MASTER 2 MPM **Université Paris XIII**
Année 2013-2014 **Institut Galilée**

LANGAGE C AVANCÉ POUR LA SIMULATION NUMÉRIQUE

EXAMEN DU 9 JANVIER 2014

Durée : 3h

Aucun document autorisé
Aucun appareil electronique autorisé

EXERCICE 1 : (13 points)

L'objectif de cet exercice est de calculer numériquement l'intégrale d'une fonction à deux variables sur un rectangle maillé par des triangles.

La structure de données associée aux maillages est imposée : elle est constituée des champs $\mathrm{n_q}$, $\mathrm{n_{me}}$, Q, me et areas correspondant à

nom	type	dimension	descriptif
$\mathrm{n_q}$	entier	1	nombre total de nœuds du maillage, (*number of nodes*)
$\mathrm{n_{me}}$	entier	1	nombre de triangles, (*number of mesh elements*)
Q	réels	$2 \times \mathrm{n_q}$	$Q[\alpha][l-1]$ est la α-ème coordonnée du l-ème nœud, $\alpha \in \{0,1\}$ et $l \in \{1,\ldots,\mathrm{n_q}\}$. Le l-ème nœud sera aussi noté q^l avec $\mathrm{q}^l_x = Q[0][l-1]$ et $\mathrm{q}^l_y = Q[1][l-1]$, (*nodes*)
me	entier	$3 \times \mathrm{n_{me}}$	$\mathrm{me}[\beta][k-1]$ est l'indice de stockage, dans le tableau Q, du β-ème sommet du k-ème triangle , $\beta \in \{0,1,2\}$ et $k \in \{1,\ldots,\mathrm{n_{me}}\}$. (*mesh elements*)
areas	réels	$\mathrm{n_{me}}$	$\mathrm{areas}[k-1]$ aire du k-ème triangle, $k \in \{1,\ldots,\mathrm{n_{me}}\}$.

Q. 1 *Définir les types* `struct mesh` *et* `Mesh` *correspondant à la structure décrite au préalable.* ▪

On note $(x_i)_{i\in[\![0,m]\!]}$ et $(y_j)_{j\in[\![0,n]\!]}$ les discrétisations **régulières** des intervalles $[a,b]$ et $[c,d]$ avec respectivement $m+1$ et $n+1$ points. On a $x_0 = a < x_1 < \ldots x_m = b$, $y_0 = c < y_1 < \ldots < y_n = d$. Un maillage **régulier** du rectangle $\Omega = [a,b] \times [c,d]$ est défini à partir de ces discrétisations régulières et il est représenté en figure 1. Les **nœuds** du maillage, noté q^l, sont représentés en figure 1 par le symbole •. Le maillage est constitué de triangles *inférieurs* (sommets (x_i,y_j), (x_{i+1},y_j), (x_{i+1},y_{j+1})) et *supérieurs* (sommets (x_i,y_j), (x_{i+1},y_{j+1}), (x_i,y_{j+1})). Soit $k \in \{1,\ldots,\mathrm{n_{me}}\}$, on note T_k le k-ème triangle (ou triangle numéro k) du maillage et on a $\Omega = \bigcup_{k=1}^{\mathrm{n_{me}}} T_k$.

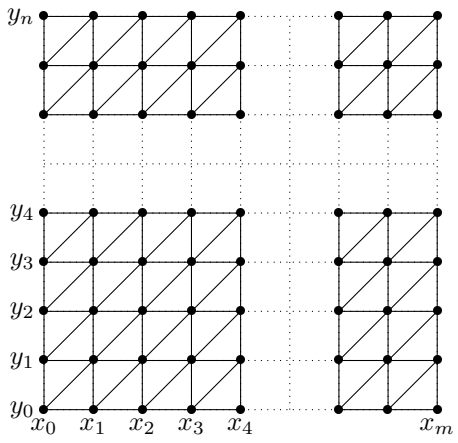

FIGURE 1 – Maillage **régulier** d'un rectangle

Q. 2 *1. Expliquer comment déterminer les x_i et y_j.*

2. *Déterminer le nombre de nœuds distincts du maillage.*
3. *Déterminer le nombre de triangles distincts (inférieurs et supérieurs) du maillage.* ▪

La numérotation des nœuds pour ce type de maillage commence par le nœud en bas à gauche pour finir en haut à droite (le parcourt s'effectue de gauche à droite et de bas en haut). On défini les carrés $\mathcal{C}_{i,j} = [x_i, x_{i+1}] \times [y_j, y_{j+1}]$. Pour numéroter les triangles, on parcourt les carrés en partant du bas à gauche pour finir en haut à droite sachant que pour chaque carré on numérote le triangle inférieur puis le triangle supérieur. Pour illustrer cela, on donne un exemple de numérotation des nœuds et des triangles en figure 2.

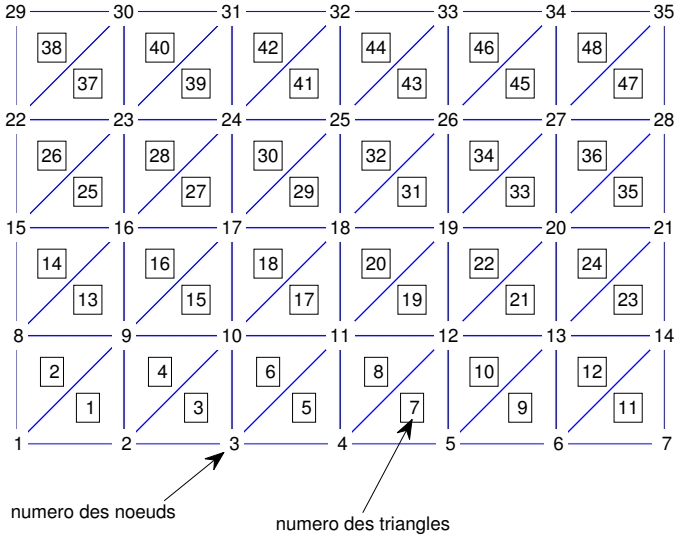

FIGURE 2 – Numérotation des nœuds et triangles ($m = 6$, $n = 4$, version algorithmique)

A titre indicatif on donne les versions algoritmiques des tableaux Q et me associés à la figure 2 :

$$Q = \begin{pmatrix} q_x^1 & q_x^2 & q_x^3 & \dots & q_x^{\mathrm{n_q}} \\ q_y^1 & q_y^2 & q_y^3 & \dots & q_y^{\mathrm{n_q}} \end{pmatrix}, \quad \mathrm{me} = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 & \dots \\ 2 & 9 & 3 & 10 & 4 & 11 & \dots \\ 9 & 8 & 10 & 9 & 11 & 10 & \dots \end{pmatrix}$$

avec $q_x^1 = x_0$, $q_x^2 = x_1$, $q_x^3 = x_2$, $q_y^1 = q_y^2 = q_y^3 = y_0$.

Q. 3 *Soient $i \in [\![0, m]\!]$ et $j \in [\![0, n]\!]$. Déterminer $l \in [\![1, \mathrm{n_q}]\!]$ tel que $\mathrm{q}^l = (x_i, y_j)$.* ▪

Q. 4 *Connaissant a, b, c, d, m et n, la fonction* `RectMesh` *va permettre d'initialiser les champs* $\mathrm{n_q}$, $\mathrm{n_{me}}$, Q *et* me *d'une variable de type* `Mesh`*. Le champ* areas *ne sera pas utilisé ici.*

1. *Quelles sont les «données» de la fonction* `RectMesh` *?*
2. *Proposer deux prototypes non trivialement similaires pour cette fonction.*
3. *Choisir un prototype et écrire la fonction* `RectMesh`*.* ▪

On rappelle que pour un triangle T de sommets q^1, q^2 et q^3, l'aire du triangle T notée $|T|$ est donnée par

$$|T| = \frac{1}{2} \left| \det \begin{pmatrix} \mathrm{q}_x^2 - \mathrm{q}_x^1 & \mathrm{q}_x^3 - \mathrm{q}_x^1 \\ \mathrm{q}_y^2 - \mathrm{q}_y^1 & \mathrm{q}_y^3 - \mathrm{q}_y^1 \end{pmatrix} \right|$$

Q. 5 *Ecrire la fonction* `InitMeshAreas` *permettant d'initialiser le champs* areas *d'une variable de type* `Mesh` *(les autres champs étant déjà initialisés).* ▪

Q. 6
1. *Ecrire la fonction* `PrintElement` *permettant d'afficher pour un triangle (donné par son indice dans le tableau* me*) les composantes de ses trois sommets ainsi que son aire.*
2. *Ecrire la fonction* `PrintMesh` *permettant d'afficher pour chaque triangle d'un maillage les composantes de ses trois sommets ainsi que son aire.* ▪

Q. 7 *Ecrire la fonction* `FreeMesh` *permettant de libérer la mémoire utilisée par une variable de type* `Mesh` *sachant que les champs* $\mathrm{n_q}$ *et* $\mathrm{n_{me}}$ *seront mis à zéro.* ▪

On suppose par la suite que l'ensemble des fonctions précédentes (et uniquement les fonctions) ont été écrites dans le fichier `mesh.c`.

Q. 8 *1. Ecrire le fichier header* `mesh.h` *associé au fichier* `mesh.c`.

2. Ecrire un programme principal permettant de
- *Créer une variable de type* `Mesh` *correspondant à un maillage du rectangle* $[-1,1]\times[-2,2]$ *avec* m *et* n *demandés à l'utilisateur,*
- *Initialiser le champ* `areas` *de cette variable,*
- *Afficher pour chaque triangle du maillage les composantes de ses trois sommets ainsi que son aire.*
- *Afficher les composantes des trois sommets et l'aire du triangle inférieur en bas à droite.* ▪

Soit f une fonction intégrable sur le rectangle Ω. Par propriétés du maillage on a

$$\int_\Omega f(\mathrm{q})d\mathrm{q} = \sum_{k=1}^{\mathrm{n_{me}}} \int_{T_k} f(\mathrm{q})d\mathrm{q}. \tag{1.1}$$

Pour un triangle T de sommets q^1, q^2 et q^3, on approchera l'intégrale de f sur ce triangle par

$$\int_T f(\mathrm{q})d\mathrm{q} \approx \frac{|T|}{3}(f(\mathrm{q}^1)+f(\mathrm{q}^2)+f(\mathrm{q}^3)d\mathrm{q}. \tag{1.2}$$

Q. 9 *Ecrire la fonction* `QuadMeshP1` *permettant de calculer une approximation de l'intégrale d'un fonction donnée sur* Ω *en utilisant les formules (1.1) et (1.2).* ▪

Q. 10 *Ecrire la fonction* `QuadMeshP1` *permettant de calculer une approximation de l'intégrale d'un fonction donnée (donc passée en argument à la fonction) sur* Ω *en utilisant les formules (1.1) et (1.2).* ▪

On suppose par la suite que l'ensemble des fonctions précédentes (et uniquement les fonctions) ont été écrites dans le fichier `quadmesh.c`

Q. 11 *1. Ecrire le fichier header* `quadmesh.h` *associé au fichier* `quadmesh.c`.

2. Ecrire un programme principal permettant de calculer une approximation de

$$\int_{-1}^{1}\int_{-1}^{1} cos(x+y)dxdy$$

sachant que l'on demandera à l'utilisateur de donner les valeurs de m *et* n *permettant de génerer le maillage.* ▪

Exercice 2 : (6 points)

On rappelle qu'un nombre copmlexe $z \in \mathbb{C}$ s'écrit sous la forme $z = a + \imath b$ où a et b sont les nombres réels appelés respectivement partie réelle et partie imaginaire.
Il n'existe pas en Langage C ANSI, de type `complex` permettant de manipuler les nombres complexes. Dans cet exercice, on va créer un tel type à l'aide d'une structure et écrire quelques fonctions permettant de manipuler les nombres complexes.

Q. 1 *1. Ecrire explicitement les types* `struct complex` *et* `Complex` *correspondant à une structure contenant deux champs de type* `double` *nommés* `Re` *(partie réelle) et* `Im` *(partie imaginaire). Un nombre complexe sera alors représenté en langage C à l'aide d'une variable de type* `Complex`

2. En trois instructions maximum, écrire la déclaration d'une variable z *de type* `Complex` *et son initialisation à* $1+i$. ▪

Q. 2 *1. Ecrire une fonction* `InitComplex` *permettant, à partir de deux* `double` a *et* b, *d'initialiser un «nombre complexe» à* $a+ib$.

2. Ecrire une fonction `ReadComplex` *permettant une saisie au clavier d'un «nombre complexe».*

3. Ecrire une fonction `PrintComplex` *permettant d'afficher un «nombre complexe».*

4. Ecrire une fonction `axpbyComplex` *permettant de calculer le nombre complexes* $z \leftarrow \lambda z_1 + \mu z_2$ *avec* z_1, z_2 *deux complexes et* λ, μ *deux réels.*

5. *Ecrire une fonction* `modulusComplex` *permettant de calculer le module d'un nombre complexe.*
6. *Ecrire une fonction* `PermutComplex` *permettant de permutter les valeurs de deux variables de type* `Complex`. ▪

Dans le plan complexe, un triangle T est représenté par ses trois sommets z_1, z_2 et z_3

Q. 3 1. *Ecrire une fonction* `areaComplexT` *permettant de calculer l'aire du triangle T.*
2. *Ecrire une fonction* `MaxLengthComplexT` *permettant de calculer la plus grande des longueurs des arêtes du triangle T.* ▪

Q. 4 *Soient A, B et C trois réels, écrire une fonction* `solve` *permettant de déterminer z_1 et z_2 les deux racines complexes de $Az^2 + Bz + C = 0$.*
N.B. z_1 et z_2 peuvent être identiques dans le cas d'une racine double et les racines réelles seront aussi représentées à l'aide de nombres complexes. ▪

On suppose que les fonctions précédentes sont écrites dans le fichier `Complex.c`.

Q. 5 1. *Ecrire le fichier header* `Complex.h` *associé au fichier* `Complex.c`.
2. *Ecrire le fichier* `PPcomplex.c`*, utilisant le fichier header* `Complex.h` *et contenant le programme principal effectuant les opérations suivantes en utilisant au mieux les fonctions déjà écrites*
   - *Saisir au clavier des nombres réels A, B et C.*
   - *Déterminer z_1 et z_2 les nombres complexes solution de $Az^2 + Bz + C = 0$.*
   - *Permutter z_1 et z_2.*
   - *Calculer la somme de ces deux nombres.*
   - *Afficher cette somme.*
3. *Donner la ou les commandes permettant de créer, avec le compilateur* `gcc`*, le programme exécutable.* ▪

Exercice 3 : (3 points)

Listing 1 – exo8a.c

```
1  #include <stdio.h>
2
3  int h1(int k, int *pj);
4  int h2(int *pj, int k);
5
6  int main(){
7    int k=1,i=2,j=3;
8
9    j=h1(k,&i);
10   printf("main: k=%2d, i=%2d, j=%2d\n",k,i,j);
11   i=h2(&k,j);
12   printf("main: k=%2d, i=%2d, j=%2d\n",k,i,j);
13   return 1;
14 }
15
16 int h1(int k, int *pj){
17   k++;
18   *pj+=1;
19   return *pj*k;
20 }
21
22 int h2(int *pj, int k){
23   k++;
24   (*pj)*=2;
25   return k+*pj;
26 }
```

Q. 1 *A l'aide de diagrammes de représentations mémoires, expliquer le déroulement du programme et donner les resultats affichés.* ▪