

## TPs EDP

### Travaux Pratiques N° 3

#### Méthodes des différences finies Equation de la chaleur 1D instationnaire

Version du 5 novembre 2025

## 1 Conditions de Dirichlet

On souhaite résoudre numériquement le problème aux limites (*boundary value problem* ou B.V.P.) suivant

$$\frac{\partial u}{\partial t}(t, x) - \nu \frac{\partial^2 u}{\partial x^2}(t, x) = f(t, x), \quad \forall (t, x) \in [t_0; t_0 + T] \times ]a; b[, \quad (1.1)$$

$$u(t_0, x) = u_0(x), \quad \forall x \in [a; b], \quad (1.2)$$

$$u(t, a) = g_a(t), \quad \forall t \in [t_0; t_0 + T], \quad (1.3)$$

$$u(t, b) = g_b(t), \quad \forall t \in [t_0; t_0 + T]. \quad (1.4)$$

avec  $\nu$  un réel strictement positif,  $t_0 \in \mathbb{R}$ ,  $T > 0$ ,  $(a, b) \in \mathbb{R}^2$ ,  $a < b$ . Bien évidemment des conditions de compatibilité sont imposées entre les conditions aux limites et la condition initiale.

On note  $t^n$ ,  $n \in \llbracket 0, N_t \rrbracket$  et  $x_i$ ,  $i \in \llbracket 0, N_x \rrbracket$  les discrétisations régulières des intervalles  $[t_0; t_0 + T]$  et  $[a; b]$  avec  $N_t$  pas de discrétisation en temps et  $N_x$  pas de discrétisation en espace.

On souhaite implémenter deux schémas de résolution de (1.1) :

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} - \nu \frac{u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}}{\Delta x^2} = f_i^{n+1}. \quad (1.5)$$

et

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} - \nu \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{\Delta x^2} = f_i^n. \quad (1.6)$$

où  $\Delta t = T/N_t$ ,  $\Delta x = (b - a)/N_x$ ,  $f_i^n = f(t^n, x_i)$  et  $u_i^n \approx u(t^n, x_i)$ .

On rappelle que le premier schéma est le **schéma d'Euler implicite** et le second le **schéma d'Euler explicite**. Le schéma d'Euler implicite est inconditionnellement stable et le schéma d'Euler explicite est stable sous la condition de C.F.L.

$$\nu \frac{\Delta t}{\Delta x^2} \leq \frac{1}{2}.$$

On note,  $\forall n \in \llbracket 0, N_t \rrbracket$ ,  $\mathbf{U}^n$  les vecteurs de dimension  $N_x + 1$ , de composantes  $\mathbf{U}_i^n = u_{i-1}^n$ ,  $\forall i \in \llbracket 1, N_x + 1 \rrbracket$ .

#### Q. 1 [rapport] ★★☆☆

- Expliquez en détails comment le schéma **implicite** (1.5) a été obtenu.
- Donner une discrétisation complète du problème (1.1) à (1.4) utilisant (1.5).
- Donner une méthodologie permettant de résoudre numériquement ce problème. Si un ou plusieurs systèmes linéaires sont à résoudre, il faudra expliciter les matrices et vecteurs second membre associés.
- Déterminer, en expliquant la manière d'opérer, la taille en **Octets** de la matrice du système lors d'un stockage plein et d'un stockage creux pour  $N_x + 1$  valant  $5 \times 10^2$ ,  $5 \times 10^3$  et  $5 \times 10^4$ .<sup>a</sup>

a. voir mini-cours sur les matrices creuses

#### Q. 2 [rapport] ★★☆☆

- Expliquez en détails comment le schéma **explicite** (1.6) a été obtenu.
- Donner une discrétisation complète du problème (1.1) à (1.4) utilisant (1.6).
- Donner une méthodologie permettant de résoudre numériquement ce problème. Si un ou plusieurs systèmes linéaires sont à résoudre, il faudra expliciter les matrices et vecteurs second membre associés.

## 2 Implémentation et validation

On étudie ce problème aux limites avec les données  $t_0 = 0$ ,  $T = 2$ ,  $a = 0$ ,  $b = 2\pi$ ,  $\nu = 2$ ,  $k = 5$

$$\begin{aligned}f(t, x) &= -k \sin(kt) \cos(x) + \nu \cos(kt) \cos(x), \\u_0(x) &= \cos(kt_0) \cos(x), \\g_a(t) &= \cos(kt) \cos(a), \\g_b(t) &= \cos(kt) \cos(b).\end{aligned}$$

Dans ce cas, la solution exacte est donnée par  $u_{\text{ex}}(t, x) = \cos(kt) \cos(x)$ .

Q. 3 [rapport] ★★☆☆

Proposer un autre exemple non trivialement similaire avec une solution exacte (non identiquement nulle!).

Q. 4 [code] ★★☆☆

- a. Pour résoudre le B.V.P. par un schéma d'**Euler implicite**, le programme `heat1d.implicit_DirichletCL` est fourni, ainsi que les fonctions `fsftx1d.eval*`, `fsftx1d.norm` et `fsftx1d.plot` (voir l'archive fournie : **TP3.tar.gz**). Dans les codes fournis, il manque le fichier `+head1d/+implicit/Euler_DirichletCL.m` correspondant à la fonction :

```
[t,x,u]=heat1d.implicit.Euler_DirichletCL(BVP,Nt,Nx)
```

résolvant le B.V.P. par un schéma d'Euler implicite avec

- BVP : structure, définie dans `heat1d.implicit_DirichletCL`, contenant l'ensemble des données du B.V.P. à résoudre.
- Nt : nombre de pas de discrétisation en temps,
- Nx : nombre de pas de discrétisation en espace,
- t : discrétisation en temps (dimension  $N_t+1$ ),  $t(n) = t^{n-1}$ ,  $\forall n \in \llbracket 1, N_t + 1 \rrbracket$ ,
- x : discrétisation en espace (dimension  $N_x+1$ ),  $x(i) = x_{i-1}$ ,  $\forall i \in \llbracket 1, N_x + 1 \rrbracket$ ,
- u :  $u(i,n)$  solution approchée au temps  $t(n)$  et point  $x(i)$  (dimension  $(N_x+1, N_t+1)$ ).

Cette fonction devra de plus utiliser l'objet `inputParser`<sup>a</sup> de Matlab/Octave pour permettre de spécifier le stockage creux (sparse matrix) ou le stockage plein (full matrix) de la matrice associée au système linéaire à résoudre à chaque itération. Par défaut, le stockage sera creux, et, pour utiliser le stockage plein, on utilisera :

```
[t,x,u]=heat1d.implicit.Euler_DirichletCL(BVP,Nt,Nx,'sparse',false)
```

Ecrire cette fonction.

- b. Ecrire un programme (fichier `+head1d/+implicit/bench_mat.m` comparant les temps d'exécution de la fonction `heat1d.implicit.Euler_DirichletCL` utilisant un stockage plein ou un stockage creux de la matrice pour différentes valeurs de  $(N_t, N_x)$ .

---

a. voir mini-cours Matlab avancé

Q. 5 ★★☆☆

- a. [code] Pour résoudre le B.V.P. par un schéma d'**Euler explicite**, le programme `heat1d.explicit_DirichletCL` est fourni, ainsi que les fonctions `fsftx1d.eval`, `fsftx1d.norm` et `fsftx1d.plot` (voir fichier **TP3.tar.gz**). Dans les codes fournis, il manque le fichier `+head1d/+explicit/Euler_DirichletCL.m` correspondant à la fonction :

```
[t,x,u]=head1d.explicit.Euler_DirichletCL(EDP,Nt,Nx)
```

résolvant le B.V.P. par un schéma d'Euler explicite. Ecrire cette fonction, sachant que ses paramètres sont identiques à ceux de la fonction `heat1d.implicit.Euler_DirichletCL` et qu'il n'y pas de paramètres optionnels.

- b. [rapport] Dans le programme `heat1d.explicit_DirichletCL`, changer le paramètre `Nt` de 2100 à 2000. Que se passe-t-il lors de la résolution ? Quel est le résultat théorique qui intervient ?

---

\*. `fsftx1d` abréviation de *functionnal space of functions in time and 1d in space*

### 3 Stabilité : approche matricielle

Soient  $\mathbf{U}^{[0]} \in \mathbb{R}^d$ ,  $\mathbb{E} \in \mathcal{M}_d(\mathbb{R})$  inversible et  $\mathbb{F} \in \mathcal{M}_d(\mathbb{R})$ . On note  $(\mathbf{U}^{[n]})_{n \in \mathbb{N}}$ , la suite de  $\mathbb{R}^d$ , définie par le schéma générique

$$\mathbb{E}\mathbf{U}^{[n+1]} = \mathbb{F}\mathbf{U}^{[n]} + \mathbf{c}^{[n]}, \quad \forall n \in \mathbb{N} \quad (3.1)$$

avec  $\mathbf{c}^{[n]} \in \mathbb{R}^d$  indépendant des  $(\mathbf{U}^{[n]})_{n \in \mathbb{N}}$ .

En notant  $\mathbb{B} \stackrel{\text{def}}{=} \mathbb{E}^{-1}\mathbb{F}$  et  $\mathbf{g}^{[n]} = \mathbb{E}^{-1}\mathbf{c}^{[n]}$ , on a alors

$$\mathbf{U}^{[n+1]} = \mathbb{B}\mathbf{U}^{[n]} + \mathbf{g}^{[n]}.$$

Le schéma (3.1) est dit stable si une petite perturbation de la donnée initiale  $\mathbf{U}^{[0]}$  n'engendre pas de grandes erreurs en itérant.

On note  $\boldsymbol{\varepsilon} \in \mathbb{R}^d$  une petite perturbation et  $\mathbf{V}^{[0]} = \mathbf{U}^{[0]} + \boldsymbol{\varepsilon}$ . On note  $(\mathbf{V}^{[n]})_{n \in \mathbb{N}}$ , la suite de  $\mathbb{R}^d$ , définie par

$$\mathbf{V}^{[n+1]} = \mathbb{B}\mathbf{V}^{[n]} + \mathbf{g}^{[n]}.$$

Une simple récurrence permet d'établir que

$$\mathbf{V}^{[n]} = \mathbf{U}^{[n]} + \mathbb{B}^n \boldsymbol{\varepsilon}.$$

Si  $\rho(\mathbb{B}) > 1$ , alors  $\lim_{n \rightarrow \infty} \|\mathbb{B}^n \boldsymbol{\varepsilon}\| = +\infty$ . Il est donc nécessaire d'avoir  $\rho(\mathbb{B}) \leq 1$ .

Pour calculer numériquement le rayon spectral d'une matrice avec Matlab/Octave, on peut utiliser la fonction `eigs` avec comme argument `'lm'` (ou `'largestabs'`) pour déterminer les valeurs propres de plus grand module.

Pour calculer les 4 plus grandes valeurs propres de  $\mathbb{B} \stackrel{\text{def}}{=} \mathbb{E}^{-1}\mathbb{F}$ , il suffit d'utiliser la commande

$$[V,D] = \text{eigs}(\text{inv}(E)*F, 4, 'lm');$$

mais le calcul de  $\mathbb{E}^{-1}$  est trop coûteux et cette matrice n'est pas forcément creuse même si  $\mathbb{E}$  l'est ! Par exemple, la matrice  $\mathbb{K} \in \mathcal{M}_d(\mathbb{R})$  définie par

$$\mathbb{K} = \begin{pmatrix} -2 & 1 & 0 & \dots & 0 \\ 1 & -2 & 1 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 1 \\ 0 & \dots & 0 & 1 & -2 \end{pmatrix}$$

a pour inverse une matrice dont tous les éléments sont non nuls.

On propose ici deux stratégies : la première utilisant un problème aux valeurs propres généralisé et la seconde utilisant une possibilité de la fonction `eigs` permettant de remplacer la matrice fournie en premier paramètre par une fonction retournant le produit de cette matrice par un vecteur.

#### 3.1 Problème aux valeurs propres généralisé

On peut noter que si  $(\lambda, \mathbf{w})$  est un mode propre de  $\mathbb{B} \stackrel{\text{def}}{=} \mathbb{E}^{-1}\mathbb{F}$ , on a

$$\mathbb{B}\mathbf{w} = \lambda\mathbf{w} \Leftrightarrow \mathbb{F}\mathbf{w} = \lambda\mathbb{E}\mathbf{w}.$$

**Définition 3.1** (Problème aux valeurs propres généralisé). Soient  $\mathbb{A}$  et  $\mathbb{B}$  dans  $\mathcal{M}_d(\mathbb{C})$ . On cherche  $(\lambda, \mathbf{u}) \in \mathbb{C} \times \mathbb{C}^d \setminus \{0\}$  tel que

$$\mathbb{A}\mathbf{u} = \lambda\mathbb{B}\mathbf{u}.$$

Le scalaire  $\lambda$  est appelée valeur propre généralisée et  $\mathbf{u}$  est le vecteur propre généralisé associé.

Si  $\mathbb{B}$  est inversible alors  $(\lambda, \mathbf{u})$  est un mode propre de  $\mathbb{B}^{-1}\mathbb{A}$ .

Pour calculer numériquement les 4 plus grandes valeurs propres généralisées (en module) décrit en Définition 3.1, on utilise avec **Matlab** la commande

$$[V,D] = \text{eigs}(A,B,4, 'largestabs');$$

On a alors  $V \in \mathcal{M}_{d,4}(\mathbb{C})$ , et  $D \in \mathcal{M}_{4,4}(\mathbb{C})$  diagonale vérifiant

$$A*V = B*V*D$$

c'est à dire  $(D(i,i), V(:,i))$  est un mode propre généralisé.

Avec **Octave**, la matrice  $B$  doit être hermitienne définie positive (à vérifier). Or le problème aux valeurs propres généralisé est équivalent à

$$\mathbb{B}^* \mathbb{A} \mathbf{u} = \lambda \mathbb{B}^* \mathbb{B} \mathbf{u}.$$

On pourra utiliser avec **Octave** la commande

$$[V,D]=\text{eigs}(B'*A,B'*B,4,'lm');$$

L'inconvénient de cette stratégie est que le caractère creux  $B'*A$  et de  $B'*B$  n'est pas forcément assuré même si  $B$  et  $A$  le sont.

### 3.2 Fonctions `eigs` et $\mathbf{x} \mapsto \mathbb{B}\mathbf{x}$

Sous Matlab et Octave, il est possible d'utiliser la commande (trop couteuse)

$$[V,D]=\text{eigs}(\text{inv}(E)*F,4,'lm');$$

pour calculer numériquement les 4 plus grandes valeurs propres en module de  $\mathbb{B} \stackrel{\text{def}}{=} \mathbb{E}^{-1}\mathbb{F}$ . Cette commande peut aussi s'écrire à l'aide d'une fonction en premier paramètre suivi de la dimension

$$[V,D]=\text{eigs}(@(\mathbf{x}) \text{inv}(E)*F*\mathbf{x}, d, 4, 'lm');$$

On peut alors remarquer qu'il n'est pas nécessaire de calculer l'inverse de  $\mathbb{E}$ ! En effet, pour tout  $\mathbf{x} \in \mathbb{C}^d$ , calculer

$$\mathbf{y} = \mathbb{E}^{-1}\mathbb{F}\mathbf{x}$$

revient à résoudre le système linéaire

$$\mathbb{E}\mathbf{y} = \mathbb{F}\mathbf{x}.$$

On pourra alors utiliser la commande

$$[V,D]=\text{eigs}(@(\mathbf{x}) \mathbb{E} \setminus (\mathbb{F}*\mathbf{x}), 4, 'lm');$$

ou

$$[V,D]=\text{eigs}(@(\mathbf{x}) \text{mldivide}(E,F*\mathbf{x}), 4, 'lm');$$

Q. 6 ★★☆☆

- [rapport]** Montrer que la discrétisation complète du problème (1.1) à (1.4) utilisant (1.5) (voir Q.1) peut s'écrire sous la forme (3.1) où l'on explicitera les matrices  $\mathbb{E}$  et  $\mathbb{F}$ , ainsi que le vecteur  $\mathbf{c}^{[n]}$ .
- [code]** Ecrire un programme `head1d.implicit_Euler_DirichletCL_spradius` permettant de calculer numériquement le rayon spectral de la matrice  $\mathbb{B}$  à  $N_x$  fixé et en faisant varier  $N_t$  de telle sorte que la condition C.F.L. ne soit pas toujours vérifiée. On affichera pour chaque valeur de  $N_t$ , le rayon spectral de  $\mathbb{B}$  et la valeur de  $\nu \frac{h_t}{h_x^2}$ .
- [rapport]** Que peut-on tirer/conjecturer des résultats du programme précédent ?

Q. 7 ★★☆☆

- [rapport]** Montrer que la discrétisation complète du problème (1.1) à (1.4) utilisant (1.6) (voir Q.2) peut s'écrire sous la forme (3.1) où l'on explicitera les matrices  $\mathbb{E}$  et  $\mathbb{F}$ , ainsi que le vecteur  $\mathbf{c}^{[n]}$ .
- [code]** Ecrire un programme `head1d.explicit_Euler_DirichletCL_spradius` permettant de calculer numériquement le rayon spectral de la matrice  $\mathbb{B}$  à  $N_x$  fixé et en faisant varier  $N_t$  de telle sorte que la condition C.F.L. ne soit pas toujours vérifiée. On affichera pour chaque valeur de  $N_t$ , le rayon spectral de  $\mathbb{B}$  et la valeur de  $\nu \frac{h_t}{h_x^2}$ .
- [rapport]** Que peut-on tirer/conjecturer des résultats du programme précédent ?

## 4 Ordres

Les schémas d'Euler explicite et d'Euler implicite sont d'ordre 2 en espace et d'ordre 1 en temps, c'est à dire que l'erreur se comporte comme un  $\mathcal{O}(h_x^2) + \mathcal{O}(h_t)$ .

Les figures 1 sont une représentation possible de ce comportement. Elles ont été établies avec  $a = -1$ ,  $b = 1$ ,  $t_0 = 0$ ,  $T = 1$ ,  $\nu = 1/2$ , et les autres données ont été choisies pour avoir

$$u(t, x) = \cos(kt) \cos(t + kx), \quad k = 2$$

comme solution exacte.

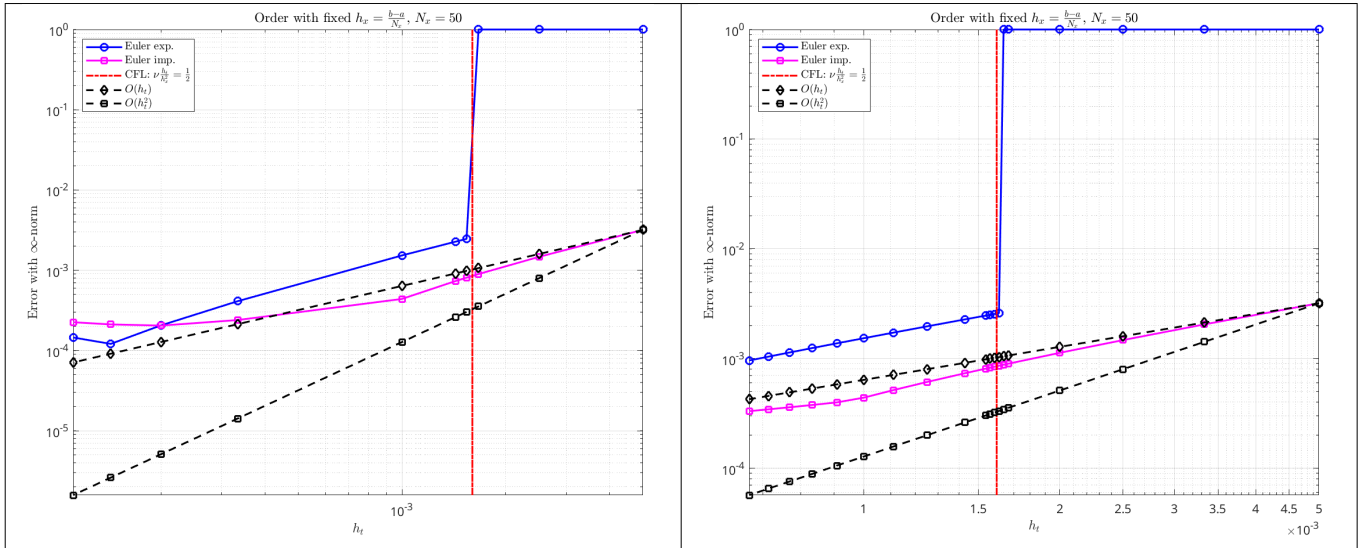


FIGURE 1 – Ordre, figure de gauche  $N_t$  dans  $[200:200:600, 650, 700, 1000:2000:10000]$  et, à droite, dans  $[200:100:600, 610:10:650, 700:100:1500]$

Q. 8 ★★☆☆

- [rapport]** Expliquer tout ce que l'on peut interpréter/comprendre à partir de ces deux figures.
- [code]** Ecrire un programme `head1d.ordre_DirichletCL` permettant de générer ces deux figures le plus exactement possible.

## 5 Application

Comme application possible nous allons résoudre numériquement par le schéma implicite d'Euler le problème de conduction thermique dans une barre de longueur  $L = 6$  :

$$\frac{\partial u}{\partial t}(t, x) - \nu \frac{\partial^2 u}{\partial x^2}(t, x) = 0, \quad \forall (t, x) \in [0; T] \times ]0; L[, \quad (5.1)$$

$$u(t_0, x) = u_0(x), \quad \forall x \in [0; L], \quad (5.2)$$

$$u(t, 0) = u_g(t), \quad \forall t \in [0; T], \quad (5.3)$$

$$u(t, L) = u_d(t), \quad \forall t \in [0; T]. \quad (5.4)$$

avec  $T = 10$ ,  $u_0(x) = 10, \forall x \in [0, L]$

$$u_g(t) = \begin{cases} 10 + 90t, & \forall t \in [0, 1] \\ 100, & \forall t > 1 \end{cases}$$

$$u_d(t) = \begin{cases} 10 + 70t, & \forall t \in [0, 1] \\ 80, & \forall t > 1 \end{cases}$$

Q. 9 ★★☆☆

- [code]** Ecrire le programme `heat1d.bar01` permettant de résoudre ce problème par le schéma d'Euler implicite.
- [rapport]** Executer ce programme pour différentes valeurs de  $\nu$  (par exemple  $\nu = 0.1$ ,  $\nu = 1$  et  $\nu = 3$ . Qu'observe-t-on ?

## 6 Conditions aux limites génériques

On souhaite résoudre numériquement le B.V.P. suivant

$$\frac{\partial u}{\partial t}(t, x) - \nu \frac{\partial^2 u}{\partial x^2}(t, x) = f(t, x), \quad \forall (t, x) \in ]t_0; t_0 + T] \times ]a; b[, \quad (6.1)$$

$$u(t_0, x) = u_0(x), \quad \forall x \in [a; b], \quad (6.2)$$

$$\delta_a u(t, a) + \mu_a \frac{\partial u}{\partial x}(t, a) = g_a(t), \quad \forall t \in [t_0; t_0 + T], \quad (6.3)$$

$$\delta_b u(t, b) + \mu_b \frac{\partial u}{\partial x}(t, b) = g_b(t), \quad \forall t \in [t_0; t_0 + T], \quad (6.4)$$

avec  $\nu$  un réel strictement positif,  $t_0 \in \mathbb{R}$ ,  $T > 0$ ,  $(a, b) \in \mathbb{R}^2$ ,  $a < b$ ,  $(\delta_a, \delta_b) \in \llbracket 0, 1 \rrbracket^2$ . Pour s'assurer de l'existence et l'unicité d'une solution à ce problème, on prend

$$\begin{cases} \mu_a \leq 0 & \text{si } \delta_a = 1, \\ \mu_a < 0 & \text{si } \delta_a = 0, \end{cases} \quad \text{et} \quad \begin{cases} \mu_b \geq 0 & \text{si } \delta_b = 1, \\ \mu_b > 0 & \text{si } \delta_b = 0. \end{cases} \quad (6.5)$$

Q. 10 [rapport] ★★★★★

- Ecrire de **de manière détaillée** une discrétisation complète de (6.1) à (6.4) utilisant un **schéma implicite**.
- Montrer que cette discrétisation complète s'écrit sous la forme (3.1) où l'on explicitera les matrices  $\mathbb{E}$  et  $\mathbb{F}$ , ainsi que le vecteur  $\mathbf{c}^{[n]}$ .

Q. 11 ★★★★★

- [code]** En utilisant une méthodologie similaire à celle du problème avec conditions de Dirichlet, écrire un programme `heat1d.implicit_genCL` permettant de
  - résoudre par un schéma implicite le problème (6.1) à (6.4).
  - représenter les solutions exactes et numériques,
  - représenter l'erreur commise.
 Lors de la programmation, les données associées aux conditions aux limites seront intégrées à la structure BVP.
- [code]** Ecrire un programme `heat1d.implicit_genCL_spradius` permettant de calculer numériquement le rayon spectral de la matrice  $\mathbb{B} = \mathbb{E}^{-1}\mathbb{F}$  à  $N_x$  fixé et en faisant varier  $N_t$  de telle sorte que la condition C.F.L. ne soit pas toujours vérifiée. On affichera pour chaque valeur de  $N_t$ , le rayon spectral de  $\mathbb{B}$  et la valeur de  $\nu \frac{h_t}{h_x^2}$ .
- [rapport]** Que peut-on tirer/conjecturer des résultats du programme précédent ?

## 7 Schéma de Crank-Nicolson

Le schéma de Crank-Nicolson pour la discrétisation de l'équation de la chaleur (6.1) est donné par

$$\frac{u_{i+1}^{n+1} - u_i^n}{\Delta t} - \frac{\nu}{2} \left( \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{\Delta x^2} + \frac{u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}}{\Delta x^2} \right) = \frac{1}{2} (f_i^n + f_i^{n+1}). \quad (7.1)$$

Ce schéma est d'ordre 2 en temps et en espace et il est inconditionnellement stable.

Q. 12 [rapport] ★★★★★

- Ecrire de **de manière détaillée** une discrétisation complète de (6.1) à (6.4) utilisant le schéma de Crank-Nicolson (7.1).
- Explicitiez le système linéaire à résoudre à chaque itération en temps.

Q. 13 ★★★★★

- [code]** En utilisant une méthodologie similaire à celle décrite en section 6, Q.10 et Q.11, écrire un programme `heat1d.CrankNicolson_genCL` permettant de
  - résoudre par le schéma de Crank-Nicolson le problème (6.1) à (6.4).
  - représenter les solutions exactes et numériques,
  - représenter l'erreur commise.
- [code]** Ecrire un programme `heat1d.CrankNicolson_genCL_spradius` permettant de calculer numériquement

le rayon spectral de la matrice  $\mathbb{B} = \mathbb{E}^{-1}\mathbb{F}$  à  $N_x$  fixé et en faisant varier  $N_t$  de telle sorte que la condition C.F.L. ne soit pas toujours vérifiée. On affichera pour chaque valeur de  $N_t$ , le rayon spectral de  $\mathbb{B}$  et la valeur de  $\nu \frac{h_t}{h_x^2}$ .

c. **[rapport]** Que peut-on tirer/conjecturer des résultats du programme précédent ?

La figure 2 est une représentation possible du comportement des schémas d'Euler implicite et de Crank-Nicolson. Elle a été établie avec  $a = 0$ ,  $b = 2\pi$ ,  $t_0 = 0$ ,  $T = 2$ ,  $\nu = 2$ ,  $\delta_1 = \delta_b = 1$ ,  $\mu_a = -1$ ,  $\mu_b = 2$  et les autres données ont été choisies pour avoir

$$u(t, x) = \cos(kt) \cos(x), \quad k = 5$$

comme solution exacte. On peut noter que le schéma explicite d'Euler n'est pas proposé car il n'a pas été implémenté avec des conditions aux limites générique ! Toutefois, la *barrière* C.F.L. a été représentée.

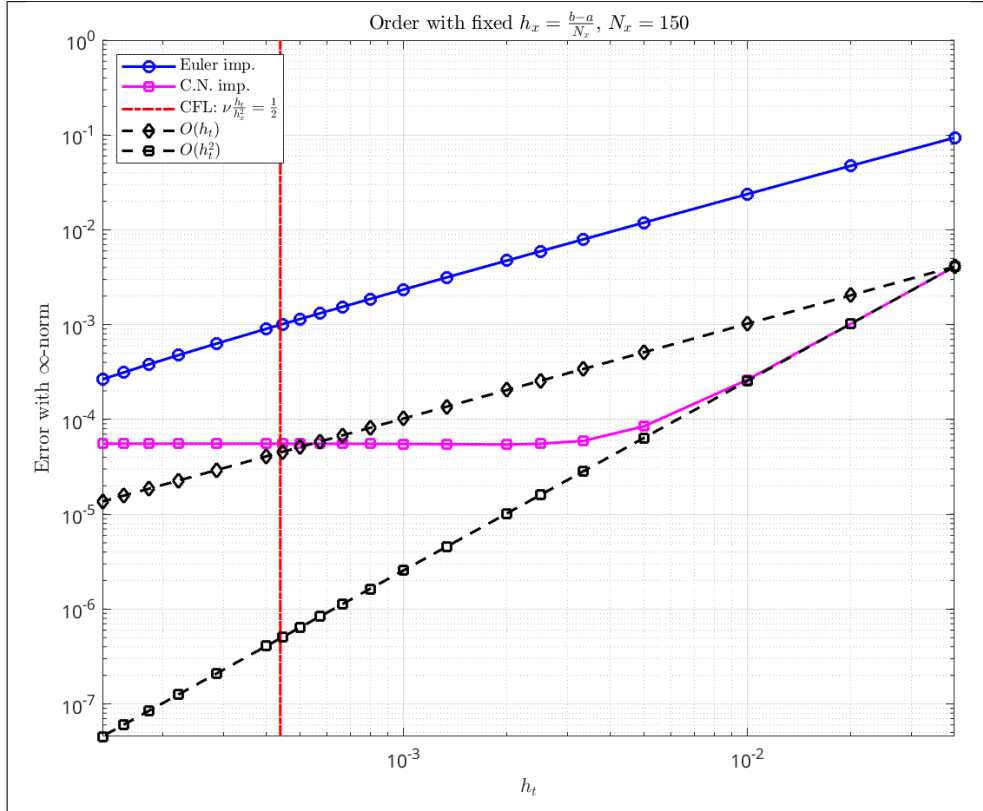


FIGURE 2 – Ordre,  $N_t$  dans [50,100,200:200:1000,1500:500:5000,7000:2000:15000]

Q. 14 ★★☆☆

a. **[rapport]** Expliquer tout ce que l'on peut interpréter/comprendre à partir de la figure 2.

b. **[code]** Ecrire un programme `head1d.ordre_genCL` permettant de la figure 2 le plus exactement possible.