Initiation à FreeFem++

O. Pantz – F. Nataf 18/01/2016

Introduction

L'objectif de ce TP consiste à résoudre numériquement, par la méthode des éléments finis, divers problèmes variationnels en dimensions 2 et 3 d'espace. A cet effet, on va utiliser le logiciel libre `FreeFem++` développé au Laboratoire Jacques-Louis Lions de Paris 6. Ce dernier permet de résoudre très simplement de nombreux problèmes variationnels.

Une documentation de `FreeFem++` est accessible sur `www.freefem.org`, à l'adresse suivante

`http://www.freefem.org/ff++/ftp/freefem++doc.pdf`

1 Le Laplacien

1.1 Prise en main

- Décompresser le fichier `TP1.zip`, en effectuant l'une des actions suivantes:
  - Double-cliquez sur l'icône du fichier `TP1.zip`
  - Sinon, ouvrir une fenêtre console ("terminal"), se déplacer dans le dossier de téléchargement, et taper la ligne de commande suivante:

    ```
    > unzip TP1.zip
    ```

    A ce stade, vous devez avoir un nouveau dossier nommé `TP1`.
- Ouvrir une fenêtre console ("terminal"), se déplacer dans le dossier `TP1` en tapant la commande

  ```
  > cd TP1
  ```

  Puis lancer l'exécution du script `chaleur.edp` par `FreeFem++` en tapant la ligne suivante:

  ```
  > FreeFem++ chaleur.edp
  ```

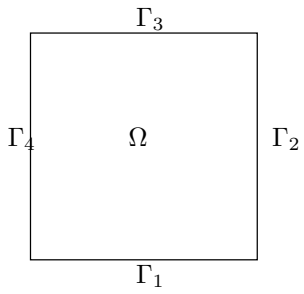
  Vous devez voir apparaître une fenêtre graphique affichant un maillage assez raffiné.
- Cette fenêtre étant au premier plan, le programme est en attente. Appuyiez sur la touche `Return` (↩) pour avancer.
- Vous devez voir apparaître une courbe 3D en couleurs représentant la solution d'un problème de Poisson.
- Appuyez sur la touche `Escape` (*esc*) pour terminer le programme.

1.2 Explication du code source `chaleur.edp`

On se propose de résoudre numériquement le problème consistant à déterminer u tel que

$$\begin{cases} -\Delta u = f & \text{dans } \Omega \\ u = 0 & \text{sur } \partial\Omega \end{cases} \tag{1}$$

dans le carré $\Omega =]0,1[^2$:



Le script `FreeFem++` ci-dessous résout précisément ce problème en dix lignes seulement (sans comptabiliser les commentaires) ! On utilise le rouge pour les commandes `FreeFem++`.

```
//Nombre de mailles suivant x et y
int Nbnoeuds=10;
```

Le texte suivi de `//` constitue des commentaires ignorés par `FreeFem++`. Chaque nouvelle variable introduite doit être précédée de son type (ici `int`, c'est à dire un entier).

```
//Definition du maillage
mesh Th=square(Nbnoeuds,Nbnoeuds,[x,y]);
```

La fonction `square` retourne un maillage structuré. Les deux premiers arguments fixent le nombre de noeuds suivant x et y respectivement. Le troisième argument est une paramétrisation de Ω pour x et y variant entre 0 et 1 (dans notre cas, c'est l'identité). Les cotés du carré sont numérotés de 1 à 4, dans le sens trigonométrique, le coté inférieur portant le label 1 (voir la figure).

```
//Fonction de x et de y
func f=x*y;

//Definition de l'espace des elements finis P1 associe
//au maillage Th
fespace Vh(Th,P1);

//uh et vh sont des elements de Vh
Vh uh,vh;
```

Les fonctions u_h et v_h appartiennent à l'espace V_h des fonctions P_1. Notons que si l'on souhaite utiliser des éléments finis P_2 et non P_1, il suffit de remplacer `P1` par `P2` dans la définition de `Vh` .

```
//Definition du probleme variationnel
problem chaleur(uh,vh,solver=LU)=
        int2d(Th)(dx(uh)*dx(vh)+dy(uh)*dy(vh))
        -int2d(Th)(f*vh)
        +on(1,2,3,4,uh=0)
;
```

La fonction `problem` permet de définir un problème variationnel, que nous dénommons ici `chaleur` . Ici, on définit le problème consistant à déterminer

$$u_h \in V_h^0 = \{w_h \in V_h \ : \ w_h = 0, \text{ sur } \Gamma_1 \cup \Gamma_2 \cup \Gamma_3 \cup \Gamma_4\}$$

tel que

$$\int_\Omega \nabla u_h . \nabla v_h dx - \int_\Omega f v_h dx = 0,$$

pour tout $v_h \in V_h^0$. Notons que cette commande définit le problème mais ne le résout pas. Le problème (1) est résolu numériquement par la commande

```
//Resolution du probleme
chaleur;

//On affiche le resultat
plot(uh,wait=1);
```

Remarque- La méthode de résolution utilisée pour résoudre le système est la factorisation LU.

Recopier ce script (c'est à dire les parties rouges) à l'aide de votre éditeur de texte favori et sauver le dans un fichier (sous le nom `chaleur.edp` par exemple). **Remarque-** `acroread` est muni d'une fonction permettant de faire des copier-coller.

Pour exécuter le script sous `FreeFem++`, il suffit de taper la commande shell

```
FreeFem++ chaleur.edp
```

Le résultat du calcul s'affiche dans une fenêtre graphique. Pour reprendre la main dans le shell, cliquer dans cette dernière.

1.3 Quelques variantes

On peut aisément modifier le script initial pour résoudre des problèmes variationnels du même type, avec conditions de Neumann, ou de Fourier par exemple.

Ainsi, le problème suivant

$$\begin{cases} -\Delta u + u = f & \text{dans } \Omega \\ \frac{\partial u}{\partial n} = 0 & \text{sur } \Gamma_1 \\ u = 0 & \text{sur } \Gamma_2 \\ \frac{\partial u}{\partial n} + \alpha u = g & \text{sur } \Gamma_3 \cup \Gamma_4 \end{cases} \tag{2}$$

où f et g sont des fonctions quelconques que vous choisirez (non toutes deux nulles tout de même), et α est un réel strictement positif.

Dans la formulation variationnelle du problème (2) apparaît une intégrale sur le bord de Ω. Une intégrale sur bord $\Gamma_3 \cup \Gamma_4$ se définit sous `FreeFem++` par la commande `int1d(Th,3,4)(`expression à intégrer `);`
Ainsi, pour notre problème, la formulation variationnelle consiste à déterminer

$$u_h \in W_h = \{w_h \in V_h \ : \ w_h = 0 \text{ sur } \Gamma_2\}$$

tel que

$$\int_\Omega \nabla u_h . \nabla v_h dx + \int_{\Gamma_3 \cup \Gamma_4} \alpha u_h v_h - \int_{\Gamma_3 \cup \Gamma_4} g v_h - \int_\Omega f v_h dx = 0,$$

pour tout $v_h \in W_h^0$.
Ce problème est défini par les commandes du fichier `chaleurrobin.edp`.

2 FreeFem++ et l'algèbre linéaire

Le programme précédent `chaleurrobin.edp` permet de calculer la solution éléments finis du problème. Le programme ci-dessous `chaleurmatrice.edp` montre comment récupérer la matrice de rigidité associée à la formulation variationnelle ainsi que le vecteur associé au second membre du problème variationnelle. Puis, la solution est calculée par la résolution d'un système linéaire. On pourra vérifier que les solutions calculées par les programmes `chaleurrobin.edp` et `chaleurmatrice.edp` sont les mêmes. L'intérêt de passer par le système matriciel est de pouvoir mettre en oeuvre des méthodes de décomposition de domaine.

Le mot-clé varf permet de définir une formulation variationnelle.

```
varf chaleur(uh,vh)=
int2d(Th)(dx(uh)*dx(vh)+dy(uh)*dy(vh))
+int1d(Th,3,4)(alpha*uh*vh)
-int1d(Th,3,4)(g*vh)
-int2d(Th)(f*vh)
+on(2,uh=0)
;
//Recuperation de la matrice et du second membre
  matrix Aglobal;// matrice creuse
  Vh rhsglobal;// rhsglobal est une fonction element fini de l'espace Vh
```

```
                // rhsglobal[] est le vecteur de ses degres de liberte
  Aglobal = chaleur(Vh,Vh,solver=UMFPACK);// matrice de rigidité du problème
                                          //solveur direct de la librairie UMFPACK
```

La fonction $rhsglobal_h$ est une fonction éléments finis et le vecteur des valeurs de ses degrés de liberté est noté $rhsglobal_h[]$.

```
  rhsglobal[] = chaleur(0,Vh);//second membre
```

La résolution se fait par une méthode de factorisation de type Gauss (LU).

```
//Resolution du probleme par une méthode directe de type LU creux
uh[] = Aglobal^-1*rhsglobal[];
```