Le schéma implicite appliqué à l'équation mouvement de la couche limite linéarisée avec la méthode des coefficients isolés.

$$B_i U_{i-1}^{j+1} + D_i U_i^{j+1} + A_i U_{i+1}^{j+1} = C_i \quad , i = 1 : Imax - 1, \ldots, j = 1 : Jmax - 1 \tag{1}$$

Avec

$$\begin{cases} U_1 = 0 \qquad U_{Imax} = Ue \\ B_i = -\frac{V_i^j}{2\Delta y} - \frac{\nu}{(\Delta y)^2} \\ D_i = \frac{U_i^j}{\Delta x} + \frac{2\nu}{(\Delta y)^2} \\ A_i = \frac{V_i^j}{2\Delta y} - \frac{\nu}{(\Delta y)^2} \\ C_i = \frac{(U_i^j)^2}{\Delta x} + U_e^j \frac{(U_e^{j+1} - U_e^j)}{\Delta x} \end{cases}$$

et l'équation de Continuité :

$$V_i^{j+1} = V_{i-1}^{j+1} - e_i(U_{i-1}^{j+1} - U_i^{j+1}) + d_j, \quad i = 1 : Imax, \ldots, j = 1 : Jmax \tag{2}$$

$$\begin{cases} V_1^{j+1} = 0 \qquad V_{Imax}^{j+1} = 0 \\ d_j = \frac{\Delta y}{2\Delta x}(U_{i-1}^j + U_i^j) \\ e_i = \frac{\Delta y}{\Delta x} \end{cases}$$

Remarque 1. *On calcule V_i^{j+1} à partir de U_i^{j+1}.*

Je dois appliquer la méthode TDMA(Thomas Direct Method Algorithm) à ce système à deux composantes(U,V) en dimension deux.
Après le processus habituel, si on pose :

$$U_i^{j+1} = P_i U_{i+1}^{j+1} + Q_i \qquad , i = 1, \ldots, Imax - 1 \tag{3}$$

le système (1) devient :

$$\begin{cases} U_i^{j+1} = P_i U_{i+1}^{j+1} + Q_i \quad i = 2 : Imax - 1, \ldots, j = 2 : Jmax - 1 \\ V_i^{j+1} = V_{i-1}^{j+1} - e_i(U_{i-1}^{j+1} - U_i^{j+1}) + d_j \qquad i = 1 : Imax, \ldots, j = 1 : Jmax \end{cases}$$

Avec

$$\begin{cases} P_i = -A_i \left(D_i + B_i P_{i-1}\right)^{-1} \\ Q_i = \left(C_i - B_i Q_{i-1}\right)\left(D_i + B_i P_{i-1}\right)^{-1} \\ B_i = -\frac{V_i^j}{2\Delta y} - \frac{\nu}{(\Delta y)^2} \\ D_i = \frac{U_i^j}{\Delta x} + \frac{2\nu}{(\Delta y)^2} \\ A_i = \frac{V_i^j}{2\Delta y} - \frac{\nu}{(\Delta y)^2} \\ C_i = \frac{(U_i^j)^2}{\Delta x} + U_e^j \frac{(U_e^{j+1} - U_e^j)}{\Delta x} \\ d_j = \frac{\Delta y}{2\Delta x}(U_{i-1}^j + U_i^j) \\ e_i = \frac{\Delta y}{\Delta x} \\ P_1 = Q_1 = 0 \qquad P_{Imax} = 0 \quad Q_{Imax} = U_e \quad U_e < 0.99) \\ V_1^{j+1} = 0, \quad V_{Imax}^{j+1} = 0 \end{cases}$$

Pour l'instant,j'ai choisi $x = 0.5, 1, 1.5, \ldots,$ et $dx = x_{i+1} - xi/2$, $Ue = sin(x)$ $dy = 0.02$ $U_1 = 0$ et $U_N = Ue$, $nu = 1e-3$

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<string.h>
#define Jmax 11
#define Imax 12
#define VELOCITE "VELOCITE"
int main()
{
int i,it,l,k,j;
//****les constantes
double nu=1e-3;
double dy=0.02;
//le fichier de sortie(pour gnuplot)
FILE *f;
//les tableaux
double a[Imax],b[Imax],c[Imax],d[Imax];
double Ue[Imax];
double x[Imax];
double dx[Imax];
```

```
//***************
double P[Imax];
double Q[Imax];
// on crée les tableaux à deux dimensions U et V
double U[Imax][Jmax];
double V[Imax][Jmax];
//*****************************************************
for(i=1;i<Imax;i++){
x[i]=i+0.5;
}
for(i=1;i<=Imax;i++){
dx[i]=x[i+1]-x[i]/2;
}
//
for(i=1;i<Imax;i++)
{
Ue[i]=sin(x[i]);
}
//
//Conditions aux limites*****************************************************************
//bord gauche
for(i=1;i<=Imax;i++)
{
U[i][1]=0.000;
V[i][1]=0.000;
}
//bord droite
for(i=1;i<=Imax;i++)
{
U[i][Jmax]=Ue[i];
V[i][Jmax]=0.0;
}
//bord extérieur(en haut du maillage)
for(j=1;j<Jmax;j++)
{
U[1][j]=0.000;
V[1][j]=0.000;
}
//*****************************************************************************************
//*************************Application : Méthode TDMA*************************************
//*****************************************************************************************
//*******sur les lignes Nord-Sud ***********************************************************
//*****************************************************************************************
```

```
P[1]=0.0;
for(i=2;i<Imax;i++)
{
Q[1]=U[i][1];    //tous nuls(Conditions aux limites gauche)
for(j=2;j<Jmax;j++)
{
//calcul des coef. ligne par ligne
a[j]=(V[i][j]/2*dy)-(nu/(dy*dy));
b[j]=-(V[i][j]/2*dy)-(nu/(dy*dy));
d[j]=(U[i][j]/dx[i])+(2*nu/(dy*dy));
c[j]=(U[i][j]*U[i][j])/dx[i]+(Ue[i]*(Ue[i+1]-Ue[i])/dx[i]);
//*************************************************************
//on Calcule P(.,j) et Q(.,j) pour chaque ligne de j=2:Jmax
//*************************************************************
P[j]=-a[j]/(d[j]+b[j]*P[j-1]);
Q[j]=(c[j]-b[j]*Q[j-1])/(d[j]+b[j]*P[j-1]);
//*************************************************************
//puis on calcule  U(.,j) pour chaque ligne de j=Jmax-1 à 1
//*************************************************************
do{
U[i][j]=P[j]*U[i][j-1]+ Q[j];
j=j-1;
}while(j!=1);
}
//*****************************************************************
//une fois U(.,j) obtenu, on calcule V(.j) j=1:Jmax-1
//*****************************************************************
for(j=1;j<Jmax;j++)
{ V[i][j]=V[i][j-1]-(dy/dx[i])*(U[i][j-1]-U[i][j])+(dy/2.*dx[i])*(U[i][j-1]-U[i][j]);}
}//**********************fin Nord-Sud*************************************************
//*****************************************************************************************
//*********************sur les colonnes Est-Ouest*****************************************
P[1]=0.0;
for(j=2;i<Jmax;j++)
{
Q[1]=U[1][j];    //tous nuls(Conditions aux limites gauche)
for(i=2;i<Imax;i++)
{
//calcul des coef. colonne par colonne
a[i]=(V[i][j]/2*dy)-(nu/(dy*dy));
b[i]=-(V[i][j]/2*dy)-(nu/(dy*dy));
d[i]=(U[i][j]/dx[j])+(2*nu/(dy*dy));
c[i]=(U[i][j]*U[i][j])/dx[j]+(Ue[j]*(Ue[j+1]-Ue[j])/dx[j]);
```

```
//*********************************************************
//on Calcule P(i,.) et Q(i,.) pour chaque colonne de i=2:Imax-1
//*********************************************************
P[i]=-a[i]/(d[i]+b[i]*P[i-1]);
Q[i]=(c[i]-b[i]*Q[i-1])/(d[i]+b[i]*P[i-1]);
//*********************************************************
//puis on calcule  U(i,.) pour chaque colonne de i=Imax-1:2
//*********************************************************
do{
U[i][j]=P[i]*U[i-1][j]+ Q[i];
printf("U[%d][%d]\n",i,j,U[i][j]);
i=i-1;
}while(i!=2);
}
//*************************************************************
//une fois U(i,.) obtenu, on calcule V(i,.) i=1:Imax-1
//*************************************************************
for(i=1;i<Imax;i++)
{ V[i][j]=V[i-1][j]-(dy/dx[j])*(U[i-1][j]-U[i][j])+(dy/2.*dx[j])*(U[i-1][j]-U[i][j]);
printf("V[%d][%d]\n",i,j,V[i][j]);
}
}//**********************fin Est-Ouest*****************************************
//*****************************************************************************
//***************************FIN TDMA-2D(U,V)**********************************
//*****************************************************************************
//********************écriture des données pour gnuplot************************
//***********************je veux simuler le profil de vitesse(i.e U/Ue en fonction x/c=0.5
f=fopen("meca-result.dat","w");
for(i=1;i<=Imax;i++)
{
for(j=1;j<=Jmax;j++)
{
fprintf(f,"%lf %lf \n",U[i][j]/Ue[i],x[i]);
}
}
fclose(f);
return 1;
}
```