Méthodes numériques, travaux pratiques

le rapport devra être envoyé par mail (guillaume.fuhr@univ-amu.fr)
avant le 15 février 2016 au format pdf

Contents

1 1D equation **2**
- 1.1 Diffusion . . . 2
- 1.2 Advection . . . 3
- 1.3 Advection-Diffusion . . . 3

2 Heat equation in 2D **4**
- 2.1 travail préliminaire . . . 4

3 Travail préparatoire analytique **4**

4 Traitement explicite **4**

5 Traitement implicite et semi-implicite **5**

6 Conclusions **6**

7 Appendices **6**
- 7.1 Résultats analytiques . . . 6
  - 7.1.1 Solution générale de l'équation de la chaleur dans un domaine de taille finie . . . 6
- 7.2 Méthodes numériques . . . 7
  - 7.2.1 Dérivés . . . 7
  - 7.2.2 Euler explicite . . . 7
  - 7.2.3 Euler implicite . . . 7
  - 7.2.4 Cranck-Nicholson . . . 8
  - 7.2.5 Runge-Kutta 4 . . . 8
- 7.3 Code AdvDiff1D . . . 8
  - 7.3.1 téléchargement et compilation . . . 8
  - 7.3.2 Initial Conditions . . . 8
- 7.4 H2D code . . . 9
  - 7.4.1 Download . . . 9
  - 7.4.2 Compilation . . . 9
  - 7.4.3 I/O . . . 9
- 7.5 usefull commands . . . 11

Introduction

Le but du sujet proposé est d'étudier une équation d'advection-diffusion du point de vue des méthodes numériques de résolution. L'équation de base étudiée est la suivante :

$$\frac{\partial u(x,t)}{\partial t} + V\nabla u(x,t) = C\nabla^2 u(x,t) \tag{1}$$

$$\nabla := \begin{cases} \partial_x \\ \partial_y \\ \partial_z \end{cases}$$

Le domaine radial sera de taille $[L_x \times L_y] = [2\pi \times 2\pi]$, pour les diverses méthodes étudiées, il faudra choisir des valeurs pour Δt et Δx permettant d'atteindre un temps dans la simulation de l'ordre de $\simeq \mathbf{10}$ fois le temps caractéristique de la dynamique considérée.
Diffusion : $\tau_D = L^2/C$
Advection : $\tau_A = L/V$

Les coefficients et paramètres de grille ne sont pas imposés mais peuvent-être choisi librement. Un point de départ pour ces diverses valeurs vous pouvez utiliser une grille de taille $[8 \times 8, 16 \times 16, 32 \times 32, \cdots, 256 \times 256]$ et des valeurs pour C, V de l'ordre de $\left[10^{-2}, 10\right]$

1 1D equation

1.1 Diffusion

Dans cette partie, seul le terme de diffusion sera considéré :

$$\frac{\partial u(x,t)}{\partial t} = C\frac{\partial^2 u(x,t)}{\partial x^2} \tag{2}$$

1. Écrivez l'équation discrétisée sous forme matricielle pour les méthodes d'Euler explicites et implicites.
$$AU^{t+1} = BU^t + S$$
2. Dérivez le critère de stabilité dans les 2 cas
3. Partant de $u(0,t) = u(L_x,t) = 0$ et $u(X,0) = u_0(x)$ donnez la solution analytique.
4. Simplifiez la solution si $u_0(x) = A\sin\left(k\frac{\pi}{L_x}x\right)$
5. Résolvez l'équation utilisant la méthode d'Euler explicite. Utilisez le graphe de la solution pour déduire les conditions de bords utilisées. Comment cela est-il implémenté dans le code?
6. Comparez les solutions obtenues quand $C\Delta t/\Delta x^2$ est $< 1/2, = 1/2 - \epsilon, = 1/2, = 1/2 + \epsilon, > 1/2$
7. Calculez le taux de croissance obtenu numérique et comparez le au taux analytique.
8. Pour Δx donné, observez vous une limite de stabilité pour le schéma RK4 en fonction du pas Δt? Comparez cette valeur au Δt correspondant à la méthode d'Euler explicite.

9. Théoriquement, il n'y a pas de limitation sur le pas de temps lors de l'utilisation d'un schéma implicite, est-ce vérifié numériquement? Commentez.

1.2 Advection

On s'interesse maintenant uniquement à l'équation d'advection :

$$\frac{\partial u(x,t)}{\partial t} + V\frac{\partial u(x,t)}{\partial x} = 0$$

1. Ecrire l'équation discrète sous forme matricielle pour les méthodes d'Euler explicites et implicites.
   - en utilisant une dérivée arrière
   - en utilisant une dérivée centrée
2. Calculez le critére de stabilité associée aux 2 méthodes d'Euler.
3. Comment sont codées les dérivées dans le code?
4. Au sens physique, que fait une advection? en particulier, est-ce que la structure radiale de la solution est modifiée?
   - Quand on prend une fonction sin pour $u_0(x)$
   - Quand on prend une fonction de heavyside pour $u_0(x)$
5. Comparez les méthodes d'Euler et Runge-Kutta. En particulier, concernant la stabilité et les implémentations possibles pour le terme de dérivé spatiale.
6. Comment évolue le système quand $V\Delta t/\Delta x$ is $< 1, \simeq 1, = 1, > 1$

1.3 Advection-Diffusion

Dans cette troisième partie, nous nous interessons à l'équation complète :

$$\frac{\partial u(x,t)}{\partial t} + V\frac{\partial u(x,t)}{\partial x} = C\frac{\partial^2 u(x,t)}{\partial x^2}$$

1. Ecrire l'équation sous forme discrète en utilisant des dérivées centrées en espace.
2. Calculez l'erreur locale de troncature?
3. Quel type de dérivée est utilisée pour le terme d'advection dans le schéma d'Euler implicite implémenté?
4. Écrire cette équation sous forme matricielle. Ce résultat peut-il être mis sous la forme de 2 matrices, une pour le terme de diffusion plus une pour le terme d'advection?
5. Comme dans le cas avec uniquement la diffusion, calculez le coefficient de diffusion à partir des données obtenues, ce résultat correspond-il à la valeur attendue?
6. Calculez à partir de vos résultats, le taux de croissance suivant qu'une dérivée centrée ou arrière est utilisée en espace pour le terme d'advection? Ce résultat dépend-il de la valeur du coefficient d'advection?
7. Dans les parties précédentes, deux critères de stabilités ont pu être mis en évidence, que se passe-t-il si l'on choisit des paramètres n'en vérifiant que l'un des 2?

2 Equation de diffusion en 2D

Dans cette seconde partie, nous allons continuer à nous interesser au phénomène de diffusion en présence d'un terme de source, en 2 dimensions. L'équation à résoudre s'écrit alors :

$$\frac{\partial u(x,y,t)}{\partial t} = S(x,y) + C\frac{\partial^2 u(x,y,t)}{\partial x^2} + C\frac{\partial^2 u(x,y,t)}{\partial y^2}$$

2.1 travail préliminaire

3 Travail préparatoire analytique

Nous allons regarder comment nous pouvons modéliser un tel système au niveau numérique.

- Nous voulons simuler un domaine de taille infinie suivant la direction y, quelles méthodes pouvons nous utiliser?
- Supposons que nous calculions la série de Fourier suivant la direction y de notre champs:

  $$u(x,y,t) \rightarrow \sum_{m=0}^{M} u_m(x,t)\exp(\mathrm{i}mk_y y)$$

  Comment s'écrit alors notre équation?
- Quelles hypothèses supplémentaires cela entraine sur la physique associée à notre problème?
- Si nous supposons que $S(x,y) = 0$ et $u_0(x,y) = A\sin(k_x x)\sin(k_y y)$ calculez analytiquement le taux de diffusion du système. Peut on faire la même chose si nous utilisons la série de Fourier définie precedemment?

4 Traitement explicite

Dans cette partie nous allons nous intéresser à la résolution de notre équation avec 2 méthodes différentes,

- méthode d'Euler explicite
- méthode de Runge-Kutta 4
- Écrire la formulation discrète de l'équation de la chaleur considérée ici
- Calculez la condition de stabilité lorsque le laplacien est exprimé dans l'espace réel ainsi que si nous appliquons une transformée de Fourier suivant la direction y?
- Si nous augmentons le nombre de points/nombre de modes, les 2 méthodes donnent-elles les mêmes résultats? Sont-elles aussi précises? Commentez.
- Si nous changeons l'expression de $u_0(x,y)$ cela influence-t-il les résultats précédents? Pourquoi?
- Estimez l'erreur commise par rapport à la solution analytique pour les différents schémas en fonction des paramètres que vous avez choisis.

- Pour un même jeu de paramètres, tracez le temps de calcul nécessaire au passage de t à $t+1$ pour les résolutions suivantes : 8×8,16×16,32×32,64×64,128×128, 256×256. La relation entre le temps de calcul et la résolution de la grille est-elle linéaire?

- Quelles sont les différentes conditions de bord possible présentes dans le code? Ajoutez le code de la fonction null_bc du fichier boundary.c afin d'implementer les conditions suivantes suivant y:

$$\partial_y u(x,0,t) = 0, \quad \partial_y u(x,Ly,t) = 0$$

- Par rapport aux conditions actuellement présentes suivant la direction y, la solution obtenue avec ces nouvelles conditions est-elle différente? pourquoi?

5 Traitement implicite et semi-implicite

Dans cette partie nous allons nous intéresser à la résolution de notre équation avec 2 méthodes différentes,

- méthode d'Euler implicite
- méthode de Cranck-Nicholson

- Retrouvez l'expression du facteur d'amplification dans le cas d'une équation de diffusion $1D$ avec les méthodes d'Euler implicite et de Cranck-Nicholson.

- Avons nous encore une condition de stabilité comme dans le cadre des méthodes explicites?

- Quelles sont les différences entre la méthode d'Euler implicite et la méthode de Cranck-Nicholson?

- Écrire l'équation de la chaleur sans source, sous forme matricielle, pour une utilisation de la la méthode d'Euler implicite dans les cas 1D et 2D. Le résultat doit se présenter sous la forme :

$$AU^{t+1} = S + BU^t$$

- Quelles différences a-t-on entre les cas 1D et 2D?

- Les conditions de bords entrent-elles en ligne de compte sur la formulation matricielle? Que devons nous changer si nous voulons passer de conditions de type Dirichlet à des conditions de type Von Neumann.

- Et si nous utilisons une décomposition en série de Fourier suivant la direction y?

- Dans le code, dans la fonction *euli*, comment se calcule U^{t+1} en fonction de U^t? Est-ce une méthode directe ou itérative?

- Typiquement, une inversion matricielle a une complexité proportionnelle à N^3 Tracez le temps de calcul nécessaire au passage de t à $t+1$ pour les résolutions suivantes : 8×8,16×16,32×32,64×64,128×128, 256×256.

- Est-ce que le temps de calcul est proportionnel à N^3 dans ce cas? conclusion?

- Même question concernant la méthode de relaxation et l'utilisation d'une discretisation utilisant les séries de Fourier.

- Á partir d'une condition initiale égale à la condition initiale utilisée en 2.2.5, comparez l'erreur obtenue avec la solution théorique en fonction du temps.

6 Conclusions

Sur les différentes méthodes proposées, si vous ne pouviez choisir qu'une seule méthode de résolution pour résoudre l'équation de la chaleur 2D, laquelle choisiriez vous? pourquoi?

7 Appendices

7.1 Résultats analytiques

7.1.1 Solution générale de l'équation de la chaleur dans un domaine de taille finie

cas 1D

$$
\begin{aligned}
\partial_t u(x,t) &= C\partial_x^2 u(x,t) + Du(x,t) + S(x) && (3)\\
\text{Domaine :} \quad & 0 \leq x \leq Lx && (4)\\
u(x,0) &= u_0(x) && (5)\\
u(0,t) &= 0 && (6)\\
u(0,Lx) &= 0 && (7)\\
u(x,t) &= \int_0^{Lx} u_0(x)G(x,\xi,t)\mathrm{d}\xi + \int_0^t \int_0^{Lx} S(\xi)G(x,\xi,t-\tau)\mathrm{d}\xi\mathrm{d}\tau && (8)\\
G(x,\xi,t) &= \frac{2}{Lx}e^{(Dt)} \sum_{n=1}^{\infty} \sin\left(\frac{n\pi x}{Lx}\right) \sin\left(\frac{n\pi \xi}{Lx}\right) \exp\left(-\frac{Cn^2\pi^2 t}{Lx^2}\right) && (9)\\
& && (10)
\end{aligned}
$$

cas 2D

$$
\begin{aligned}
\partial_t u(x,y,t) &= C\partial_x^2 u(x,t) + C\partial_y^2 u(x,y,t) + S(x,y) && (11)\\
\text{Domaine :}\quad & 0 \le x \le Lx && (12)\\
& -\infty \le y \le \infty && (13)\\
u(x,y,0) &= u_0(x,y) && (14)\\
u(0,y,t) &= 0 && (15)\\
u(Lx,y,t) &= 0 && (16)\\
u(x,t) &= \int_{-\infty}^{\infty}\int_{0}^{Lx} u_0(\xi,\eta)G(x,y,\xi,\eta,t)\mathrm{d}\xi\mathrm{d}\eta && (17)\\
& + \int_0^t \int_0^{Lx}\int_{-\infty}^{\infty} S(\xi,\eta)G(x,y,\xi,\eta,t-\tau)\mathrm{d}\xi\mathrm{d}\eta\mathrm{d}\tau && (18)\\
G(x,\xi,t) &= G_1(x,\xi,t)G_2(y,\eta,t) && (19)\\
G_1(x,\xi,t) &= \frac{2}{Lx}\sum_{n=1}^{\infty}\sin\left(\frac{n\pi x}{Lx}\right)\sin\left(\frac{n\pi\xi}{Lx}\right)\exp\left(-\frac{an^2\pi^2 t}{Lx^2}\right) && (20)\\
G_2(y,\eta,t) &= \frac{1}{2\sqrt{\pi Ct}}\left[\exp\left(-\frac{(y-\eta)^2}{4Ct}\right) - \exp\left(-\frac{(y+\eta)^2}{4Ct}\right)\right] && (21)
\end{aligned}
$$

7.2 Méthodes numériques

La convention suivante est utilisée dans la suite :

$$f(x,t) \to f(x_i,t_j) \to f(x+i\Delta x, t+j\Delta t) \to f_{t+i}^{x+j}$$

7.2.1 Dérivés

Partant d'une équation type,

$$\frac{\partial f(x,t)}{\partial t} = L(t, f(x,t)) \tag{22}$$

$$
\begin{aligned}
\text{dérivé avant :}\quad & \partial_t f(t) &=& \tfrac{f^{t+1}-f^t}{\Delta t}\\
\text{dérivé arrière :}\quad & \partial_t f(t) &=& \tfrac{f^{t}-f^{t-1}}{\Delta t}\\
\text{dérivé centrée :}\quad & \partial_t f(t) &=& \tfrac{f^{t+1}-f^{t-1}}{2\Delta t}
\end{aligned} \tag{23}
$$

7.2.2 Euler explicite

$$
\begin{aligned}
\frac{f^{t+1}-f^t}{\Delta t} &= L(t,f^t) && (24)\\
f^{t+1} &= f^t + \Delta t L(t,f^t) && (25)
\end{aligned}
$$

7.2.3 Euler implicite

$$
\begin{aligned}
\frac{f^{t+1}-f^t}{\Delta t} &= L(t+\Delta t, f^{t+1}) && (26)\\
f^{t+1} &= f^t + \Delta t L(t+\Delta t, f^{t+1}) && (27)
\end{aligned}
$$

7.2.4 Cranck-Nicholson

$$\frac{f^{t+1}-f^t}{\Delta t} = \frac{1}{2}L(t,f^t) + \frac{1}{2}L(t+\Delta t, f^{t+1}) \tag{28}$$

$$f^{t+1} = f^t + \Delta t\left(\frac{1}{2}L(t,f^t) + \frac{1}{2}L(t+\Delta t, f^{t+1})\right) \tag{29}$$

7.2.5 Runge-Kutta 4

$$f^{t+1} = f^t + \frac{\Delta t}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right) \tag{30}$$

with (31)

$$k_1 = \Delta t L(t, f^t) \tag{32}$$

$$k_2 = \Delta t L(t + \frac{\Delta t}{2}, f^t + \frac{1}{2}k_1) \tag{33}$$

$$k_3 = \Delta t L(t + \frac{\Delta t}{2}, f^t + \frac{1}{2}k_2) \tag{34}$$

$$k_4 = \Delta t L(t + \Delta t, f^t + k_3) \tag{35}$$

(36)

7.3 Code AdvDiff1D

7.3.1 téléchargement et compilation

adresse de téléchargement : **https://github.com/GFuhr/MF_FCM6/zipball/master**
Code source se trouve dans le dossier TP1.

Comment compiler

- Sous linux (Mac?) , la compilation se fait via la commande **make**.
- Pour éxécuter le programme généré sous linux, tapez **./advdiff.exe**
- Sous Windows, dans un éditeur type VisualStudio ou Code::Blocks, chargez les fichiers *advdiff.c* et *advdiff.h*.

7.3.2 Initial Conditions

temps

	$advection$	$diffusion$
$t=0$	$u^0(x)$	$u^0(x)$

espace

	$advection$	$diffusion$
$x=0$	$u(t,0)$	$u(t,0)$
$x=L_x$		$u(t,L_x)$

Champs initiaux

$u^0(x)$	$A\sin(\sigma\frac{\pi}{L_x}(x-x_0))$
$u^0(x)$	$A\exp(-(x-x_0)^2/\sigma^2)$
$u^0(x)$	$\begin{cases} A \text{ if } x\in[x_0-\sigma/2,\, x_0+\sigma/2] \\ 0 \text{ else} \end{cases}$

I/O Inputs :
Signification des entrées initiales.

- "Nx=?" : nombre de points en espace
- "Npas=?" : nombre d'itérations en temps
- "Nout=?" : chaque $Nout$ iterations, $u(t, x)$ sera exporté dans un fichier
- "C=?" : coefficient de diffusion
- "V=?" : coefficient d'advection
- "A=?" : Amplitude initiale, comme décrit en 7.3.2
- "x0=?": Paramètre x_0 défini dans 7.3.2
- "sigma=?": Paramètre σ défini dans 7.3.2
- "Dx=?" : valeur du pas Δx
- "Dt=?" : pas de temps Δt

Sorties :
À chaque éxécution du programme, *Nout* fichiers de sortie sont générés **out_XXXX.dat**. Remarque, ces fichiers ne sont pas écrasés entre les runs. Chaque fichier est codé en format texte (ASCII) et contient $(Nx - 2)$ points avec les valeurs de u_i^j au temps $t = j * Nout$.

7.4 H2D code

7.4.1 Download

Code is in the same archive as previous one :
https://github.com/GFuhr/MF_FCM6/zipball/master,
Le code H2D est dans le dossier TP2.

7.4.2 Compilation

- on linux, with make command.
- To run it on linux,after compilation step through **make**, type **./bin/h2d_gcc.exe**
- on windows, a "project" file usable with *Code::Blocks* and a "solution" file for *Visual Studio* can be found in folder *TP2/H2D/*

7.4.3 I/O

Inputs :
Initial parameters must be put in file params/params.h. Following parameters can be modified :

- "C" : diffusion coefficient
- "NX" : nombre de points dans la direction x
- "NY" : nombre de points dans la direction y

- "DT" : pas de temps

- "ITER" : nombre d'itérations en temps

- "LX" : Taille de la boîte suivant x

- "LY" : Taille de la boîte suivant y

- "discret" : discretisation spatiale, peut-être "real" ou "fourier"

- "scheme" : schéma numérique
  "eule" : explicit Euler
  "euli" : implicit Euler
  "eulis" : implicit Euler with relaxations
  "rk4" : Runge-Kutta 4
  "cn" : Cranck-Nicholson

Profil initial ainsi que la source peuvent être modifié dans le fichier params/functions.c

Outputs :
2 fichiers sont générés à chaque fois : **H2D_GPLOT_XXXX.dat** and **H2D_OCT_XXXX.dat**. La seule différence est que les fichiers ($\_OCT\_$) peuvent être utilisés avec octave et les fichiers ($\_GPLOT\_$) avec gnuplot.

Files $\_OCT\_$, contain Ny lines and Nx columns with values of $u(x, y, t)$. Files $\_GPLOT\_$ contain $Nx * Ny$ lines and 3 columns with values of $u(x, y, t)$.

Output format $u(x_i, y_j) \rightarrow u_{i,j}$**:**

Case where "discret=real", files $GPLOT$ **:**

$$\text{Nx*Ny lines, 3 columns} \begin{cases} x_0 & y_0 & u_{i,j} \\ x_1 & y_0 & u_{i,j} \\ \vdots & \vdots & \vdots \\ x_{Nx-1} & y_0 & u_{Nx-1,Ny-1} \\ x_0 & y_1 & u_{0,1} \\ \vdots & \vdots & \vdots \\ x_{Nx-1} & y_{Ny-1} & u_{Nx-1,Ny-1} \end{cases} \tag{37}$$

Case where "discret=fourier", files $GPLOT$:

$$\text{2*Nx*Ny lines, 3 columns} \begin{cases} x_0 & m_0 & \Re(u_{i,j}) \\ x_1 & m_0 & \Re(u_{i,j}) \\ \vdots & \vdots & \vdots \\ x_{Nx-1} & m_0 & \Re(u_{Nx-1,Ny-1}) \\ x_0 & m_1 & \Re(u_{0,1} \\ \vdots & \vdots & \vdots \\ x_{Nx-1} & m_{Ny-1} & \Re(u_{Nx-1,Ny-1}) \\ x_0 & m_0 & \Im(u_{i,j}) \\ x_1 & m_0 & \Im(u_{i,j}) \\ \vdots & \vdots & \vdots \\ x_{Nx-1} & m_0 & \Im(u_{Nx-1,Ny-1}) \\ x_0 & m_1 & \Im(u_{0,1}) \\ \vdots & \vdots & \vdots \\ x_{Nx-1} & m_{Ny-1} & \Im(u_{Nx-1,Ny-1}) \end{cases} \tag{38}$$

Case where "discret=real", files OCT :

$$\text{Ny lines, Nx columns} \begin{cases} u_{0,0} & u_{1,0} & \cdots & u_{Nx-1,0} \\ u_{0,1} & u_{1,1} & \cdots & u_{Nx-1,1} \\ \vdots & \vdots & \vdots & \\ u_{0,Ny-1} & u_{1,Ny-1} & \cdots & u_{Nx-1,Ny-1} \end{cases} \tag{39}$$

Case where "discret=fourier", files OCT :

$$\text{2*Ny lines, Nx columns} \begin{cases} \Re(u_{0,0}) & \Re(u_{1,0}) & \cdots & \Re(u_{Nx-1,0}) \\ \Re(u_{0,1}) & \Re(u_{1,1}) & \cdots & \Re(u_{Nx-1,1}) \\ \vdots & \vdots & \vdots & \\ \Re(u_{0,Ny-1}) & \Re(u_{1,Ny-1}) & \cdots & \Re(u_{Nx-1,Ny-1}) \\ \Im(u_{0,0}) & \Im(u_{1,0}) & \cdots & \Im(u_{Nx-1,0}) \\ \Im(u_{0,1}) & \Im(u_{1,1}) & \cdots & \Im(u_{Nx-1,1}) \\ \vdots & \vdots & \vdots & \\ \Im(u_{0,Ny-1}) & \Im(u_{1,Ny-1}) & \cdots & \Im(u_{Nx-1,Ny-1}) \end{cases} \tag{40}$$

7.5 usefull commands

- pour mesurer le temps d'éxécution, la commande **time** est utilisée :
  time ./bin/h2d_gcc.exe
  Remarque : une mesure ne peut être considérée comme fiable que si elle est supérieure à 10 secondes.

- How to plot 3D data with gnuplot. In following, we suppose NX=64 and NY=64

```
gnuplot> set dgrid3d 64,64
gnuplot> set hidden3d
gnuplot> splot "H2D_0000.dat" u 1:2:3 with lines
```

- How to plot 1D data with gnuplot.

  ```
  gnuplot> plot "out_0000.dat" with lines
  ```

- with Octave
  - to read datas use function *load*
  - to plot datas $u(x, y)$, use function *surf*

    ```
    octave> data=load('H2D_0000.dat');
    octave> surf(data)
    ```

- download code in a terminal :
  wget https://github.com/GFuhr/MF_FCM6/zipball/master
- unzip archive :
  unzip master
- put unzipped files in a directory with a "friendly name" :
  mv GFuhr-MF_FCM6* *newname*
- change directory :
  cd
- create directory mkdir
  mkdir directory_name
- delete file :
  rm filename
- list file in a directory :
  ls directory_name
- list file in current directory :
  ls ./