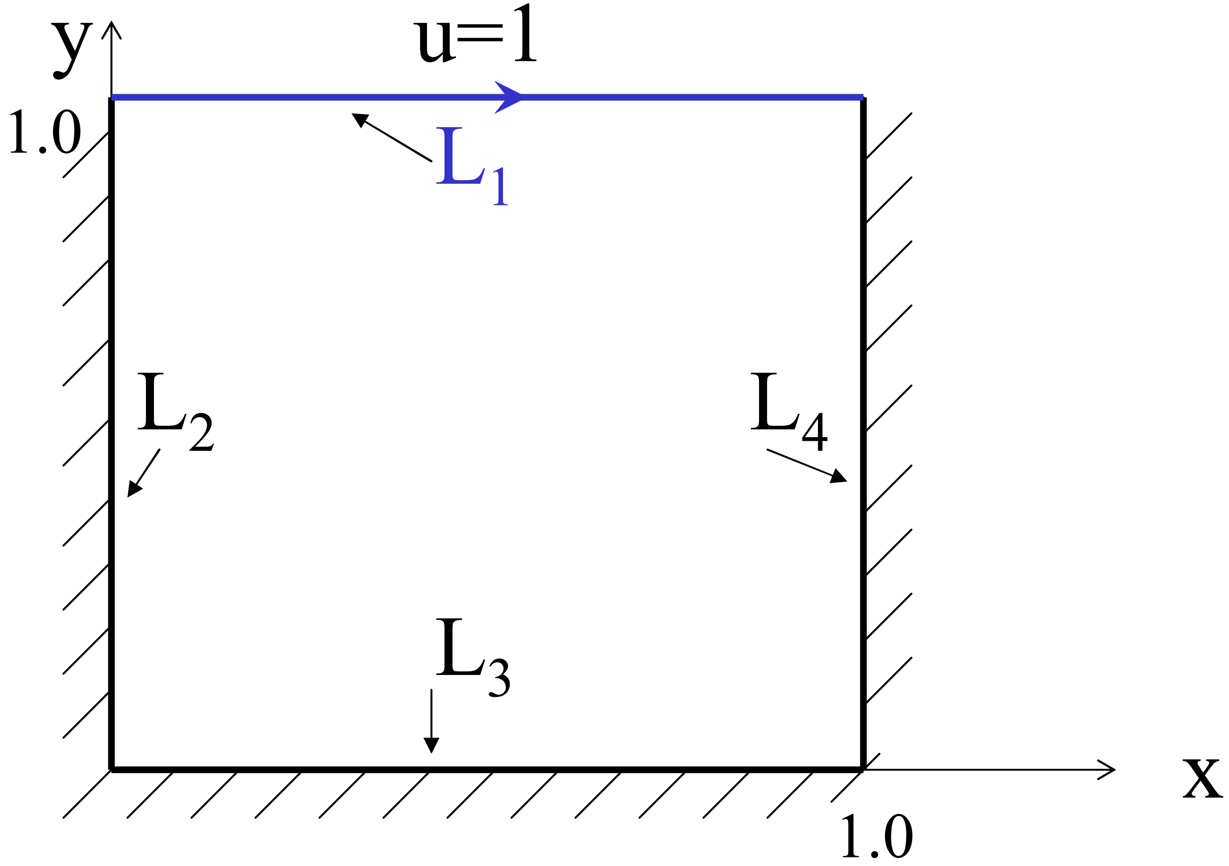
# FDM流れ解析　No.5
# 速度-圧力法でキャビティ流れ

2007年6月

後　保範 (東京工芸大学)

# 目次

# 1. 解析対象



# 2. 基礎方程式

- ベクトル表示の基礎方程式

$$div(v) = 0$$

$$\frac{\partial v}{\partial t} + (v \cdot \nabla)v = -\nabla p + \frac{1}{\mathrm{Re}} \Delta v$$

# 2. 1 2次元の具体的基礎方程式

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0$$

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}=-\frac{\partial p}{\partial x}+\frac{1}{\mathrm{Re}}\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)$$

$$\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}=-\frac{\partial p}{\partial y}+\frac{1}{\mathrm{Re}}\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}\right)$$

# 3. 速度-圧力方程式

- 圧力(p)に関する式の導出

$$\frac{\partial v}{\partial t}+(v\cdot\nabla)v=-\nabla p+\frac{1}{\mathrm{Re}}\Delta v$$

の発散$(rot)$をとり$, D=div(v)$を代入する。

$$-\Delta p=div((v\cdot\nabla)v)+\frac{D^{(n+1)}-D^{(n)}}{\Delta \mathrm{t}}-\frac{1}{\mathrm{Re}}\Delta D^{(n)}$$

本来は$D=0$であり、$D^{(n+1)}$と右辺第3項は無視できる。

$$-\Delta p=div((v\cdot\nabla)v)-\frac{D}{\Delta t}$$

# 3.1 速度-圧力による方程式

$$-\Delta p = div((v \cdot \nabla)v) - \frac{div(v)}{\Delta t}$$

$$\frac{\partial v}{\partial t} + (v \cdot \nabla)v = -\nabla p + \frac{1}{\mathrm{Re}}\Delta v$$

# 3.2 2次元速度-圧力の具体的方程式

$$-\frac{\partial^2 p}{\partial x^2}-\frac{\partial^2 p}{\partial y^2}=\left(\frac{\partial u}{\partial x}\right)^2+\left(\frac{\partial v}{\partial y}\right)^2+2\frac{\partial v}{\partial x}\frac{\partial u}{\partial y}-\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)/\Delta t$$

$$\frac{\partial u}{\partial t}+\frac{\partial u^2}{\partial x}+\frac{\partial (uv)}{\partial y}=-\frac{\partial p}{\partial x}+\frac{1}{\mathrm{Re}}\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)$$

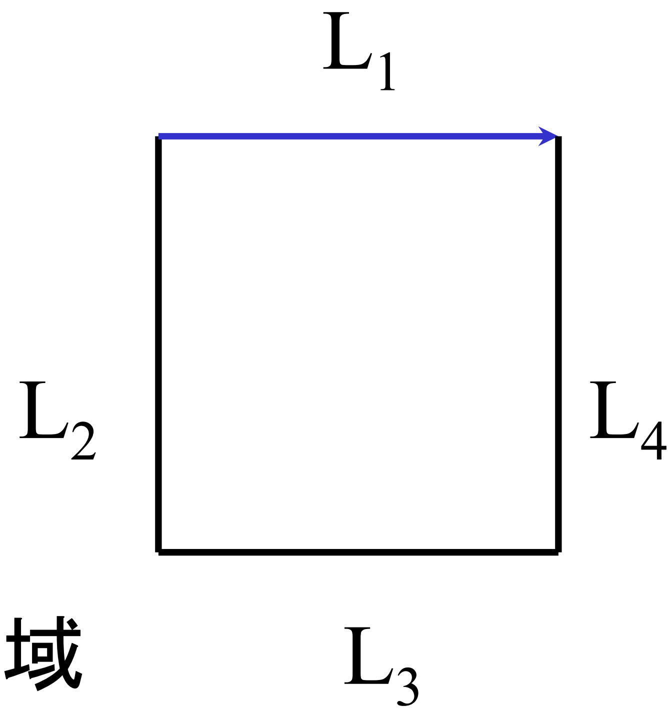
$$\frac{\partial v}{\partial t}+\frac{\partial (uv)}{\partial x}+\frac{\partial v^2}{\partial y}=-\frac{\partial p}{\partial y}+\frac{1}{\mathrm{Re}}\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}\right)$$

# 4. 境界条件と初期条件

- u,vの境界条件

$u=1,\ v=0$ on $L_1$

$u=v=0$ on $L_2, L_3, L_4$




- 初期条件 ; $u=v=0$ in 領域

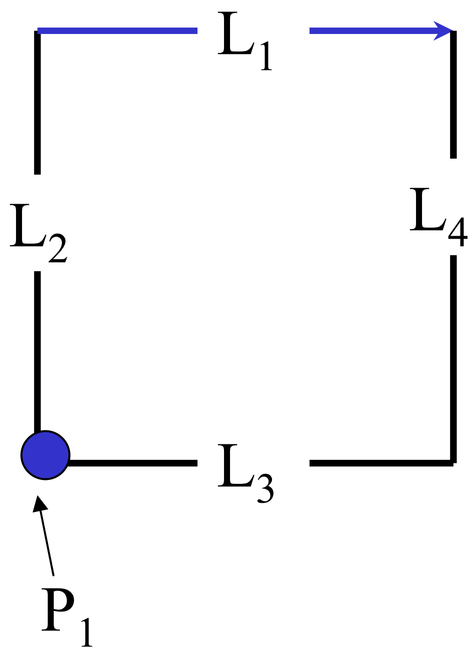
# 4. 1　圧力の境界条件と初期条件

- pの境界条件

$$p = 0.0 \quad on \quad P_1$$

$$\frac{\partial p}{\partial y} = 0.0 \quad on \quad L_1, L_3$$

$$\frac{\partial p}{\partial x} = 0.0 \quad on \quad L_2, L_4$$




- pの初期条件

p=0.0　in 内部領域

# 4.2 pの計算用外側境界条件

$$u_{i,ny+1} = 1,\ v_{i,ny+1} = 0,$$

$$p_{i,ny+1} = p_{i,ny-1} \quad on \quad L_1$$

$$u_{-1,j} = u_{1,j},\ v_{-1,j} = -v_{1,j},$$

$$p_{-1,j} = p_{1,j} \quad on \quad L_2$$

$$u_{i,-1} = -u_{i,1},\ v_{i,-1} = v_{i,-1},$$

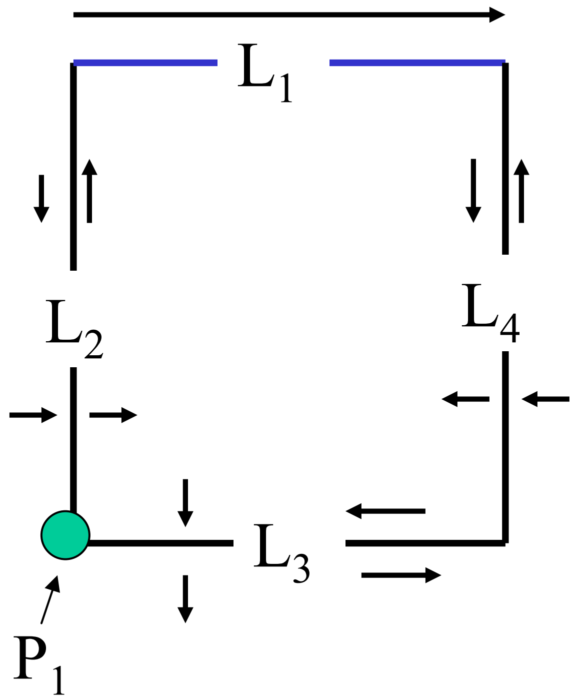
$$p_{i,-1} = p_{i,1} \quad on \quad L_3$$

$$u_{ny+1,j} = u_{ny-1,j},$$

$$v_{ny+1,j} = -v_{ny-1,j},$$

$$p_{ny+1,j} = p_{ny-1,j} \quad on \quad L_4$$

nx×ny分割



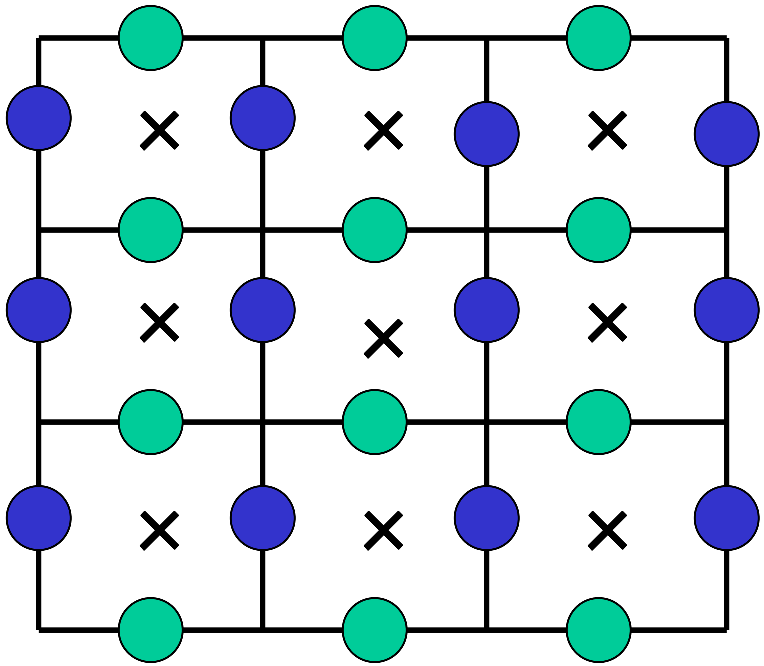
# 5. 差分法による離散化方法

- スタガード格子

● ; u格子

● ; v格子

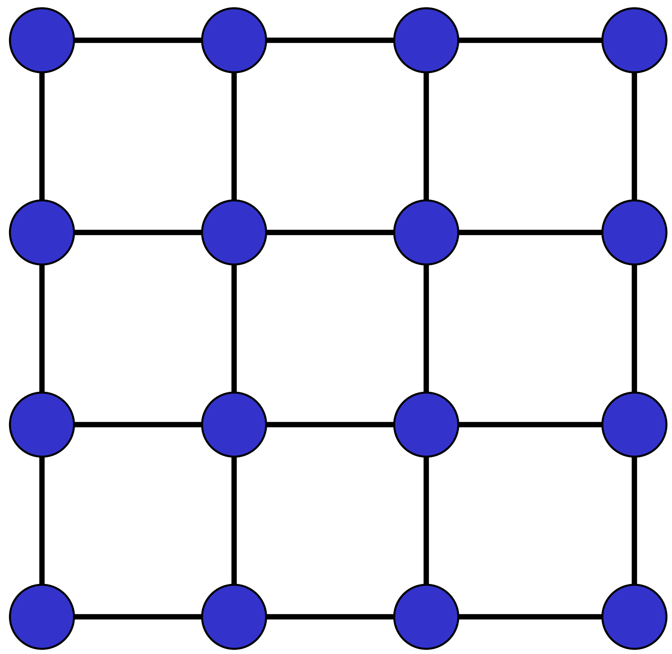
× ; p格子



# 5.1 通常格子による離散化

- 通常格子

; u,v,p格子



# 5.2　pの離散化(通常格子)

$$\frac{2p_{i,j}-p_{i-1,j}-p_{i+1,j}}{(\Delta x)^2}+\frac{2p_{i,j}-p_{i,j-1}-p_{i,j+1}}{(\Delta y)^2}=$$

$$\frac{(u_{i+1,j}-u_{i-1,j})^2}{4(\Delta x)^2}+\frac{(v_{i,j+1}-v_{i,j-1})^2}{4(\Delta y)^2}+\frac{(v_{i+1,j}-v_{i-1,j})(u_{i,j+1}-u_{i,j-1})}{2\Delta x\Delta y}$$

$$-\left[\frac{u_{i+1,j}-u_{i-1,j}}{2\Delta x}+\frac{v_{i,j+1}-v_{i,j-1}}{2\Delta y}\right]/\Delta t$$

# 5.3　uの離散化(通常格子)

$$\frac{u_{i,j}^{(k+1)}-u_{i,j}}{\Delta t}=-\frac{u_{i+1,j}^2-u_{i-1,j}^2}{2\Delta x}-\frac{u_{i,j+1}v_{i,j+1}-u_{i,j-1}v_{i,j-1}}{2\Delta y}$$

$$-\frac{p_{i+1,j}-p_{i-1,j}}{2\Delta x}-$$

$$\left[\frac{2u_{i,j}-u_{i-1,j}-u_{i+1,j}}{(\Delta x)^2}+\frac{2u_{i,j}-u_{i,j-1}-u_{i,j+1}}{(\Delta y)^2}\right]/\mathrm{Re}$$

# 5.4　vの離散化(通常格子)

$$\frac{v_{i,j}^{(k+1)}-v_{i,j}}{\Delta t}=-\frac{u_{i+1,j}v_{i+1,j}-u_{i-1,j}v_{i-1,j}}{2\Delta x}-\frac{v_{i,j+1}^{2}-v_{i,j-1}^{2}}{2\Delta y}$$

$$-\frac{p_{i,j+1}-p_{i,j-1}}{2\Delta y}-$$

$$\left[\frac{2v_{i,j}-v_{i-1,j}-v_{i+1,j}}{(\Delta x)^{2}}+\frac{2v_{i,j}-v_{i,j-1}-v_{i,j+1}}{(\Delta y)^{2}}\right]/\,\mathrm{Re}$$

# 6. 計算手順

- u,v,pの計算

k=0,1,2 …と指定回数まで計算

$t = t + \Delta t$

$u^{(k+1)}$,$v^{(k+1)}$の計算($p^{(k)}$,$u^{(k)}$,$v^{(k)}$を使用)

$p^{(k+1)}$を反復法で計算($u^{(k+1)}$,$v^{(k+1)}$を使用)

# 6.1 プログラム全体計算手順

NX,Re数, Δt, NT(反復回数),MT(出力間隔)入力

u,v,pの初期条件設定

k=1,2,・・・,NTまで反復計算

t＝t＋ Δt

u,vの計算 (7.1を使用)

if(発散(Over変数で判定)) --> 発散として停止

SOR法でpの計算 (7.2～7.4を使用)

if(mod(k,MT)==0) --> u,v,pの出力

# 7. プログラムの主要部分

- u,v,pの初期条件(u,vの境界条件を含む)

```
for (i=0; i<=NX; i++) {
  for (j=0; j<=NY; j++) {
    P[i][j] = 0.0 ;  U[i][j] = 0.0 ;  V[i][j] = 0.0 ;  }   }
for (i=0; i<=NX; i++) {
  U[i][NY] = 1.0 ;  }
```

- pの境界条件

  pのSOR法の計算時に考慮

# 7.1 u,vの計算(通常格子)

- h=Δx=Δy, DT=Δt,D5=1/2h,D2=1/h²とする。

```
for (i=1; i<NX; i++) {
  for (j=1; j<NY; j++) {
    U[i][j] = U[i][j] + DT*((U[i-1][j]*U[i-1][j]-U[i+1][j]*U[i+1][j]
            + U[i][j-1]*V[i][j-1] - U[i][j+1]*V[i][j+1]
            + P[i-1][j] - P[i+1][j])*D5 - (4.0*U[i][j] - U[i][j-1]
            - U[i-1][j] - U[i+1][j] - U[i][j+1])*D2/Re ) ;
    V[i][j] = V[i][j] + DT*((V[i][j-1]*V[i][j-1]-V[i][j+1]*V[i][j+1]
            + U[i-1][j]*V[i-1][j] - U[i+1][j]*V[i+1][j]
            + P[i][j-1] - P[i][j+1])*D5 - (4.0*V[i][j] - V[i][j-1]
            - V[i-1][j] - V[i+1][j] - V[i][j+1])*D2/Re ) ;    }    }
```

# 7.2 pのSOR法による計算

- 計算概要

-Δp=f(u,v)の右辺ベクトルbを作成

k=1,2,...,NX*NYが収束まで以下を反復

-Δp=bのpをSOR法で計算

$err = \| r \|_2 / \| b \|_2$

if(err <= EPS) 収束

# 7.3 pの計算(右辺ベクトル作成)

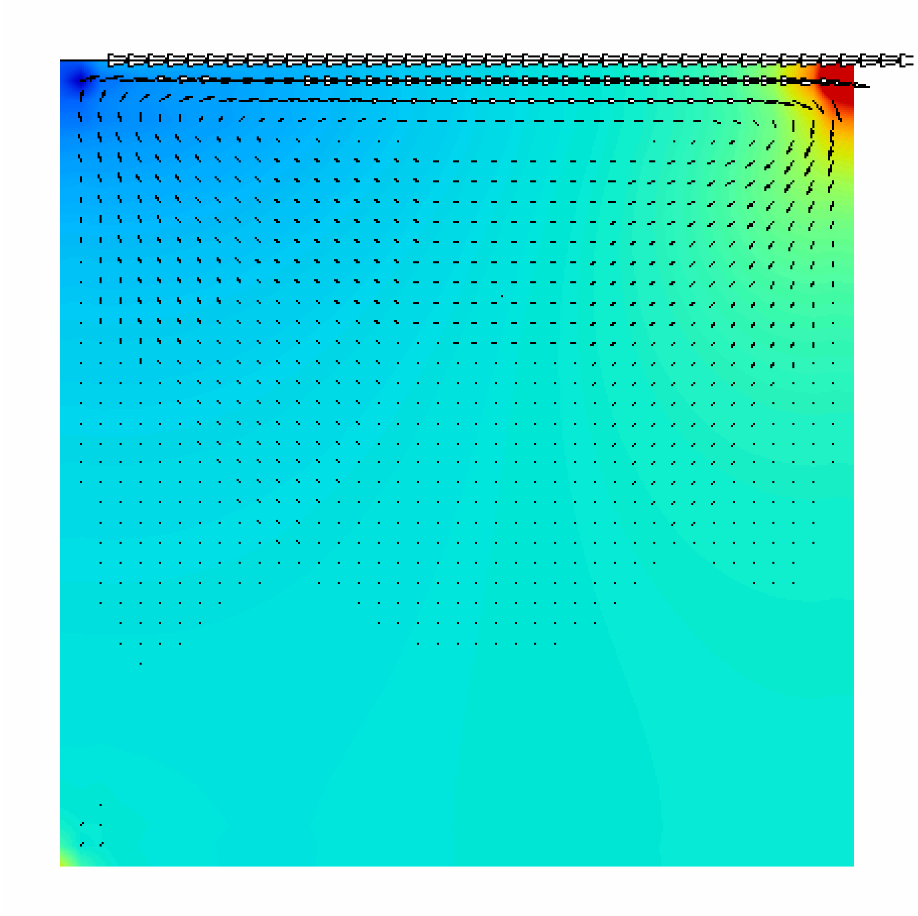
```
for (i=0; i<=NX; i++) {
   for (j=0; j<=NY; j++) {
//   Boundary on L1
       if(j != NY) { U0P=U[i][j+1] ;  V0P=V[i][j+1] ;  }
         else { U0P=1.0 ;  V0P=0.0 ;  }
//   Boundary on L2
       if(i != 0)  { UM0=U[i-1][j] ;  VM0=V[i-1][j] ;  }
         else { UM0=U[1][j] ;  VM0=-V[1][j] ;  }
//   Boundary on L3,L4
         . . . . . . . .
//   Computation
        B[i][j] = ((UP0-UM0)*(UP0-UM0) + (V0P-V0M)*(V0P-V0M))/4.0
                   + (VP0 - VM0)*(U0P - U0M)/2.0
                   - (UP0 - UM0 + V0P - V0M)*H/(2.0*DT) ;     }    }
```

# 7.4 pのSOR法による計算本体

```
RN = 0.0 ;
for (i=0; i<=NX; i++) {
   IP = i + 1 ;   IM = i - 1 ;          // Boundary  on L2, L4
   if(i == NX) { IP = NX - 1 ; }    if(i == 0)  { IM = 1 ;  }
   for (j=0; j<=NY; j++) {
      JP = j + 1 ;  JM = j - 1 ;          // Boundary  on L1,L3
      if(j == NY) { JP = NY - 1 ; }    if(j == 0)  { JM = 1 ; }
      R = (B[i][j]+P[i][JM]+P[IM][j]+P[IP][j]+P[i][JP])/4.0-P[i][j] ;
      P[i][j] = P[i][j] + ALP*R ;
      if(i==0 & j==0) { P[i][j]=0.0 ;  R=0.0 ;  }      // Boundary on P1
      RN = RN + R*R*16.0 ;
 }  }
 ERR = sqrt(RN/BN) ;
```

誤差でこの値が小さいと収束

# 8. 計算結果 (Re=400, t=0.5)

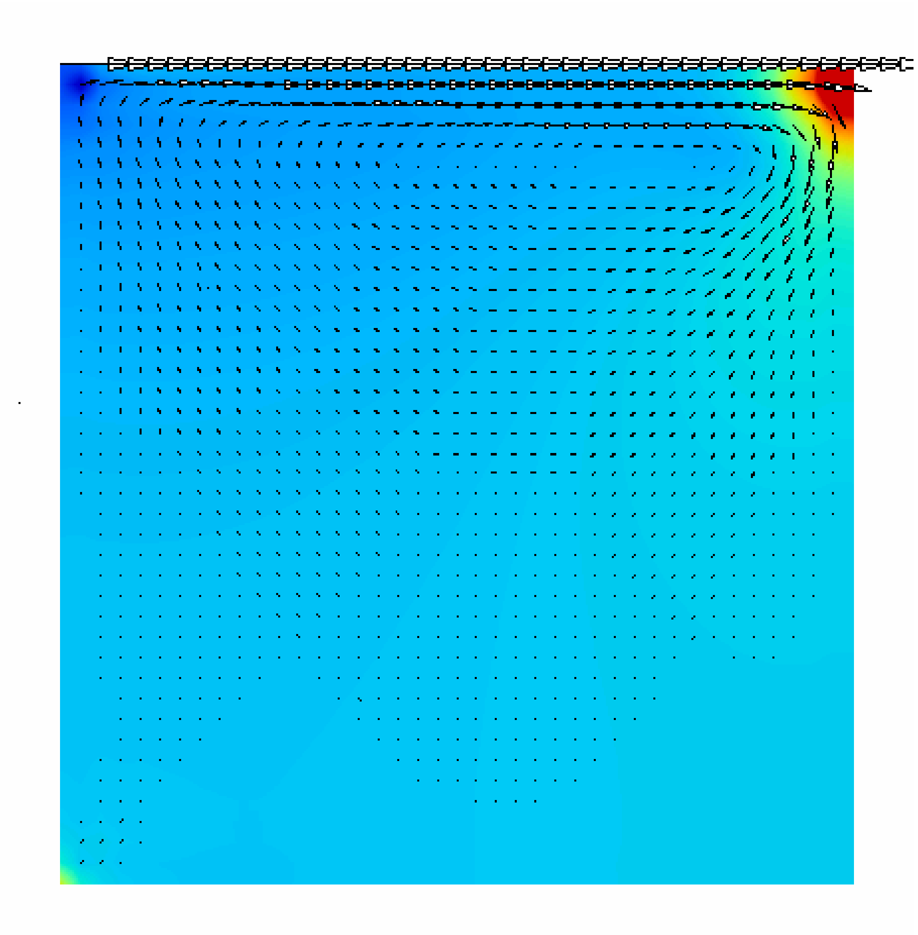


40×40分割

Δt = 0.01

流速ベクトル
と圧力分布

# 8.1 計算結果 (Re=400, t=1.0)

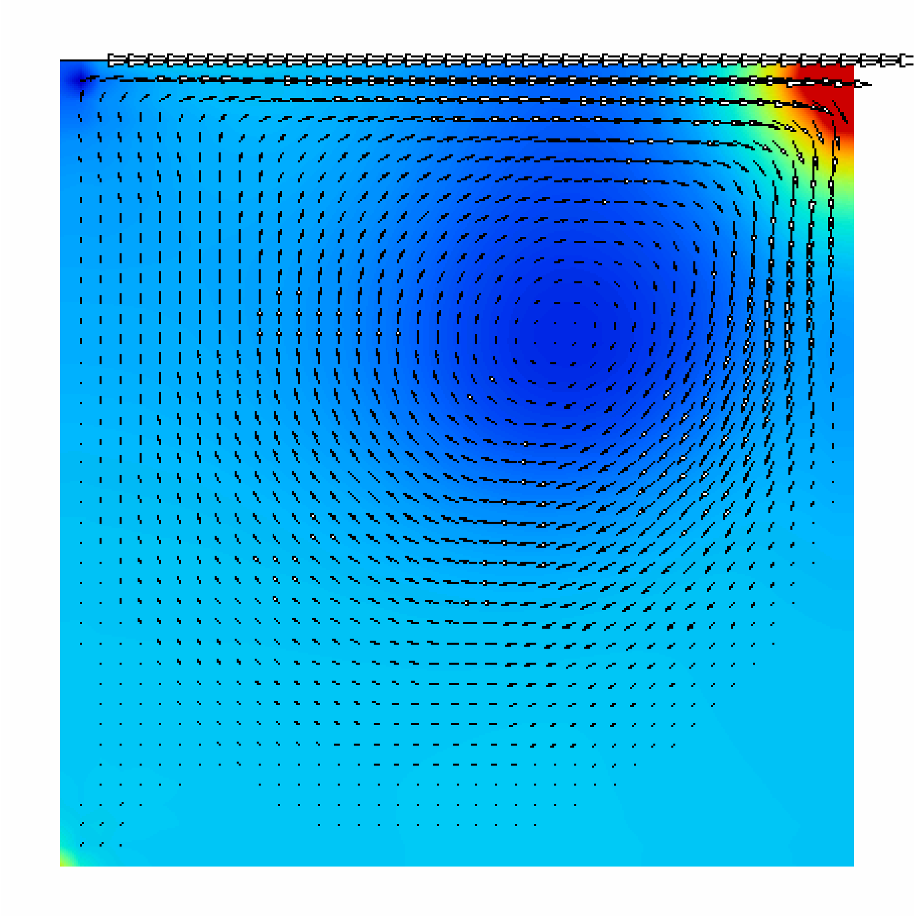


40×40分割

Δt = 0.01

流速ベクトル
と圧力分布

# 8.2 計算結果 (Re=400, t=10.0)



40×40分割

Δt = 0.01

流速ベクトルと圧力分布