Correction 3 : Initiation aux éléments finis 2D

Ex 1 *Ecoulements de Poiseuille*

1)

La formulation faible

$$\int_0^1 \int_0^1 \left(\frac{\partial u}{\partial x}\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y}\frac{\partial v}{\partial y} \right) \, dx \, dy = \int_0^1 \int_0^1 f v \, dx \, dy + \int_0^1 \left[\frac{\partial u}{\partial x} \, v \right]_{x=0}^{x=1} \, dy + \int_0^1 \left[\frac{\partial u}{\partial y} \, v \right]_{y=0}^{y=1} \, dx$$

Sur chaque élément, on obtient le système élémentaire `Nno x Nno` suivant

$$A^k U^k = F^k \tag{1}$$

où :

$$a_{ij}^k = \int\int_{\mathbf{K}} \left[\nabla \psi_j^k\right]^T \left[\nabla \psi_i^k\right] \, dx \, dy \quad \text{avec} \quad [\nabla \psi_j^k] = \begin{pmatrix} \frac{\partial \psi_j^k}{\partial x} \\ \\ \frac{\partial \psi_j^k}{\partial x} \end{pmatrix}$$

$$f_i^k = \int\int_{\mathbf{K}} \psi_i^k \, dx \, dy$$

2)

La matrice Jacobienne

$$DT_k = \begin{pmatrix} \sum_{j=1}^4 x_j^k \frac{\partial L_j}{\partial \xi} & \sum_{j=1}^4 x_j^k \frac{\partial L_j}{\partial \eta} \\ \\ \sum_{j=1}^4 y_j^k \frac{\partial L_j}{\partial \xi} & \sum_{j=1}^4 y_j^k \frac{\partial L_j}{\partial \eta} \end{pmatrix}$$

Ce qui donne le programme

```
x  = COOR(CONNEC(iel,:),1); y = COOR(CONNEC(iel,:),2);

[F,DxiFP,DetaFP] = Q1(xg,yg);  // on evalue les fonctions de Lagrange au point (xg,yg)

 j11 = x(1:4)'*DxiFP;     // = sum_j  x_j^k dL_j/dxi
 j12 = x(1:4)'*DetaFP;
 j21 = y(1:4)'*DxiFP;
 j22 = y(1:4)'*DetaFP;
//
 DT  = [j11 j12; j21 j22];
```

3)

Listing du programme

```
clear; xdel(0); xdel(1);
//  on charge les programmes dont on a besoin
//
exec('Q1.sci'); exec('Q2.sci'); exec('gauss.sci');
exec('maillage_2D_Q2.sci');
exec('cal_indice.sci');  exec('interpol.sci');
exec('trace_maillage_2D.sci'); exec('trace_u_2D.sci');

//
//  nbre de points pour l'integration numerique
Ng = 3;
[XG,PG] = gauss(Ng);
//
nx = 10;  ny = 10;  // nbre d'element dans les directions x et y
lx = 1.; ly = 1.; // taille du carre

[CONNEC,COOR,Nelt,Nno,Npt] =  maillage_2D_Q2(1,1,nx,ny);
trace_maillage_2D(CONNEC,COOR,0);

//
//  creation de la matrice  NUMER et ADRESS
//
//  Conditions aux limites
//  u = 0 sur tout le bord
NUMER=zeros(Npt,1);
NI = 0;
nc = 0;
for i=1:Npt
   x = COOR(i,1); y = COOR(i,2);
   if x <> 0 & x <> 1 & y <> 0 & y <> 1 then
     NI = NI+1;
     NUMER(i) = NI;
   else
     NUMER(i) = Npt-nc;
     nc = nc + 1;
   end;
end;
//
//  creation de la matrice ADRESS
//
ADRESS = zeros(Nelt,Nno);
for i =1:Nelt
  for j = 1:Nno
    ADRESS(i,j) = NUMER(CONNEC(i,j));
  end;
end;
//
//  creation du vecteur UC
```

```
//
UC = zeros(Npt-NI,1);

//
//    Fin initialisation des tableaux

A_glob = spzeros(Npt,Npt); F_glob = zeros(Npt,1);

for iel = 1:Nelt  // boucle sur les elements
//
// ------------------------------------------------------------------------
//  PARTIE   A ECRIRE
//--------------------------------------------------------------------
// calcul du syteme elementaire sur chaque element
//
   A_el = zeros(Nno,Nno);
   F_el = zeros(Nno,1);

   x  = COOR(CONNEC(iel,:),1); y = COOR(CONNEC(iel,:),2);


   for ig=1:Ng    // double boucle pour l'integration numerique
    xg = XG(ig);
    for jg=1:Ng
       yg = XG(jg);
//
// calcul de la matrice jacobienne
//
          [F,DxiFP,DetaFP] = Q1(xg,yg);
          j11 = x(1:4)'*DxiFP;
          j12 = x(1:4)'*DetaFP;
          j21 = y(1:4)'*DxiFP;
          j22 = y(1:4)'*DetaFP;
//
          DT  = [j11 j12; j21 j22];
          jac = det(DT);
          B = inv(DT)';
//
       [FP,DxiFP,DetaFP] = Q2(xg,yg);

       for i=1:Nno
          DFPi = B*[DxiFP(i); DetaFP(i)];
          F_el(i) = F_el(i) + PG(ig)*PG(jg)*FP(i)*jac;
          for j=1:Nno
            DFPj = B*[DxiFP(j); DetaFP(j)];
            A_el(i,j) = A_el(i,j) + PG(ig)*PG(jg)*( DFPj' * DFPi) *jac;
          end;
       end;
    end;
    end;
//
// fin calcul systeme elementaire pour l'element iel
//
```

```
//-------------------------------------------------------------
//  FIN PARTIE A ECRIRE
//-------------------------------------------------------------
// on doit assembler

   for i=1:Nno
      i_glob = ADRESS(iel,i);
      F_glob(i_glob) = F_glob(i_glob) + F_el(i);
    end;

   A_el=sparse(A_el);
   [ij,v,mn] = spget(A_el);
   for i = 1:length(v)
     ij(i,1) = ADRESS(iel,ij(i,1));
     ij(i,2) = ADRESS(iel,ij(i,2));
   end;
   A_el = sparse(ij,v,[Npt Npt]);
   A_glob = A_glob + A_el;

end;  // fin boucle sur les elements


A = A_glob(1:NI,1:NI);  F = F_glob(1:NI);
M12 = A_glob(1:NI,NI+1:$);
F = F - M12*UC;

[h,rk] = lufact(A);
U = lusolve(h,F);
ludel(h);

u_cal = [U; UC];

[u,x,y]=trace_u_2D(u_cal,COOR,NUMER,1,20,20);
```

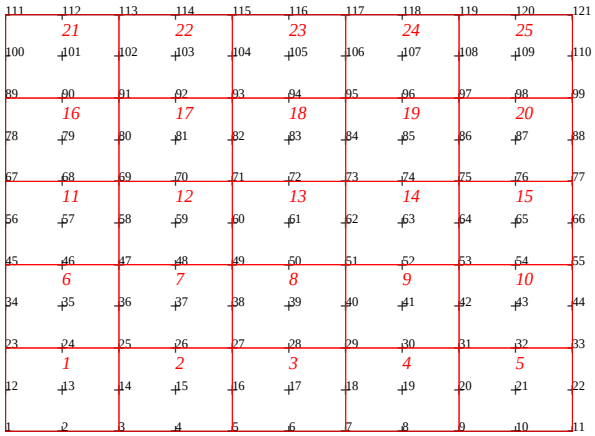

FIG. 1 – Exemple de maillage avec `Nelt = 25` et `Npt = 121`.

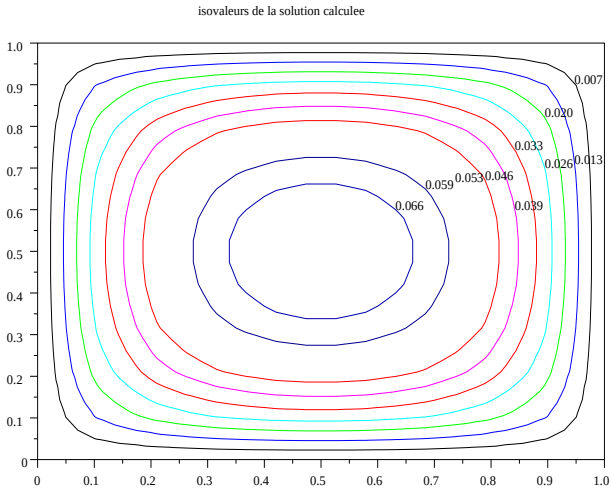

FIG. 2 – Solution du problème pour avec `Nelt = 100` et `Npt = 441`.

4)
La formule qui donne le débit est :

$$q = \int_0^1 \int_0^1 u(x,y)\ dx\ dy = \sum_{k=1}^{\texttt{Nelt}} \left(\sum_{j=1}^{\texttt{Nno}} u_j^k \int_{-1}^{1} L_j(\xi,\eta)\ Jac^k\ d\xi\ d\eta \right)$$

avec $u|_{\mathbf{K_k}} = \sum_{j=1}^{\texttt{Nno}} u_j^k \psi_j^k(x,y)$.

On trouve $q = .0351$ avec le programme suivant :

```
function q = cal_debit_2D(U,COOR, CONNEC, NUMER, ADRESS)
//
[Nelt, Nno] = size(CONNEC);
Npt = length(COOR);

q = 0.;
for iel = 1:Nelt

   x  = COOR(CONNEC(iel,:),1); y = COOR(CONNEC(iel,:),2);
   u_el = U(ADRESS(iel,:))

   for ig=1:Ng    // double boucle pour l'integration numerique
      xg = XG(ig);
      for jg=1:Ng
         yg = XG(jg);
//
// calcul de la matrice jacobienne
//
          [F,DxiFP,DetaFP] = Q1(xg,yg);
          j11 = x(1:4)'*DxiFP;
          j12 = x(1:4)'*DetaFP;
          j21 = y(1:4)'*DxiFP;
          j22 = y(1:4)'*DetaFP;
//
          DT  = [j11 j12; j21 j22];
          jac = det(DT);
//
          [FP,DxiFP,DetaFP] = Q2(xg,yg);
//
          for i=1:Nno
             q  = q + PG(ig)*PG(jg) * u_el(i)*FP(i) * jac;
          end;
      end;
   end;
end;   // fin boucle sur les elements

endfunction
```

5)
Il faut changer le calcul de la matrice `NUMER` à la ligne 28. Il faut "libérer" les degrés de liberté qui sont situés sur les bords $x = 0$ et 1. on remplace la condition

```
if x <> 0 & x <> 1 & y <> 0 & y <> 1 then
```

par

```
if y <> 0 & y <> 1 then
```

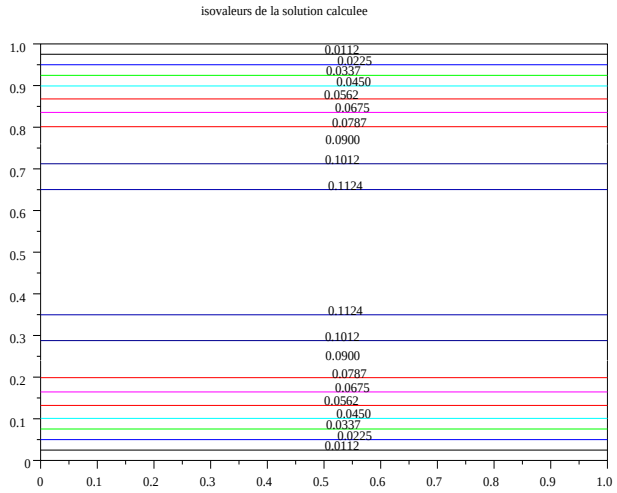

FIG. 3 – Solution de la question **5)** pour avec `Nelt = 100` et `Npt = 441`.

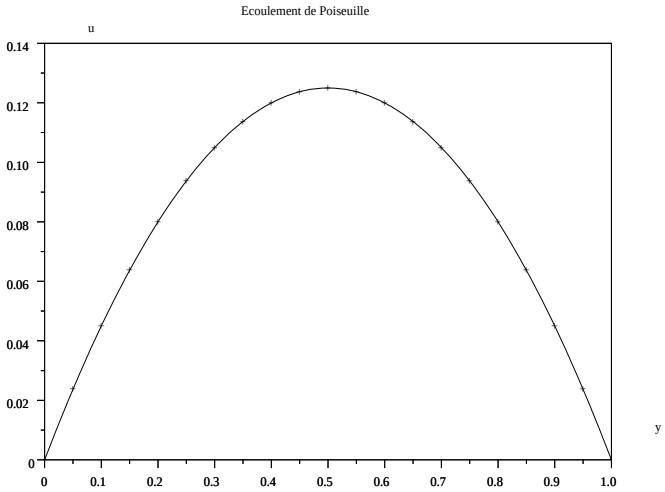

FIG. 4 – Comparaison entre la solution calculée (+) et la solution analytique $u_e = y(y-1)/2$ pour une valeur de x fixée.

Ex 2 *Ecoulement analytique*

1)

La formulation faible

$$\int_0^1 \int_0^1 \left(\frac{\partial u}{\partial x}\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y}\frac{\partial v}{\partial y} \right) \, dx \, dy + 3 \int_0^1 \int_0^1 u \, v \, dx \, dy = 0$$

2)

Le programme est essentiellement le même que pour l'exercice précédent. Il suffit d'imposer les conditions aux limites de Diriclet correctement

```
UC = zeros(Npt-NI,1);
for i=1:Npt
  x = COOR(i,1); y = COOR(i,2);
  if y == 0 | y == 1 | x == 0 | x == 1 then
     UC(NUMER(i)-NI) = exp(2*x) * sin(y);
  end;
end;
```

et de calculer correctement la matrice élémentaire

```
 A_el(i,j) = A_el(i,j) + ...
             PG(ig)*PG(jg)*( DFPj(1)*DFPi(1) + DFPj(2)*DFPi(2) + 3*FP(j)*FP(i) ) *jac;
```

Si on utilise des éléments $\mathbf{P_1}$, il faut utiliser la quadrature de Hammer et la fonction `hammer.sci` pour avoir les points d'intégration.

Quelques exemples de calcul sont rassemblés dans le tableau suivant :

	Npt	Nelt	Nno	$\|\|u_c - u_a\|\|/\|\|u_a\|\|$
$\mathbf{Q_1}$	81	64	4	$5.05\ 10^{-4}$
$\mathbf{Q_2}$	81	16	9	$1.40\ 10^{-5}$
$\mathbf{P_1}$	81	128	3	$5.15\ 10^{-4}$

TAB. 1 – Erreur relative $||u_c - u_a||/||u_a||$ en fonction des éléments choisis.

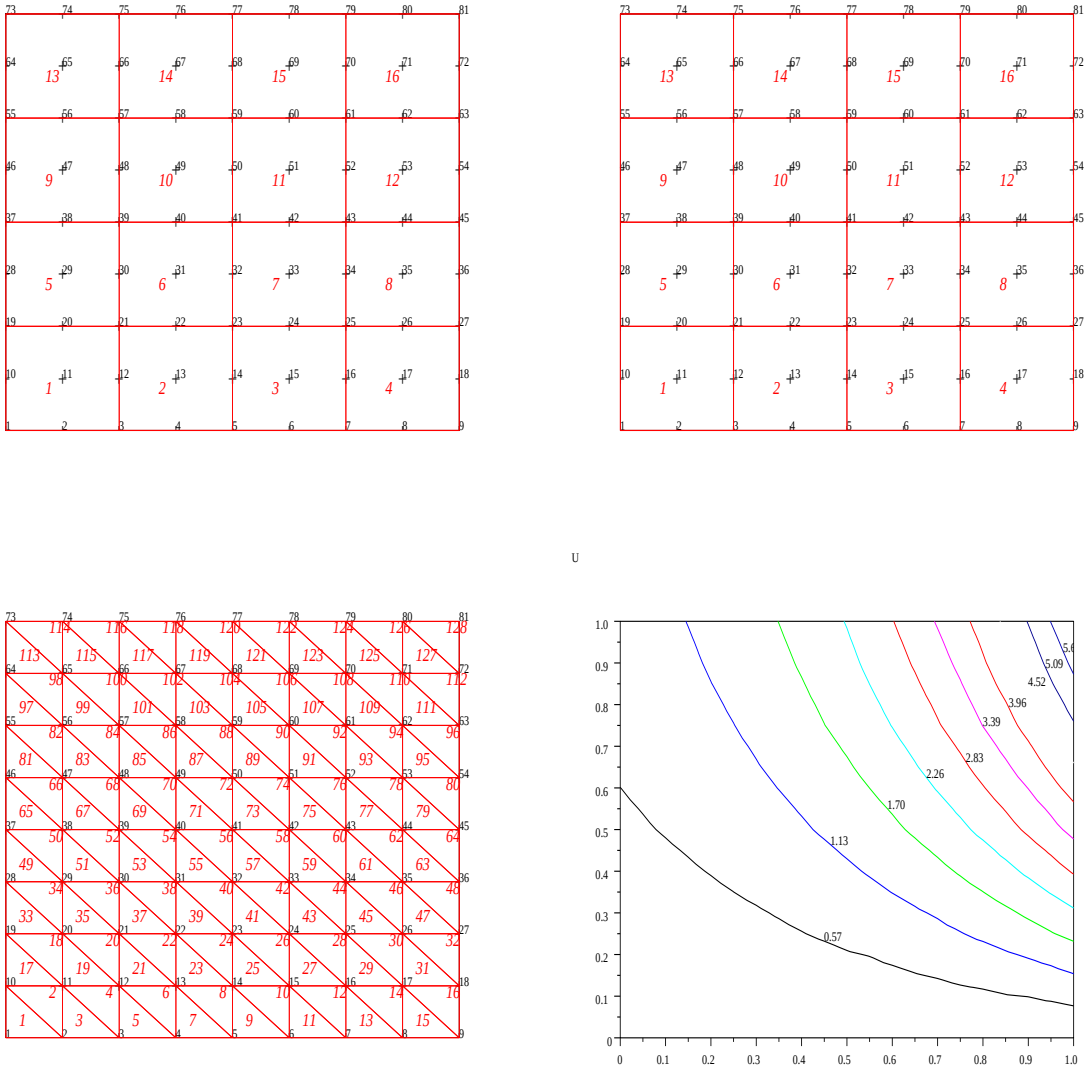

FIG. 5 – Les différents maillages et la solution : (a) $\mathbf{Q_1}$; (b) $\mathbf{Q_2}$; (c) $\mathbf{P_1}$; (d) $u(x, y)$.