψ	La fonction de courant	$[m^2.S^{-1}]$
ν	La Viscosité cinématique	$[m^2.S^{-1}]$
ω	La Vorticité	$[S^{-1}]$
ρ	La densité volumique	$[kg.m^{-3}]$
g	Accélération de pesanteur	$[m^2.S^{-1}]$
t	Le temps	[S]
x	Distance, position suivant la direction horizontale	$[m]$
y	Distance, position suivant la direction verticale	$[m]$
u	La vitesse du fluide	$[m.S^{-1}]$
u_x	La composante de itesse verticale du fluide	$[m.S^{-1}]$
u_y	La composante de vitesse horizontal du fluide	$[m.S^{-1}]$
L_x	La distance entre les deux plans	$[m]$
L_y	La longueur des plans	$[m]$
$\triangle x$	Pas d'espace suivant la direction horizontale	$[m]$
$\triangle y$	Pas d'espace suivant la direction verticale	$[m]$
$\triangle t$	Pas temporelle	$[S]$
k	La conductivité thermique du fluide	$[W.m^{-1}.K^{-1}]$
c	La capacité calorifique massique ou chaleur spécifique	$[J.kg^{-1}.K^{-1}]$
μ	La viscosité dynamique $\mu = \rho\nu$	$[Pa.S^{-1}]$
p	La pression	$[Pa]$
Re	Nombre de Reynolds Re=$\frac{L_x u}{\nu}$	[-]
Pr	Nombre dPrandtl Pr=$\frac{\nu}{\kappa}$	[-]
Ra	Nombre de Rayleigh Ra=$\frac{\beta g L^3 \Delta T_{s,0}}{\kappa\nu}$	[-]
Gr	Nombre de Grashof Gr=$\frac{\beta g L^3 \Delta T_{s,0}}{\nu^2}$	[-]
Nu	Nombre de Nusselt Nu=$\frac{hL}{k}$	

Les abréviations

CFD	Computational Fluid Dynamics
CM	Équation de Conservation de la Masse
NS	Équation de Navier Stoks
SOR	Successive Over-Relaxation
ECF	Équation de la Chaleur dans un Fluide
CFL	Courant-Friedrichs-Lewy
RS	Régime Stationnaire
CL	Convection Libre
CN	Convection Naturelle

Introduction générale

En ingénierie on utilise de plus en plus les simulations numériques au détriment des simulations expérimentales qui sont très couteuses que se soit en temps ou en argent. C'est pourquoi on a souvent un recours à des approches de modélisation numériques en utilisant les méthodes numériques comme les différences finies (FDM), les éléments finis (FEM), les volumes finis (FVM), les méthodes spectrales...
En tant qu'élèves ingénieurs de la filière Modélisation pour l'Industrie et les Services (MInds) spécialité modélisation et simulation numérique, nous avons acquis dans notre cursus les connaissances nécessaires pour la modélisation et la simulation des problèmes très rencontrés en ingénierie, en transfert thermique, en mécanique et mécanique des fluides...
La convection thermique est un mode de transfert de chaleur qui a une importance vitale aussi bien dans le domaine fondamental que dans le domaine des applications telles que, la climatisation et le chauffage dans le domaine d'habitat, le refroidissement dans les thermosiphons et les systèmes mécanique ou électronique, les système de chauffage de fluides (eau ou air) impliquant les capteur solaires, le séchage des produit agro-alimentaires, les échangeurs de chaleur et les serres dans le domaine agricole. Pour ce mode de transfert de chaleur (convection) on distingue : la convection forcée, pour laquelle le mouvement du fluide est produit par une action extérieur (pompe, ventilateur, vent...) et le convection naturelle ou libre, pour laquelle le mouvement du fluide résulte simplement de la différence de la masse volumique entre les régions chaude et froide du fluide en présence d'un champ de gravité.
Depuis plusieurs décennies, le transfert de chaleur par convection naturelle a fait l'objet de nombreux travaux de recherche et offre également un champ d'application diversifié. Dans ce projet, on se propose de faire l'étude d'un écoulement de poiseuille bidimensionnel, le phénomène de convection naturelle et de convection forcée en se basant sur la méthode de vortex qu'est une méthode parmi plusieurs autres qui va nous permette de simplifier les équations mathématique qui modélisent ces phénomènes physiques, tout en utilisant la méthodes de différences finies (FDM) qui va nous permette la résolution et l'implémentation numérique avec Python, un langage de programmation portable, dynamique, extensible, gratuit, qui permet (sans l'imposer) une approche modulaire et orientée objet de la programmation. Python est développe depuis 1989 par Guido van Rossum et de nombreux contributeurs bénévoles **[2]** Dont les domaines d'application sont : l'apprentissage de la programmation objet, les développement liés à l'Internet et en particulier au Web, l'accès aux bases de données, la réalisation d'interfaces graphiques utilisateurs, le prototypage rapide d'applications et bien évidemment le calcul scientifique.

CHAPITRE

1 Écoulement de poiseuille

La majorité des écoulements des fluides sont modélisés par l'équation de navier stokes (NS), et dans cette même perspective, on suppose que nous avons un écoulement d'un fluide incompressible, bidimensionnel (dans le plan (X,Y))et entre deux plans parallèles de longueurs infinies comme le montre la figure 1.1.

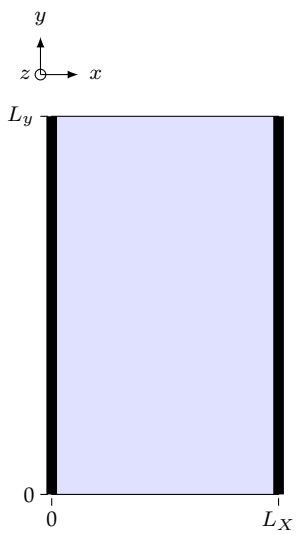


FIGURE 1.1 – Écoulement entre deux plans parallèles avec Lx << Ly.

Donc le problème qu'on cherche à résoudre est modélisé par le système des équations suivant **[1]** :

$$(S_1) : \begin{cases} \nabla.\vec{u} = 0 & (C.M) \\ \rho\partial_t\vec{u} + \rho(\vec{u}.\nabla)\vec{u} = -\nabla p + \rho\vec{g} + \mu\nabla^2\vec{u} & (N.S) \end{cases}$$

1.1 La méthode de vorticité

Exprimer les équations incompressibles de Navier-Stokes en termes de variables autres que la vitesse et la pression devient une bonne alternative pour les résoudre numériquement.
Donc pour résoudre le système (S_1), on va appliquer une méthode appelée la méthode de vortex qui consiste à appliquer le rotationnel à l'équation de (N.S) pour faire disparaitre le terme de pression, afin d'obtenir une équation simple et indépendante de la pression inconnue **[3]** :

$$\rho\partial_t\nabla\times\vec{u} + \rho(\vec{u}.\nabla)(\nabla\times\vec{u}) - \rho((\nabla\times\vec{u}).\nabla)\vec{u} = -\nabla\times\nabla p + \rho\nabla\times\vec{g} + \mu\nabla\times\nabla^2\vec{u} \qquad (1.1)$$

Pour simplifier on prend $\vec{g} = \vec{0}$:

$$\rho\partial_t \nabla \times \vec{u} + \rho(\vec{u}.\nabla)\nabla \times \vec{u} - \rho((\nabla \times \vec{u}).\nabla)\vec{u} = \mu\nabla^2\nabla \times \vec{u} \tag{1.2}$$

Or, on a le vecteur de vortcité qu'est définit par cette expression :

$$\vec{\omega} = \nabla \times \vec{u} = (\partial_x u_y - \partial_y u_x)\vec{e_z} = \omega\vec{e_z} \tag{1.3}$$

Donc [1] :

$$\partial_t\vec{\omega} + (\vec{u}.\nabla)\vec{\omega} = \nu\nabla^2\vec{\omega} \tag{1.4}$$

On introduit aussi le vecteur de la fonction de courant $\vec{\psi} = \psi(x, y)\vec{e_z}$ tel que :

$$\vec{u} = \nabla \times \vec{\psi} = \partial_y\psi(x, y)\vec{e_x} - \partial_x\psi(x, y)\vec{e_y} \tag{1.5}$$

On remarque que :

$$\nabla^2\vec{\psi} = \triangle\vec{\psi} = (\partial_{xx}\psi + \partial_{yy}\psi)\vec{e_z} = -\vec{\omega} \tag{1.6}$$

L'équation (1.4) devient alors après projection :

$$\partial_t\omega + \partial_y\psi\partial_x\omega - \partial_x\psi\partial_y\omega = \nu\triangle\omega \tag{1.7}$$

D'où le système qu'on cherche à résoudre devient :

$$(S_1') : \begin{cases} \partial_t\omega + \partial_y\psi\partial_x\omega - \partial_x\psi\partial_y\omega = \nu(\partial_{xx}\omega + \partial_{yy}\omega) \\ \partial_{xx}\psi + \partial_{yy}\psi = -\omega \end{cases}$$

1.2 Résolution analytique et numérique 1D

Dans cette section nous faisons des simplification sur le champ de vitesse $\vec{u}$ selon la géométrie que nous avons dans la figure 1.1 :
Puisque on a un écoulement parallèle alors $\vec{u} = u_y(x, y)\vec{e_y}$.
D'après l'équation de conservation de la masse (CM), on a $\nabla.\vec{u} = 0$ alors $\partial_y u_y = 0$, donc on aura $\vec{u} = u_y(x)\vec{e_y}$.
Par suite, les équations deviennent :

$$\omega = \partial_x u_y \tag{1.8}$$

$$\partial_x\psi = -u_y \tag{1.9}$$

$$\partial_t\omega = \nu\partial_{xx}\omega \tag{1.10}$$

$$\partial_{xx}\psi = -\omega \tag{1.11}$$

1. Nous avons un écoulement bidimensionnel donc $((\nabla \times \vec{u}).\nabla)\vec{u} = \vec{0}$

1.2.1 Solution analytique 1D en RS

On calcule la solution analytique en régime stationnaire pour pouvoir la comparer avec la solution numérique.
En régime stationnaire l'équation (1.10) donne :

$$\partial_{xx}\omega = 0 \tag{1.12}$$

Ce qui implique que ω s'écrit sous la forme suivante :

$$\omega = ax + b \tag{1.13}$$

Les condition de Dirichlet suivantes $\omega(0) = 1$ et $\omega(L_x) = -1$, permettent de déterminer a et b et par suite ω :

$$\omega = \frac{-2}{L_x}x + 1 \tag{1.14}$$

Et on a :

$$\partial_x u_y = w \tag{1.15}$$

Donc :

$$u_y = \frac{-1}{L_x}x^2 + x \tag{1.16}$$

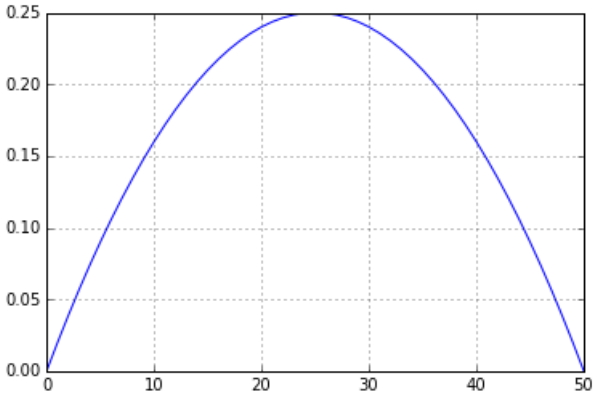


FIGURE 1.2 – Solution analytique 1D en RS de la composante de vitesse u_y pour Lx=1.

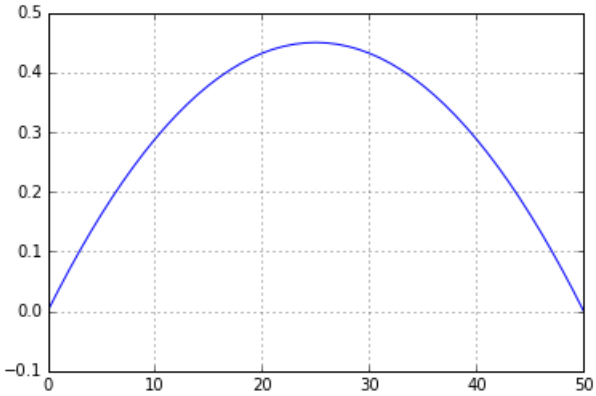


FIGURE 1.3 – Solution analytique 1D en RS de la composante de vitesse u_y pour Lx=1.8.

1.2.2 Solution numérique 1D

Il est clair qu'il suffit d'utiliser les trois équations (1.8), (1.9) et (1.10) pour trouver ω, u_y et ψ. Pour résoudre ces équation numériquement on utilise la méthode de différences finies : on considère ici un schéma implicite en temps, notant i l'indice spatial (i $\in$ [0, Nx]) et n l'indice temporel (n > 0), on discrétise les différentes dérivées partielles des équations sous la forme suivante :

La dérivée partielle	La discrétisation	La dérivée partielle	La discrétisation
$\partial_t\omega$	$\frac{\omega_i^{n+1}-\omega_i^n}{\triangle t}$	$\partial_x u_y$	$\frac{u_{yi+1}^n-u_{yi}^n}{\triangle x}$
$\partial_x\psi$	$\frac{\psi_{i+1}^n-\psi_i^n}{\triangle x}$	$\partial_{xx}\omega$	$\frac{\omega_{i+1}^{n+1}-2\omega_i^{n+1}+\omega_{i-1}^{n+1}}{(\triangle x)^2}$

TABLE 1.1 – La discrétisation des dérivées partielles des équations de l'écoulement de poiseuille 1D.

La vorticité ω :

$$\partial_t \omega = \nu \partial_{xx} \omega \rightsquigarrow \frac{\omega_i^{n+1} - \omega_i^n}{\triangle t} = \nu \frac{\omega_{i+1}^{n+1} - 2\omega_i^{n+1} + \omega_{i-1}^{n+1}}{(\triangle x)^2} \tag{1.17}$$

Alors :

$$\omega_i^{n+1} = \omega_i^n + \nu \frac{\triangle t}{(\triangle x)^2} (\omega_{i+1}^{n+1} - 2\omega_i^{n+1} + \omega_{i-1}^{n+1}) \tag{1.18}$$

Et on pose $\alpha = \nu \frac{\triangle t}{(\triangle x)^2}$:

$$\omega_i^n = -\alpha \omega_{i+1}^{n+1} + (1 + 2\alpha)\omega_i^{n+1} - \alpha \omega_{i-1}^{n+1} \tag{1.19}$$

Cette relation de récurrence est valable pour i $\in$[1,$N_x - 1$] car ω_{-1}^n et $\omega_{N_x+1}^n$ ne sont pas définis.
Donc :

$$\begin{pmatrix} 1+2\alpha & -\alpha & 0 & \cdots & \cdots & 0 \\ -\alpha & 1+2\alpha & -\alpha & 0 & \cdots & \vdots \\ 0 & -\alpha & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & -\alpha & 0 \\ \vdots & \cdots & 0 & -\alpha & 1+2\alpha & -\alpha \\ 0 & \cdots & \cdots & 0 & -\alpha & 1+2\alpha \end{pmatrix} \begin{pmatrix} \omega_1^{n+1} \\ \omega_2^{n+1} \\ \omega_3^{n+1} \\ \vdots \\ \vdots \\ \omega_{N_x-2}^{n+1} \\ \omega_{N_x-1}^{n+1} \end{pmatrix} = \begin{pmatrix} \omega_1^n \\ \omega_2^n \\ \omega_3^n \\ \vdots \\ \vdots \\ \omega_{N_x-2}^n \\ \omega_{N_x-1}^n \end{pmatrix} + \alpha \begin{pmatrix} \omega_0^{n+1} \\ 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \\ \omega_{N_x}^{n+1} \end{pmatrix}$$

On pose :

$$A = \begin{pmatrix} 1+2\alpha & -\alpha & 0 & \cdots & \cdots & 0 \\ -\alpha & 1+2\alpha & -\alpha & 0 & \cdots & \vdots \\ 0 & -\alpha & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & -\alpha & 0 \\ \vdots & \cdots & 0 & -\alpha & 1+2\alpha & -\alpha \\ 0 & \cdots & \cdots & 0 & -\alpha & 1+2\alpha \end{pmatrix}$$

$$W^{n+1} = \begin{pmatrix} \omega_1^n \\ \omega_2^n \\ \omega_3^n \\ \vdots \\ \vdots \\ \omega_{N_x-2}^n \\ \omega_{N_x-1}^n \end{pmatrix} \text{ et } W_D^{n+1} = \begin{pmatrix} \omega_0^{n+1} \\ 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \\ \omega_{N_x}^{n+1} \end{pmatrix}$$

D'où :

$$W^{n+1} = A^{-1}(W^n + \alpha W_D^{n+1}) \tag{1.20}$$

L'intervalle [0,1] est divisé en N_x segments identiques de longueur $\triangle x = \frac{1}{N_x}$; W^{n+1}, W^n des matrices colonnes de N_x-1 lignes et W_D^{n+1} la matrice colonne des conditions aux limites de Dirichlet.

Le composante de la vitesse u_y :

$$\omega = \partial_x u_y \rightsquigarrow \frac{u^n_{yi+1} - u^n_{yi}}{\triangle x} = w^n_i \tag{1.21}$$

Donc :

$$u^n_{yi+1} - u^n_{yi} = \triangle x w^n_i \tag{1.22}$$

Cette relation de récurrence est valable pour i $\in$[0,$N_x - 1$] car $u^n_{yN_x+1}$ n'est pas défini.
On a alors :

$$\begin{pmatrix} 1 & 0 & 0 & \cdots & \cdots & 0 \\ -1 & 1 & 0 & 0 & \cdots & \vdots \\ 0 & -1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 & 0 \\ \vdots & \cdots & 0 & -1 & 1 & 0 \\ 0 & \cdots & \cdots & 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} u^{n+1}_{y1} \\ u^{n+1}_{y2} \\ u^{n+1}_{y3} \\ \vdots \\ \vdots \\ u^{n+1}_{yN_x-1} \\ u^{n+1}_{yN_x} \end{pmatrix} = \triangle x \begin{pmatrix} \omega^n_0 \\ \omega^n_1 \\ \omega^n_2 \\ \vdots \\ \vdots \\ \omega^n_{N_x-2} \\ \omega^n_{N_x-1} \end{pmatrix} + \begin{pmatrix} u^n_{y0} \\ 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \\ 0 \end{pmatrix}$$

On pose :

$$B = \begin{pmatrix} 1 & 0 & 0 & \cdots & \cdots & 0 \\ -1 & 1 & 0 & 0 & \cdots & \vdots \\ 0 & -1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 & 0 \\ \vdots & \cdots & 0 & -1 & 1 & 0 \\ 0 & \cdots & \cdots & 0 & -1 & 1 \end{pmatrix}$$

$$U^{n+1}_y = \begin{pmatrix} u^{n+1}_{y1} \\ u^{n+1}_{y2} \\ u^{n+1}_{y3} \\ \vdots \\ \vdots \\ u^{n+1}_{yN_x-1} \\ u^{n+1}_{yN_x} \end{pmatrix} \quad W^n_0 = \begin{pmatrix} \omega^n_0 \\ \omega^n_1 \\ \omega^n_2 \\ \vdots \\ \vdots \\ \omega^n_{N_x-2} \\ \omega^n_{N_x-1} \end{pmatrix} \quad et\ U^n_{yD} = \begin{pmatrix} u^n_{y0} \\ 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \\ 0 \end{pmatrix}$$

D'où :

$$U^{n+1}_y = B^{-1}(\triangle x W^n_0 + U^n_{yD}) \tag{1.23}$$

La Fonction de courant ψ :

$$-u_y = \partial_x \psi \rightsquigarrow \frac{\psi^n_{i+1} - \psi^n_i}{\triangle x} = -u^n_{yi} \tag{1.24}$$

Donc :

$$\psi^n_{i+1} - \psi^n_i = -\triangle x u^n_{yi} \tag{1.25}$$

Cette relation de récurrence est valable pour i $\in$[0,$N_x - 1$] car $\psi^n_{N_x+1}$ n'est pas défini.
On a alors :

$$\begin{pmatrix} 1 & 0 & 0 & \cdots & \cdots & 0 \\ -1 & 1 & 0 & 0 & \cdots & \vdots \\ 0 & -1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 & 0 \\ \vdots & \cdots & 0 & -1 & 1 & 0 \\ 0 & \cdots & \cdots & 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} \psi_1^{n+1} \\ \psi_2^{n+1} \\ \psi_3^{n+1} \\ \vdots \\ \vdots \\ \psi_{N_x-1}^{n+1} \\ \psi_{N_x}^{n+1} \end{pmatrix} = -\triangle x \begin{pmatrix} u_{y0}^n \\ u_{y1}^n \\ u_{y2}^n \\ \vdots \\ \vdots \\ u_{yN_x-2}^n \\ u_{yN_x-1}^n \end{pmatrix} + \begin{pmatrix} \psi_0^n \\ 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \\ 0 \end{pmatrix}$$

On pose :

$$\Psi^{n+1} = \begin{pmatrix} \psi_1^{n+1} \\ \psi_2^{n+1} \\ \psi_3^{n+1} \\ \vdots \\ \vdots \\ \psi_{N_x-1}^{n+1} \\ \psi_{N_x}^{n+1} \end{pmatrix} \quad U_{y0}^n = \begin{pmatrix} u_{y0}^n \\ u_{y1}^n \\ u_{y2}^n \\ \vdots \\ \vdots \\ u_{yN_x-2}^n \\ u_{yN_x-1}^n \end{pmatrix} \quad et \ \Psi_0^n = \begin{pmatrix} \psi_0^n \\ 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \\ 0 \end{pmatrix}$$

D'où :

$$\Psi^{n+1} = B^{-1}(-\triangle x U_{y0}^n + \Psi_0^n) \tag{1.26}$$

Les résultats en utilisant Python :

Ces résultats [2] numériques sont réalisés par la langage de programmation Python **[2]** dont le code se trouve dans l'annexe A1.

2. Avec des conditions initiaux nulles c'est-à-dire à l'instant t=0 on prend ω=0, ψ=0 et u_y=0.

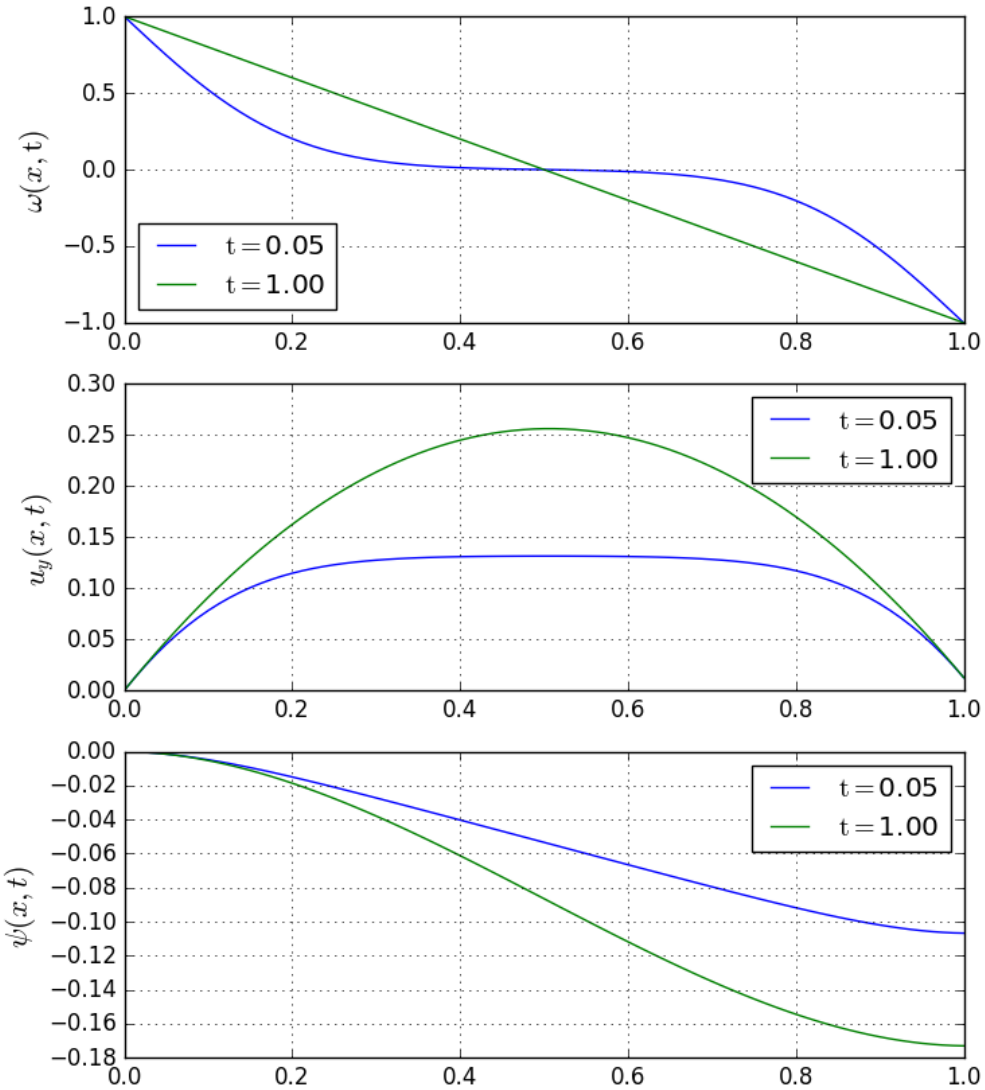


FIGURE 1.4 – Résultats de la simulation 1D : la vorticité w, la composante de la vitesse u_y et la fonction de courant ψ.

Commentaires : D'après ces résultats, on voit bien qu'après un temps t=1 (400 S) le régime permanent est établi et la vitesse[3] u_y prend la forme de poiseuille. Et en utilisant les résultats de la solution théorique on prouve la performance de la méthode que nous avons utilisé pour résoudre les équations.

3. C'est la même chose pour la vorticité et la fonction de courant.

1.3 Résolution numérique 2D

Dans cette partie, on ne va pas faire les simplifications sur le champ de vecteur vitesse [4].
On va faire la résolution de ce problème par un schéma explicite, décentré en temps et centré en espace qui sera stable selon la condition de CFL : $\frac{\nu\triangle t}{\triangle x^2} + \frac{\nu\triangle t}{\triangle y^2} \leq \frac{1}{2}$ **[4]**.

1.3.1 La discrétisation des équations

on discrétise les différentes dérivées partielles des équations sous la forme suivante :

La dérivée partielle	La discrétisation	La dérivée partielle	La discrétisation
$\partial_t\omega$	$\frac{\omega_{i,j}^{n+1}-\omega_{i,j}^n}{\triangle t}$	$\partial_{xx}\omega$	$\frac{\omega_{i+1,j}^n-2\omega_{i,j}^n+\omega_{i-1,j}^n}{(\triangle x)^2}$
$\partial_x\omega$	$\frac{\omega_{i+1,j}^n-\omega_{i-1,j}^n}{2\triangle x}$	$\partial_{yy}\omega$	$\frac{\omega_{i,j+1}^n-2\omega_{i,j}^n+\omega_{i,j-1}^n}{(\triangle y)^2}$
$\partial_y\psi$	$\frac{\psi_{i,j+1}^n-\psi_{i,j-1}^n}{2\triangle y}$	$\partial_{xx}\psi$	$\frac{\psi_{i+1,j}^n-2\psi_{i,j}^n+\psi_{i-1,j}^n}{(\triangle x)^2}$
$\partial_x\psi$	$\frac{\psi_{i+1,j}^n-\psi_{i-1,j}^n}{2\triangle x}$	$\partial_{yy}\psi$	$\frac{\psi_{i,j+1}^n-2\psi_{i,j}^n+\psi_{i,j-1}^n}{\triangle y^2}$
$\partial_y\omega$	$\frac{\omega_{i,j+1}^n-\omega_{i,j-1}^n}{2\triangle y}$	X	X

TABLE 1.2 – La discrétisation des dérivées partielles des équations de l'écoulement de poiseuille 2D.

On remplace dans les deux équations de (S_1') :

$$\frac{\omega_{i,j}^{n+1}-\omega_{i,j}^n}{\triangle t} + \frac{\psi_{i,j+1}^n-\psi_{i,j-1}^n}{2\triangle y} \times \frac{\omega_{i+1,j}^n-\omega_{i-1,j}^n}{2\triangle x} - \frac{\psi_{i+1,j}^n-\psi_{i-1,j}^n}{2\triangle x} \times \frac{\omega_{i,j+1}^n-\omega_{i,j-1}^n}{2\triangle y} = \nu(\frac{\omega_{i+1,j}^n-2\omega_{i,j}^n+\omega_{i-1,j}^n}{(\triangle x)^2} + \frac{\omega_{i,j+1}^n-2\omega_{i,j}^n+\omega_{i,j-1}^n}{(\triangle y)^2}) \tag{1.27}$$

$$\omega_{i,j}^{n+1}-\omega_{i,j}^n + \frac{\triangle t}{4\triangle y\triangle x}((\psi_{i,j+1}^n-\psi_{i,j-1}^n)\times(\omega_{i+1,j}^n-\omega_{i-1,j}^n)-(\psi_{i+1,j}^n-\psi_{i-1,j}^n)\times(\omega_{i,j+1}^n-\omega_{i,j-1}^n)) = \frac{\nu\triangle t}{(\triangle x)^2}(\omega_{i+1,j}^n-2\omega_{i,j}^n+\omega_{i-1,j}^n) + \frac{\nu\triangle t}{(\triangle y)^2}(\omega_{i,j+1}^n-2\omega_{i,j}^n+\omega_{i,j-1}^n) \tag{1.28}$$

Pour simplifier, on prend $\triangle x = \triangle y = h$, on obtient :

$$\omega_{i,j}^{n+1} = \omega_{i,j}^n - \frac{\triangle t}{4h^2}((\psi_{i,j+1}^n-\psi_{i,j-1}^n)\times(\omega_{i+1,j}^n-\omega_{i-1,j}^n)-(\psi_{i+1,j}^n-\psi_{i-1,j}^n)\times(\omega_{i,j+1}^n-\omega_{i,j-1}^n)) + \frac{\nu\triangle t}{h^2}(\omega_{i+1,j}^n+\omega_{i-1,j}^n+\omega_{i,j+1}^n+\omega_{i,j-1}^n-4\omega_{i,j}^n) \tag{1.29}$$

4. Autrement dit les inconnus ω, ψ, u_x et u_y dépendent des cordonnés de l'espace x et y.

Et :

$$\frac{\psi^n_{i+1,j} - 2\psi^n_{i,j} + \psi^n_{i-1,j}}{(\triangle x)^2} + \frac{\psi^n_{i,j+1} - 2\psi^n_{i,j} + \psi^n_{i,j-1}}{(\triangle y)^2} = -\omega^n_{i,j} \tag{1.30}$$

Pour $\triangle x = \triangle y = h$:

$$\frac{\psi^n_{i+1,j} + \psi^n_{i-1,j} + \psi^n_{i,j+1} + \psi^n_{i,j-1} - 4\psi^n_{i,j}}{h^2} = -\omega^n_{i,j} \tag{1.31}$$

On applique la méthode de Successive Over-Relaxation (SOR) [5], avec $\zeta \in [1, 2]$:

$$\psi^{n+1}_{i,j} = 0.25\zeta(\psi^n_{i+1,j} + \psi^n_{i-1,j} + \psi^n_{i,j+1} + \psi^n_{i,j-1} + h^2\omega^n_{i,j}) + (1-\zeta)\psi^n_{i,j} \tag{1.32}$$

Et pour les composantes de la vitesse u_x et u_y :

$$u_{xi,j} = \frac{\psi_{i,j+1} - \psi_{i,j-1}}{2h} \tag{1.33}$$

$$u_{yi,j} = -\frac{\psi_{i+1,j} - \psi_{i-1,j}}{2h} \tag{1.34}$$

1.3.2 Les conditions aux limites

Pour bien résoudre les équations du problème et trouver les bonnes résultats, il est nécessaire de trouver les conditions aux limites adéquates avec notre domaine.
Puisque le système est symétrique par rapport à x=$\frac{L_x}{2}$, il doit y avoir une anti-symétrie pour la vorticité par rapport à x=$\frac{L_x}{2}$.
A partir du profil théorique calculé précédemment de la vorticité, on peut voir qu'une condition de Neuman ne convient pas donc elle s'agit bien de Dirichlet pour w avec le caractère anti-symétrique : $\omega(0, y) = -\omega(L_x, y)$. Il suffit de prendre par exemple $\omega(0, y) = 1$ et $\omega(L_x, y) = -1$.
Et pour y=0 et y=L_y une condition de Neuman nous convient. Il suffit de prendre par exemple $\partial_y\omega = 0$

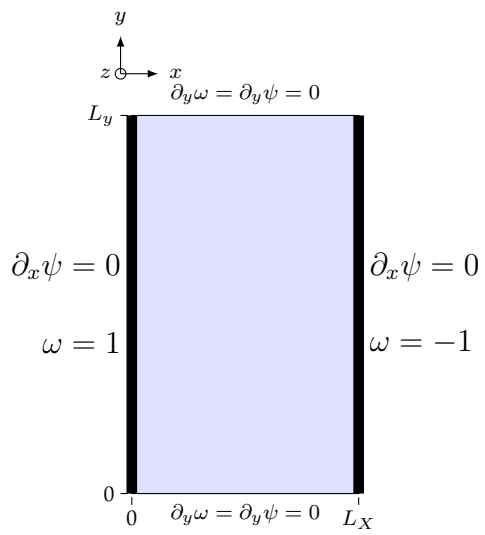


FIGURE 1.5 – Conditions aux limites de ω et ψ 2D.

Pour la fonction de courant, on prend une condition de Neuman partout, c'est-à-dire par exemple pour x=0 et x=L_x, on prend $\partial_x\psi = 0$ et pour y=0 et y=L_y, on prend $\partial_y\psi = 0$.

1.3.3 Algorithme de résolution et visualisation des résultats

Suite à la discrétisation des équations qui modélisent le problème et le choix convenable des conditions aux limites, on passe à l'étape de programmation numérique, pour ce faire an applique l'Algorithme 1.[6]

5. Ou bien Sur-relaxation successive : c'est une variante de la méthode de Gauss-Seidel qui nous permet de résoudre un système d'équations linéaires, elle se caractérise généralement par une convergence plus rapide.

Algorithme 1 ÉCOULEMENT DE POISEUILLE 2D

1 Initialisation de ω, ψ, et la vitesse.

2 Calculer les conditions aux limites pour vorticité et pour la fonction de courant.

3 Résoudre l'équation de transport de vorticité (1.7) pour trouver ω.

4 Résoudre l'équation de Poisson pour trouver la fonction de courant (1.6) en utilisant (SOR).

5 Calculer le champ de vitesse en utilisant (1.5).

6 Revenir à l'étape 2 et répétez le calcul pour une autre étape de temps jusqu'à l'attente le RS .

Les résultats numérique 2D :

Après l'implémentation de cet algorithme et lorsque le régime stationnaire est atteint, on obtient les résultats suivantes :

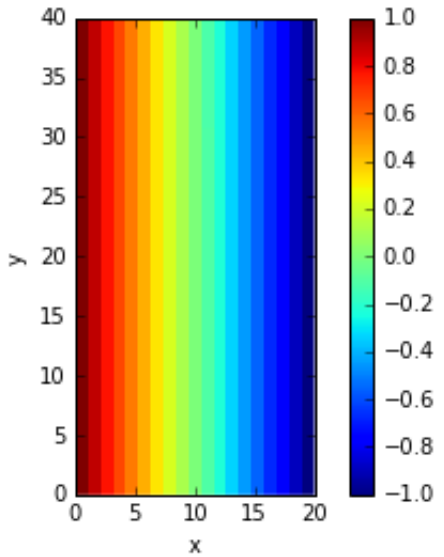


FIGURE 1.6 – La vorticité ω avec un nombre de reynolds Re=21.6.

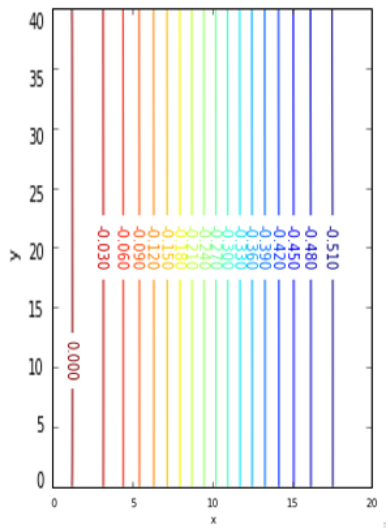


FIGURE 1.7 – La fonction de courant ψ, avec Re=21.6.

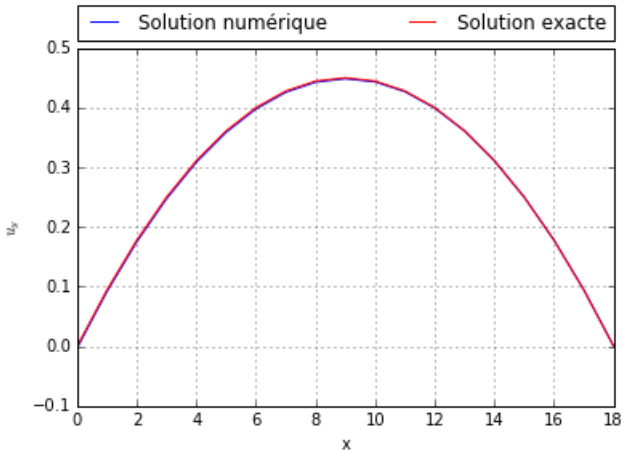


FIGURE 1.8 – La composantes de la vitesse u_y numérique et analytique, pour Re=21.6.

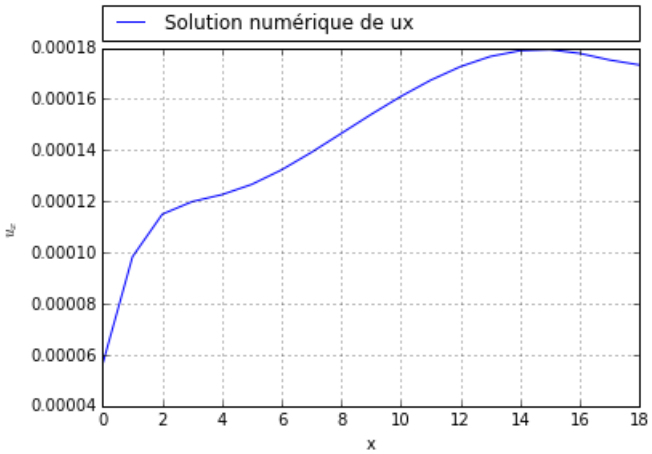


FIGURE 1.9 – La composantes de la vitesse u_x, pour Re=21.6.

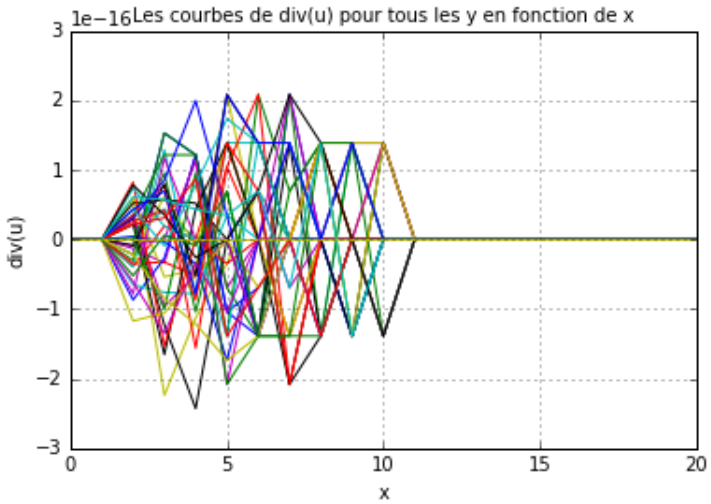


FIGURE 1.10 – La divergence de la vitesse $\vec{u}$.

Commentaires :

1. On observe bien d'après les figures 1.6 et 1.7 que ψ et ω gardent les mêmes valeurs indépendamment de y. La figure 1.9 montre que la composante de la vitesse selon $\vec{e_x}$ est pratiquement nulle. Ce qui justifier les simplifications que nous avons faites sur $\vec{u}$ dans la section précédente [6].

2. En regardant la figure (1.8), on constate que la composante de vitesse u_y converge vers la solution exacte avec un erreur pratiquement nul et c'est ce qu'on va démontrer dans la section suivante.

3. Puisque nous avons appliqué le rotationnel sur l'équation de N.S pour un fluide incompressible donc il faut confirmer la validité de l'hypothèse d'incompressibilité du fluide, on calcule numériquement la divergence de $\vec{u}_x$ (div($\vec{u}_x$)), la figure 1.10 montre bien qu'elle est de l'ordre de grondeur de 10^{-16} ce qui est négligeable.

6. Même remarque pour la composante de la vitesse u_y.

1.4 Étude des erreurs entre les courbes numériques et analytique de u_y

Pour déterminer la qualité du code que nous avons développé, on calcule différents type d'erreurs entre les courbes numérique et analytique de u_y tout en variant le valeur du nombre de Reynolds.

1.4.1 Erreur $\mid u_{ythe} - u_{ynume} \mid$

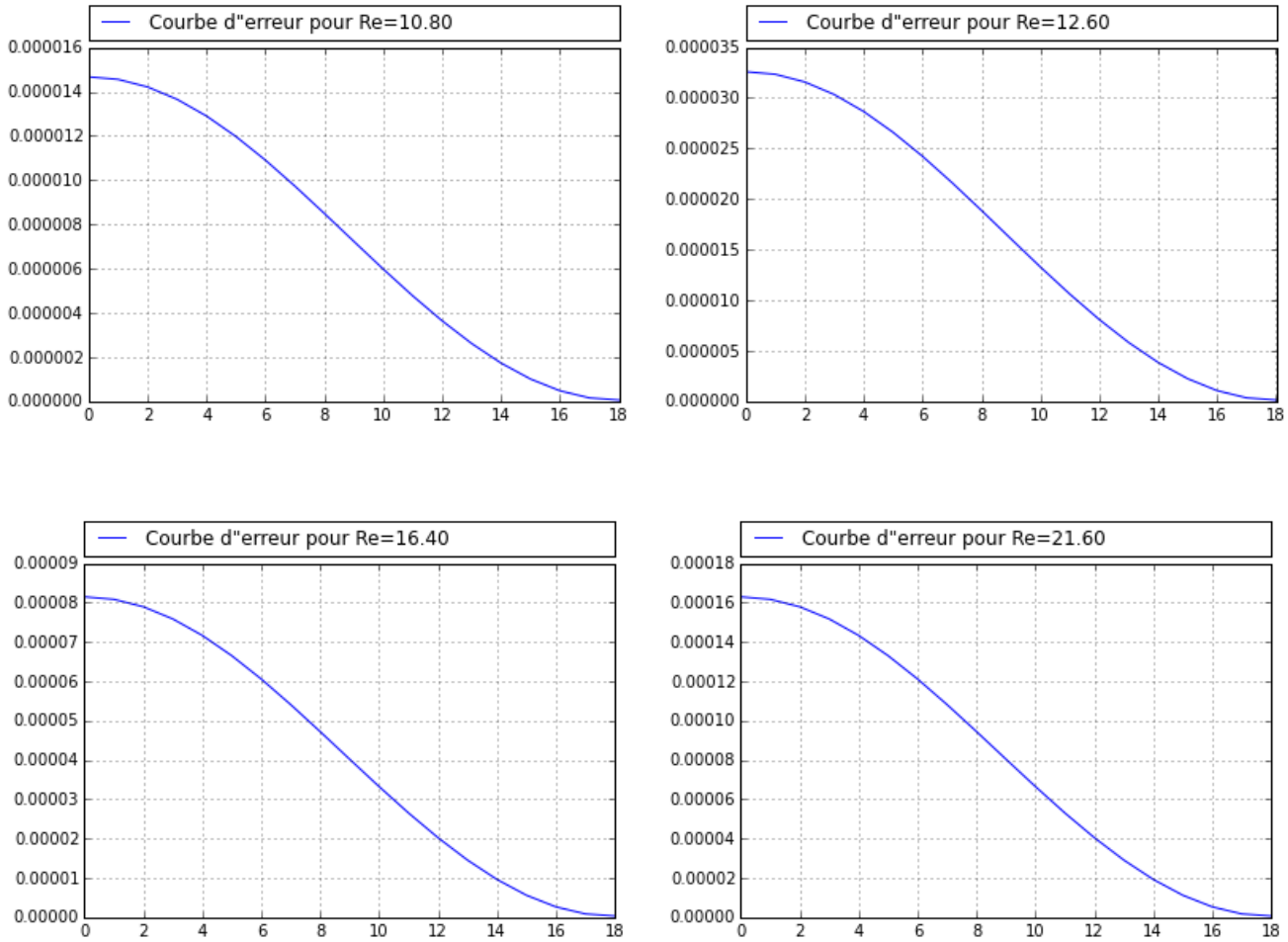


FIGURE 1.11 – Les courbes d'erreur $\mid u_{ythe} - u_{ynume} \mid$ en fonction de x et pour un y fixé, pour des nombres de Reynolds varient de 10.8 et à 21.6.

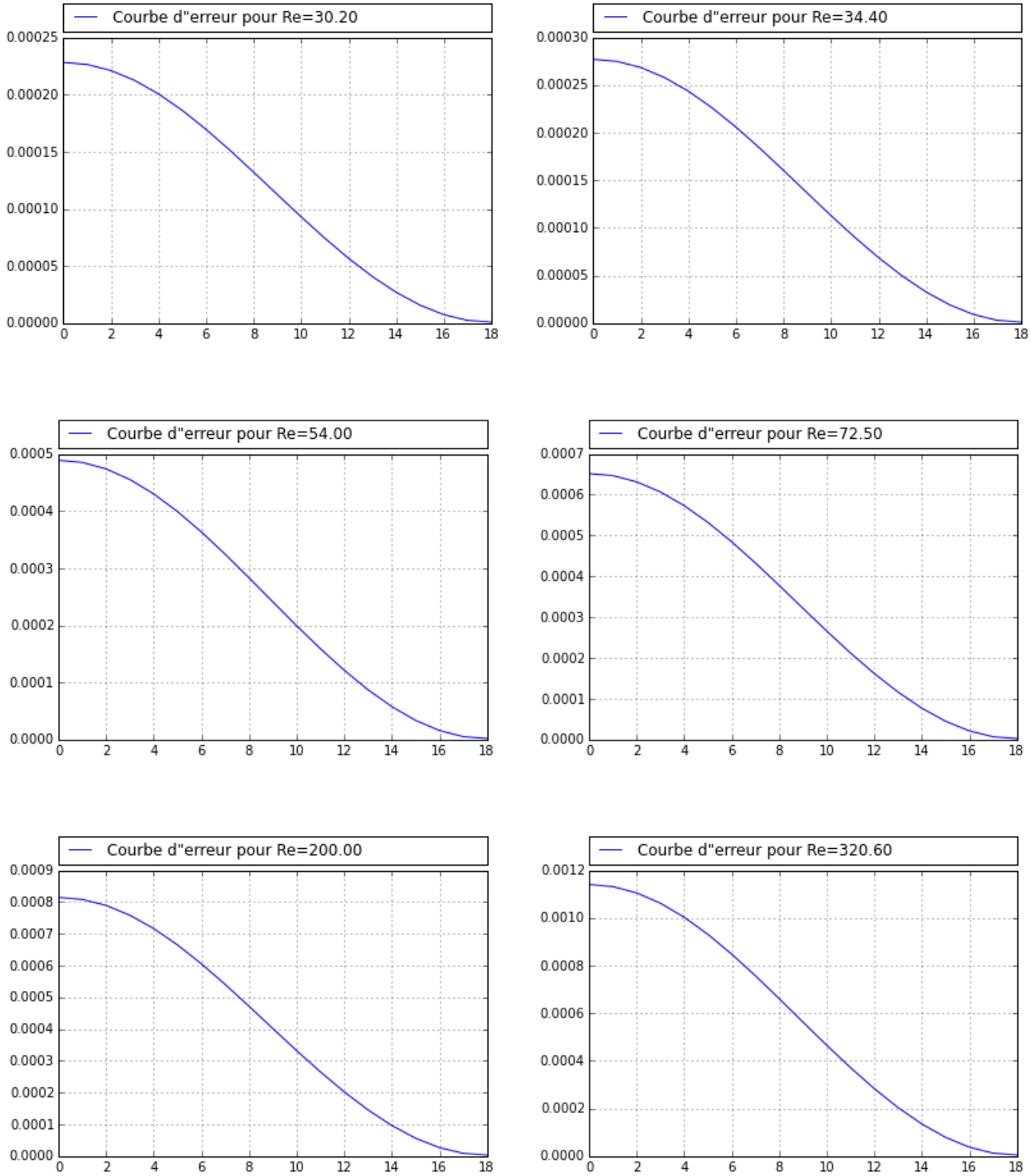


FIGURE 1.12 – Les courbes d'erreur $\mid u_{ythe} - u_{ynume} \mid$ en fonction de x et pour un y fixé, pour des nombres de Reynolds varient de 30.2 à 320.6.

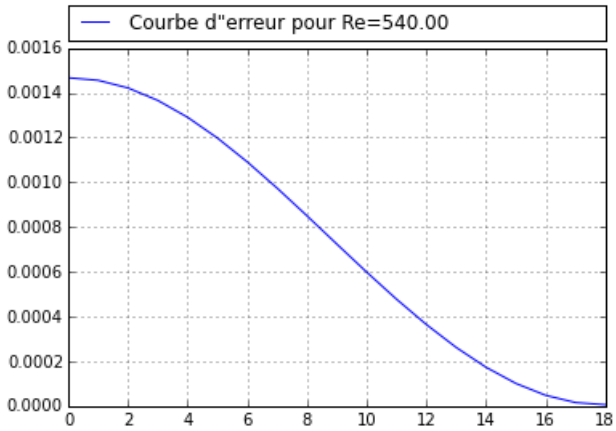


FIGURE 1.13 – Les courbes d'erreur $\mid u_{ythe} - u_{ynume} \mid$ en fonction de x et pour un y fixé, pour un nombre de Reynolds Re=540.

1.4.2 Erreur $max(\mid u_{ythe} - u_{ynume} \mid)$

On calcule le maximum des courbes précédentes, on obtient le tableau suivant :

Re	10.8	12.6	16.4	21.6	30.2	34.4	54
Erreur	$1.47.10^{-5}$	$3.26.10^{-5}$	$8.15.10^{-5}$	$1.8.10^{-4}$	$2.28.10^{-4}$	$2.7.10^{-4}$	$4.9.10^{-4}$

Re	72.50	200	320.6	540
Erreur	$6.5.10^{-4}$	$8.14.10^{-4}$	$11.4.10^{-4}$	$14.6.10^{-4}$

TABLE 1.3 – Erreur= max($\mid u_{yth} - u_{ynum} \mid$) en fonction de Re.

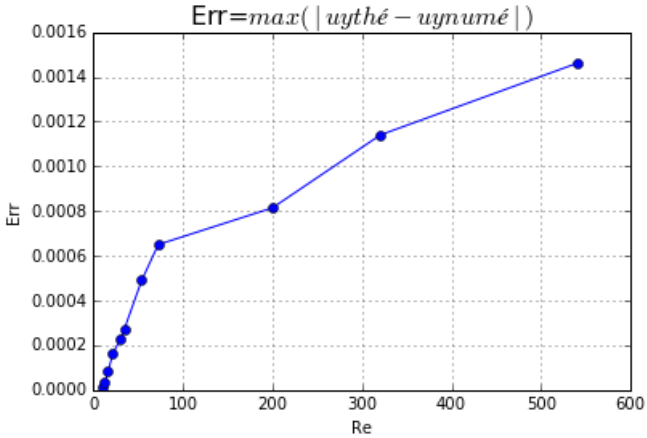


FIGURE 1.14 – Erreur=$max(\mid u_{ythe} - u_{ynume} \mid)$ en fonction de Re.

1.4.3 Commentaires

1. On constate que les deux types d'erreur sont faibles.
2. L'erreur $max(\mid u_{ythe} - u_{ynume} \mid)$ augmente quand Re augmente, ce qui est prévu puisque on s'approche du régime turbulent.

CHAPITRE

2 La convection forcée

Après l'étude de la dynamique de fluide[1] que nous avons fait dans le chapitre précédent, maintenant on va entamer l'étude thermique du fluide.

2.1 Mise en équations

Pour pouvoir faire l'étude de la convection forcé il est impératif d'ajouter l'équation de chaleur dans un fluide au système (S_1) **[1]** pour qu'il devient :

$$(S_2) : \begin{cases} \nabla.\vec{u} = 0 & (C.M) \\ \rho\partial_t\vec{u} + \rho(\vec{u}.\nabla)\vec{u} = -\nabla p + \rho\vec{g} + \rho\nu\nabla^2\vec{u} & (N.S) \\ \rho c\partial_t T + \rho c(\vec{u}.\nabla)T = \nabla.(k\nabla T) & (E.C.F) \end{cases}$$

L'équation de chaleur dans un fluide (E.C.F) nous donne :

$$\rho c\partial_t T + \rho c(u_x\partial_x T + u_y\partial_y T) = k(\partial_{xx}T + \partial_{yy}T) \tag{2.1}$$

D'où le système qu'on cherche à résoudre devient :

$$(S_2') : \begin{cases} \partial_t\omega + \partial_y\psi\partial_x\omega - \partial_x\psi\partial_y\omega = \nu(\partial_{xx}\omega + \partial_{yy}\omega) \\ \partial_{xx}\psi + \partial_{yy}\psi = -\omega \\ \rho c\partial_t T + \rho c(u_x\partial_x T + u_y\partial_y T) = k(\partial_{xx}T + \partial_{yy}T) \end{cases}$$

2.2 La discrétisation de l'équation de chaleur

La dérivée partielle	La discrétisation	La dérivée partielle	La discrétisation
$\partial_x T$	$\frac{T^n_{i+1,j} - T^n_{i-1,j}}{2\triangle x}$	$\partial_{xx}T$	$\frac{T^n_{i+1,j} - 2T^n_{i,j} + T^n_{i-1,j}}{(\triangle x)^2}$
$\partial_y T$	$\frac{T^n_{i,j+1} - T^n_{i,j-1}}{2\triangle y}$	$\partial_{yy}T$	$\frac{T^n_{i,j+1} - 2T^n_{i,j} + T^n_{i,j-1}}{(\triangle y)^2}$
$\partial_t T$	$\frac{T^{n+1}_{i,j} - T^n_{i,j}}{\triangle t}$	X	X

TABLE 2.1 – La discrétisation des dérivées partielles de l'équation de chaleur.

1. Écoulement de poiseuille.

On remplace dans l'équation (2.1) :

$$\begin{aligned}&\rho c\frac{T_{i,j}^{n+1}-T_{i,j}^{n}}{\triangle t}+\rho c(u_{xi,j}^{n}\frac{T_{i+1,j}^{n}-T_{i-1,j}^{n}}{2\triangle x}+u_{yi,j}^{n}\frac{T_{i,j+1}^{n}-T_{i,j-1}^{n}}{2\triangle y})\\&=k(\frac{T_{i+1,j}^{n}-2T_{i,j}^{n}+T_{i-1,j}^{n}}{(\triangle x)^2}+\frac{T_{i,j+1}^{n}-2T_{i,j}^{n}+T_{i,j-1}^{n}}{(\triangle y)^2})\end{aligned} \tag{2.2}$$

On prend $\triangle x=\triangle y=h$, donc on obtient l'équation finale discrétisée [2]

$$\begin{aligned}T_{i,j}^{n+1}&=T_{i,j}^{n}-\frac{\triangle t}{2h}(u_{xi,j}^{n}(T_{i+1,j}^{n}-T_{i-1,j}^{n})+u_{yi,j}^{n}(T_{i,j+1}^{n}-T_{i,j-1}^{n}))\\&+\frac{k\triangle t}{\rho ch^2}(T_{i+1,j}^{n}+T_{i-1,j}^{n}+T_{i,j+1}^{n}+T_{i,j-1}^{n}-4T_{i,j}^{n})\end{aligned} \tag{2.3}$$

2.3 Les conditions aux limites

On prend des conditions aux limites pour la température de type Dirichlet en x=0 et x=L_x et on calorifuge les deux bords en y=0 et y=L_y.

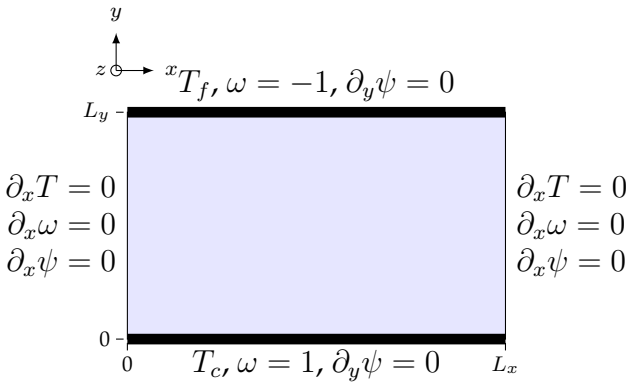


FIGURE 2.1 – Convection forcée : Conditions aux limites.

Remarques :

1. Dans cette partie du projet, on change la disposition des deux plans pour qu'ils soient situés horizontalement comme le montre la figure 1.2.
2. Pour les autres inconnus c'est-à-dire ω et ψ, on utilise les mêmes conditions aux limites du chapitre 1 qui nous permettent de trouver un écoulement de poiseuille anisotherme sans couplage thermique.

2.4 Algorithme de résolution et visualisation de résultat

Après la discrétisation des équations et choisir d'une manière convenable les conditions aux limites qui modélise la convection forcée, on passe à l'étape de programmation numérique, pour ce faire on applique l'Algorithme 2.

2. Pour les autres équations de (S_2') en utilise (1.29) et (1.31).

Algorithme 2 LA CONVECTION FORCÉE

1 Initialisation de ω, ψ, la vitesse et T.

2 Calculer les conditions aux limites pour la vorticité, la fonction de courant et T.

3 Résoudre l'équation de transport de vorticité (1.7) pour trouver ω.

4 Résoudre l'équation de poisson pour trouver la fonction de courant (1.6) en utilisant (SOR).

5 Calculer le champ de la vitesse en utilisant (1.5).

6 Calculer la température en utilisant (2.2).

7 Revenir à l'étape 2 et répétez le calcul pour une autre étape de temps.

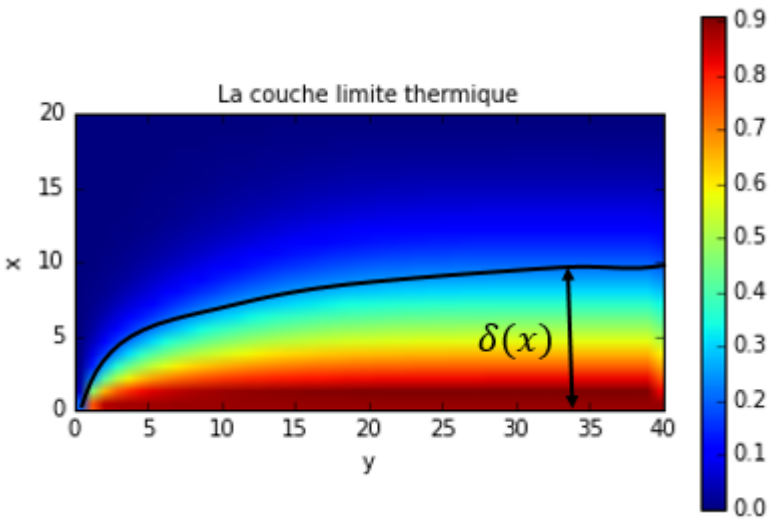


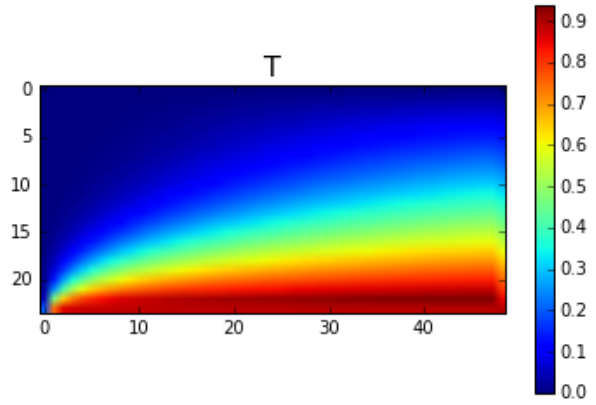
FIGURE 2.2 – Résultat de la simulation de T lorsque le RS est atteint.

On voit d'après la figure 2.2 que la température ne varie sensiblement dans le liquide que sur une couche de faible épaisseur $\delta(x)$ au voisinage immédiat de la surface solide. Cette couche de faible épaisseur est appelée couche limite thermique[3]. A l'origine, le profil de température dans le fluide est uniforme (0 dans notre cas), puis l'influence de T_c se fait sentir progressivement dans le fluide, ce qui se traduit par l'apparition d'un gradient de température, accompagné d'un flux de chaleur dirigé du plan y=0 vers le fluide.

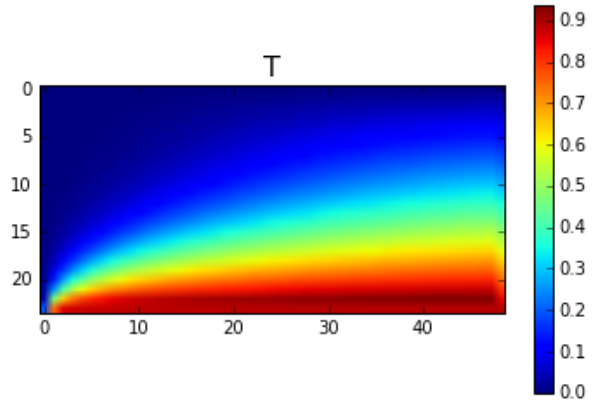
2.5 Influence de Re sur la couche limite thermique

Dans cette partie nous allons voir comment varie l'épaisseur de la couche limite thermique ($\delta(x)$) lorsque le nombre de Reynolds varie.

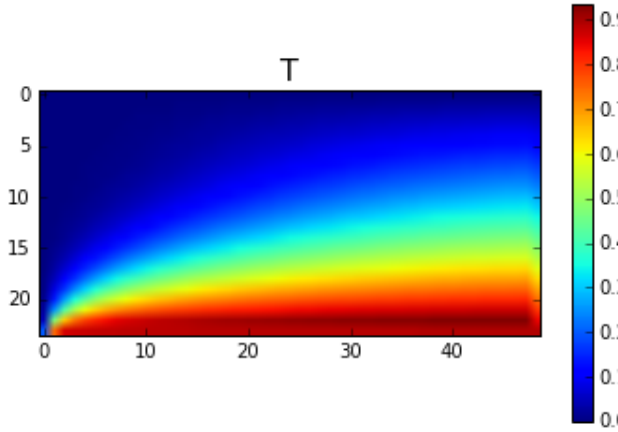
3. L'épaisseur $\delta(x)$ de la couche limite est la distance, mesurée à partir du plan y=0, pour laquelle la température du fluide atteint la valeur T_c à 1% prés.



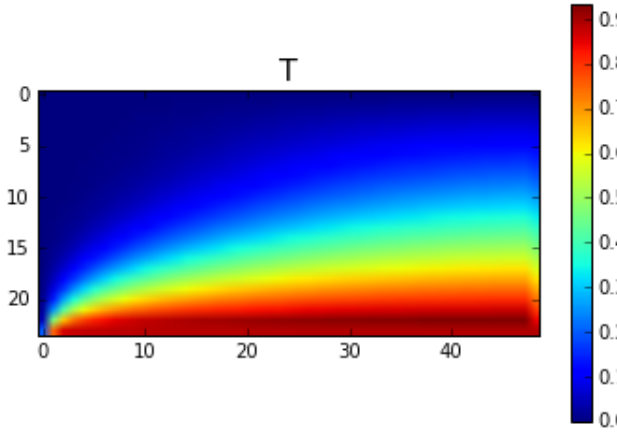
La CL thermique pour Re=12.



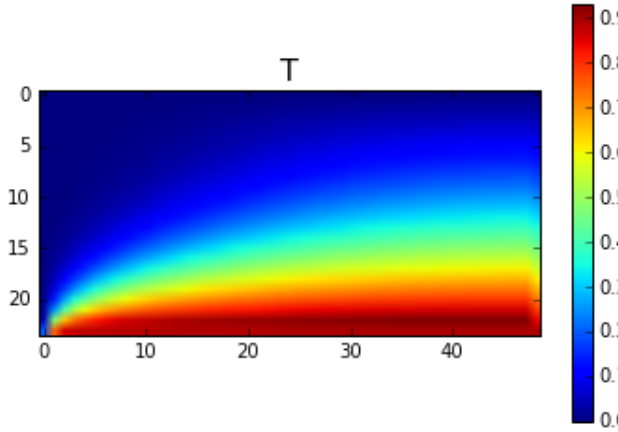
La CL thermique pour Re=30.2.



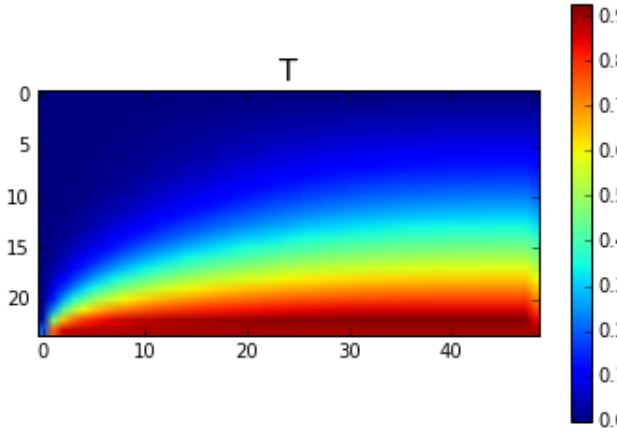
La CL thermique pour Re=39.



La CL thermique pour Re=57.

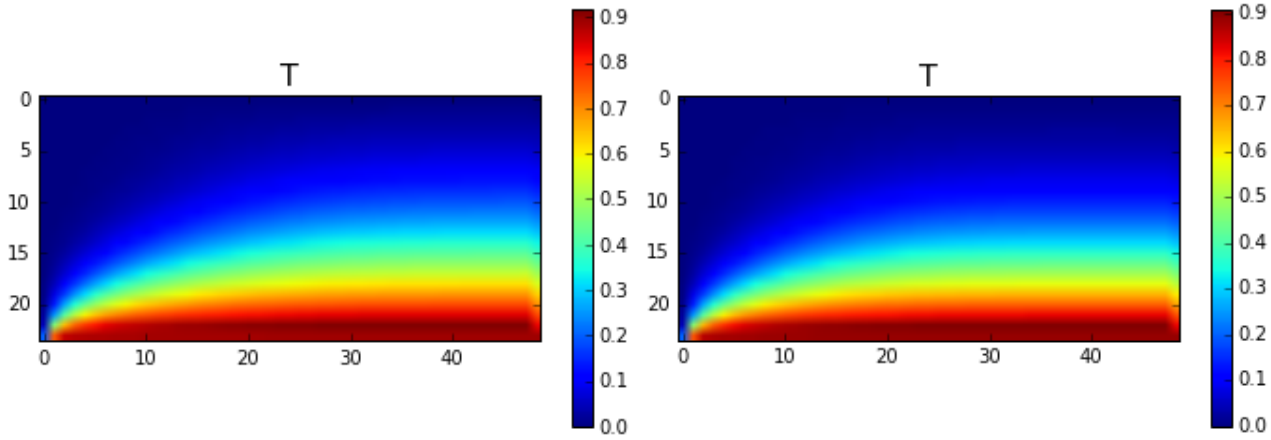


La CL thermique pour Re=70.



La CL thermique pour Re=280.

FIGURE 2.3 – Le champ de Température pour Re de 12 à 280 avec x en abscisse et y en ordonné.



La CL thermique pour Re=330. La CL thermique pour Re=600.

FIGURE 2.4 – Le champ de Température pour Re=330 et Re=600, avec x est en abscisse et y en ordonné.

D'après les figures 2.3 et 2.4, on remarque que lorsque le nombre de Reynolds Re augmente, l'épaisseur de la couche limite diminue. Donc l'épaisseur de la couche limite thermique dépend des propriétés thermophysiques du fluide : viscosité cinématique ν.
Les domaines d'application de la couche limite thermique sont très large par exemples : en mécanique des fluides (aérodynamique, hydrodynamique), en océanographie, en météorologie...

CHAPITRE

3 La convection naturelle

La convection naturelle est un phénomène extrêmement répandu. Il peut être observé à petite échelle, par exemple dans une casserole d'eau sur une cuisinière, mais aussi à grande échelle, par exemple sur la terre, où l'énergie solaire est absorbée par le sol avant de réchauffer les couches inférieures de l'atmosphère provoquant ainsi des vents thermiques.

3.1 Mise en équations

Les équations permettant de décrire le transfert thermique en présence d'un écoulement par convection naturelle au sein d'un liquide incompressible sont : la conservation de la matière au sein du fluide (CM), l'équation de Navier-Stokes (NS) et l'équation de conservation de l'énergie (CE).

$$\vec{\nabla}_{\mathrm{M}} \cdot \vec{u} = 0 \quad (\mathrm{CM}) \tag{3.1}$$

$$\rho_0 \partial_\tau \vec{u} + \rho_0 (\vec{u} \cdot \vec{\nabla}_{\mathrm{M}}) \vec{u} = -\vec{\nabla}_{\mathrm{M}}(p) + \mu \Delta_{\mathrm{M}} \vec{u} + \vec{f}_{\mathrm{ext}} \quad (\mathrm{NS}) \tag{3.2}$$

$$\rho_0 c \partial_\tau T + \rho_0 c (\vec{v} \cdot \vec{\nabla}_{\mathrm{M}}) T = \vec{\nabla}_{\mathrm{M}} \cdot \left(k \vec{\nabla}_{\mathrm{M}} T \right) + \dot{w} \quad (\mathrm{CE}) \tag{3.3}$$

où $\vec{f}_{\mathrm{ext}}$ est la densité massique de force extérieure appliquée à l'eau, $\dot{w}$ est la densité volumique de puissance, qui s'écrit $\dot{w} = \dot{w}_{\mathrm{v}} + \dot{w}_{\mathrm{e}}$ avec $\dot{w}_{\mathrm{v}}$ et $\dot{w}_{\mathrm{e}}$ les densités volumiques de puissance respectivement dissipative visqueuse et d'excitation externe et $\vec{\nabla}_{\mathrm{M}}$ l'opérateur nabla au point M de coordonnées (X, Y), avec $X, Y \in [0, L_{\mathrm{X}}], [0, L_{\mathrm{Y}}]$.
Nous faisons les hypothèses suivantes dans la présente étude :

- la densité volumique de puissance dissipative visqueuse est négligeable devant les transferts conductifs au sein de l'eau, qui est un liquide peu visqueux : $\dot{w}_{\mathrm{v}} \approx 0$; il n'y a pas de terme d'excitation externe (terme Joule, chauffage micro-onde, ...) : $\dot{w}_{\mathrm{e}} = 0$ dans la présente étude ;
- les variations de température rencontrées dans cette étude sont suffisamment faibles pour supposer que les caractéristiques physiques du fluide c et k sont constantes ;
- nous appliquons l'approximation de **Boussinesq** qui consiste à supposer que la masse volumique de l'eau est constante partout (et égale à ρ_0), excepté dans le terme de force externe que nous écrivons comme :

$$\vec{f}_{\mathrm{ext}} = \rho_0 \left[1 - \beta \left(T - T_0 \right) \right] \vec{g} \tag{3.4}$$

  avec $\beta > 0$ le coefficient de dilatation thermique de l'eau.

L'indice "0" indique ici l'état initial du fluide (grandeurs uniformes et fluide au repos). En utilisant les hypothèses précédentes, nous pouvons ré-écrire les équations bilan (NS) et (CE)

sous la forme suivante :

$$\partial_\tau \vec{u} + (\vec{u} \cdot \vec{\nabla}_{\mathrm{M}})\vec{u} = -\vec{\nabla}_{\mathrm{M}}(p'/\rho_0) + \nu\Delta_{\mathrm{M}}\vec{u} - \beta\,(T - T_0)\,\vec{g} \quad (\mathrm{NS}) \tag{3.5}$$

$$\partial_\tau T + (\vec{u} \cdot \vec{\nabla}_{\mathrm{M}})T = \kappa\Delta_{\mathrm{M}}T \quad (\mathrm{CE}) \tag{3.6}$$

avec $\kappa = k/\rho c$ la diffusivité thermique et $p' = p - \rho_0 g Y$.

3.1.1 Adimensionnement des équations

Nous adimensionnons les équations (CM), (NS) et (CE) en introduisant les trois grandeurs sans dimensions suivantes :

$$\left|\begin{array}{l} \mathrm{M} \to \mathrm{m} = \mathrm{M}/L = (x, y) \\ t = \tau\dfrac{\kappa}{L^2} \\ \Theta = \dfrac{T - T_0}{T_s - T_0} \end{array}\right. \tag{3.7}$$

avec T_0 la température initiale, supposée uniforme, et T_s une température de référence qui sera précisée ultérieurement. En utilisant ces changements de variables, on obtient :

$$\left|\begin{array}{l} \vec{\nabla}_{\mathrm{M}} \to \dfrac{1}{L}\vec{\nabla}_{\mathrm{m}} \\ \Delta_{\mathrm{M}} \to \dfrac{1}{L^2}\Delta_{\mathrm{m}} \\ \partial_\tau \to \dfrac{\kappa}{L^2}\partial_t \\ \vec{u} \to \dfrac{\kappa}{L}\vec{U} \\ p' \to \dfrac{\kappa^2\rho_0}{L^2}P \end{array}\right. \tag{3.8}$$

Les équations non dimensionnées du problème :

$$\vec{\nabla} \cdot \vec{U} = 0 \quad (\mathrm{CMA}) \tag{3.9}$$

$$\partial_t\vec{U} + (\vec{U} \cdot \vec{\nabla})\vec{U} = -\vec{\nabla}P + \mathrm{Pr}\Delta\vec{U} - \mathrm{PrRa}\Theta\vec{e}_y \quad (\mathrm{NSA}) \tag{3.10}$$

$$\partial_t\Theta + (\vec{U} \cdot \vec{\nabla})\Theta = \Delta\Theta \quad (\mathrm{CEA}) \tag{3.11}$$

Avec [1] :

$$\mathrm{Pr} = \frac{\nu}{\kappa} \quad \text{et} \quad \mathrm{Ra} = \frac{\beta g L^3 \Delta T_{s,0}}{\kappa\nu} \tag{3.12}$$

3.1.2 La méthode de vorticité

On applique le rotationnel à l'équation (11) :

$$\partial_t\vec{\nabla} \times \vec{U} + \vec{\nabla} \times ((\vec{U} \cdot \vec{\nabla})\vec{U}) = -\vec{\nabla} \times (\vec{\nabla}P) + \mathrm{Pr}\vec{\nabla} \times (\Delta\vec{U}) - \mathrm{PrRa}\vec{\nabla} \times (\Theta\vec{e}_y) \tag{3.13}$$

1. Pr est le nombre de Prandtl qui est un nombre sans dimension, représente le rapport entre la diffusivité de quantité de mouvement ν (ou viscosité cinématique) et la diffusivité thermique et Ra est le nombre de Rayleigh qui est un nombre sans dimension utilisé en mécanique des fluides et caractérisant le transfert de chaleur au sein d'un fluide.

Soient la vorticité et la fonction de courant sans dimensions :

$$\left| \begin{array}{l} \vec{\omega} \rightarrow \dfrac{\kappa}{L^3}\vec{\Omega} \\ \vec{\psi} \rightarrow \dfrac{\kappa}{L}\vec{\Psi} \end{array} \right. \tag{3.14}$$

Donc :

$$\partial_t\vec{\Omega} + (\vec{U}\cdot\vec{\nabla})\vec{\Omega} = \mathrm{Pr}\Delta\vec{\Omega} - \mathrm{PrRa}\vec{\nabla}\times(\Theta\vec{e}_y) \tag{3.15}$$

$$\Delta\vec{\Psi} = -\vec{\Omega} \tag{3.16}$$

$$\vec{U} = \vec{\nabla}\times\vec{\Psi} \tag{3.17}$$

D'où le problème se modélise par ces quatre équations :

$$\partial_t\Omega + \partial_y\Psi\partial_x\Omega - \partial_x\Psi\partial_y\Omega = \mathrm{Pr}(\partial_{xx}\Omega + \partial_{yy}\Omega) - \mathrm{PrRa}\partial_x\Theta \tag{3.18}$$

$$\partial_{xx}\Psi + \partial_{yy}\Psi = -\Omega \tag{3.19}$$

$$U_x = \partial_y\Psi \tag{3.20}$$

$$U_y = -\partial_x\Psi \tag{3.21}$$

$$\partial_t\Theta + U_x\partial_x\Theta + U_y\partial_y\Theta = \partial_{xx}\Theta + \partial_{yy}\Theta \tag{3.22}$$

3.2 La discrétisation des équations

La dérivée partielle	La discrétisation	La dérivée partielle	La discrétisation
$\partial_t\Omega$	$\frac{\Omega^{n+1}_{i,j}-\Omega^n_{i,j}}{\triangle t}$	$\partial_y\Omega$	$\frac{\Omega^n_{i,j+1}-\Omega^n_{i,j-1}}{2\triangle y}$
$\partial_t\Theta$	$\frac{\Theta^{n+1}_{i,j}-\Theta^n_{i,j}}{\triangle t}$	$\partial_{xx}\Omega$	$\frac{\Omega^n_{i+1,j}-2\Omega^n_{i,j}+\Omega^n_{i-1,j}}{(\triangle x)^2}$
$\partial_x\Omega$	$\frac{\Omega^n_{i+1,j}-\Omega^n_{i-1,j}}{2\triangle x}$	$\partial_{yy}\Omega$	$\frac{\Omega^n_{i,j+1}-2\Omega^n_{i,j}+\Omega^n_{i,j-1}}{(\triangle y)^2}$
$\partial_y\Psi$	$\frac{\Psi^n_{i,j+1}-\Psi^n_{i,j-1}}{2\triangle y}$	$\partial_{xx}\Psi$	$\frac{\Psi^n_{i+1,j}-2\Psi^n_{i,j}+\Psi^n_{i-1,j}}{(\triangle x)^2}$
$\partial_x\Psi$	$\frac{\Psi^n_{i+1,j}-\Psi^n_{i-1,j}}{2\triangle x}$	$\partial_{yy}\Psi$	$\frac{\Psi^n_{i,j+1}-2\Psi^n_{i,j}+\Psi^n_{i,j-1}}{(\triangle y)^2}$
$\partial_y\Theta$	$\frac{\Theta^n_{i,j+1}-\Theta^n_{i,j-1}}{2\triangle y}$	$\partial_{xx}\Theta$	$\frac{\Theta^n_{i+1,j}-2\Theta^n_{i,j}+\Theta^n_{i-1,j}}{(\triangle x)^2}$
$\partial_x\Theta$	$\frac{\Theta^n_{i+1,j}-\Theta^n_{i-1,j}}{2\triangle x}$	$\partial_{yy}\Theta$	$\frac{\Theta^n_{i,j+1}-2\Theta^n_{i,j}+\Theta^n_{i,j-1}}{(\triangle y)^2}$

TABLE 3.1 – La discrétisation des dérivées partielles des équations pour la convection naturelle

On remplace dans les équations de (3.18) à (3.22) :

$$\begin{aligned} &\frac{\Omega^{n+1}_{i,j}-\Omega^n_{i,j}}{\triangle t} + \frac{\Psi^n_{i,j+1}-\Psi^n_{i,j-1}}{2\triangle y}\times\frac{\Omega^n_{i+1,j}-\Omega^n_{i-1,j}}{2\triangle x} - \frac{\Psi^n_{i+1,j}-\Psi^n_{i-1,j}}{2\triangle x}\times\frac{\Omega^n_{i,j+1}-\Omega^n_{i,j-1}}{2\triangle y} \\ &= \mathrm{Pr}\Big(\frac{\Omega^n_{i+1,j}-2\Omega^n_{i,j}+\Omega^n_{i-1,j}}{(\triangle x)^2} + \frac{\Omega^n_{i,j+1}-2\Omega^n_{i,j}+\Omega^n_{i,j-1}}{(\triangle y)^2}\Big) - \mathrm{PrRa}\frac{\Theta^n_{i+1,j}-\Theta^n_{i-1,j}}{2\triangle x} \end{aligned} \tag{3.23}$$

$$\frac{\Psi^n_{i+1,j} - 2\Psi^n_{i,j} + \Psi^n_{i-1,j}}{(\triangle x)^2} + \frac{\Psi^n_{i,j+1} - 2\Psi^n_{i,j} + \Psi^n_{i,j-1}}{(\triangle y)^2} = -\Omega^n_{i,j} \tag{3.24}$$

$$U_{xi,j} = \frac{\Psi_{i,j+1} - \Psi_{i,j-1}}{2\triangle y} \tag{3.25}$$

$$U_{yi,j} = -\frac{\Psi_{i+1,j} - \Psi_{i-1,j}}{2\triangle x} \tag{3.26}$$

$$\begin{aligned} &\frac{\Theta^{n+1}_{i,j} - \Theta^n_{i,j}}{\triangle t} + (U^n_{xi,j}\frac{\Theta^n_{i+1,j} - \Theta^n_{i-1,j}}{2\triangle x} + U^n_{yi,j}\frac{\Theta^n_{i,j+1} - \Theta^n_{i,j-1}}{2\triangle y}) \\ &= \frac{\Theta^n_{i+1,j} - 2\Theta^n_{i,j} + \Theta^n_{i-1,j}}{(\triangle x)^2} + \frac{\Theta^n_{i,j+1} - 2\Theta^n_{i,j} + \Theta^n_{i,j-1}}{(\triangle y)^2} \end{aligned} \tag{3.27}$$

Pour simplifier, on prend $\triangle x = \triangle y = h$, on obtient :

$$\begin{aligned} \Omega^{n+1}_{i,j} &= \Omega^n_{i,j} - \frac{\triangle t}{4h^2}((\Psi^n_{i,j+1} - \Psi^n_{i,j-1})(\Omega^n_{i+1,j} - \Omega^n_{i-1,j}) - (\Psi^n_{i+1,j} - \Psi^n_{i-1,j})(\Omega^n_{i,j+1} - \Omega^n_{i,j-1})) \\ &+ \frac{\triangle t \mathrm{Pr}}{h^2}(\Omega^n_{i+1,j} + \Omega^n_{i-1,j} + \Omega^n_{i,j+1} + \Omega^n_{i,j-1} - 4\Omega^n_{i,j}) - \frac{\triangle t \mathrm{PrRa}}{2h}(\Theta^n_{i+1,j} - \Theta^n_{i-1,j}) \end{aligned} \tag{3.28}$$

$$\frac{1}{h^2}(\Psi^n_{i+1,j} + \Psi^n_{i-1,j} + \Psi^n_{i,j+1} + \Psi^n_{i,j-1} - 4\Psi^n_{i,j}) = -\Omega^n_{i,j} \tag{3.29}$$

$$U_{xi,j} = \frac{\Psi_{i,j+1} - \Psi_{i,j-1}}{2h} \tag{3.30}$$

$$U_{yi,j} = -\frac{\Psi_{i+1,j} - \Psi_{i-1,j}}{2h} \tag{3.31}$$

$$\begin{aligned} \Theta^{n+1}_{i,j} &= \Theta^n_{i,j} - \frac{\triangle t}{2h}[U^n_{xi,j}(\Theta^n_{i+1,j} - \Theta^n_{i-1,j}) + U^n_{yi,j}(\Theta^n_{i,j+1} - \Theta^n_{i,j-1})] \\ &+ \frac{\triangle t}{h^2}(\Theta^n_{i+1,j} + \Theta^n_{i-1,j} + \Theta^n_{i,j+1} + \Theta^n_{i,j-1} - 4\Theta^n_{i,j}) \end{aligned} \tag{3.32}$$

On applique la méthode de (SOR) à l'équation (3.29) et en prenant $\zeta \in [1, 2]$:

$$\Psi^{n+1}_{i,j} = 0.25\zeta(\Psi^n_{i+1,j} + \Psi^n_{i-1,j} + \Psi^n_{i,j+1} + \Psi^n_{i,j-1} + h^2\Omega^n_{i,j}) + (1-\zeta)\Psi^n_{i,j} \tag{3.33}$$

3.3 Algorithme de résolution

Pour résoudre les équations, on applique l'algorithme 3 avec les conditions aux limites celles de la section précédente.

Algorithme 3 LA CONVECTION NATURELLE

1 Initialisation de Ω, Ψ, U_x, U_y et Θ.
2 Calculer les conditions aux limites pour la vorticité, la fonction de courant et la température.
3 Résoudre l'équation (3.33) pour trouver la fonction de courant Ψ.
4 Résoudre l'équation (3.28) discrétisée pour trouver Ω.
5 Calculer le champ de vitesse en utilisant (3.30) et (3.31).
6 Résoudre l'(EC) en utilisant (3.32) pour trouver Θ.
7 Revenir à l'étape 2 et répétez le calcul pour une autre étape de temps jusqu'à l'attente le RS .

3.4 Les conditions aux limites et les résultats

Dans cette section, nous considérons deux configuration différentes.

3.4.1 Première configuration

Dans cette configuration les deux plans sont situés verticalement figure 3.1, nous choisissons comme conditions aux limites pour la vitesse u=0 sur les bords et pour la température comme il est indiqué dans cette figure :

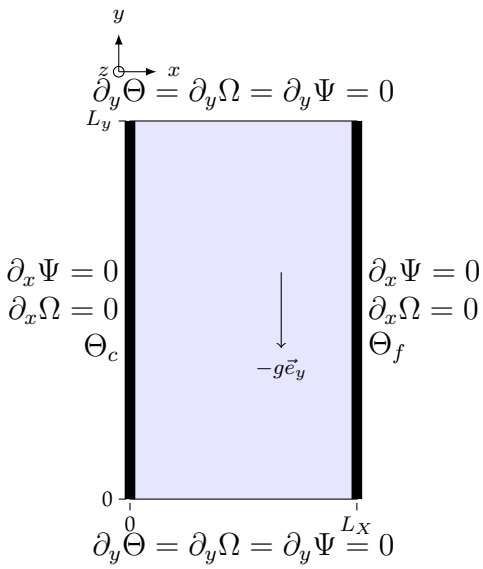
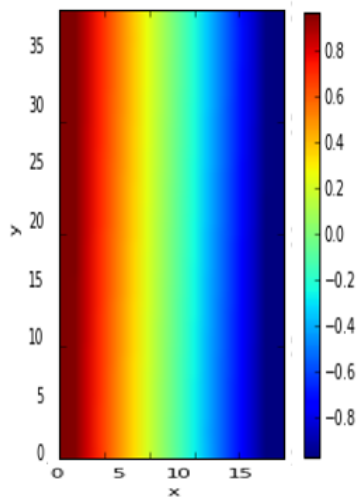


FIGURE 3.1 – Conditions aux limites de la configuration 1.

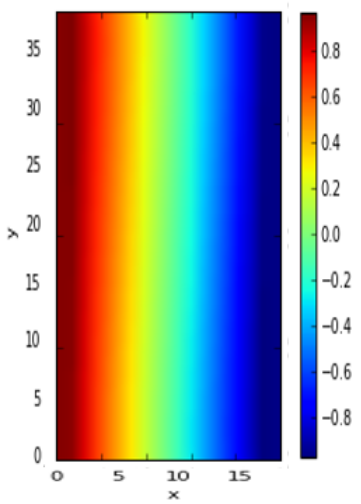
Les résultats de la première configuration :

Dans ces figures on trouve le profil du champ des températures dans un fluide (air et eau) siège d'une convection naturelle pour différents valeurs de Ra lorsque le régime stationnaire est atteint (3000 itérations temporelles).

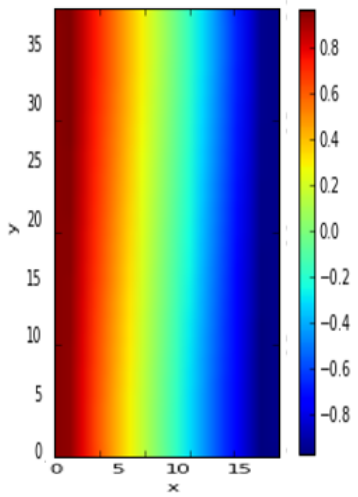
Pour l'air



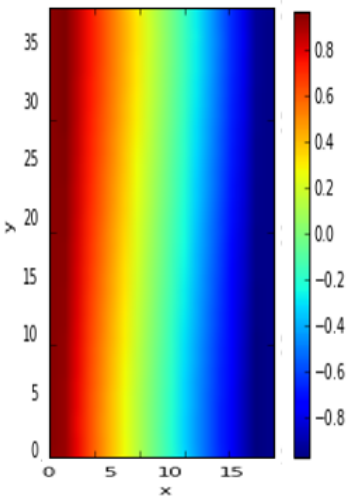
La température avec Ra=10 et Nu=0.11



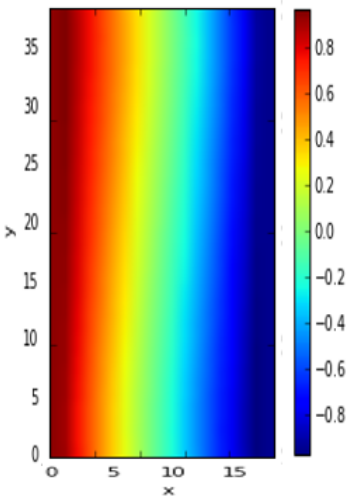
La température avec Ra=20 et Nu=0.13



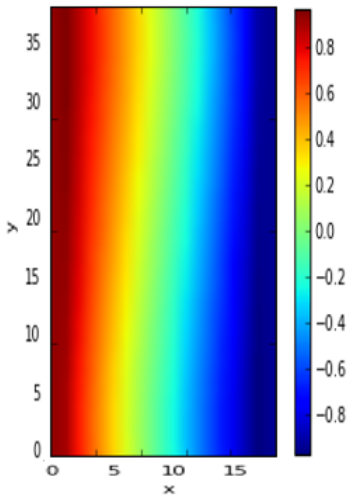
La température avec Ra=30 et Nu=0.14



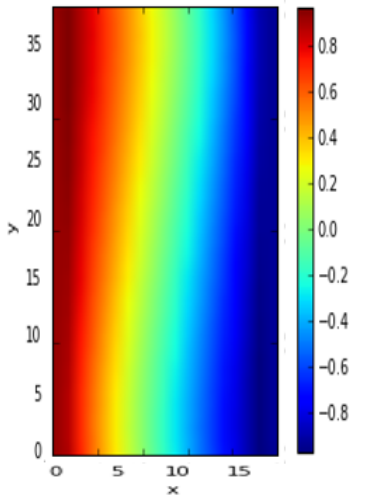
La température avec Ra=40 et Nu=0.15



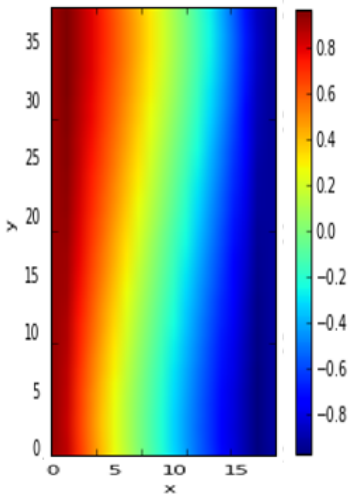
La température avec Ra=50 et Nu=0.159



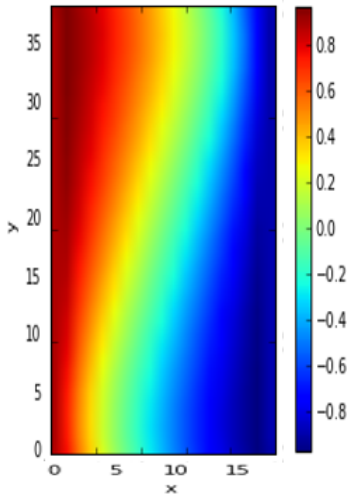
La température avec Ra=60 et Nu=0.167



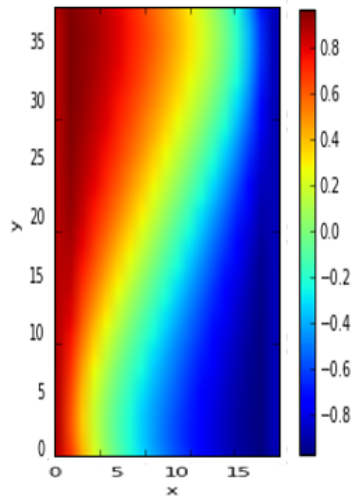
Ra=80 et Nu=0.179



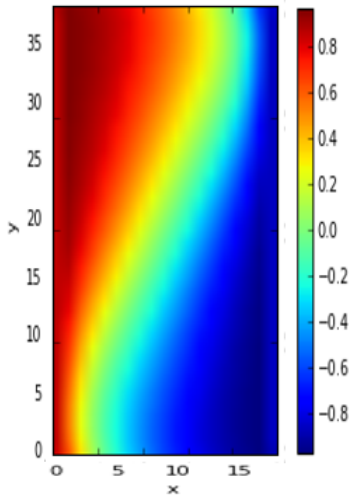
Ra=100 et Nu=0.189



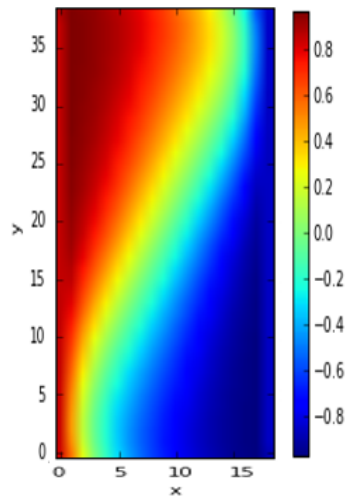
Ra=200 et Nu=0.226



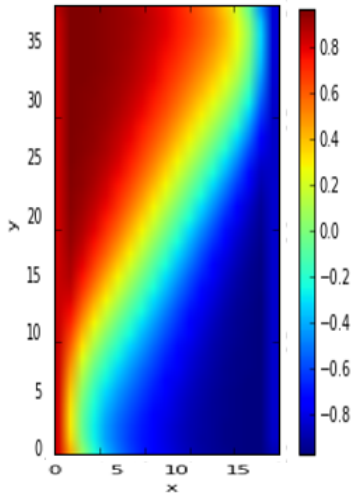
Ra=300 et Nu=0.25



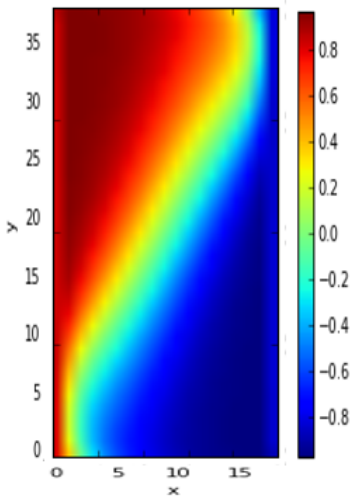
Ra=400 et Nu=0.268



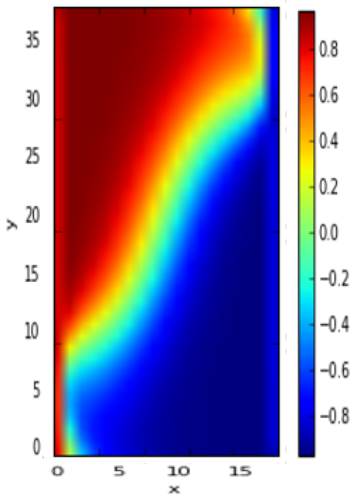
Ra=500 et Nu=0.284



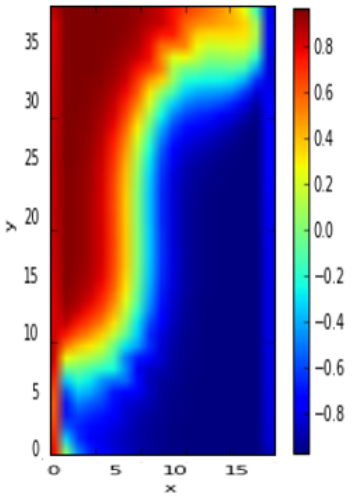
Ra=800 et Nu=0.319



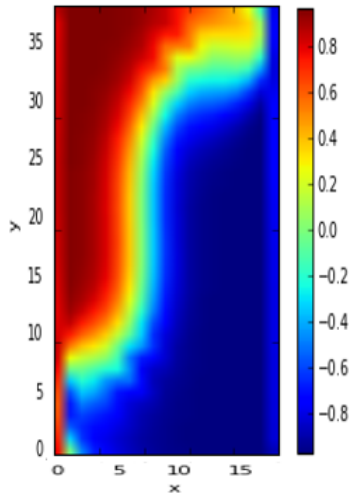
Ra=1000 et Nu=0.337



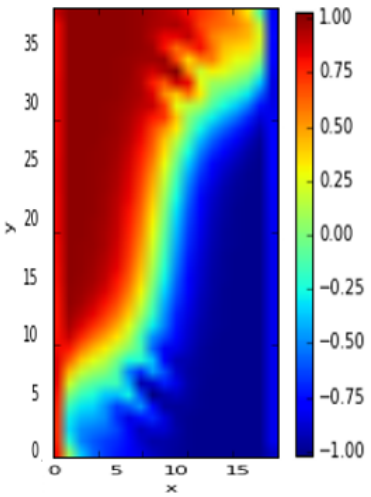
Ra=2000 et Nu=0.337



Ra=3000 et Nu=0.44

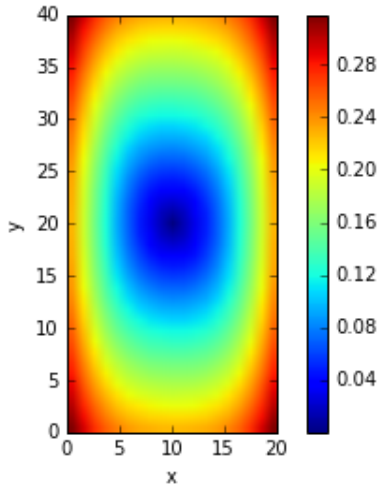


Ra=3000 et Nu=0.44

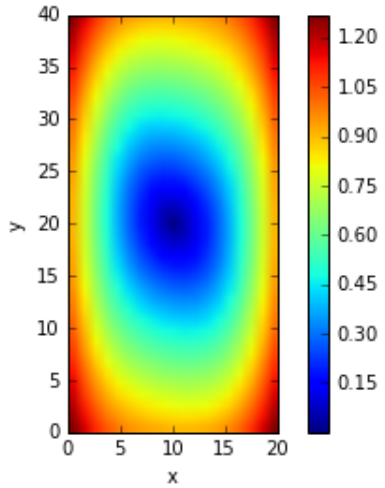


Ra=4000 et Nu=0.477

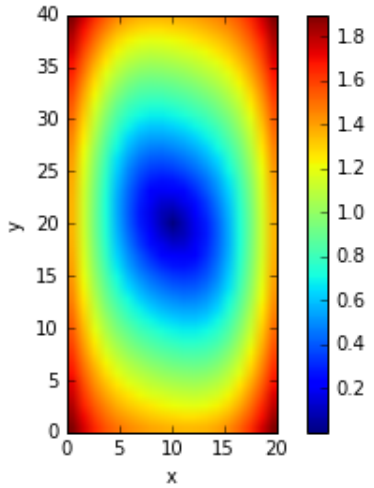
FIGURE 3.2 – La température dans l'air siège d'une CN pour des Ra allant de 300 à 4000.



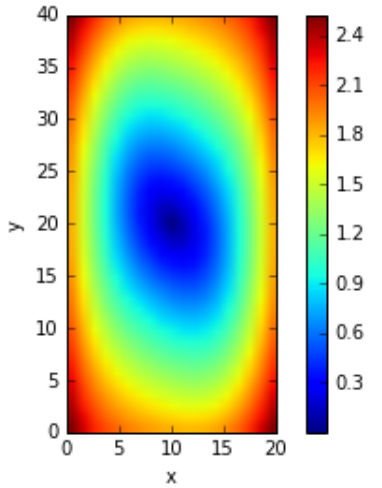
Le champ de la vitesse pour l'air avec Ra=10



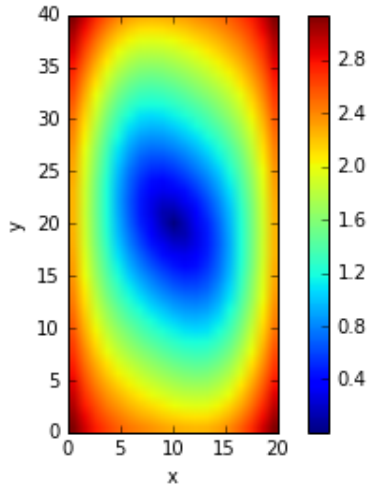
Le champ de la vitesse pour l'air avec Ra=40



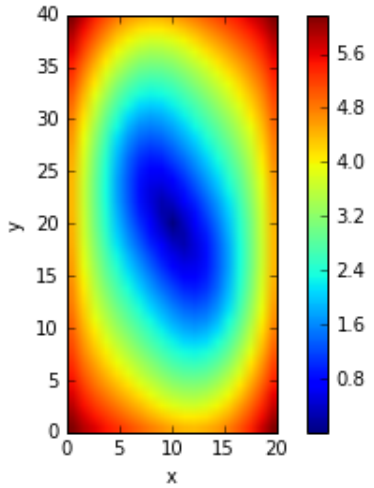
Le champ de la vitesse pour l'air avec Ra=60



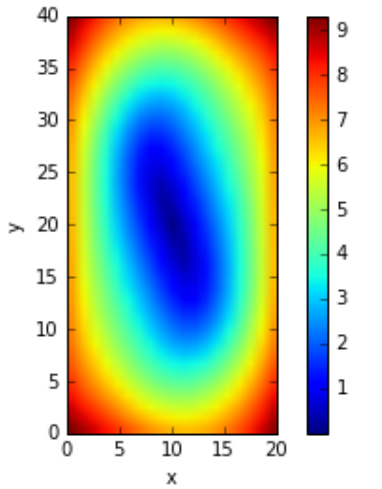
Le champ de la vitesse pour l'air avec Ra=80



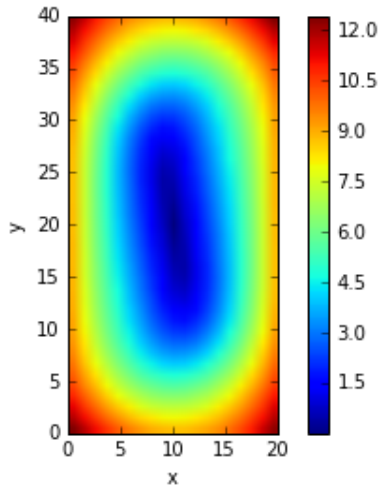
Le champ de la vitesse pour l'air avec Ra=100



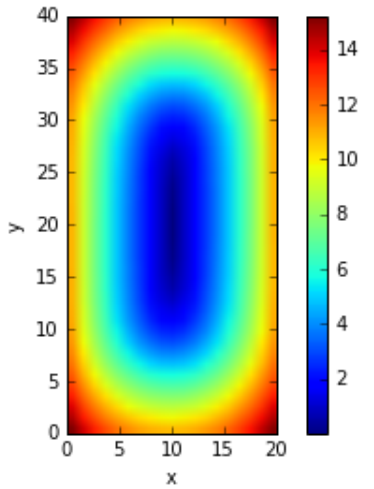
Le champ de la vitesse pour l'air avec Ra=200



La champ de la vitesse pour l'air avec Ra=300

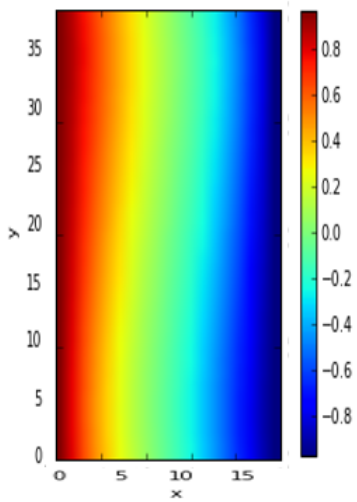


Le champ de la vitesse pour l'air avec Ra=400

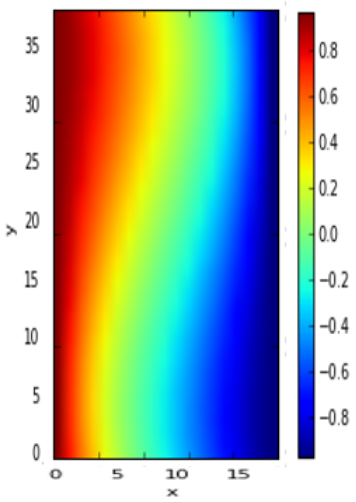


Le champ de la vitesse pour l'air avec Ra=500

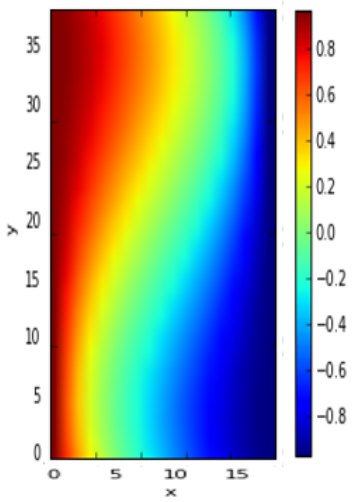
Pour l'eau :



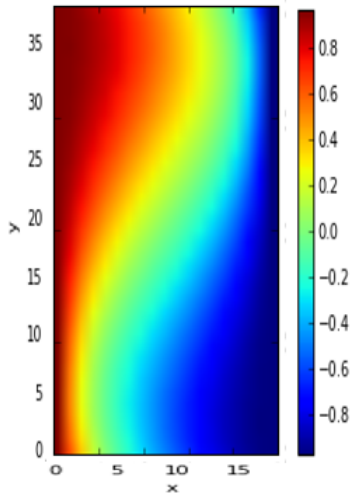
Ra=10 et Nu=0.89



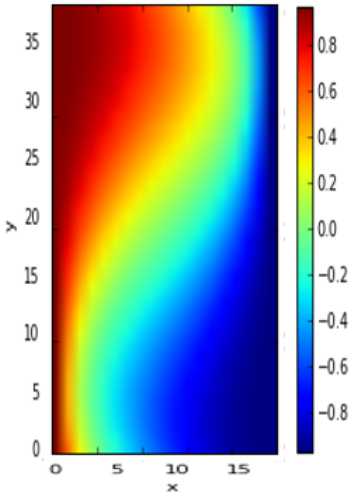
Ra=20 et Nu=1.057



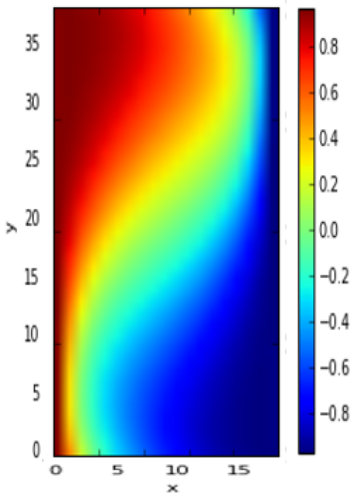
Ra=30 et Nu=1.17



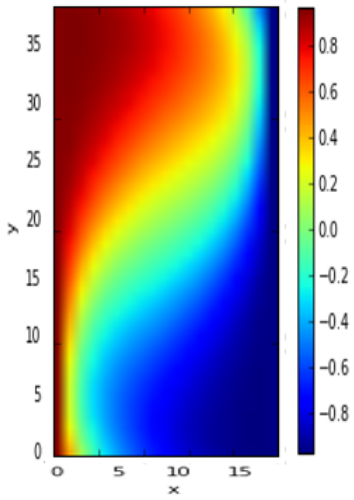
Ra=40 et Nu=1.26



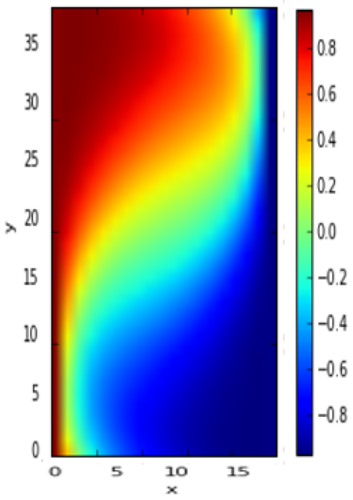
Ra=50 et Nu=1.33



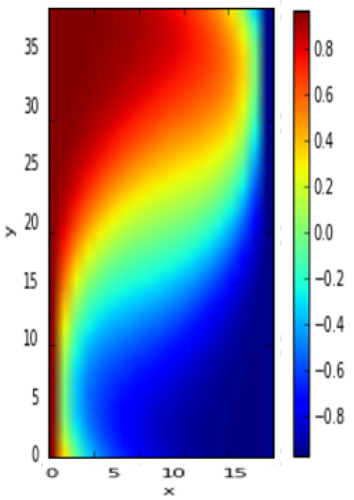
Ra=60 et Nu=1.39



Ra=70 et Nu=1.45

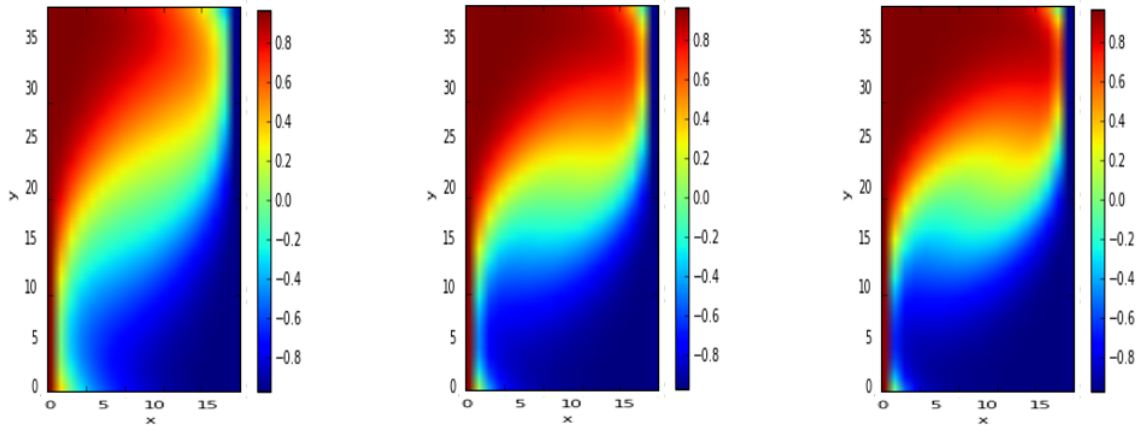


Ra=80 et Nu=1.495



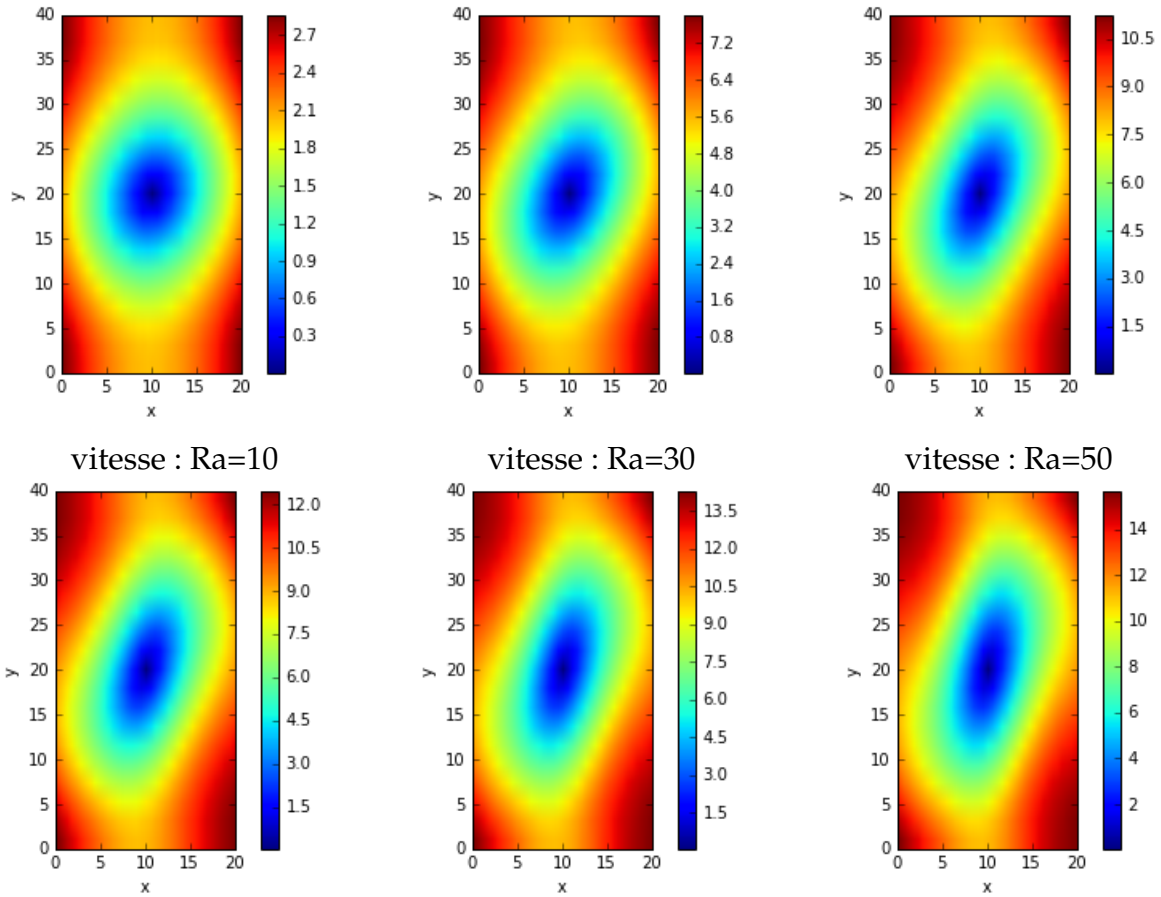
Ra=90 et Nu=1.54

FIGURE 3.3 – La température dans l'eau siège d'une CN pour des Ra allant de 10 à 90



Ra=100 et Nu=1.58 Ra=150 et Nu=1.75 Ra=200 et Nu=1.88

FIGURE 3.4 – La température dans l'eau siège d'une CN pour des Ra allant de 100 à 300



vitesse : Ra=10 vitesse : Ra=30 vitesse : Ra=50

vitesse : Ra=60 vitesse : Ra=80 vitesse : Ra=100

FIGURE 3.5 – Le champ de la vitesse dans l'eau siège d'une CN pour des Ra allant de 100 à 300 lorsque le RS est atteint.

Dans le cours transfert thermique **[1]** on définit le nombre de Nusselt local :

$$\mathrm{N}u_y = \frac{h_y y}{k} \tag{3.34}$$

Avec h_y le coefficient d'échange conducto-convectif local :

$$h_y = \frac{k}{T_c - T_f}\left|\left(\frac{\partial T}{\partial x}\right)_{x=0}\right| = k\left|\left(\frac{\partial \Theta}{\partial x}\right)_{x=0}\right| \tag{3.35}$$

Nous obtenons la valeur moyenne Nu_{Ly} par l'intégration suivante :

$$\mathrm{N}u_{Ly} = \frac{1}{L_y}\int_0^{L_y} \mathrm{N}u_y dy = \frac{1}{L_y}\int_0^{L_y} \frac{h_y y}{k} dy \tag{3.36}$$

Nous discrétisons l'équation (3.36) :

$$h_{yj} = k\left|\frac{\Theta_{2,j} - \Theta_{0,j}}{2h}\right| \tag{3.37}$$

Nous calculons numériquement $\mathrm{N}u_{Ly}$ en fonction de Ra pour l'air :

Ra	10	20	30	40
$\mathrm{N}u_{Ly}$	0.004	0.008	0.0130	0.0173
Ra	50	60	80	100
$\mathrm{N}u_{Ly}$	0.0216	0.0258	0.034	0.0419
Ra	200	300	400	800
$\mathrm{N}u_{Ly}$	0.073	0.0895	0.0913	0.10

TABLE 3.2 – La valeur numérique de $\mathrm{N}u_{Ly}$ en fonction de Ra pour l'air.

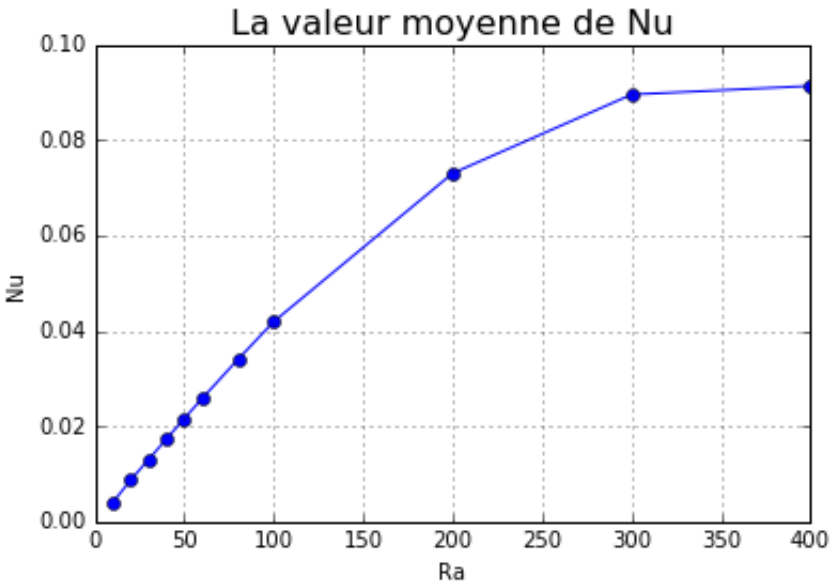


FIGURE 3.6 – Le nombre de Nusselt moyen $\mathrm{N}u_{Ly}$ en fonction de Ra pour l'air.

Nous calculons numériquement $\mathrm{N}u_{Lx}$ en fonction de Ra pour l'eau :

Ra	10	20	30	40
Nu_{Ly}	2.07	2.08	2.15	2.24
Ra	50	60	80	100
Nu_{Ly}	2.416	2.62	2.89	3.41

TABLE 3.3 – La valeur numérique de Nu_{Ly} en fonction de Ra pour l'eau.

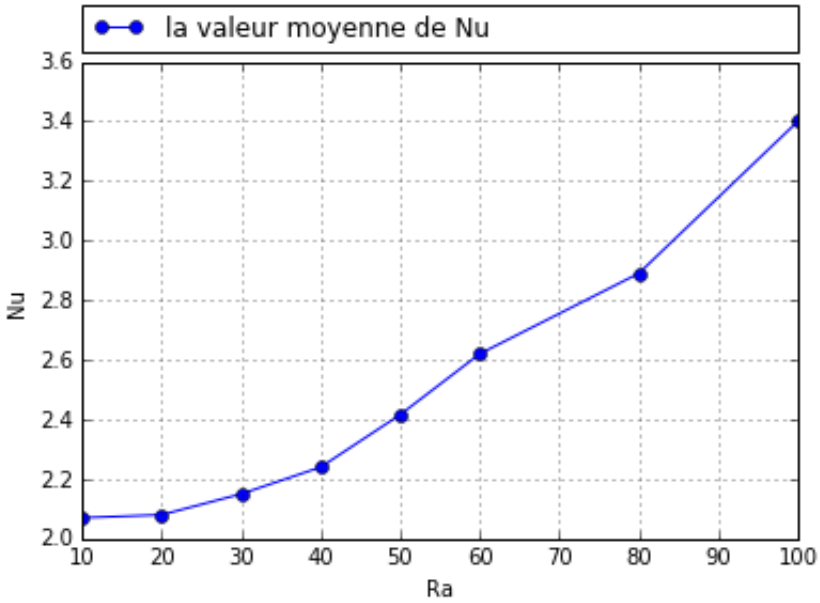


FIGURE 3.7 – Le nombre de Nusselt moyen Nu_{Ly} en fonction de Ra pour l'eau.

3.4.2 Deuxième configuration

Dans cette configuration les deux plans sont situés horizontalement figure 3.7 :

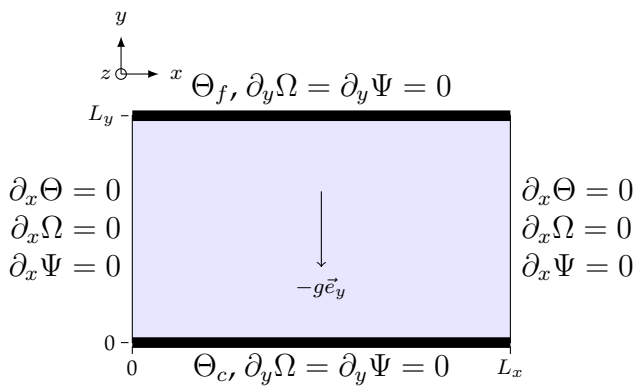


FIGURE 3.8 – Conditions aux limites de la configuration 2.

Les résultats de la deuxième configuration :

Après l'application de l'algorithme 3 on obtient [2] :

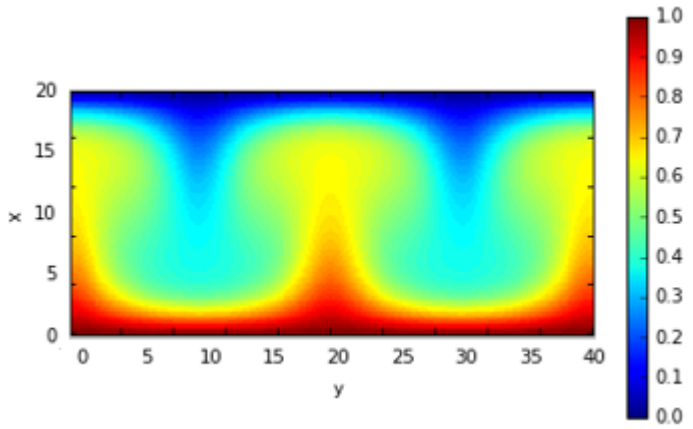


FIGURE 3.9 – Convection libre pour Nx=20, Ny=40, Pr=1, Ra=20000, $T_c = 1$ et $T_f = 0$.

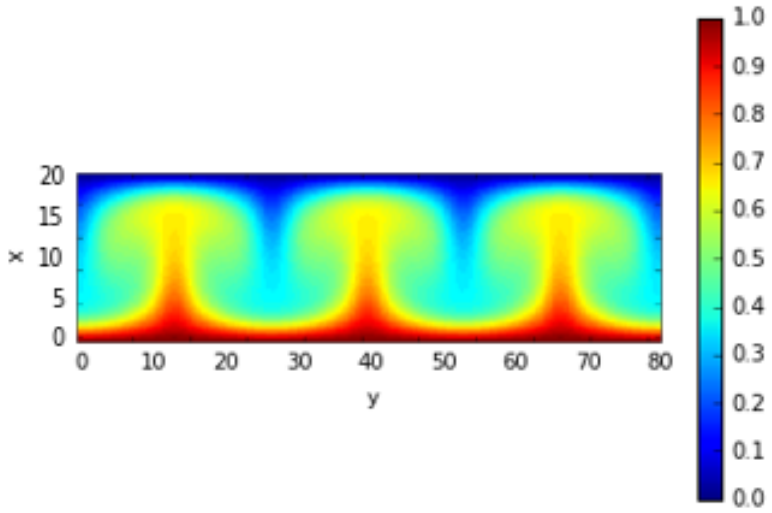


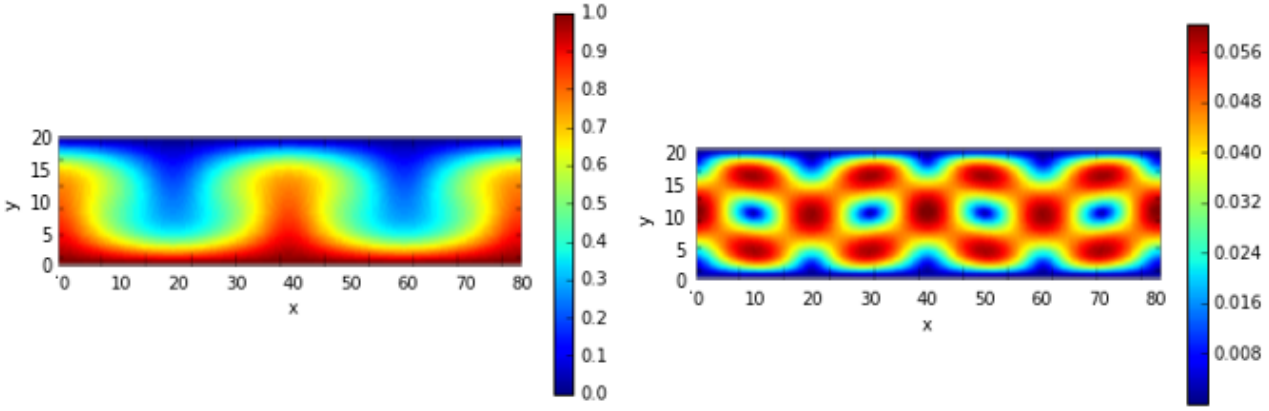
FIGURE 3.10 – Convection libre pour Nx=20, Ny=80, Pr=1, Ra=20000, $T_c = 1$ et $T_f = 0$

Dans le champ de pesanteur, cette configuration est instable en raison du gradient de densité engendré par la différence de température entre le haut et le bas de l'enceinte : le fluide du bas se dilate, devient plus léger que le fluide du haut et tend à s'élever sous l'effet de la poussée d'Archimède.

Influence du nombre de Rayleigh sur le champ de température :

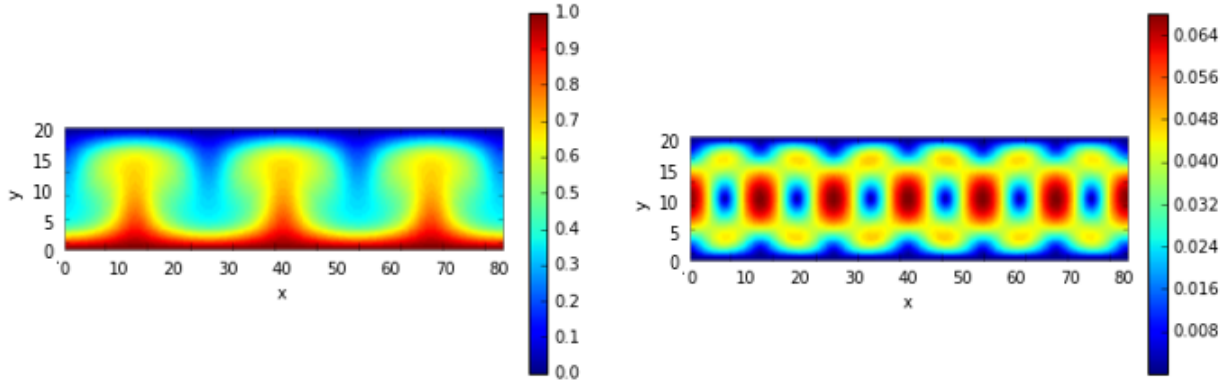
On varie la valeur de Ra pout $T_c = 1$ et $T_f = 0$, nous obtenons les résultats suivantes :

2. Ces résultats sont obtenus lorsque le régime stationnaire est atteint pour un nombre d'itérations temporelles 30000.



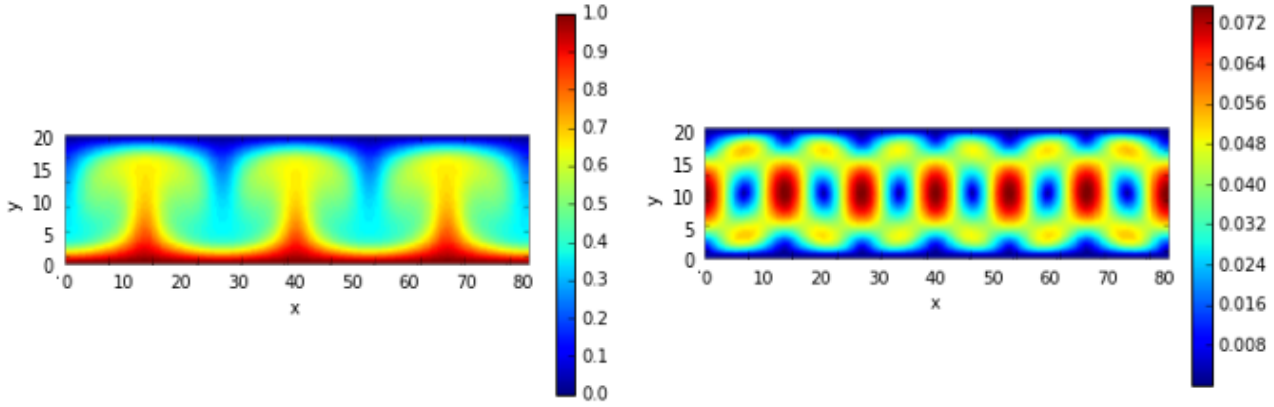
Le champ de la température. Le champ de la vitesse.

FIGURE 3.11 – Les champs de la Température et de la vitesse pour Pr=1 et Ra=5000.



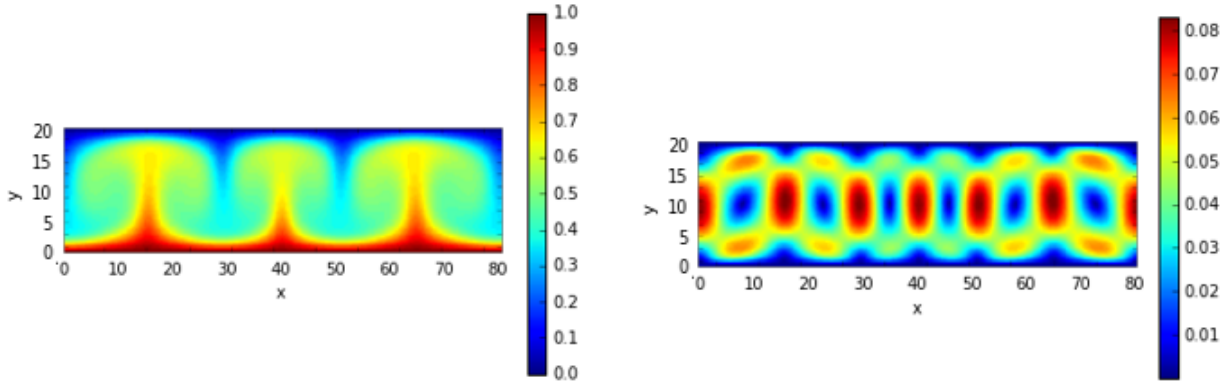
Le champ de la température. Le champ de la vitesse.

FIGURE 3.12 – Les champs de la Température et de la vitesse pour Pr=1 et Ra=10^4.



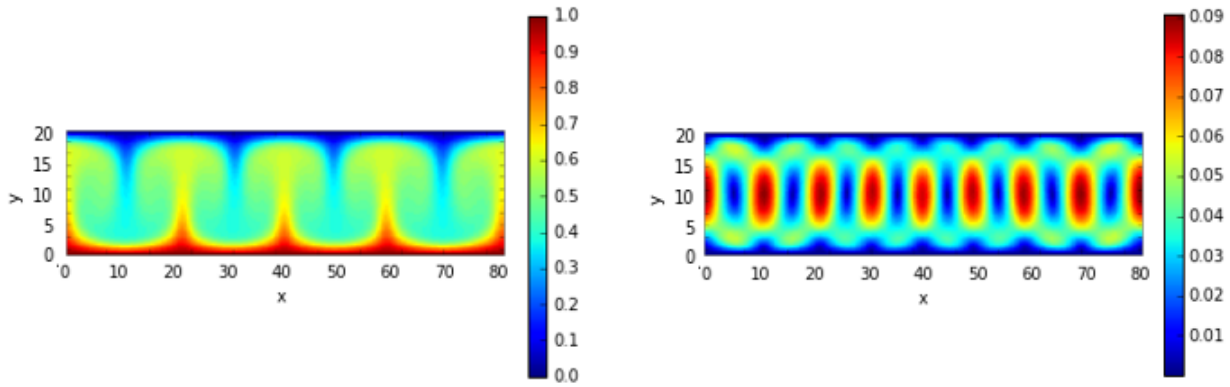
Le champ de la température. Le champ de la vitesse.

FIGURE 3.13 – Les champs de la Température et de la vitesse pour Pr=1 et Ra=2.10^4.



Le champ de la température. Le champ de la vitesse.

FIGURE 3.14 – Les champs de la Température et de la vitesse pour Pr=1 et Ra=4.10^4



Le champ de la température. Le champ de la vitesse.

FIGURE 3.15 – Les champs de la Température et de la vitesse pour Pr=1 et Ra=8.10^4

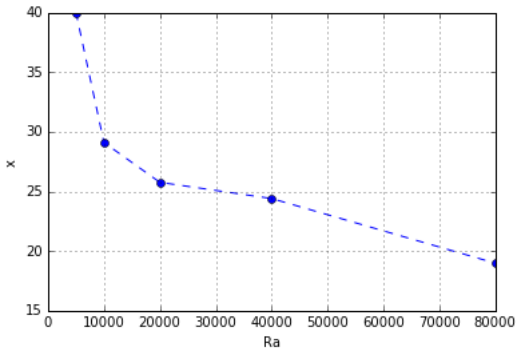


FIGURE 3.16 – La période du motif en fonction de Ra.

On voit bien d'après ces figures que nous avons un motif thermique qui se répète d'une façon périodique. Et que la période diminue lorsque le nombre de Reynolds augmente c'est-à-dire ces motifs se rapprochent quand Re augment.

CHAPITRE

4 Conclusion

Dans ce projet nous avons simulé numériquement des modèles mathématiques correspondant à des phénomènes physiques très répondus en ingénierie (un écoulement de poiseuille et La convection) en utilisant une méthode parmi plusieurs autres qui est la méthode de vorticité. Cette méthode nous a permis de se débrouiller du terme en pression qui figure dans l'équation de Navier Stockes et d'avoir une formulation moins compliqué de l'équation avec d'autres termes qui sont la vorticité et la fonction de courant, en revanche les conditions aux limites associés à la vorticité ne sont pas accessible facilement par cette méthode.
Dans le cadre de ce projet de fin de fin d'année, nous avons simulé les phénomènes de convections naturelle et forcée dans le cas des géométries simples par la méthode des différences finis qui a donné des résultats de simulation qui approchent bien les solutions analytiques. L'étude peut s'étendre à des géométries plus complexes afin de s'approcher plus aux géométries réelles existant dans les industries et la vie en générale, et dans ce cas nous pouvons utiliser la méthode des éléments finis aux lieu des différences finies avec les logiciels comme FreeFem++ et Fenics.
Ce projet a été une opportunité pour mieux maitriser, les connaissances qui figurent dans notre cursus. Pour se familiariser au langage de programmation Python.
Finalement, ce projet a été l'occasion de comprendre toutes les qualités que doit avoir un ingénieur spécialisé en mécanique des fluides numérique : un sens de l'organisation et des méthodes de travail propres à tout ingénieur, une maîtrise des outils numériques et enfin un savoir théorique en mécanique des fluides.

Bibliographie

[1] Rodolphe HEYD : Cours transfert thermique, École nationale d'ingénieurs de Tunis, 2016.

[2] Rodolphe HEYD : Cours PYTHON, École nationale d'ingénieurs de Tunis, 2016.

[3] T. J. CHUNG «COMPUTATIONAL FLUID DYNAMICS» 2002.

[4] Jérémie GRESSIER : Calcul d'écoulements par méthode Vortex, mars 2007.

[5] Laurent RISSER : Différences finies pour la résolution numérique des équations de la mécanique des fluides, 4 février 2006.

[6] A. SALIH : Streamfunction-Vorticity Formulation,Indian Institute of Space Science and Technology, Thiruva nanthapuram, 2013

[7] Maciej MATYKA :Solution to two-dimensional Incompressible Navier-Stokes Equations with SIMPLE, SIMPLER and Vorticity-Stream Function Approaches. Driven-Lid Cavity Problem : Solution and Visualization, 2004.

ANNEXE

A Codes Python accompagnant le rapport

Cette partie rassemble les scripts Python utilisés dans les chapitres précédent.

A.1 Écoulement de poiseuille 1D

Ce code permet de résoudre un écoulement de poiseuille en faisant des simplifications sur le champ de vitesse selon le domaine que nous avons.

```
#importation des bibliotheques
import numpy as np
import matplotlib.pyplot as plt
from numpy import linalg as LA

visc=0.25  #la viscosit  du fluide
#condtion aux limites de w
w0=1.0
wlx=-1.0
#condition initiale de w
wi=0

Lx=80 #nombre d'it rations spatialle
Nt=400 #nombre d'it rations temporelle
deltat=1/Nt #pas de discr tisation temporelle
deltax=1/80 #pas de discr tisation spatialle
alpha=(visc*deltat)/(deltax**2) #alpha
# Creation de la matrice position adimensionne X
X=np.zeros(Lx+1)
for i in range(0,Lx+1):
    X[i]=0+i*deltax
#Implementation de la matrice A tridiagonale
A=(1+2*alpha)*np.identity(Lx-1)
xAs=-alpha*np.ones(Lx-2)
yAs=np.diagflat(xAs,1)
yAi=np.diagflat(xAs,-1)
A=A+yAs+yAi
#Implementation de la matrice B bidiagonale
B=np.identity(Lx)
xBs=-np.ones(Lx-1)
yBi=np.diagflat(xBs,-1)
B=B+yBi
#Definition des matrices vorticit
wA=np.ones((Lx-1,1))
wA0L=np.zeros((Lx-1,1))
wn=np.zeros((Lx+1,1))
```

```
#Prise en compte des conditions aux limites
wA0L[0]=w0
wA0L[Lx-2]=wlx
# Conditions initiales
wA=wA*wi
#Inversion de la matrice tridiagonale A
InvA=LA.inv(A)
wn=np.append(wA0L[0],np.append(wA,wA0L[Lx-2]))
#Inversion de la matrice bidiagonale B
InvB=LA.inv(B)

#Calcul des matrices finales
fig = plt.figure()
ax1=fig.add_subplot(311)
ax2=fig.add_subplot(312)
ax3=fig.add_subplot(313)
for k in range(1,Nt):
    wA=np.dot(InvA,wA+alpha*wA0L)
    wn=np.append(wA0L[0],np.append(wA,wA0L[Lx-2]))
    vB=np.dot(InvB,deltax*np.delete(wn,Lx))
    vn=np.append(0,vB)
    psiB=np.dot(InvB,-deltax*vB)
    psin=np.append(0,psiB)
    if (k==20):  # Affichage des courbes anant le RS
        ax1leg='{:.2f}'.format(k*deltat)
        ax1.plot(X,wn,label=r'$\mathrm{t= }$'+ax1leg)
        ax1.grid()
        ax2leg='{:.2f}'.format(k*deltat)
        ax2.plot(X,vn,label=r'$\mathrm{t= }$'+ax2leg)
        ax2.grid()
        ax3leg='{:.2f}'.format(k*deltat)
        ax3.plot(X,psin,label=r'$\mathrm{t= }$'+ax3leg)
        ax3.grid()

#Affichage final des courbes
ax1leg='{:.2f}'.format(k*deltat)
ax1.plot(X,wn,label=r'$\mathrm{t= }$'+ax1leg)
ax2leg='{:.2f}'.format(k*deltat)
ax2.plot(X,vn,label=r'$\mathrm{t= }$'+ax2leg)
ax3leg='{:.2f}'.format(k*deltat)
ax3.plot(X,psin,label=r'$\mathrm{t= }$'+ax3leg)
ax1.legend(loc=3)
ax2.legend(loc=0)
ax3.legend(loc=0)
ax1.set_ylabel(r'$\omega(x,\mathrm{t})}$',size=16)
ax2.set_ylabel(r'$uy(x,t)}$ ',size=16)
ax3.set_ylabel(r'$\psi(x,t)}$ ',size=16)
plt.show()

# La fin du programe
print('End')
```

A.2 Écoulement de poiseuille 2D

Ce code nous permet de résoudre un écoulement de poiseuille sans faire des simplifications sur le vitesse d'écoulement.

```
#Imortation des bibliotheques
import numpy as np
import matplotlib.pylab as plt

Nx=20 #nombre des iterations selon x
Ny=40 #nombre des itrations selon y
Maxt1=2000 #nombre des iterations temporelle
Visc=0.025 #La viscosite de fluide nu
Maxt2=50 #nombre des iteration pour SOR
zeta=1.5 #coefficien de relaxation
p=0.1 #Pas spatial
dt=0.01 #pas temporelle

#Inisialisation des inconnus    0
psi=np.zeros((Nx+1,Ny+1))
w=np.zeros((Nx+1,Ny+1))
k=np.zeros((Nx+1,Ny+1))
u=np.zeros((Nx+1,Ny+1))
v=np.zeros((Nx+1,Ny+1))
div=np.zeros((Nx+1,Ny+1))
X=np.zeros(Nx+1)
Y=np.zeros(Nx+1)
Err=np.zeros(Nx-2)

#les distances carecteristiques du domaine
Lx=p*(Nx-2)
Ly=p*(Ny-2)

#Le nombre de reynolds
Re=(1.0*(Lx**2))/(6*Visc)

#La condition de CFL
CFL=Visc*dt/(p**2)
for i in range(0,Nx+1):
    X[i]=0+i*p

for it1 in range(1,Maxt1+1): #Boucle des iterations temporelles
    w[1,:]=1.0                #Les conditions aux limites pour w
    w[Nx-1,:]=-1.0
    w[:,0]=w[:,2]
    w[:,Ny]=w[:,Ny-2]
    psi[2,:]=psi[0,:]        #Les conditions aux limites pour psi
    psi[Nx,:]=psi[Nx-2,:]
    psi[:,0]=psi[:,2]
    psi[:,Ny]=psi[:,Ny-2]
    for it2 in range(1,Maxt2+1): #Boucle des iterations pour la methode SOR
        for i in range(1,Nx):
            for j in range(1,Ny):
                psi[i,j]=0.25*zeta*(psi[i+1,j]+psi[i-1,j]+psi[i,j+1]+psi[i,j-1]+
    p*p*w[i,j])+(1.0-zeta)*psi[i,j]
    for i in range(1,Nx):
        for j in range(1,Ny):
            k[i,j]=-0.25*((psi[i,j+1]-psi[i,j-1])*(w[i+1,j]-w[i-1,j])-(psi[i+1,j
    ]-psi[i-1,j])*(w[i,j+1]-w[i,j-1]))/(p*p)+Visc*(w[i+1,j]+w[i-1,j]+w[i,j+1]+w[
    i,j-1]-4.0*w[i,j])/(p*p)
    w[1:Nx,1:Ny]=w[1:Nx,1:Ny]+dt*k[1:Nx,1:Ny] #mettre e jour w
    for i in range(1,Nx):                          #calcul de ux et uy
        for j in range(1,Ny):
            u[i,j]=(psi[i,j+1]-psi[i,j-1])/(2*p)
```

```
            v[i,j]=-(psi[i+1,j]-psi[i-1,j])/(2*p)
w[1,:]=1.0
w[Nx-1,:]=-1.0

#clacul de div(u)
for i in range(1,Nx):
        for j in range(1,Ny):
            div[i,j]=(u[i+1,j]-u[i-1,j])/(2*p) +(v[i,j+1]-v[i,j-1])/(2*p)

#calcul de la solution exacte uy
for i in range(Nx+1):
        Y[i]=-(1.0/Lx)*(X[i]*X[i])+1.0*X[i]

#calcul de l'erreur entre la vitesse theorique et la vitesse pratique
Err=abs(Y[0:Nx-1]-v[1:Nx,20])

#Affichage de fonction de courant psi
CS=plt.contour(np.transpose(psi[1:Nx,1:Ny]),18)
plt.clabel(CS, inline=1, fontsize=10)
plt.xlabel('x')
plt.ylabel('y')
plt.show()

#Affichage de la Vorticite w
plt.imshow(np.transpose(w[1:Nx,1:Ny]),interpolation='none',extent=[0,20,0,40])
plt.colorbar()
plt.xlabel('x')
plt.ylabel('y')
plt.show()

#Affichage de uy numerique et analytique
plt.plot(v[1:Nx,20],label='Solution num rique')
plt.plot(abs(Y[0:Nx-1]),'r',label='Solution exacte')
plt.legend(bbox_to_anchor=(0.,1.02,1.,.102),loc=3,ncol=2,mode="expand",
    borderaxespad=0.)
plt.xlabel('x')
plt.ylabel('$u_y$')
plt.grid()
plt.show()

#Affichage de ux
plt.plot(u[1:Nx,5],label='Solution num rique de ux')
plt.legend(bbox_to_anchor=(0.,1.02,1.,.102),loc=3,ncol=2,mode="expand",
    borderaxespad=0.)
plt.xlabel('x')
plt.ylabel('$u_x$')
plt.grid()
plt.show()

# Affichage de la courbe d'erreur
plt.plot(Err,label='Courbe d"erreur pour Re={:.2f}'.format(Re))
plt.grid()
plt.legend(bbox_to_anchor=(0.,1.02,1.,.102),loc=3,ncol=2,mode="expand",
    borderaxespad=0.)
plt.show()

#Affichage des courbes de div(u)
plt.plot(div[1:Nx,1:Ny])
plt.title(u"Les courbes de div(u) pour tous les y en fonction de x",fontsize=10)
```

```
plt.xlabel('x')
plt.ylabel('div(u)')
plt.grid()
plt.show()

# La fin du programe
print('End')
```

A.3 La Convection Forcée

Ce code nous permet de trouver le profile de la couche limite thermique.

```
#Importations des bibliotheques
import numpy as np
import matplotlib.pylab as plt


Nx=22 #nombre des iterations selon x
Ny=42  #nombre des iterations selon y
Maxt1=400 #nombre des iterations temporelle
Visc=0.25 #La viscosite de fluide nu
dt=0.01 #pas tompporelle
Maxt2=100 #nombre des iterations SOR
zeta=1.5 #parametre de SOR

#Inisialisation des inconnus    0
psi=np.zeros((Nx+1,Ny+1))
w=np.zeros((Nx+1,Ny+1))
k=np.zeros((Nx+1,Ny+1))
u=np.zeros((Nx+1,Ny+1))
v=np.zeros((Nx+1,Ny+1))
S=np.zeros((Nx+1,Ny+1))
T=np.zeros((Nx+1,Ny+1))

p=0.1 #pas spacialle
#les distances carecteristiques du domaine
Lx=p*(Nx-2)
Ly=p*(Ny-2)
Re=(Lx**2)/(6*Visc) #Le nombre de reynolds
CFL=Visc*dt/(p**2) #La condition de CFL

for it1 in range(1,Maxt1+1): #Boucle des iterations temporelles
    w[1,:]=1.0                  #Les conditions aux limites pour w
    w[Nx-1,:]=-1.0
    w[:,0]=w[:,2]
    w[:,Ny]=w[:,Ny-2]
    psi[2,:]=psi[0,:]        #Les conditions aux limites pour psi
    psi[Nx,:]=psi[Nx-2,:]
    psi[:,0]=psi[:,2]
    psi[:,Ny]=psi[:,Ny-2]
    T[1,:]=0                    #Les conditions aux limites pour psi
    T[Nx-1,:]=1
    T[:,1]=0#T[:,2]
    T[:,Ny-1]=T[:,Ny-2]
    for it2 in range(1,Maxt2+1):    #Boucle des iterations pour la methode SOR
        for i in range(1,Nx):       #calcul de psi
            for j in range(1,Ny):
```

```
                psi[i,j]=0.25*zeta*(psi[i+1,j]+psi[i-1,j]+psi[i,j+1]+psi[i,j-1]+p*p*w[i,j])+(1.0-zeta)*psi[i,j]
    for i in range(1,Nx):  #calcul de w
        for j in range(1,Ny):
            k[i,j]=-0.25*((psi[i,j+1]-psi[i,j-1])*(w[i+1,j]-w[i-1,j])-(psi[i+1,j]-psi[i-1,j])*(w[i,j+1]-w[i,j-1]))/(p*p)+Visc*(w[i+1,j]+w[i-1,j]+w[i,j+1]+w[i,j-1]-4.0*w[i,j])/(p*p)
    w[1:Nx,1:Ny]=w[1:Nx,1:Ny]+dt*k[1:Nx,1:Ny] #mettre e jour w
    for i in range(1,Nx):    #calcul des composantes de la ditesse
        for j in range(1,Ny):
            u[i,j]=(psi[i,j+1]-psi[i,j-1])/(2*p)
            v[i,j]=-(psi[i+1,j]-psi[i-1,j])/(2*p)
    for i in range(1,Nx):   #calcul de T
        for j in range(1,Ny):
            S[i,j]=-0.5*(u[i,j]*(T[i+1,j]-T[i-1,j])+v[i,j]*(T[i,j+1]-T[i,j-1]))/p+0.1*(T[i,j+1]+T[i,j-1]+T[i-1,j]+T[i+1,j]-4*T[i,j])/(p*p)
    T[1:Nx,1:Ny]=T[1:Nx,1:Ny]+dt*S[1:Nx,1:Ny]  #mettre e jour T
w[1,:]=1.0
w[Nx-1,:]=-1.0

#Affichage de la Vorticite w
plt.imshow(w[1:Nx,1:Ny])
plt.title(u"$\omega$",fontsize=16)
plt.colorbar()
plt.show()

#Affichage de la Temperature T
plt.imshow(T[1:Nx,1:Ny],extent=[0,40,0,20])
plt.title(u"T",fontsize=16)
plt.colorbar()
plt.title(u"La couche limite thermique",fontsize=10)
plt.xlabel('x')
plt.ylabel('y')
plt.show()

# La fin du programe
print('End')
```

A.4 La Convection Naturelle

Ce code nous permet de trouver le profile de la température siège d'une convection naturelle.

```
#Imortation des bibliotheques
import numpy as np
import matplotlib.pylab as plt

Nx=20 #nombre des iterations selon x
Ny=40 #nombre des itrations selon y
Maxt1=400 #nombre des iterations temporelle
Maxt2=200 #nombre des iteration pour SOR
zeta=1.5 #coefficien de relaxation
p=0.1 #Pas spatial
dt=0.001 #pas temporelle

#initialisation des donn es
Psi=np.zeros((Nx+1,Ny+1))
```

```
Omega=np.zeros((Nx+1,Ny+1))
k=np.zeros((Nx+1,Ny+1))
Ux=np.zeros((Nx+1,Ny+1))
Uy=np.zeros((Nx+1,Ny+1))
S=np.zeros((Nx+1,Ny+1))
Theta=np.zeros((Nx+1,Ny+1))
Nu=np.zeros(Ny+1)
hy=np.zeros(Ny+1)
NuL=np.zeros(Ny+1)
Nuy=np.zeros(Ny+1)
#CFL=Visc*dt/(p**2)

Pr=0.1#Visc/kappa
Ra=100#(beta*g*(L**3)*deltaT)/(kappa*Visc)

for it1 in range(1,Maxt1+1):
    Psi[0,:]=Psi[2,:]     #Les conditions aux limites pour Psi
    Psi[Nx,:]=Psi[Nx-2,:]
    Psi[:,0]=Psi[:,2]
    Psi[:,Ny]=Psi[:,Ny-2]
    Omega[0,:]=Omega[2,:]    #Les conditions aux limites pour Omega
    Omega[Nx,:]=Omega[Nx-2,:]
    Omega[:,0]=Omega[:,2]
    Omega[:,Ny]=Omega[:,Ny-2]
    Theta[0,:]=-1#Theta[2,:]    #Les conditions aux limites pour Theta
    Theta[Nx,:]=1#Theta[Nx-2,:]
    Theta[:,0]=Theta[:,2]
    Theta[:,Ny]=Theta[:,Ny-2]
    for i in range(1,Nx): #Calcul de Theta
        for j in range(1,Ny):
            S[i,j]=-0.5*(((Psi[i,j+1]-Psi[i,j-1])/(2*p))*(Theta[i+1,j]-Theta[i
    -1,j])+(-(Psi[i+1,j]-Psi[i-1,j])/(2*p))*(Theta[i,j+1]-Theta[i,j-1]))/p+(
    Theta[i,j+1]+Theta[i,j-1]+Theta[i-1,j]+Theta[i+1,j]-4*Theta[i,j])/(p*p)
    Theta[1:Nx,1:Ny]=Theta[1:Nx,1:Ny]+dt*S[1:Nx,1:Ny]
    for it2 in range(1,Maxt2+1): #iteration SOR
        for i in range(1,Nx):  #Calcul de Psi
            for j in range(1,Ny):
                Psi[i,j]=0.25*zeta*(Psi[i+1,j]+Psi[i-1,j]+Psi[i,j+1]+Psi[i,j-1]+
    p*p*Omega[i,j])+(1.0-zeta)*Psi[i,j]
    for i in range(1,Nx): #calcul de Psi
        for j in range(1,Ny):
            k[i,j]=-0.25*((Psi[i,j+1]-Psi[i,j-1])*(Omega[i+1,j]-Omega[i-1,j])-(
    Psi[i+1,j]-Psi[i-1,j])*(Omega[i,j+1]-Omega[i,j-1]))/(p*p)+Pr*(Omega[i+1,j]+
    Omega[i-1,j]+Omega[i,j+1]+Omega[i,j-1]-4.0*Omega[i,j])/(p*p)-Pr*Ra*(Theta[i
    +1,j]-Theta[i-1,j])/(2*p)
    Omega[1:Nx,1:Ny]=Omega[1:Nx,1:Ny]+dt*k[1:Nx,1:Ny]
    for i in range(1,Nx):    #calcul des composantes de la ditesse
        for j in range(1,Ny):
            Ux[i,j]=(Psi[i,j+1]-Psi[i,j-1])/(2*p)
            Uy[i,j]=-(Psi[i+1,j]-Psi[i-1,j])/(2*p)

#calcul du coefficient d'echange conducto-convectif local h
for j in range(1,Ny):
    hy[j]=abs(Theta[3,j]-Theta[1,j]/(2*p))
#calcul du nombre de Nusselt
for j in range(1,Ny):
    Nuy[j]=hy[j]*j
#Affichage de Nu
print(Nuy)
```

```
#La valeur moyene de Nu
SNu=0
for j in range(2,Ny-1):
    SNu=SNu+Nuy[j]
NuL=SNu/(len(Nuy)-2)
print('La valeur moyenne du nombre de Nusselt : ',NuL)

#Affichage de champ de temp rature adimentionn
plt.imshow(Theta[1:Nx,1:Ny])
plt.colorbar()
plt.xlabel('x')
plt.ylabel('y')
plt.show()

#Affichage de norme de vitesse adimentionn
plt.imshow(((Uy[1:Nx,1:Ny])**2+(Ux[1:Nx,1:Ny])**2)**0.5,extent=[0,20,0,40])
plt.colorbar()
plt.xlabel('x')
plt.ylabel('y')
plt.show()
#la fin du programme
print('End')
```