TP4 : Calculs élémentaires et assemblage

Le présent TP a pour but d'énumérer la procédure de construction de la matrice de rigidité globale A et le vecteur global F, définis par :

$$A_{ij} = a(\varphi_i, \varphi_j), \quad F_i = l(\varphi_i), i, j = 1 \dots N.$$

Correspondant à une approximation interne par éléments finis Lagrange de la formulation variationnelle

Chercher $u_h \in V_h$ tel que $a(u_h, v_h) = l(v_h) \quad \forall v_h \in V_h$.

$(\varphi_i), i = 1 \dots N$ est une base éléments finis P_k-Lagrange ($k = 1$ ou $k = 2$).

Nous allons considérer en guise d'illustration la discrétisation du Laplacien. Dans ce cas :

$$a(u, v) = \int_\Omega \nabla u \cdot \nabla v \; dx, \quad l(v) = \int_\Omega f v \; dx.$$

Le domaine Ω est doté d'un maillage triangulaire $\tau_h = \{T_k\}_{1 \le k \le nT}$.

Exercice - 1 *Matrices et second membres élémentaires*

On considère un triangle $T_k = (S_1, S_2, S_3)$, dont la matrice des coordonnées géométriques est X_k.

Rappel : On peut encore écrire $a(u, v) = \sum_{k=1}^{nT} a^k(u, v)$. Dans l'expression $a^k(u, v)$, u et v désignent les restrictions à T_k, qui de ce fait conduisent à l'écriture équivalente $a^k(u, v) \equiv a^k(u_{|_{T_k}}, v_{|_{T_k}})$. On peut écrire ces restrictions à l'aide des fonctions de base locales $(\varphi_i^{T_k})_{i=1\dots nl}$ (dans la suite on écrira tout simplement $(\varphi_i)_{i=1\dots nl}$) : désignant par $[U^k]$ la matrice colonne des coefficients de $u_{|_{T_k}}$ dans cette base locale, on définit la matrice de rigidité locale comme la matrice A^k telle que $[V^k]^t A^k [U^k] = a^k(u_{|_{T_k}}, v_{|_{T_k}})$, où ici et dans la suite V^t désigne la transposée du vecteur ou matrice V.
Si on introduit les matrices $[PH] = [\varphi_1 \; \cdots \; \varphi_{nl}]$, $[DxPH] = \begin{bmatrix} \partial_x\varphi_1 & \cdots & \partial_x\varphi_{nl} \\ \partial_y\varphi_1 & \cdots & \partial_y\varphi_{nl} \end{bmatrix}$,

alors $u_{|_{T_k}} = [PH][U^k]$, et $A^k = \int_{T_k} [DxPH]^t [DxPH] \; dxdy$, $F^k = \int_{T_k} f(x, y)[PH]^t \; dxdy$.

On ramène le calcul de ces intégrales en un calcul d'intégrales sur le triangle de référence $\hat{T}$ à l'aide de la transformation affine F_T (*voir TP 3*). Pour cela on introduit les matrices
$[HatPH] = [\hat{\varphi}_1 \; \cdots \; \hat{\varphi}_{nl}]$, $[DxHatPH] = \begin{bmatrix} \partial_x\hat{\varphi}_1 & \cdots & \partial_x\hat{\varphi}_{nl} \\ \partial_y\hat{\varphi}_1 & \cdots & \partial_y\hat{\varphi}_{nl} \end{bmatrix} \stackrel{\text{def}}{\equiv} [\text{DXF}]^{-1} \begin{bmatrix} \partial_{\hat{x}}\hat{\varphi}_1 & \cdots & \partial_{\hat{x}}\hat{\varphi}_{nl} \\ \partial_{\hat{y}}\hat{\varphi}_1 & \cdots & \partial_{\hat{y}}\hat{\varphi}_{nl} \end{bmatrix}$,
où $[\text{DXF}] = \begin{bmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} X_k$, (voir TP 3), désigne la matrice jacobienne de la transformation.
Et alors

$$A^k = \int_{\hat{T}} [DxHatPH]^t [DxHatPH] \; J(\hat{x}, \hat{y}) \; d\hat{x}d\hat{y}, \quad F^k = \int_{\hat{T}} (f \circ F_T)(\hat{x}, \hat{y}) [HatPH]^t \; J(\hat{x}, \hat{y}) \; d\hat{x}d\hat{y}.$$

Q-1 : Écrire une fonction `matlab` : $[\mathbf{A_k}, \mathbf{F_k}] = \mathbf{elmatLaplace}(X_k, polyorder)$
qui calcule la matrice de rigidité locale et le second membre local sur un triangle T^k dont les coordonnées des sommets sont données pas X_k.

(*On pourra recourir aux formules de quadrature (voir TP 3). Le script ci-dessous construit la matrice de masse locale* $M_{i,j} = \int_T \phi_i \phi_j \, dx, 1 \leq i, j \leq ndl$, *où* $ndl = 3$ *ou* 6, *est le nombre local de degrés de liberté*).

script matlab/octave pour la matrice de masse locale

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Evaluate the elementary matrix and  vector
%% INPUTS  :  X array 3 x 2 of  coordinates of the triangle
%%            polyorder : local polonomial order (1 or 2)
%% IMPLICIT INPUT: sencond hand function frhs
%% OUTPUTS :  Ak the elementary matrix
%%            Fk the elementary right-hand side
%% (c) Jean-Baptiste Apoung Kamga (July 2007)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [Ak,Fk] = elematmass(X,polyorder,f)

sizeloc=[]; qforder=[];
if(polyorder==1) sizeloc=3; qforder =3; else sizeloc=6; qforder = 5; end
Ak= zeros(sizeloc,sizeloc); Fk=zeros(sizeloc,1);
[PhQ, dxPhQ, dyPhQ, weight,XYQ] = basis_and_deriv_on_quad(X,polyorder,qforder);
nq=length(weight);
for l = 1:nq
    wl = weight(l); xl=XYQ(l,1); yl=XYQ(l,2);
    HatPh = PhQ(:);
    Ak = Ak + wl *HatPh'*HatPh;
    Fk = Fk + wl * f(xl,yl) * HatPh';
end
```

Exercice - 2 *Assemblage de la matrice globale*

Pour cet exercice le maillage est donné par le fichier `carree.msh` à télécharger sur la page web du cours : `http:www/math.u-psud.fr/~apoung/`. Rubrique : Enseignements>Éléments Finis.

Q-1 : Assemblage sans condition aux limites. Écrire une fonction `Matlab` de prototype

```
function [A,F] = Assemble(M, polyorder)
```

qui assemble la matrice et le second membre du problème du Laplacien discrétisé par éléments finis Lagrange de degré $k = polyorder$. Une entête est donnée par :

script matlab/octave pour l' assemblage de la matrice et du second membre

```
%% INPUTS  :  mesh T
%%            polyorder: order of approximation
%%
%% IMPLICIT INPUTS : Rhs and Dirichlet boundary functions
%%
%% OUTPUTS :  global matrix A
%%            global second hand F
%%
%%  COURS D'ÉLÉMENTS FINIS
%%  M2PRO 2007-2008    Lab AN-EDP Univ d'ORSAY
%% (c) Jean-Baptiste Apoung Kamga (July 2007)

function [A, F]=assemble(T, polyorder)
```

Q-2 : Prise en compte des conditions aux limites. Écrire une fonction `Matlab` de prototype

```
function [A,F] = boundarycond(T, A, F, Dirlabel,polyorder)
```

qui modifie la matrice A et le second membre F, pour prendre en compte les condition aux limites de Dirichlet $u = g$ où g est une fonction donnée dans un fichier `g.m`. Une entête est donnée par :

script matlab/octave pour la prise en compte des conditions aux limites par terme

```
%% This function sets the boundary condition
%%
%% INPUTS  :
%%           T : the mesh
%%           A : the Matrix
%%           F : the second hand
%%           Dirlabel: vector of Dirichlet label
%%           polyorder: order of polynomial
%% IMPLICIT INPUT : frhs (right hand-side) and fdirichlet boundary condition
%% OUTPUTS :  global matrix A
%%            global second hand F
%%
%%  COURS D'ÉLÉMENTS FINIS
%%  M2PRO 2007-2008    Lab AN-EDP Univ d'ORSAY
%% (c) Jean-Baptiste Apoung Kamga (July 2007)

function [A,F] = boundarycond(T, A, F, Dirlabel,polyorder)
```