Rapport du T.P. :
5A. Mathématiques appliquées et Modélisation 2017-2018 (option C.O. et B.D.P.)

Galerkine Discontinu :
Simulation des méthodes vues en cours :

Travail effectué par : ABOUADALLAH Mohamed Anwar

École Polytechnique Lyon
15 Boulevard Latarjet 69622 Villeurbanne

"Loin d'être l'exercice ingrat ou vain que l'on imagine, les mathématiques pourraient bien être le chemin le plus court pour la vraie vie, laquelle, quand elle existe, se signale par un incomparable bonheur." Alain Badiou, 4ème de couverture de son livre Éloge des mathématiques.

Table des matières

1 Introduction :

1.1 Motivations :

Les méthodes de différences finies et de volumes finies sont des méthodes ponctuelles, qui ne nécessitent pas pour leur application la définition d'un espace. A contrario, les méthodes d'éléments finis et de Galerkin discontinues sont des méthodes variationnelles.
Ainsi les méthodes de Galerkine discontinues présentent plusieurs avantages par rapport aux autres méthodes, parmi eux on peut citer :

- Orthogonalité de l'erreur
- La qualité de l'approximation
- Forme matricielle du problème

1.2 Présentation de notre modèle :

Nous commençons notre T.P. par présenter notre équation à résoudre puis les méthodes qu'on va utiliser. Ainsi tout d'abord notre équation à résoudre, il s'agit de équation d'advection 1D qui est un cas particulier de l'équation du transport 1D donnée par :

$$\boxed{\partial_t u + a\partial_x u = 0, \forall a > 0}$$

NB : Nous allons utiliser les termes h, T et a qui seront explicités dans une section ultérieur du rapport.

Nous rappelons que $\partial_t u$ est notre C.I. et que de ce terme découle la matrice suivante :

$$M = \frac{1}{h}\begin{pmatrix} 2 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 1 \\ \vdots & 2 & 1 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \vdots \\ \vdots & 1 & 2 & \cdots & \cdots & \cdots & \cdots & \vdots & \cdots & \vdots \\ \vdots & & & \ddots & & & & & & \vdots \\ \vdots & & & & \ddots & & & & & \vdots \\ \vdots & & & & & & \ddots & & & \vdots \\ \vdots & & & & & & & \ddots & & \vdots \\ \vdots & & & & & & & 2 & 1 & \vdots \\ \vdots & & & & & & & 1 & 2 & \vdots \\ 1 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 2 \end{pmatrix}$$

Ensuite en fait passer le second terme "$a\partial_x u$" sous la forme matricielle ce qui nous donne :

$$G_d = \frac{a}{2}\begin{pmatrix} 1 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & -T \\ -2 & 1 & 1 & \dots & \dots & \dots & \dots & \dots & \dots & \vdots \\ \vdots & -1 & 1 & \dots & \dots & \dots & \dots & \vdots & \dots & \vdots \\ \vdots & & -2 & 1 & 1 & & & & & \vdots \\ \vdots & & & -1 & 1 & & & & & \vdots \\ \vdots & & & & & & \ddots & & & \vdots \\ \vdots & & & & & & & \ddots & & \vdots \\ \vdots & & & & & & -2 & 1 & 1 & \vdots \\ \vdots & & & & & & & -1 & 1 & \vdots \\ 1 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & -2 & 1 \end{pmatrix}$$

2 Partie théorique :

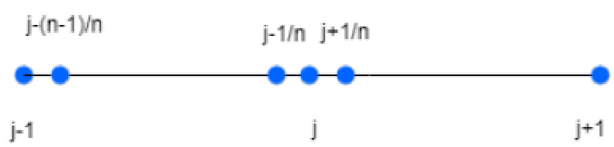
Durant cette section, nous allons développer la théorie à l'origine de notre démarche, pour cela nous allons commencer par expliquer les méthodes :

- Á ce dessein nous expliciterons la formulation variationnelle de notre problème.
- Ensuite, on discrétise notre problème.
- S'ensuivra le calcul de G.
- Enfin nous allons définir nos cas tests.

2.1 Formulation variationnelle :

2.1.1 Espace de discrétisation :

On se donne un espace de discrétisation suivant :
Soit un maillage τ_h de m points, avec périodicité c'est à dire que le nœud 1 est équivalent au nœud m + 1 :
On appel $e_j = [x_j, x_{j+1}]$



2.1.2 Formulation variationnelle :

On commence par multiplier par une fonction test puis on intègre sur les nœuds $e_j, j = 1, \ldots, n$ ce qui nous donne la formulation variationnelle suivante pour notre problème :

$$\boxed{\sum_{e_j} \int_{e_j} \partial_t u \Phi dx - a \int_{e_j} u \partial_x \Phi dx + a\hat{u}\Phi_{x_j}^{x_{j+1}}}$$

Ici on considère un flux décentré donc on ne prend que les valeurs à gauche $\Rightarrow$ $\boxed{\hat{u} = \lim_{\substack{\varepsilon \to 0 \\ \varepsilon > 0}} u(x_j - \varepsilon)}$

2.2 Discrétisation de l'espace :

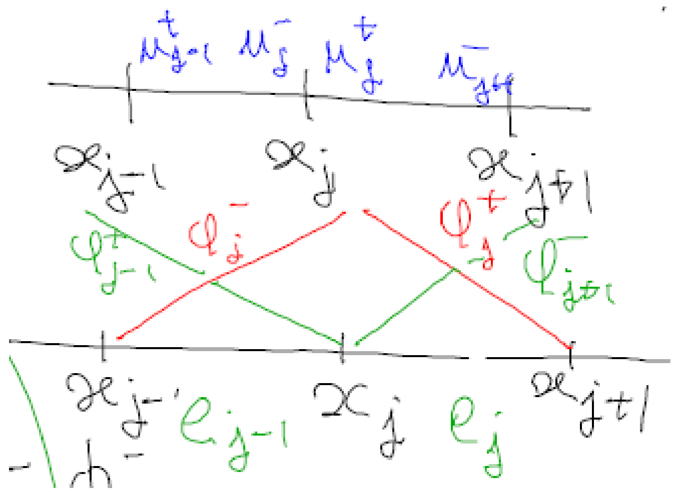
On fait une discrétisation de l'espace et on considère un Euler explicite. Cela nous permet d'avoir :

$$\boxed{\frac{M(u^{n+1} - u^n)}{\Delta t} = Gu^n}$$

avec $u = \begin{pmatrix} u_1^- & u_1^+ & u_2^- & u_2^+ & \ldots & u_n^- & u_n^+ \end{pmatrix}^T$
Ceci nous donne :

$$\boxed{u^{n+1} = u^n + \Delta t M^{-1} G u^n}$$

2.3 Calcul de G dans le cas décentré :



On commence notre calcul de cette manière :
On réécrit notre formulation variationnelle telle cela :

$$\sum_{e_j} \int_{e_j} u\partial_t \Phi dx \text{ -} a \int_{e_j} u\partial_x \Phi dx + a\hat{u}\Phi_{x_j}^{x_{j+1}}$$

- Premièrement, sur x_j, il y a deux contribution, celle de e_{j-} ainsi que celle de e_{j+} ainsi notre fonction Φ_j^- correspond à notre fonction teste.

$$\Rightarrow$$

  Ainsi le terme u peut s'écrire : $u = u_{j-1}^+ \Phi_{j-1}^+ + u_j^- \Phi_j^-$
- Ensuite, le terme en rouge correspond aux noeuds x_j et peut se réécrire de cette manière :

$$\boxed{\sum_{e_j} \int_{e_j} (u_{j-1}^+ \Phi_{j-1}^+ + u_j^- \Phi_j^-) \partial_t \Phi_j^- dx}$$ [1]

- Enfin, le terme vert corresponds à :

$$\boxed{a\hat{u}\Phi_{x_j}^{x_{j+1}} = au_{j-}}$$

Maintenant on va utiliser les deux propriétés suivantes à dessein d'exprimer notre G :

Proposition 1 *La contribution en* e_{j-1} *est :* $\frac{a}{2}(u_j^- - u_{j-1}^+)$

Cette proposition nous conduit vers une autre qui permet de calculer la contribution de e_j au noeud x_j. Elle est donnée par :

$$\boxed{\frac{a}{2}(-2u_j^- + u_{j+1}^- + u_j^+)}$$

$$\boxed{G = -\frac{a}{h} u_{j-1}^+ \int_{e_j} \Phi_{j-1}^+ + \frac{a}{h} u_j^- \int_{e_j} \Phi_j^-) \partial_t \Phi_j^- dx + au_{j-}}$$

1. Avec $\partial_t \Phi_j^- = \frac{1}{h}$

2.4 Différents cas tests pris en compte :

Dans le cadre de ce t.p. nous avons mobilisé cinq cas tests.

- La fonction : $U_0 = 1, \forall x \in [0,5]$
- $u_0(c) = \exp(-10(x-1)^2)$
- $$u_0 = \begin{cases} 1 & \text{si } x < \frac{1.}{3} \\ 0.5 & \text{si } \frac{1.}{3} \leq x < \frac{2.}{3} \\ 1 & \text{si } x > \frac{2.}{3} \end{cases}$$
- $$u_0 = \begin{cases} 1 & \text{si } x < \frac{1.}{2} \\ 1-5(x-0.5) & \text{si } \frac{1.}{2} \leq x < 0.6 \\ 0.5 & \text{si } x > 0.6 \end{cases}$$
- $u_0(x) = \sin(\frac{2\pi x}{L-h})$

3 Implémentation en matlab :

3.1 Présentation du Logiciel utilisé :

"MATLAB (« matrix laboratory ») est un langage de programmation de quatrième génération émulé par un environnement de développement du même nom ; il est utilisé à des fins de calcul numérique. Développé par la société The MathWorks, MATLAB permet de manipuler des matrices, d'afficher des courbes et des données, de mettre en œuvre des algorithmes." [2]

3.2 Implémentation :

Durant ce t.p., nous allons effectuer une démarche simple et très efficace, similaire à celle qu'on a utilisé durant les t.p. de modélisation. Dans la suite de cette sous-section, nous allons mettre les étapes de notre démarche ainsi que les codes utilisées :

- Nous commencerons notre implémentation par la définition de nos constantes :

```
c = 1;
m = 100;
T = 10;
h = 1/m;
a=1;
cfl=0.1;
dt=h*cfl/a;
lambda=1;
```

- Ensuite nous définissons les différentes fonction testes :
  - Second cas test : $u_0(c) = \exp(-10(x-1)^2)$

```
function [ out] =condition_init2(x)

out = exp(-10 .*(x-1).^2);

end
```

2. Source : Wikipédia.

♠ Troisième cas test :

$$u_0 = \begin{cases} 1 & \text{si } x < \frac{1}{3} \\ 0.5 & \text{si } \frac{1}{3} \leq x < \frac{2}{3} \\ 1 & \text{si } x > \frac{2}{3} \end{cases}$$

```
function [ out] =test_3(x,n)
 out=zeros(n,1)
   for i=1:n
     if(x(i)<1/3.)
       out(i)=2.;
     else
       if x(i)<2/3.
         out(i)=1/2.;
       else
         out(i)=1.;
       end
     end
   end
end
```

♠

$$u_0 = \begin{cases} 1 & \text{si } x < \frac{1}{2} \\ 1-5(x-0.5) & \text{si } \frac{1}{2} \leq x < 0.6 \\ 0.5 & \text{si } x > 0.6 \end{cases}$$

```
function [ out] =condition_init_4(x,n)
out=zeros(n,1)
   for i=1:n
    if(x(i)<1/2)
      out(i)=1;
    else
      if x(i)<0.6
         out(i)=1-5*(i-1/2);
      else
         out(i)=1/2;
      end
    end
   end
end
```

3.2.1 Première résolution :

♣ Dans un premier temps, nous allons commencer par faire une discrétisation via une méthode d'éléments finis. Cette méthode consiste à écrire pour chaque nœud : m nœuds (bords des intervalles), m(n-1) nœuds intérieurs : mn nœuds donc mn équations discrètes.

```
function [M, G] = MatDiscretisationEspaceFE (h, c, m)

  M = sparse(diag(diag(ones(m) * (2/3) )) + diag(diag(ones(m-1) * (1/6)) , -1)
     + diag(diag(ones(m-1) * (1/6)) , 1));
```

```
  M(1 , m) = 1/6;
  M(m, 1) = 1/6;


  G = sparse(diag(-diag(ones(m-1) ) , -1) + diag(diag(ones(m-1) ) , 1));
  G(1 , m) = -1;
  G(m, 1) = 1;
  G = G*c*1/(2*h);

end
```

3.2.2 Discrétisation dans l'espace :

- Dans cette sous-sous section nous allons subdiviser notre script en plusieurs sous scriptes afin de mieux l'expliquer.
  - Premièrement discrétisation via galerkine continue, elle corresponds à a méthode des éléments finis décrite auparavant.
  - Galerkine Discontinue

```
G = zeros(m,m);
 I = eye((m/2)-1);
GradD = a/2 * [1,1;-1,-1]; % Grad decentre; Avec conditions aux limites,
    et les morceaux de flux
G(2:m-1,2:m-1) = kron(I, GradD);
G(1,1) = a/2 * (2*lambda-1);
G(1,m) = a/2 * (-1);
G(m,m) = a/2 * 1;
G(m,1) = a/2 * 1;
```

  - Rajout du flux calculé précédemment :

```
for i = 3:2:m-1
    G(i,i)= G(i,i) + a;
end

for i = 2:2:m
    G(i,i-1) = G(i,i-1) - a;
end
```

- Ensuite nous appliquons notre méthode sur notre problème.

```
M_Bloc=[h/3,h/6;h/6,h/3];
M=kron(eye(m),M_Bloc);

G = zeros(2*m,2*m);
G=speye(2*m)*(2*lambda-1);


for i = 2:2:(2*m)-1
```

```
    G(i,i-1)=1;
    G(i-1,i)=-1;
    G(i,i+1)=-2*lambda;
    G(i+1,i)=2*(1-lambda);
end

G(1,2)=-1;
G(1,2*m)=2*(1-lambda);
G(2*m,1)=-2*lambda;
G(2*m,(2*m)-1)=1;
G((2*m)-1,2*m)=-1;

G=(a/2)*G;
```

♣ Affichage :

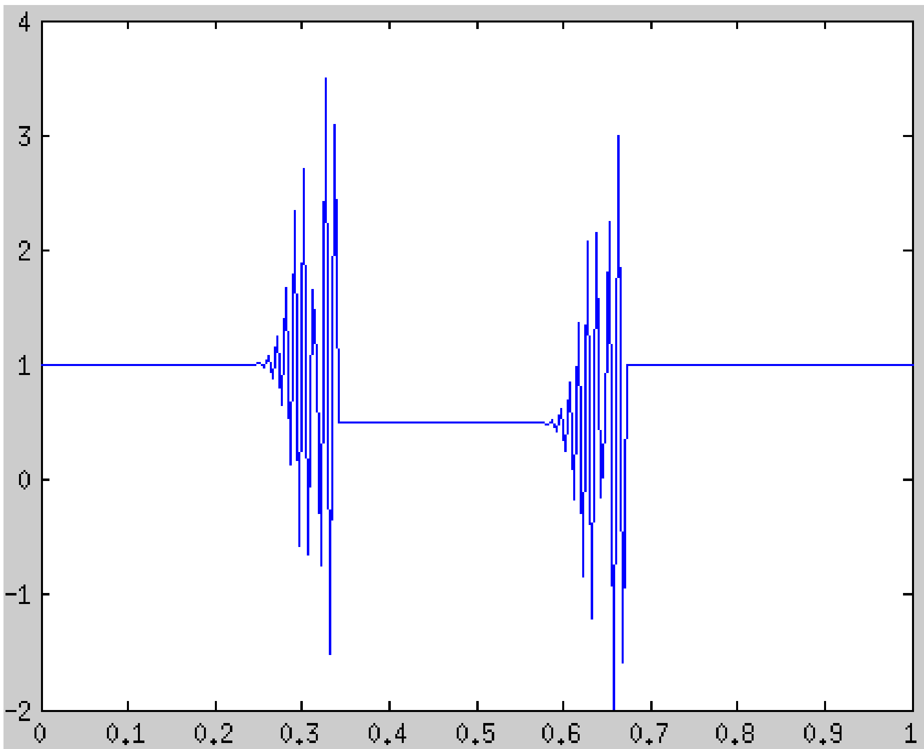
```
u1 = zeros(m, 1)';
u2 = zeros(m, 1)';

for i = 1:T
    u1 = M\(M*u + dt * G * u);
    u2 = M\(3/4 * M*u + 1/4 * M*u1+ dt/4 * G * u1);
    u = M\ (1/3 * M*u + 2/3 * M*u2  + 2*dt/3 * G * u2  );
plot(x, u);
pause(0.1);
end
```

4 Résultats :

Suite à plusieurs problèmes survenus dans X2go, je n'ai pas pu tester mes codes sur l'ensemble des fonctions tests. Par ce fait, je compte seulement exposer les résultats obtenus au début des t.p..

4.1 Première Simulation :



Pour ce graphique nous avons évaluer le troisième cas test, ceci nous donne des discontinuité sur les bords du domaine. Tandis que la solution est bien approximer dans le reste.
Nous avons aussi réussi à faire l'assemblage des matrices dans le cas des éléments finis et galerkine discontinue.

4.2 Méthode des éléments finis :

Dans le cadre de ce T.p., nous avons effectué un stockage creux de cette matrice et nous avons obtenus les élements non nuls de cette matrice :
Voici le résultat obtenu [3] :

3. nous avons utilisé R pour afficher nos résultats

(1,1)	0.6667	(9,8)	0.1667	(16,16)	0.6667
(2,1)	0.1667	(8,9)	0.1667	(17,16)	0.1667
(100,1)	0.1667	(9,9)	0.6667	(16,17)	0.1667
(1,2)	0.1667	(10,9)	0.1667	(17,17)	0.6667
(2,2)	0.6667	(9,10)	0.1667	(18,17)	0.1667
(3,2)	0.1667	(10,10)	0.6667	(17,18)	0.1667
(2,3)	0.1667	(11,10)	0.1667	(18,18)	0.6667
(3,3)	0.6667	(10,11)	0.1667	(19,18)	0.1667
(4,3)	0.1667	(11,11)	0.6667	(18,19)	0.1667
(3,4)	0.1667	(12,11)	0.1667	(19,19)	0.6667
(4,4)	0.6667	(11,12)	0.1667	(20,19)	0.1667
(5,4)	0.1667	(12,12)	0.6667	(19,20)	0.1667
(4,5)	0.1667	(13,12)	0.1667	(20,20)	0.6667
(5,5)	0.6667	(12,13)	0.1667	(21,20)	0.1667
(6,5)	0.1667	(13,13)	0.6667	(20,21)	0.1667
(5,6)	0.1667	(14,13)	0.1667	(21,21)	0.6667
(6,6)	0.6667	(13,14)	0.1667	(22,21)	0.1667
(7,6)	0.1667	(14,14)	0.6667	(21,22)	0.1667
(6,7)	0.1667	(15,14)	0.1667	(22,22)	0.6667
(7,7)	0.6667	(14,15)	0.1667	(23,22)	0.1667
(8,7)	0.1667	(15,15)	0.6667	(22,23)	0.1667
(7,8)	0.1667	(16,15)	0.1667	(23,23)	0.6667

4.3 Galerkine discontinue :

Comme ce sont des matrices de grande taille nous n'allons afficher qu'une partie :

0.003333333	0.001666667	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
0.001666667	0.003333333	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
0.000000000	0.000000000	0.003333333	0.001666667	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
0.000000000	0.000000000	0.001666667	0.003333333	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
0.000000000	0.000000000	0.000000000	0.000000000	0.003333333	0.001666667	0.000000000	0.000000000	0.000000000	0.000000000
0.000000000	0.000000000	0.000000000	0.000000000	0.001666667	0.003333333	0.000000000	0.000000000	0.000000000	0.000000000
0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.003333333	0.001666667	0.000000000	0.000000000
0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.001666667	0.003333333	0.000000000	0.000000000
0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.003333333	0.001666667
0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.001666667	0.003333333
0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000

Cette matrice est similaire à celle ci qu'on a présenté durant notre introduction.

$$M = \frac{1}{h}\begin{pmatrix} 2 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & 1 \\ \vdots & 2 & 1 & \dots & \dots & \dots & \dots & \dots & \dots & \vdots \\ \vdots & 1 & 2 & \dots & \dots & \dots & \dots & \vdots & \dots & \vdots \\ \vdots & & & \ddots & & & & & & \vdots \\ \vdots & & & & \ddots & & & & & \vdots \\ \vdots & & & & & & \ddots & & & \vdots \\ \vdots & & & & & & & \ddots & & \vdots \\ \vdots & & & & & & & 2 & 1 & \vdots \\ \vdots & & & & & & & 1 & 2 & \vdots \\ 1 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & 2 \end{pmatrix}$$

5 Conclusion :

À dessein de conclure ce projet nous a permis d'effectuer cette analogie cours/réalité sur un domaine très prisé par les mathématiciens et les physiciens : La résolution des edp.
De plus, grâce à ce t.p. nous avons manipulé les différentes notions vu en cours afin de pouvoir faire cette compréhension de la théorie et passer à l'application des notions étudiées dans le contexte d'un projet un peu plus orienté ingénierie.