Création de la librairie d'éléments finis (2)

1 Les éléments finis

Pour la fabrication des éléments finis à proprement parler, nous allons procéder comme dans le TP1. Pour cela, il nous faudra construire des matrices creuses qui permettent, étant donnée une fonction discrète u_h de l'espace discret considéré, de donner, en fonction des valeurs de ses degrés de liberté, ses valeurs, ou celles de ses dérivées aux points d'intégration.

Attention : Il ne faut pas confondre points d'intégration et degrés de liberté. Les points d'intégration, sont, nous l'avons vu, les points permettant d'intégrer sur le maillage des fonctions (presque) quelconques. Ils dépendent de la précision que l'on souhaite pour *l'intégration*. Les degrés de liberté sont les coordonnées d'une fonction dans l'espace discret. Si l'on fait des éléments finis $\mathbb{P}_1$, ceux-ci sont situés aux sommets du maillage (étant données des valeurs aux sommets du maillage, il n'y a qu'une seule façon de les interpoler à l'intérieur des triangles de manière affine).

2 Le triangle de référence

Comme pour les structures d'intégration, on travaille tout d'abord dans le triangle de référence, puis on "transportera" les éléments finis sur les triangles du maillage. On rappelle que les degrés de liberté des éléments finis $\mathbb{P}_1$ sont les 3 valeurs aux 3 sommets, celui de $\mathbb{P}_0$ est l'unique valeur au centre de gravité, et pour $\mathbb{P}_2$ on a les 3 sommets et les 3 milieux des arêtes du triangle (ce dernier cas est relativement plus complexe, nous l'aborderons à la fin de ce TP).

Dans le triangle de référence, il y a trois fonctions de base, associées respectivement aux sommets A, B, C. Ce sont les fonctions affines qui valent 1 au sommet respectif et 0 aux deux autres sommets.

Question 1. Donner l'expression analytique des fonctions de base du triangle de référence pour $\mathbb{P}_0$ et $\mathbb{P}_1$. Construire une fonction `Lagrange2D` qui calcule pour des points $(x_i, y_i)_{1\leq i\leq N}$ quelconques du triangle de référence donnés en argument, la valeur des trois fonctions de base ainsi que celle de leurs dérivées en x et en y en ces N points.

Les arguments d'entrée sont `x, y, ordre` où `x` et `y` sont des tableaux $N \times 1$ et `ordre` est l'ordre de l'élément fini (0, 1), et les arguments de retour sont `phi`, `dxphi` et `dyphi` de taille $N \times N_b$ où N_b est le nombre de fonctions de base, contenant respectivement les valeurs des fonctions de base et de leurs dérivées aux points fournis en entrée. (Cette famille d'éléments finis s'appelle les "éléments finis de Lagrange").

Attention : On veillera à retourner un message d'erreur dans le cas où l'ordre de l'élément fini demandé n'est pas (encore) disponible. Noter que pour les éléments $\mathbb{P}_0$ il n'y a qu'une fonction de base. Pour $\mathbb{P}_1$, il y en a 3 que l'on numérotera dans l'ordre des points A, B, C du triangle de référence ($A = (0,0), B = (1,0), C = (0,1)$).

Question 2. Au passage, écrire la fonction `Lagrange1D` permettant de créer des éléments finis en dimension 1. Les arguments d'entrée seront `ordre, x` et les arguments de sortie `phi, dxphi`.

3 Les éléments finis globaux

Les éléments finis sont définis sur tout un maillage. Pour l'assemblage des matrices d'éléments finis, on aura besoin, comme lors du TP1, des matrices creuses permettant de calculer les valeurs d'une fonction discrète ou de ses dérivées aux points d'intégration du maillage. Il faudra également un tableau permettant de passer du *local* au *global*. Il s'agit, pour chaque triangle du maillage de connaître le numéro global de la fonction de base correspondant au numéro local (1 pour $\mathbb{P}_0$ et de 1 à 3 pour du $\mathbb{P}_1$) de la fonction de base locale définie sur $\hat{K}$. Il s'agit donc d'un tableau à deux dimensions de taille $N_{tri} \times N_b$.

Question 3. Écrire une fonction qui renvoie la structure (resp. le dictionnaire) d'éléments finis `ef`.
Syntaxe : `FEspace(mesh, typeEF, ordre, formule)`

Cette structure (resp. ce dictionnaire) doit contenir la numérotation des degrés de liberté. On utilisera la numérotation suivante :

- Pour $\mathbb{P}_0$ les degrés de liberté sont numérotés comme les triangles. On doit alors construire le tableau `ef.numDdl` (resp. `ef['numDdl']`) comme un tableau de taille $N_{tri} \times 1$ contenant les nombres de 1 à N_{tri}.
- Pour $\mathbb{P}_1$ les degrés de liberté sont numérotés comme les sommets. Pour chaque triangle les numéros des trois fonctions de base sont les numéros des sommets du triangle. On doit alors construire le tableau `ef.numDdl` (resp. `ef['numDdl']`) comme un tableau de taille $N_{tri} \times 3$ contenant les numéros des sommets de chaque triangle (remarquer que l'on connaît déjà ce tableau).

Question 4. Construire aussi les coordonnées des degrés de liberté. On construira le tableau `ef.ddl` (resp. `ef['ddl']`) de taille $N_{ddl} \times 2$ qui donne les coordonnées du point correspondant aux degrés de liberté (le centre de gravité du triangle pour $\mathbb{P}_0$, les sommets pour $\mathbb{P}_1$).

Question 5. Enfin, construire les trois matrices `ef.u`, `ef.dxu`, `ef.dyu` (resp. `ef['u']`, `ef['dxu']`, `ef['dyu']`) donnant respectivement les valeurs de la fonction et de ses dérivées.

Dans la fonction précédente, `mesh` est un maillage, `typeEF` est le type de l'élément fini considéré (pour l'instant on n'acceptera que la chaîne de caractères 'Lagrange'), `ordre` est l'ordre de l'élément fini et `formule` est la formule d'intégration à utiliser. Afin de créer les matrices d'éléments finis, il faudra d'abord appeler la formule d'intégration désirée, puis construire les

fonctions de base aux points d'intégration de l'élément de référence. Ensuite, il y aura deux cas :

- Pour la matrice `ef.u` (resp. `ef['u']`) : Il n'y a pas de calcul de dérivée. Il faudra donc ici transporter la structure *locale* dans une matrice creuse *globale*.

  On utilisera la construction de matrices creuses `sparse` de Matlab et Python dont la syntaxe `B = sparse(i,j,a,m,n)` en Matlab (resp. `B = scipy.sparse.csc_matrix((a, (i,j)), shape=(m,n))` en Python) permet de construire la matrice creuse `B` de taille `m x n` et dont les coefficients non nuls sont donnés dans le vecteur `a`, la ligne et la colonne correspondantes au coefficient étant dans les vecteurs `i` et `j` respectivement. (Autrement dit on aura le coefficient $B_{i(k),j(k)} = a(k)$ pour tout k. Les vecteurs `i,j,a` auront donc la même taille.) Ainsi, on suivra la démarche suivante :

  - On allouera les vecteurs `i,j,a` à la bonne taille ($N_{tri} \times N_b \times N_{int}$ car nous avons N_{int} points d'intégration pour chaque triangle et chaque fonction de base).
  - On fera ensuite 2 boucles imbriquées sur chaque point d'intégration par triangle (de 1 à N_{int}) et chaque fonction de base (de 1 à N_b), et on traitera tous les triangles correspondant de manière vectorielle.
  - On calculera directement N_{tri} coefficients de `i,j,a` d'un seul coup en utilisant la numérotation globale des degrés de liberté.
  - On terminera par la commande `sparse` (resp. scipy.sparse.csc_matrix).

- Pour les matrices `ef.dxu` (resp. `ef['dxu']`) et `ef.dyu` (resp. `ef['dyu']`) c'est plus compliqué. Les dérivées en x et y doivent être calculées en fonction des dérivées des fonctions de base en $\hat{x}$ et $\hat{y}$.

  On a par exemple si $(x, y) = T_K(\hat{x}, \hat{y})$ et $u(x, y) = u(T_K(\hat{x}, \hat{y})) = \hat{u}(\hat{x}, \hat{y})$:

$$\frac{\partial u}{\partial x} = \frac{\partial \hat{u}}{\partial \hat{x}}\frac{\partial \hat{x}}{\partial x} + \frac{\partial \hat{u}}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial x}\,.$$

  Il faudra donc calculer les coefficients de la matrice jacobienne $\dfrac{\partial(\hat{x}, \hat{y})}{\partial(x, y)}$, qui est une matrice 2×2 pour chaque triangle. Pour ce faire, on assemblera 4 tableaux correspondant respectivement à

$$\frac{\partial x}{\partial \hat{x}}, \frac{\partial x}{\partial \hat{y}}, \frac{\partial y}{\partial \hat{x}}, \frac{\partial y}{\partial \hat{y}}$$

  pour chacun des triangles (penser aux coordonnées des points du maillage et à la transformation linéaire du triangle de référence au triangle courant) et on inversera en parallèle sur chacun des triangles la matrice précédente.

Question 6. Vérification. Pour vérifier les matrices précédentes, on peut faire deux choses :

1. Calculer des intégrales de fonctions connues sur un maillage et faire de la convergence en maillage. Pour cela, on maillera un carré $[0, 1]^2$, on évaluera la fonction

$$f(x, y) = \exp(x)$$

   sur chacun des degrés de liberté de façon à construire la fonction approchée f_h. On l'intégrera ensuite en passant sur les points d'intégration grâce à la matrice `ef.u` (resp.

`ef['u']`). Lorsque le maillage est de plus en plus fin, on doit observer que l'intégrale converge vers $e - 1$.

On vérifiera de même `ef.dxu` (resp. `ef['dxu']`) et `ef.dyu` (resp. `ef['dyu']`). Pour la convergence en maillage, on pourra tracer en fonction du pas d'espace h la quantité

$$\int_{[0,1]^2} (f(x,y) - f_h(x,y))^2 \, dx \, dy$$

en calculant l'expression à l'intérieur de l'intégrale aux points d'intégration.

2. Vérifier des formules d'intégration par parties.

$$\int_\Omega f \frac{\partial g}{\partial x} \, dx \, dy = - \int_\Omega g \frac{\partial f}{\partial x} \, dx \, dy$$

lorsque f ou g s'annule sur les bords verticaux du carré.

Question 7. Généraliser toutes les fonctions précédentes aux éléments finis $\mathbb{P}_2$ (attention, c'est délicat, en particulier pour la numérotation).