TP5 : Discrétisation de l'équation de Stokes

On se propose dans ce TP d'associer deux éléments finis $P1-$Lagrange et $P2-$Lagrange, pour la discrétisation d'un problème aux limites. Cette approche est couramment adoptée pour résoudre des problèmes à inconnues vectorielles. Nous nous intéressons ici au problème de Stokes.

Exercice - 1 *Problème de Stokes*

Soit $\Omega \subset \mathbb{R}^2$ et $\mathbf{f}, \mathbf{u}_D$ deux fonctions de $\mathbb{R}^2 \to \mathbb{R}$. On considère le problème :

Chercher deux fonctions $\mathbf{u} : \mathbb{R}^2 \to \mathbb{R}^2, p : \mathbb{R}^2 \to \mathbb{R}$ telles que

$$\left\{ \begin{array}{rlll} -\Delta \mathbf{u} + \nabla p = & \mathbf{f} & \text{dans} & \Omega, \\ \nabla \cdot \mathbf{u} = & 0 & \text{dans} & \Omega, \\ \mathbf{u} = & \mathbf{u}_D & \text{sur} & \partial\Omega. \end{array} \right. \tag{1}$$

Où, pour $\mathbf{u} = (u_1, u_2)^t$, on a posé $\Delta \mathbf{u} = (\Delta u_1, \Delta u_2)^t$, $\nabla \cdot \mathbf{u} = \frac{\partial u_1}{\partial x} + \frac{\partial u_2}{\partial y}$.
Et pour toute fonction $v : \mathbb{R}^2 \to \mathbb{R}$, $\Delta v = \frac{\partial^2 v}{\partial^2 x} + \frac{\partial^2 v}{\partial^2 y}$.

Q-1 **: Formulation variationnelle**

On pose

$$\begin{aligned} &H^1(\Omega) = \{v \in L^2(\Omega); \nabla v \in (L^2(\Omega))^2\}, \quad H_0^1(\Omega) = \{v \in H^1(\Omega); v = 0 \text{ sur } \partial\Omega\}, \\ &V = H_0^1(\Omega) \times H_0^1(\Omega), \quad M = \left\{ q \in L^2(\Omega); \int_\Omega q\, dx\, dy = 0 \right\}, \\ &a(\mathbf{u}, \mathbf{v}) = \int_\Omega \nabla \mathbf{u} : \nabla \mathbf{v}\, dx\, dy \equiv \sum_{i=1}^{2} \int_\Omega \nabla u_i \cdot \nabla v_i\, dx, \quad b(\mathbf{v}, p) = -\int_\Omega p\, \nabla \cdot \mathbf{v}\, dx\, dy, \\ &l(\mathbf{v}) = \int_\Omega \mathbf{f} \cdot \mathbf{v}\, dx\, dy. \end{aligned}$$

On admet qu'il existe une *unique* fonction $\bar{\mathbf{u}}_D \in \left(H^1(\Omega)\right)^2$ dont la restriction sur $\partial\Omega$ coïncide avec $\mathbf{u}_D$.
(*C'est l'image de* $(\mathbf{u}_D)|_{\partial\Omega}$ *par le relèvement continu des traces. Les détails sortent du cadre du présent TP.*)

Montrer que la formulation variationnelle du problème (1) est alors donnée par :

$$\left\{ \begin{array}{ll} \text{Chercher } \mathbf{u} \in \bar{\mathbf{u}}_D + V \text{ et } p \in M \text{ tels que} & \\ \quad a(\mathbf{u}, \mathbf{v}) + b(\mathbf{v}, p) = l(v) & \forall \mathbf{v} \in V, \\ \quad b(\mathbf{u}, q) = 0 & \forall q \in M. \end{array} \right. \tag{2}$$

On admettra que ce problème possède une unique solution, dotée des régularités nécessaires, permettant son évaluation ponctuelle.

Exercice - 2 *Discrétisation par éléments finis de Taylor-Hood*

Pour discrétiser le problème de Stokes, on va introduire deux espaces éléments finis, l'un pour la vitesse et l'autre pour la pression. L'association de ces espaces doit être judicieusement menée, car en effet, il faudra s'assurer que : à vitesse donnée, le problème en pression est résoluble. Le choix des éléments finis est ainsi soumis à une restriction qu'on appelle couramment *condition inf-sup ou BNB (Banach-Nečas-Babuška) discrète*. Plusieurs couples d'éléments finis ont franchi cet obstacle, nous nous intéressons ici à celui de Taylor-Hood, appelé aussi éléments finis P_2/P_1. Il consiste à utiliser les éléments finis $P2-$Lagrange pour discrétiser chaque composante de la vitesse et les éléments finis $P1-$Lagrange pour discrétiser la pression.

Q-1 **: Discrétisation par éléments finis** (p_2/p_1)

On introduit τ_h un maillage conforme de Ω, formé des triangles.

Espaces et problème discret. On introduit $V_h \subset V$ et $M_h \subset M$ des sous-espaces vectoriels de dimension finie définis de la manière suivante :

$$X_h^r = \{p_h \in C^0(\Omega) : p_h|_T \in P_r(T)\ \forall T \in \tau_h\},$$

où $P_r(T)$ est l'espace vectoriel des polynômes de degré total r sur le triangle T,

$$V_h = \{\mathbf{v}_h \in X_h^2 \times X_h^2 \text{ tel que } : \mathbf{v}_h = 0 \text{ sur } \partial\Omega\},$$

$$M_h = \{q_h \in X_h^1 \text{ tel que } \int_\Omega q_h\, dx = 0\}.$$

Le problème discret s'écrit alors

$$\left\{\begin{array}{ll} \text{Chercher } \mathbf{u}_h \in \bar{\mathbf{u}}_{hD} + V_h \text{ et } p_h \in M_h \text{ tels que} & \\ \quad a(\mathbf{u}_h, \mathbf{v}_h) + b(\mathbf{v}_h, p_h) = l(\mathbf{v}_h) & \forall \mathbf{v} \in V_h, \\ \quad b(\mathbf{u}_h, q_h) = 0 & \forall q_h \in M_h, \end{array}\right. \tag{3}$$

où $\bar{\mathbf{u}}_{hD}$ est un interpolé de $\bar{\mathbf{u}}_D$ dans $X_h^2 \times X_h^2$.

1. On suppose que le maillage a N_{so} sommets, N_{ma} triangles, N_a arêtes, dont N_{ab} arêtes frontières. Donner en fonction de ces quantités les dimensions de $X_h^r, r = 1, 2,\ V_h$, et M_h.
2. On admettra que ce problème admet une unique solution. (*Il est important de s'en assurer*).

Problème matriciel. On se propose maintenant d'écrire le problème discret sous forme matriciel.

1. On introduit une base (Φ_i) de V_h et (ψ_i) de M_h. Soit $\mathbf{U}$ respectivement P les composantes respectives de $\mathbf{u}_h$ et de p_h dans ces bases.
   Montrer que le problème matriciel peut s'écrire
   $$\begin{bmatrix} A & B \\ B^t & 0 \end{bmatrix} \begin{bmatrix} \mathbf{U} \\ P \end{bmatrix} = \begin{bmatrix} \mathbf{F} \\ 0 \end{bmatrix}. \tag{4}$$
   Expliciter $\mathbf{F}$, A, B.
2. Cette écriture matricielle peut d'avantage être simplifiée. En effet, on peut constater qu' une base de V_h est $\left\{(\varphi_i, 0), i = 1, \ldots \dim(X_{0h}^2)\right\} \cup \left\{(0, \varphi_i), i = 1, \ldots \dim(X_{0h}^2)\right\}$ où les φ_i constituent une base de $X_{0h}^2 = \{v \in C^0(\Omega); v|_T \in P_2(T), \forall T \in \tau_h, v = 0 \text{ sur } \partial\Omega\}$.
   Montrer qu'on peut encore écrire (4) sous la forme
   $$\begin{bmatrix} A_1 & 0 & B_1 \\ 0 & A_2 & B_2 \\ B_1^t & B_1^t & 0 \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ P \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \\ 0 \end{bmatrix}. \tag{5}$$
   On exprimera, $A_i, B_i, F_i, i = 1, 2$, en fonction des bases $(\varphi_i)_{1 \leq i \leq \dim(X_{0h}^2)}$ et $(\psi_i)_{1 \leq i \leq \dim(M_h)}$.

Matrices élémentaires et vecteurs élémentaires. Comme indiqué dans la procédure de discrétisation par la méthode des éléments finis, il faut d'abord construire les matrices et seconds membres élémentaires. On rappelle que chaque composante de la vitesse est approchée par les éléments finis P_2-Lagrange alors que la pression est approchée par les éléments finis P_1-Lagrange.

Q-2 : Soit T un triangle de numéro k, donner en fonction des fonctions des bases locales P1 et P2 Lagrange, les expressions des matrices $A_i^k, B_i^k, F_i^k, i = 1, 2$. Puis en utilisant les formules de quadratures (voir TP3), construire ces matrices si le triangle T est défini par les coordonnées de ses sommets $(x_i, y_i), i = 1, 2, 3)$ (i.e la matrice coordonnées $[X_i] = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \end{bmatrix}$). (*On expliquera ce que fait le script suivant*).

script Matlab qui ...

```
function [A1k,A2k,B1k,B2k,F1k,F2k] = elmatStokesp2p1(X)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% fonction qui construit les matrices et vecteurs élémentaires pour Stokes par P2/P1
%
% ENTREE: X-> la matrice des coordonées des sommets du triangle
% COURS D'ELEMENTS FINIS
% M2PRO 2007-2008    Lab AN-EDP Univ d'ORSAY
% (c) Jean-Baptiste Apoung Kamga (July 2007)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
A1k= zeros(6,6);A2k= zeros(6,6);
B1k= zeros(6,3);B2k= zeros(6,3);
F1k= zeros(6,1);F2k= zeros(6,1);
[PhQ2, dxPhQ2, dyPhQ2, weight,XYQ] = basis_and_deriv_on_quad(X,2,5);
[PhQ1, dxPhQ1, dyPhQ1, weight,XYQ] = basis_and_deriv_on_quad(X,1,5);
nq=length(weight);
for l = 1:nq
    DXPh = [dxPhQ2(l,:) ; dyPhQ2(l,:)];
    dxDXPh = dxPhQ2(l,:);
    dyDXPh = dyPhQ2(l,:);
    Ph1    = PhQ1(l,:);
    Ph2    = PhQ2(l,:);
    wl = weight(l);
    A1k = A1k + wl *DXPh'*DXPh;
    A2k = A2k + wl *DXPh'*DXPh;
    B1k = B1k - wl *dxDXPh'*Ph1;
    B2k = B2k - wl *dyDXPh'*Ph1;
    xl  = XYQ(l,1); yl = XYQ(l,2);
    F1k = F1k + wl * f1(xl,yl) *Ph2';
    F2k = F2k + wl * f2(xl,yl) *Ph2';
 end
```

Q-3 : Matrices globales et vecteurs globaux.

Il est question d'assembler les matrices globales $A_i, B_i, i = 1, 2$ et les seconds membres globaux F_1, F_2, *sans prise en compte des conditions aux limites essentielles.*

1. On suppose disposer des tables de degrés de liberté $l2g(:,:)$ pour les éléments finis $P2-$Lagrange, et $l2g1(:,:)$ pour les éléments finis $P1-$Lagrange. Expliquer comment assembler dans $A_i, B_i, F_i, i = 1, 2$, les matrices et vecteurs élémentaires $A_i^k, B_i^k, F_i^k, i = 1, 2$ construits sur un élément T de numéro k.
2. Expliquer ce que fait le script suivant

script Matlab qui ...

```
function [A1,A2,B1,B2,F1,F2] = gstokes(T)
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  % ENTREE: T -> le maillage
  % SORTIE:  .........
  %
  %  COURS D'ELEMENTS FINIS
  %  M2PRO 2007-2008    Lab AN-EDP Univ d'ORSAY
  % (c) Jean-Baptiste Apoung Kamga (July 2007)
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[l2g, nbloc, nbdof , doflab, coord] = builddof(T, 2);
[l2g1, nbloc1, nbdof1 , doflab1, coord1] = builddof(T,1);
[nt,dum]=size(T.Triangles);
%Allow size to global matrix and global right hand side
n = nbdof;
```

```
m = nbdof1;
A1=zeros(n,n);A2=zeros(n,n);
B1=zeros(n,m);B2=zeros(n,m);
F1=zeros(n,1);F2=zeros(n,1);
% assembling
for k=1:nt
  %coordinate matrix
  Xk = T.Vertices(T.Triangles(k,1:3),1:2);
  %elementaries matrices and right hand sides
  [A1k,A2k,B1k,B2k,F1k,F2k] = elmatStokesp2p1(Xk);
  %assembling into globals
  F1(l2g(k,1:nbloc)) = F1(l2g(k,1:nbloc)) + F1k(1:nbloc);
  F2(l2g(k,1:nbloc)) = F2(l2g(k,1:nbloc)) + F2k(1:nbloc);
  A1(l2g(k,1:nbloc),l2g(k,1:nbloc)) = A1(l2g(k,1:nbloc),l2g(k,1:nbloc)) + A1k(1:nbloc,1:nbloc);
  A2(l2g(k,1:nbloc),l2g(k,1:nbloc)) = A2(l2g(k,1:nbloc),l2g(k,1:nbloc)) + A2k(1:nbloc,1:nbloc);
  B1(l2g(k,1:nbloc),l2g1(k,1:nbloc1))= B1(l2g(k,1:nbloc),l2g1(k,1:nbloc1)) + B1k(1:nbloc,1:nbloc1);
  B2(l2g(k,1:nbloc),l2g1(k,1:nbloc1))= B2(l2g(k,1:nbloc),l2g1(k,1:nbloc1)) + B2k(1:nbloc,1:nbloc1);
end
```

Q-4 : Prise en compte des conditions aux limites.

1. Monter que l'introduction des conditions aux limites par la méthode de *terme unité sur la diagonale* conduit au système

$$\begin{bmatrix} A_1 & 0 & B_1 \\ 0 & A_2 & B_2 \\ B_1^t & B_1^t & 0 \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ P \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \\ F_d \end{bmatrix}. \tag{6}$$

   Quelles sont les modifications apportées à $A_1, A_2, B_1, B_2, F_1, F_2$ de la formule (5) ? Expliquer l'origine du vecteur F_d ?

2. Pourrait-on aussi utiliser la méthode de pénalisation (*très grosse valeur sur la diagonale*) ? Si oui, quelles seraient ses limites potentielles ?

3. Expliquer ce que fait le script suivant

script Matlab qui ...

```
function [A1,A2,B1,B2,F1,F2,Fp] = bcmodifstokes(T,A1,A2,B1,B2,F1,F2,Dirlabel)
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  % function function [A1,A2,B1,B2,F1,F2,Fp] = bcmodifstokes(T,A1,A2,B1,B2,F1,F2,Dirlabel)
  % introduisant les conditions aux limtes dans le problème de Stokes P2/P1,
  % par la méthode de terme unité sur la diagonale
  %  ENTREE: T->maillage
  %          Dirlabel -> vecteur des label des frontières de Dirichlet
  %          A1,A2,B1,B2,F1,F2    -> matrices et vecteurs blocs
  % SORTIE:  A1,A2,B1,B2,F1,F2,Fb -> matrices et vecteurs blocs
  %%  COURS D'ELEMENTS FINIS
  %%  M2PRO 2007-2008    Lab AN-EDP Univ d'ORSAY
  %% (c) Jean-Baptiste Apoung Kamga (July 2007)
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[l2g, nbloc, nbdof , doflab, coord] = builddof(T, 2);
H1=zeros(nbdof,1);  H2=zeros(nbdof,1); Fp=zeros(size(B1,2),1);
dirIndex =zeros(nbdof,1);
if(length(Dirlabel)>=1)
    for i=1:length(Dirlabel)
        indlist = find(doflab==Dirlabel(i));
        if(length(indlist) >= 1)
            dirIndex(indlist)=1;
        end
    end
end
N = find(dirIndex == 1);
if(length(N) >= 1)
  % l'évaluation H1(N,1) = g1(coord(N,1),coord(N,2)), est risqué car si la fonction g1
  % possède un if (y>a)  alors coord(N,2)>a n'est vraie
  % que si l'inégalité est vraie pour toutes les composantes
    for i=1:length(N)
       H1(N(i),1)= g1(coord(N(i),1),coord(N(i),2));
       H2(N(i),1)= g2(coord(N(i),1),coord(N(i),2));
    end
    F1 = F1 - A1 * H1;    F2 = F2 - A2 * H2;    Fp = -B1'*H1 - B2'*H2;
    F1(N)=H1(N);   F2(N)=H2(N);
    A1(N,:)=0;B1(N,:)=0; A2(N,:)=0; B2(N,:)=0;   A1(:,N)=0; A2(:,N)=0;
    D = zeros(nbdof,1); D(N)=1;     A1 = A1 + diag(D); A2 = A2 + diag(D);
end
```

Q-5 : Résolution du problème matriciel.

Il est maintenant question de résoudre le problème (6). On se propose d'utiliser un algorithme d'Uzawa. Cela revient à condenser le problème en un unique problème sur la variable P et à utiliser une méthode itérative pour le résoudre.

Montrer que le problème (6) est équivalent à résoudre les problèmes (7) et (8) suivants

$$\begin{cases} U_1 = A_1^{-1}(F_1 - B_1\, P), \\ U_2 = A_2^{-1}(F_2 - B_2\, P), \end{cases} \tag{7}$$

$$(B_1^t A_1^{-1} B_1 + B_2^t A_2^{-1} B_2)P = -F_d + B_1^t A_1^{-1} F_1 + B_2^t A_2^{-1} F_2. \tag{8}$$

On résout alors le problème (8), puis on détermine la vitesse grâce à la formule (7).

Bien-que ce soit déconseiller en pratique, on demande ici de construire explicitement la matrice et le vecteur

$$A = B_1^t A_1^{-1} B_1 + B_2^t A_2^{-1} B_2 \quad \text{et} \quad F = -F_d + B_1^t A_1^{-1} F_1 + B_2^t A_2^{-1} F_2.$$

Résoudre le problème matriciel $AP = F$ par un algorithme de Gradient conjugué fourni par le fichier `cg.m`. On prendra soin d'initialiser l'algorithme par $P = 0$, qui vérifie bien la condition de moyenne nulle où alors on corrigera la pression après calcul pour la rendre à moyenne nulle (*une fonction* `pressioncorrect.m` *est fournie dans ce cas*).

script Matlab pour la correction de la pression

```
function [P]=pressioncorrect(T,P)
[l2g1, nbloc1, nbdof1 , doflab1, coord1] = builddof(T,1);
nt=size(T.Triangles,1);
F=zeros(nbdof1,1);
ar = 0;
for k=1:nt
    Xk = T.Vertices(T.Triangles(k,1:3),1:2);
    [PhQ1, dxPhQ1, dyPhQ1, weight,XYQ] = basis_and_deriv_on_quad(Xk,1,3);
    nq=length(weight);
    for l = 1:nq
        Ph1   = PhQ1(l,:);
        wl = weight(l);
        F(l2g1(k,:),1)= F(l2g1(k,:),1) + wl * Ph1';
        ar = ar + wl;
    end
end
ar = dot(F,P)/ar;
P = P - ar;
```

Q-6 : Post-traitement. On affiche la solution et on calcule si possible les erreurs numériques.

1. **Représentation de la vitesse calculée**. On dispose de la vitesse par ses composantes U_1, U_2 correspondant à ses valeurs aux positions des degrés de liberté. Or pour chaque degré de liberté idl, on dispose de sa position géométrique (coordonnées) : $(Coord(idl,1), Coord(idl,2))$.
   On représente alors la vitesse en traçant au point de coordonnées $(Coord(idl,1), Coord(idl,2))$ le vecteur de composante $(U_1(idl), U_2(idl))$. Ceci peut se faire en utilisant la commande `quiver` de Matlab.

script Matlab pour la représentation de la vitesse

```
function plotvel(T,U1,U2)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% function plotvel(T,U1,U2)
% fontion qui représente la vitesse sur le maillage
%  ENTREE : T-> maillage
%           U1-> vecteur première composante de la vitesse
%           U2-> vecteur deuxieme composante de la vitesse
%
%  COURS D'ELEMENTS FINIS
%  M2PRO 2007-2008    Lab AN-EDP Univ d'ORSAY
% (c) Jean-Baptiste Apoung Kamga (July 2007)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%Recupération des degrés de liberté et leur position
[l2g, nbloc, nbdof , doflab, Coord] = builddof(T, 2);
%représentation du maillage
plot2dmesh(T,[1,1,1]); hold on;
% représentation du champ des vitesses
quiver(Coord(:,1),Coord(:,2),U1,U2,2);
```

D'autres choix sont possibles, comme celui consistant à parcourir les mailles et à tracer dans chaque maille la vitesse en un ou plusieurs points ; ceci peut nécessiter l'utilisation de l'opérateur d'interpolation local.

2. **Représentation de la pression calculée**. La pression est une grandeur scalaire, on la représente par ses isovaleurs (*comme dans le cas de la résolution du laplacien*). On utilise pour cela la fonction `plotsol` programmée au précédent TP.
3. **Exemple : cavité entraînée** On prend $\Omega =]0,1[\times]0,1[$, $f = (f_1, f_2)^t, u_D = (g_1, g_2)^t$, avec

$$\left\| \begin{array}{ll} f_1(x,y) = 1, & g_1(x,y) = 1 \text{ si } y = 1, 0 \text{ sinon}, \\ f_2(x,y) = 1, & \qquad g_2(x,y) = 0. \end{array} \right.$$

Représenter graphique la vitesse et la pression solutions du problème de Stokes, sur le maillage du carrée unité (`carree.msh`) à récupérer sur la page web du cours, où à construire (voir TP3).

4. **Estimation de l'erreur numérique.** On rappelle que l'erreur dans une certaine norme $\|u - u_h\|$ entre la solution exacte u et la solution calculée u_h est majorée par

$$\|u - u_h\| \leq \|u - \pi_h(u)\| + \|\pi_h(u) - u_h\|.$$

La première à droite est l'erreur d'interpolation ; c'est la meilleure qu'on puisse obtenir puisque selon son principe même, la méthode de Galerkin consiste à plonger “un interpolé (à coefficients inconnus)” u_h dans la formulation variationnelle. Il est clair que ce faisant, on ne peut espérer obtenir une erreur $\|u - u_h\|$ meilleure que $\|u - \pi_h(u)\|$. C'est à ce titre qu'on peut quelque fois se restreindre au calcul de $\|\pi_h(u) - u_h\|$ et à le considérer comme erreur numérique.

On considère les données suivantes : $\Omega =]0,1[\times]0,1[$, $f = (f_1, f_2)^t, u_D = (g_1, g_2)^t$, avec

$$\left\| \begin{array}{ll} f_1(x,y) = 2\sin(\pi(x+y)) + \frac{1}{\pi}\cos(\pi(x+y)), & g_1(x,y) = \frac{1}{\pi^2}\sin(\pi(x+y)), \\ f_2(x,y) = -2\sin(\pi(x+y)) + \frac{1}{\pi}\cos(\pi(x+y)), & g_2(x,y) = -\frac{1}{\pi^2}\sin(\pi(x+y)). \end{array} \right.$$

Pour lesquelles la solution exacte est

$$u_1(x,y) = \frac{1}{\pi^2}\sin(\pi(x+y)), \quad u_2(x,y) = \frac{1}{\pi^2}\sin(\pi(x+y)), \quad p(x,y) = \frac{1}{\pi^2}\sin(\pi(x+y)).$$

On se propose de calculer les erreurs en norme L^2. On veut pour ainsi dire évaluer

$$\|\pi_h^2(u_1) - u_{1h}\|_{L^2(\Omega)}, \quad \|\pi_h^2(u_2) - u_{2h}\|_{L^2(\Omega)}, \quad \|\pi_h^1(p) - p_h\|_{L^2(\Omega)},$$

où π_h^2 est l'opérateur d'interpolation $P2-$Lagrange, π_h^1 est l'opérateur d'interpolation $P1-$Lagrange, (u_{1h}, u_{2h}) sont les composantes de la vitesse calculée, et p_h est la pression calculée.
Ces erreurs peuvent être calculées au moyen des matrices de masse $M_{i,j} = \int_\Omega \varphi_i \varphi_j \, dx$.
En effet, dans le cas particulier de la pression, si on désigne par P_e le vecteur valeurs de la pression exacte p aux positions des degrés de liberté (*i.e* `P_e = p(Coord( :,1),Coord( :,2))`),
on peut écrire : $\pi_h^1(p) - p_h = [\varphi_1 \dots \varphi_{ndl}](P_{ex} - P)$, d'où

$$\|\pi_h^1(p) - p_h\|^2_{L^2(\Omega)} = (P_{ex} - P)^t \underbrace{\left(\int_\Omega [\varphi_1 \dots \varphi_{ndl}]^t [\varphi_1 \dots \varphi_{ndl}] \, dx\, dy\right)}_{M \,\equiv\, \text{matrice de masse}} (P_{ex} - P)$$

On introduit la fonction suivante qui calcule la matrice de masse pour les polynômes de Lagrange d'ordre 1 et 2.

script Matlab pour la représentation de la vitesse

```
function [M]= massmat(T,order)
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  % function [M]= massmat(T,order)
  % fonction qui calcule la matrice de masse pour les éléments
  % finis Lagrange sur un maillage triangulaire
  %
  % ENTREES: T     -> le maillage
  %          order -> l'ordre d'approximation 1 ou 2
  % SORTIE:
  %          M -> matrice de masse
  %
  %  COURS D'ELEMENTS FINIS
  %  M2PRO 2007-2008    Lab AN-EDP Univ d'ORSAY
  % (c) Jean-Baptiste Apoung Kamga (July 2007)
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%degrés de liberte et leur position
[l2g, nbloc, nbdof , doflab, coord] = builddof(T,order);
% initialisation de la matrice
M=zeros(nbdof,nbdof);
nt=size(T.Triangles,1);
%boucle sur les éléments
for k=1:1:nt
  % Triangle courant
  Xk = T.Vertices(T.Triangles(k,1:3),1:2);
  % fonctions de base et leurs dérivées premières aux point de quadrature
    [PhQ, dxPhQ, dyPhQ, weight,XYQ] = basis_and_deriv_on_quad(Xk,order,5);
    nq=length(weight);
    % construction de la matrice locale et assemblage au vif
    for l = 1:nq
        Ph   = PhQ(l,:);
        wl = weight(l);
        M(l2g(k,:),l2g(k,:) )= M(l2g(k,:),l2g(k,:) )+ wl * Ph * Ph';
    end
end
```

(a) En utilisant cette fonction, calculer en norme L^2 les erreurs numériques commises dans la discrétisation $P2/P1$ de l'équation de Stokes.

(b) Faites varier le maillage en le prenant de plus en plus fin, et représenter les courbes de variation des erreurs L^2 en fonction du pas h du maillage. Quel ordre de convergence observez-vous, pour la vitesse ? pour la pression ?.
on pourra utiliser la fonction fournie
`function [res]=maillagerectangle(x1,y1,x2,y2,nx,ny,fichier)`
qui maille uniformément un rectangle $[x_1, x_2] \times [y_1, y_2]$ *en* $2 \times n_x \times n_y$ *triangles rectangles isocèles. .*