DEA Physique et Génie des Matériaux

Mastère Matériaux et Mise en Forme

TD Elément Finis - Solutions

P. Laure (Patrice.Laure@inln.cnrs.fr)
Institut Non Linéaire de Nice

Correction 0 : un peu de mathématique.

Ex 1 *Calcul de la solution d'une edo*

1)
On cherche une solution polynômiale de la forme

$$u(x) = ax^2 + bx + c$$

qui vérifie l'équation et les conditions aux limites. Ce qui donne

$$\left\{ \begin{array}{lcl} -2a & = & 1 \\ c & = & 0 \\ 2a+b & = & 0 \end{array} \right. \tag{1}$$

et finalement

$$u(x) = x\left(-\frac{x}{2} + 1\right) \tag{2}$$

2) On discrétise sur $N = 5$ points ($x_1 = 0, x_2 = 1/4, x_3 = 1/2, x_4 = 2/3, x_5 = 1$). On a les relations suivantes avec $\Delta_x = 1/4$:

$$\begin{array}{rcl} u_1 & = & 0 \\ -u_1 + 2\ u_2 - u_3 & = & \Delta_x^2 \\ -u_2 + 2\ u_3 - u_4 & = & \Delta_x^2 \\ -u_3 + 2\ u_4 - u_5 & = & \Delta_x^2 \\ u_3 - 4\ u_4 + 3\ u_5 & = & 0 \end{array}$$

où on a approché $u'(1)$ par $\frac{3u_5 - 4u_4 + u_3}{2\Delta_x}$.
On doit finalement résoudre le système matriciel suivant :

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 1 & -4 & 3 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 1/16 \\ 1/16 \\ 1/16 \\ 0 \end{bmatrix} \tag{3}$$

ce qui donne

$$u_1 = 0\,;\ u_2 = 7/32\,;\ u_3 = 3/8\ ;\ u_4 = 15/32\,;\ u_5 = 1/2.$$

On peut vérifier que ces valeurs correspondent à la solution (2).
On a la valeur exacte car l'approximation des dérivées par des différences finies d'ordre 2 est exacte pour un polynôme d'ordre 2.

3) Les fonctions tests sont des polynômes de degré 1 qui valent 1 en un point x_i et zéro sur les autres (voir la figure 1). On peut les définir par,

$$\begin{array}{lll} \phi_i(x) = 0 & & x < x_{i-1} \\ \phi_i(x) = \frac{x - x_{i-1}}{x_i - x_{i-1}} & x_{i-1} < x < x_i & \\ \phi_i(x) = \frac{x_{i+1} - x}{x_{i+1} - x_i} & x_i < x < x_{i+1} & \\ \phi_i(x) = 0 & x_{i+1} < x & \end{array}$$

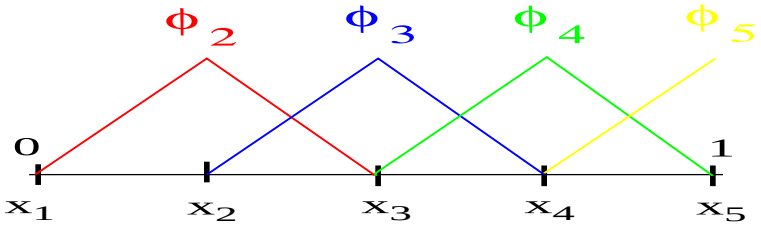

FIG. 1 – Schéma montrant les 5 fonctions tests.

Par exemple

$$\phi_2(x) = 4x \quad \text{pour} \quad x < 1/4 \quad \text{et} \quad \phi_2(x) = 2(1-2x) \quad \text{pour} \quad 1/4 < x < 1/2 \quad \text{et zéro ailleurs}$$

La solution approchée est de la forme $u = \sum_{j=2}^{5} u_j \phi_j$ puisque les conditions aux limites impose $u_1 = 0$. La formulation faible s'écrit

$$\int_0^1 u\ v\ dx = \int_0^1 v\ dx$$

où v appartient à la base des fonctions tests. on a donc

$$\sum_{j=2}^{4} u_i \int_0^1 \phi_j' \phi_i'\ dx = \int_0^1 \phi_i\ dx$$

On a par exemple

$$\int_0^1 \phi_2' \phi_2' dx = \int_0^{1/2} 16\ dx = 8$$

$$\int_0^1 \phi_2\ dx = \int_0^{1/4} 4x\ dx + \int_{1/4}^{1/2} 2(1-2x)\ dx = 1/4$$

$$\int_0^1 \phi_2' \phi_3'\ dx = \int_{1/4}^{1/2} (-16)\ dx = -4$$

$$\int_0^1 \phi_5' \phi_5'\ dx = \int_{3/4}^{1} 16\ dx = 4$$

$$\int_0^1 \phi_4' \phi_5'\ dx = \int_{3/4}^{1} (-16)\ dx = -4$$

On a finalement le système matriciel suivant :

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ -4 & 8 & -4 & 0 & 0 \\ 0 & -4 & 8 & -4 & 0 \\ 0 & 0 & -4 & 8 & -4 \\ 0 & 0 & 0 & -4 & 4 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \\ 1/8 \end{bmatrix} \tag{4}$$

Ce système est le même que celui obtenue en (3). La dernière ligne de (4) est équivalente à la somme des deux dernières lignes de (3).

Ex 2 *Les polynômes de Lagrange*

1)

(a) La surface de ce triangle est égal à

$$S = \int_{x=0}^{1} \left[\int_{0}^{1-x} dy \right] dx = \int_{x=0}^{1} (1-x) dx = \frac{1}{2}$$

(b) Par exemple pour la première fonction, on écrit $L_1(x,y) = a + bx + cy$ et on doit résoudre

$$\left\{ \begin{array}{lllll} a & & & = & 1 \\ a & +b & & = & 0 \\ a & & +c & = & 0 \end{array} \right.$$

ce qui donne $L_1(x,y) = 1 - x - y$. De la même façon, on trouve $L_2(x,y) = x$ et $L_3(x,y) = y$.

2)

On trouve $\int \int_{\mathbf{T}_{\text{ref}}} L_i \; dx \; dy = \frac{1}{6}$

3)

(a) Si (ξ, η) sont les coordonnées dans le triangle de référence, on a

$$(x,y) = \sum_{i=1}^{3} (x_i, y_i) L_i(\xi, \eta)$$

ou

$$\left(\begin{array}{c} x \\ y \end{array} \right) = \left(\begin{array}{c} (1-\xi-\eta)x_1 + \xi x_2 + \eta x_3 \\ (1-\xi-\eta)y_1 + \xi y_2 + \eta y_3 \end{array} \right)$$

(b) On utilise la formule qui permet de changer de coordonnée

$$S = \int_R dx \; dy = \int_{T^{-1}(R)} |J(T)| \; d\xi \; d\eta$$

La matrice Jacobienne de la transformation s'écrit

$$J(T) = \left[\begin{array}{cc} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{array} \right] = \left[\begin{array}{cc} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{array} \right]$$

et donc $|J(T)| = (x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1)$ qui ne dépend pas de ξ et η. Finalement

$$S = \frac{(x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1)}{2}$$

4)

(a) On a bien par contruction que $\hat{U}(A_i) = U_i$.

(b)

$$\int \int_{T_{ref}} \hat{U}(x,y) \; dx \; dy = \sum_i U_i \int \int_{\mathbf{T}_{\text{ref}}} L_i \; dx \; dy = \frac{U_1 + U_2 + U_3}{3} \; \times \; \frac{1}{2}$$

(c)

$$= \int \int_{\mathbf{T}_{\text{ref}}} \hat{U}(\xi, \eta) |J(T)| \; d\xi \; d\eta = \frac{U_1 + U_2 + U_3}{3} \frac{|J(T)|}{2} = \frac{U_1 + U_2 + U_3}{3} \; \times \text{ aire de } T_{ge}$$

On a remplacé $U(x,y)$ par sa valeur moyenne à l'intérieur du triangle

$$\int \int_{\mathbf{T}_{\text{ref}}} U(x,y) \; dx \; dy \sim \frac{U_1 + U_2 + U_3}{3} \; \times \text{ aire de } T_{ge}$$

5)

(a) On a les 4 triangles suivant définies à partir des sommet $S_1 = (0,0)$, $S_2 = (1,0)$, $S_3 = (1,1)$, $S_4 = (0,1)$ et $S_5 = (1/2, 1/2)$ (voir Figure 2)

– T_1 dont les sommets sont $S_1,\ S_2,\ S_5$.
– T_2 dont les sommets sont $S_2,\ S_3,\ S_5$.
– T_3 dont les sommets sont $S_3,\ S_4,\ S_5$.
– T_4 dont les sommets sont $S_4,\ S_1,\ S_5$.

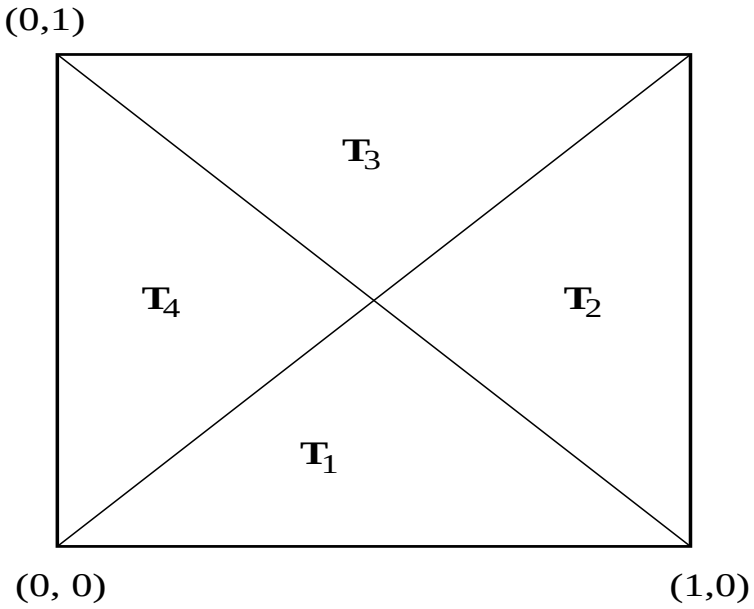

FIG. 2 – Triangulation du carré.

Comme chaque triangle a la même surface $I_T = 1/4 =$ (surface du carré)/4.
(b) La fonction U a pour valeurs en ces 5 points ($U_i = U(S_i)$) : $U_1 = 0$, $U_2 = 1$, $U_3 = 2$, $U_4 = 1$ et $U_5 = 1$, donc

$$\int\int_{\text{carré}} U(x,y)dx\ dy = \sum_1^4 \int\int_{\mathbf{T}_i} U(x,y)dxdy = I_T \frac{2U_1 + 2U_2 + 2U_3 + 2U_4 + 4U_5}{3} = \frac{1}{4}\frac{12}{3} = 1$$

(c) La formule permettant d'avoir une valeur approchée de la fonction U sur un triangle, suppose que l'on peut remplacer la fonction U par un polynôme du premier degré en x et y. Dans notre cas, on a $\hat{U} = U$ sur chaque triangle, donc avec cette formule, on obtient l'intégrale exacte.

Correction 1 : Initiation à Scilab

Ex 1 *Manipulation sur les vecteurs*

1)

```
v1 = 1:.1:3
```

2)

```
v2 = 3:-.1:1
```

3)

```
v3 = (1:10)^2
```

4)

```
n = (1:10);
un = ones(1,10);
v4 = (- un).^n .* (n^2)
```

5)

```
un = ones(1,10)
v5 = [0*un, un]
```

Ex 2 *Manipulation sur les Matrices*

1)

```
m = matrix(1:36,6,6)'
m = [1:6;7:12;13:18;19:24;25:30;31:36]
```

2)

```
n = 5
C = zeros(n,n);
for i = 1:n
   C(i,i) =2;
end;
for i = 1:n-1
   C(i,i+1) = -1;
end;
for i = 2:n
   C(i,i-1) = -1;
end;
```

ou dans un style plus `Scilab`,

```
 C = 2*eye(n,n)-diag(ones(n-1,1),1)-diag(ones(n-1,1),-1)}
```

Ex 3 *Exemple de tracé*

```
X = 0:.1:4*%pi;
Y = sin(X) + X;
plot2d(X,Y)
xtitle('Exemple','x','sin(x) + x');
```

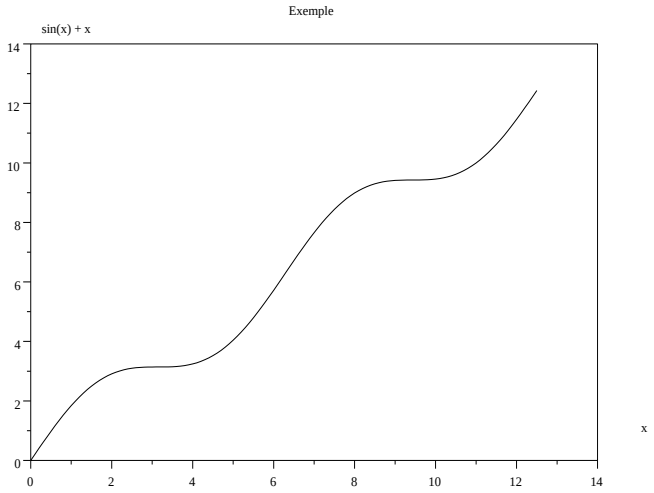

FIG. 3 – Tracé de $f(x) = \sin x + x$.

Ex 4 *Ecriture d'une fonction*

```
function C = alterne2_colonne(A,B)

[n,m] = size(A)

C = []

for i=1:m
  C = [C, A(:,i), B(:,i)];
end;
```

ou avec un style plus **scilab**

```
function C = alterne2_colonne(A,B)

[n,m] = size(A)

A = matrix(A,n*m,1);
```

```
B = matrix(B,n*m,1);

C = [A B]

C = matrix(C,n,2*m);

endfunction
```

Ex 5 *Calcul de la solution d'une edo*

1)

C'est une équation à coefficients constants : une solution particulière de l'équation est $u_s = 1$; la solution générale de l'équation sans second membre $u_0 = a\ e^x + b\ e^{-x}$, alors la solution peut s'écrire $u = u_s + u_0$.

Les conditions aux limites permettent de calculer les coefficients a et b ....

2)

```
clear
n = 50;
F = ones(n,1);
A = eye(n,n);

x = matrix(linspace(0,1,n),n,1);
Dx = 1/(n-1);
u_exact = 1- (exp(x) + exp(1-x))/(1+exp(1));

//  on impose les conditions aux limites

 F(1) = 0;
 F(n) = 0;

for i=2:n-1
   A(i,i) = A(i,i) + 2./Dx^2;
   A(i,i-1) = -1/Dx^2;
   A(i,i+1) = -1/Dx^2;
end;


u_c = A\F;
erreur = norm(u_exact-u_c)

\\ Trace de la solution

xbasc()
plot2d(x,u_c,style=[-1])

x=linspace(0,1,100);
u_exact = 1- (exp(x) + exp(1-x))/(1+exp(1));
plot2d(x,u_exact,style=[1])

xtitle('Solution','x','u(x)');
```

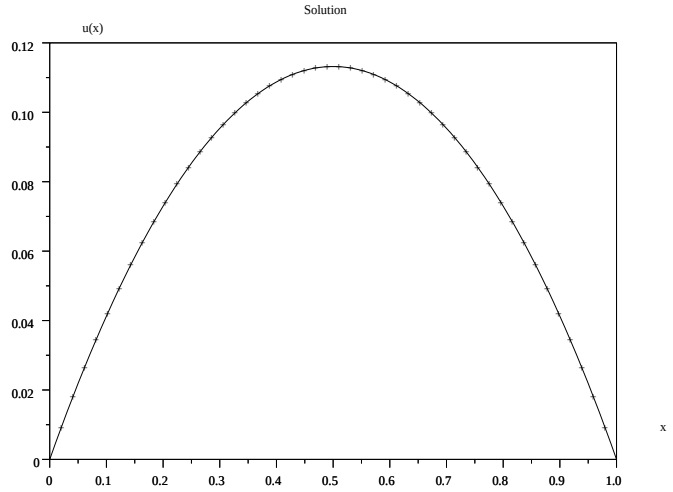

FIG. 4 – Solution du problème pour $n = 50$: solution exacte (–), solution calculée (+).

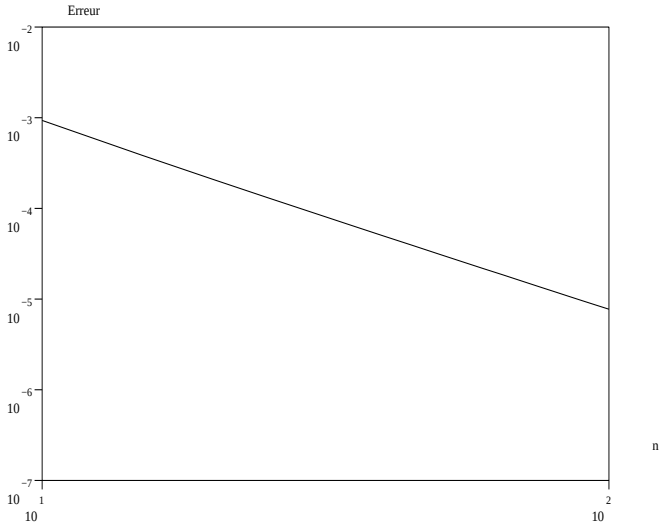

FIG. 5 – Evolution de l'erreur $||u - u_{exact}||/||u_{exact}||$ en fonction de n

Correction 2 : Initiation aux éléments finis 1D

Ex 1 *Eléments Finis Unidimensionnels*

1)
La formulation faible

$$\int_0^1 (u'\ v' + u\ v)\ dx = \int_0^1 v\ dx$$

2)

Le listing complet du programme est donné à la réponse de la question **5)**.
Pour le calcul de COOR et de CONNEC, il faut lire les lignes 8-24.
3)
Le système élémentaire sur le $k$ème élément s'écrit :

$$A^k U^k = F^k \tag{5}$$

où :

$$a_{ij}^k = \int_{x_1^k}^{x_2^k} \left[\psi_j^{k\prime}\ \psi_i^{k\prime} + \psi_j^k\ \psi_i^k\right]\ dx = \int_{-1}^{1} \left[\left(\frac{1}{Jac^k}\right)^2\ \frac{dL_j}{d\xi}\ \frac{dL_i}{d\xi} + L_j\ L_i\right]\ Jac^k\ d\xi$$

$$f_i^k = \int_{x_1^k}^{x_2^k} \psi_i^k\ dx = \int_{-1}^{1} L_i\ Jac^k\ d\xi$$

où Jac^k est le Jacobien de la transformation

$$Jac^k = \frac{x_2^k - x_1^k}{2}$$

c.f. lignes 30-50 du listing.
4)

c.f. lignes 55-62 du listing.
5)

Listing du programme permettant la résolution avec l'élément $\mathbf{P_1}$:

```
1   //  on charge les programmes dont on a besoin
2   //
3   exec('P1.sci'); exec('gauss.sci');
4   //  nbre de points pour l'integration numerique
5   Ng = 3;
6   [XG,PG] = gauss(Ng);
7   //
8   //
9   Nelt = 20;              // nbre d'elements
10
```

```
11  Nno = 2;                // nbre de noeud par element
12
13  Npt = Nelt * (Nno-1)+1; // nbre de points
14
15  Dx = 1./(Npt-1);
16
17  COOR = matrix((0:Dx:1),Npt,1);  //  Coordonnees des points
18
19  CONNEC = zeros(Nelt,Nno);
20  for i=1:Nelt
21     for j=1:Nno
22        CONNEC(i,j) = j + (i-1);  // matrice de connectivite
23     end;
24  end;
25  //
26  //    Fin initialisation des tableaux
27
28  A_glob = zeros(Npt,Npt); F_glob = zeros(Npt,1);
29
30  for iel = 1:Nelt
31     // calcul du syteme elementaire sur chaque element
32     A_el = zeros(Nno,Nno);
33     F_el = zeros(Nno,1);
34
35     x1 = COOR(CONNEC(iel,1)); x2 = COOR(CONNEC(iel,2));
36     jac = (x2 - x1)/2.;   // le jacobien est constant sur l'element
37
38     for ig=1:Ng    // boucle pour l'integration numerique
39
40        [FP,DxFP] = P1(XG(ig));  // choix des fonctions tests
41
42        DxFP = DxFP/jac;
43
44         for i=1:Nno
45            F_el(i) = F_el(i) + PG(ig)*FP(i)*jac;
46            for j=1:Nno
47               A_el(i,j) = A_el(i,j) + PG(ig)*(DxFP(j)*DxFP(i) + FP(i)*FP(j))*jac;
48            end;
49         end;
50      end;
51     // fin calcul systeme elementaire pour l'element iel
52
53  // on doit assembler
54
55     for i=1:Nno
56        i_glob = CONNEC(iel,i);
57        F_glob(i_glob) = F_glob(i_glob) + F_el(i);
58        for j=1:Nno
59           j_glob = CONNEC(iel,j);
60           A_glob(i_glob,j_glob) = A_glob(i_glob, j_glob) + A_el(i,j);
61        end;
```

```
62      end;
63  end;  // fin boucle sur les elements
64
65  //
66  //  On impose les deux conditions aux limites de Diriclet
67  //        U(0) = U(1) = 0
68
69  A = [zeros(1,Npt); A_glob(2:(Npt-1),:); zeros(1,Npt)];
70  A(1,1) = 1; A(Npt,Npt) = 1;
71  F = [0; F_glob(2:(Npt-1)); 0];
72
73  u_cal = A\F;
74
75  // comparaison avec la solution exacte
76  //
77  x = COOR;
78  u_exact = 1- (exp(x) + exp(1-x))/(1+exp(1));
79  erreur = norm(u_cal-u_exact)/norm(u_exact)
80
81  xbasc();
82  plot2d(x,u_cal,style=-1);
83  x = linspace(0,1,100);
84  u_exact = 1- (exp(x) + exp(1-x))/(1+exp(1));
85  plot2d(x,u_exact,style=1);
86  xtitle('Solution P1; Nelt ='+string(Nelt)+', Npt ='+string(Npt),'x','u');
87
```

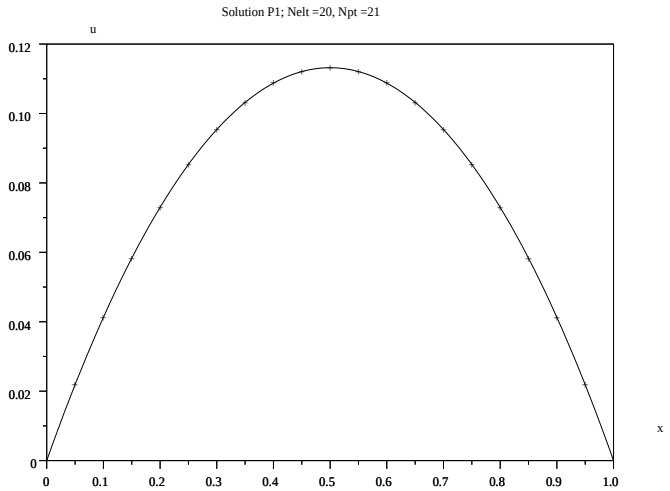

FIG. 6 – Solution du problème pour `Nelt = 20` : solution exacte (–), solution calculée (+).

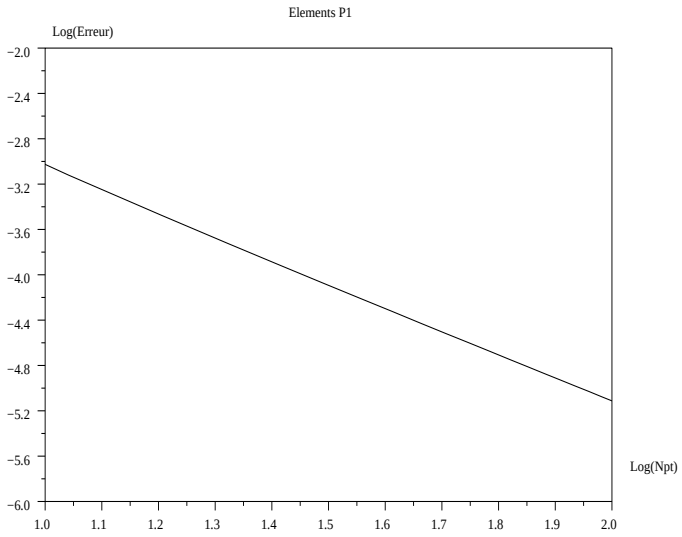

FIG. 7 – Evolution de l'erreur $||u - u_{exact}||/||u_{exact}||$ en fonction de `Npt`

6)
Si on veut utiliser des élements $\mathbf{P_2}$, les modifications sont mineures par rapport au programme précédent :

- Il faut appeller la fonction qui calcule les polynômes de Lagrange de degré 2 (Ligne 3).
- Modifier la valeur `Nno` (Ligne 11).
- Le calcul de la matrice de connectivité (Lignes 20-24).
- L'appel aux fonctions tests L_2 lors de l'intégration numérique (Ligne 40).

```
3    exec('P2.sci'); exec('gauss.sci');
```

```
11    Nno = 3; // nbre de noeuds par element
```

```
20    for i=1:Nelt
21      CONNEC(i,1) = 1 + 2*(i-1);
22      CONNEC(i,2) = 3 + 2*(i-1);
23      CONNEC(i,3) = 2 + 2*(i-1);
24    end;
```

```
40    [FP,DxFP] = P2(XG(ig));
```

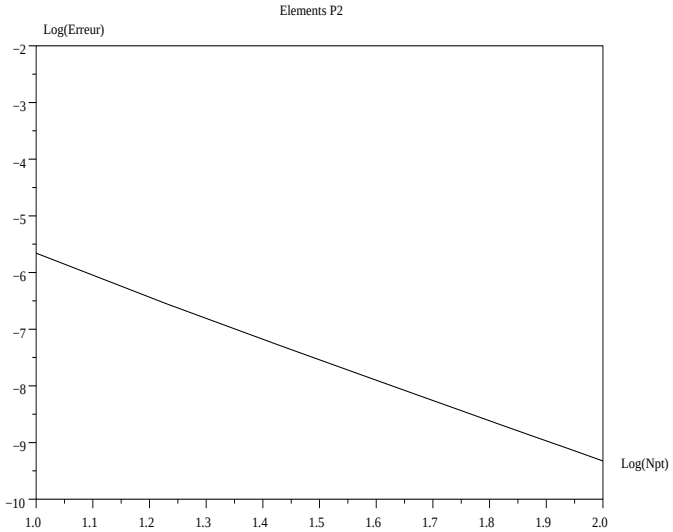

FIG. 8 – Evolution de l'erreur $||u - u_{exact}||/||u_{exact}||$ en fonction de `Npt` pour des élément $\mathbf{P_2}$.

7)
les modifications sont les suivantes :
On libère la valeur de u en $x = 1$ aux lignes 69-71.

```
69    A = [zeros(1,Npt); A_glob(2:Npt,:)];
70    A(1,1) = 1;
71    F = [0; F_glob(2:Npt)];
```

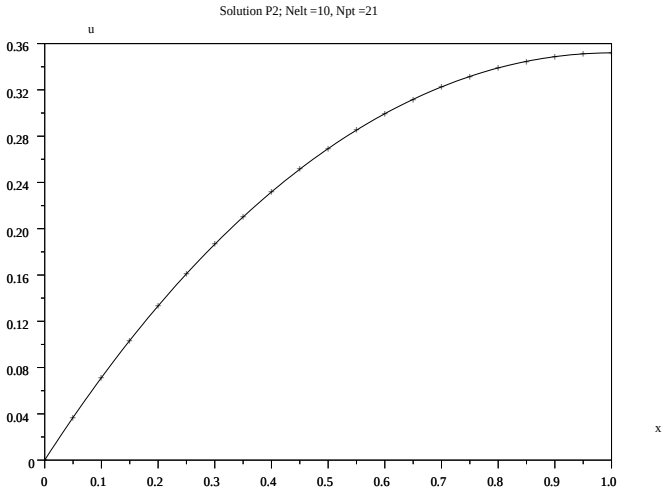

FIG. 9 – Solution du problème pour `Nel = 10` avec des élements $\mathbf{P_2}$: solution exacte (–), solution calculée (+).

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

$$A^k U^k = F^k \tag{6}$$

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;
```

```
47  //
48  //  creation du vecteur UC
49  //
50  UC = zeros(Npt-NI,1);
51
52  //
53  //    Fin initialisation des tableaux
54
55  A_glob = spzeros(Npt,Npt); F_glob = zeros(Npt,1);
56
57  for iel = 1:Nelt  // boucle sur les elements
58  //
59  // ------------------------------------------------------------------------
60  //  PARTIE   A ECRIRE
61  //---------------------------------------------------------------------
62  // calcul du syteme elementaire sur chaque element
63  //
64     A_el = zeros(Nno,Nno);
65     F_el = zeros(Nno,1);
66
67     x  = COOR(CONNEC(iel,:),1); y = COOR(CONNEC(iel,:),2);
68
69
70     for ig=1:Ng    // double boucle pour l'integration numerique
71      xg = XG(ig);
72      for jg=1:Ng
73         yg = XG(jg);
74  //
75  // calcul de la matrice jacobienne
76  //
77             [F,DxiFP,DetaFP] = Q1(xg,yg);
78             j11 = x(1:4)'*DxiFP;
79             j12 = x(1:4)'*DetaFP;
80             j21 = y(1:4)'*DxiFP;
81             j22 = y(1:4)'*DetaFP;
82  //
83             DT  = [j11 j12; j21 j22];
84             jac = det(DT);
85             B = inv(DT)';
86  //
87          [FP,DxiFP,DetaFP] = Q2(xg,yg);
88
89          for i=1:Nno
90             DFPi = B*[DxiFP(i); DetaFP(i)];
91             F_el(i) = F_el(i) + PG(ig)*PG(jg)*FP(i)*jac;
92             for j=1:Nno
93               DFPj = B*[DxiFP(j); DetaFP(j)];
94               A_el(i,j) = A_el(i,j) + PG(ig)*PG(jg)*( DFPj' * DFPi) *jac;
95             end;
96          end;
97      end;
```

```
98      end;
99  //
100 // fin calcul systeme elementaire pour l'element iel
101 //
102 //-------------------------------------------------------------
103 //  FIN PARTIE A ECRIRE
104 //-------------------------------------------------------------
105 // on doit assembler
106
107    for i=1:Nno
108       i_glob = ADRESS(iel,i);
109       F_glob(i_glob) = F_glob(i_glob) + F_el(i);
110     end;
111
112    A_el=sparse(A_el);
113    [ij,v,mn] = spget(A_el);
114    for i = 1:length(v)
115      ij(i,1) = ADRESS(iel,ij(i,1));
116      ij(i,2) = ADRESS(iel,ij(i,2));
117    end;
118    A_el = sparse(ij,v,[Npt Npt]);
119    A_glob = A_glob + A_el;
120
121 end;  // fin boucle sur les elements
122
123
124 A = A_glob(1:NI,1:NI);  F = F_glob(1:NI);
125 M12 = A_glob(1:NI,NI+1:$);
126 F = F - M12*UC;
127
128 [h,rk] = lufact(A);
129 U = lusolve(h,F);
130 ludel(h);
131
132 u_cal = [U; UC];
133
134 [u,x,y]=trace_u_2D(u_cal,COOR,NUMER,1,20,20);
```

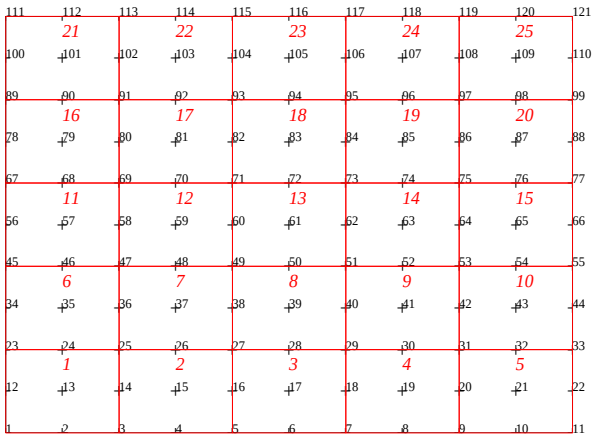

FIG. 10 – Exemple de maillage avec `Nelt = 25` et `Npt = 121`.

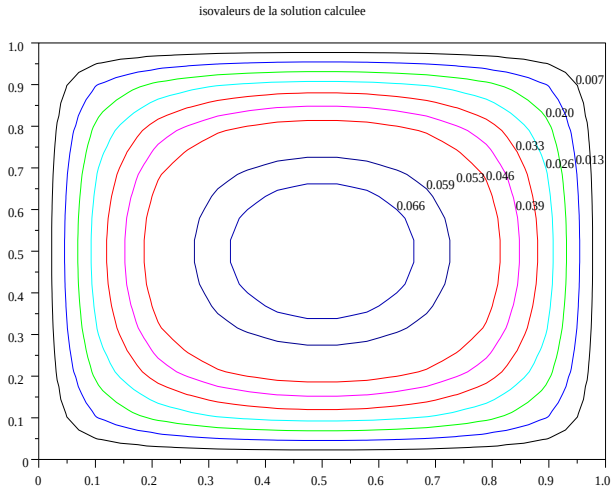

FIG. 11 – Solution du problème pour avec `Nelt = 100` et `Npt = 441`.

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
```

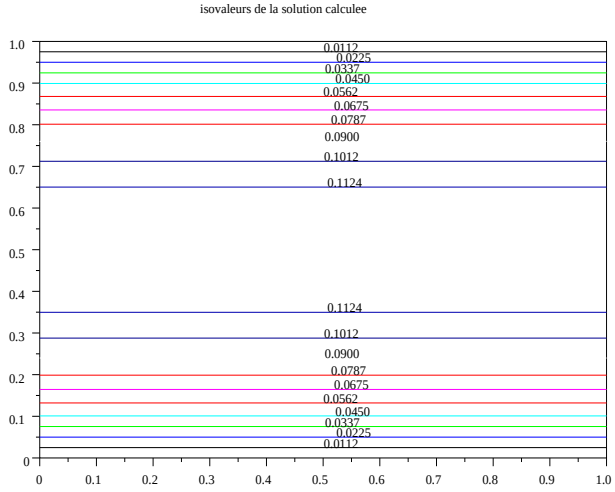

FIG. 12 – Solution de la question **5)** pour avec `Nelt = 100` et `Npt = 441`.

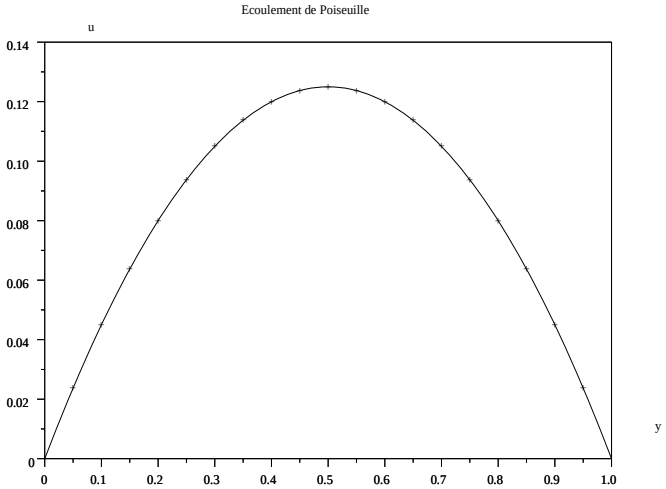

FIG. 13 – Comparaison entre la solution calculée (+) et la solution analytique $u_e = y(y-1)/2$ pour une valeur de x fixée.

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.

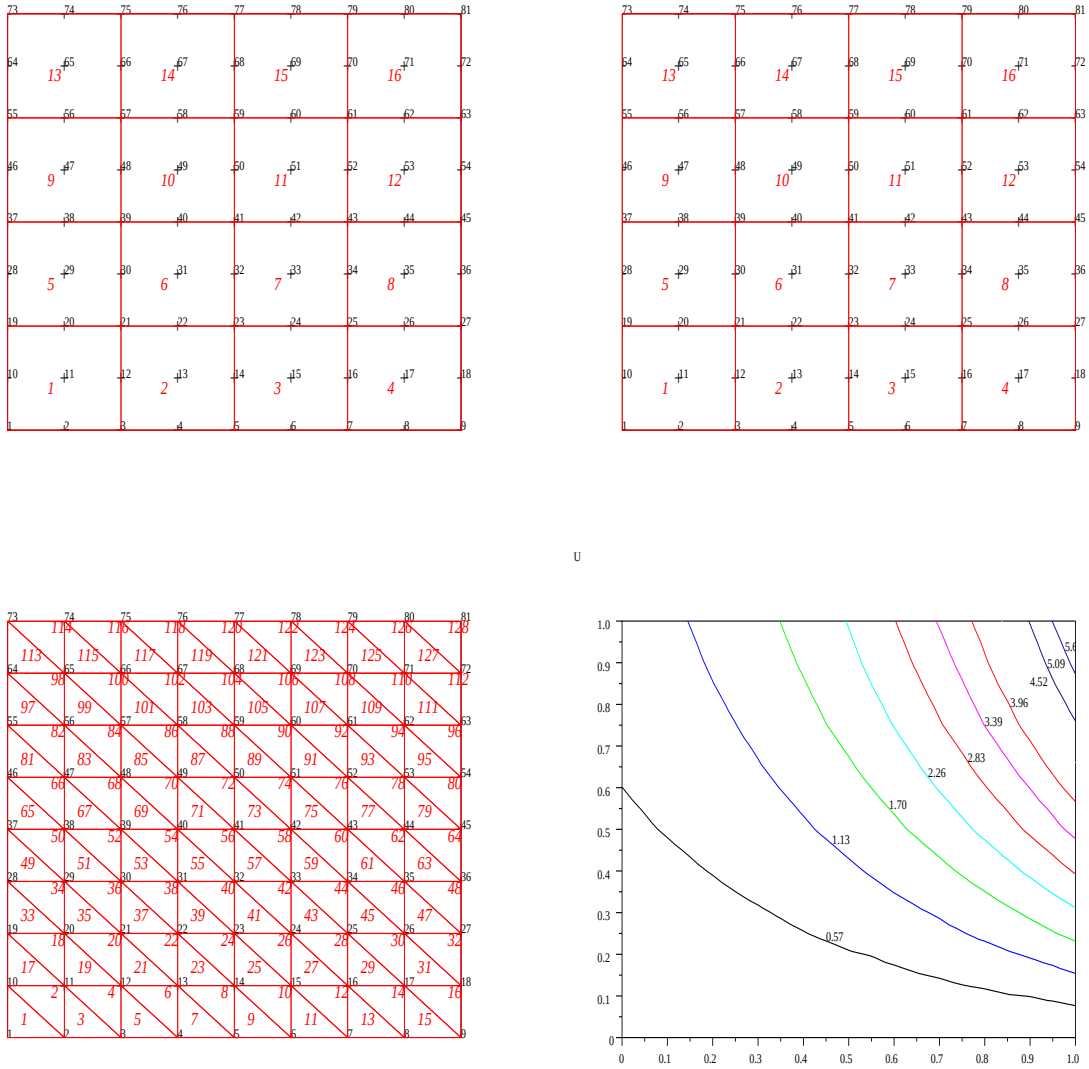

FIG. 14 – Les différents maillages et la solution : (a) $\mathbf{Q_1}$; (b) $\mathbf{Q_2}$; (c) $\mathbf{P_1}$; (d) $u(x, y)$.