

Méthodes Numériques (G3SEMN)*

Sup'Galilée, Ingénieurs Energétique, 1ère année

François Cuvelier

Laboratoire d'Analyse Géométrie et Applications
Institut Galilée
Université Paris XIII.

2026/03/24

- Chapitre I : Algorithmique numérique
- Chapitre II : Dérivation numérique
- Chapitre III : Résolution numérique des E.D.O.
- Chapitre IV : Résolution numérique des E.D.P.**

1 Exemples d'E.D.P.

2 Méthodes de résolution numérique d'EDP

3 Méthode des différences finies 1D

4 **Problème modèle évolutif**

- Schéma explicite
- Schéma implicite



EDP modèle instationnaire en dimension 1 : équation de la chaleur

Trouver $u : [0, T] \times [a, b] \longrightarrow \mathbb{R}$ telle que

$$\frac{\partial u}{\partial t}(t, x) - D \frac{\partial^2 u}{\partial x^2}(t, x) = f(t, x), \quad \forall (t, x) \in]0, T[\times]a, b[, \quad (1)$$

$$u(0, x) = u_0(x), \quad \forall x \in [a, b] \quad (2)$$

$$-D \frac{\partial u}{\partial x}(t, a) = \alpha(t), \quad \forall t \in [0, T] \quad (3)$$

$$u(t, b) = \beta(t), \quad \forall t \in [0, T] \quad (4)$$

où $a < b$, $D > 0$ (coefficient de diffusivité), $\alpha : [0, T] \longrightarrow \mathbb{R}$, $\beta : [0, T] \longrightarrow \mathbb{R}$, $u_0 : [a, b] \longrightarrow \mathbb{R}$ et $f : [0, T] \times [a, b] \longrightarrow \mathbb{R}$ donnés.

condition de compatibilité :

$$u_0(b) = \beta(0). \quad (5)$$

$$x_i = a + ih_x,$$

$$\forall i \in \llbracket 0, N_x \rrbracket,$$

$$\text{avec } h_x = (b - a)/N_x$$

$$t^n = nh_t,$$

$$\forall n \in \llbracket 0, N_t \rrbracket,$$

$$\text{avec } h_t = T/N_t.$$

Objectif: Trouver $u_i^n \approx u(t^n, x_i), \forall n \in \llbracket 0, N_t \rrbracket, \forall i \in \llbracket 0, N_x \rrbracket$

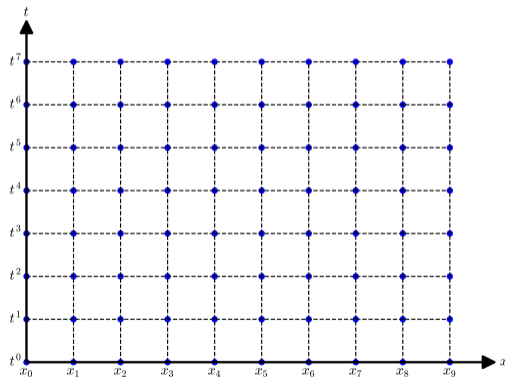


Figure: Grille espace-temps avec $N_t = 7$ et $N_x = 9$.



EDP modèle d'évolution en dimension 1 : équation de la chaleur, formulation aux points de discrétisation

Trouver $u(t^n, x_i) \in \mathbb{R}$, $\forall n \in \llbracket 0, N_t \rrbracket$, $\forall i \in \llbracket 0, N_x \rrbracket$, tels que

$$\frac{\partial u}{\partial t}(t^n, x_i) - D \frac{\partial^2 u}{\partial x^2}(t^n, x_i) = f(t^n, x_i), \quad (6)$$

$$u(t^0, x_i) = u_0(x_i), \quad \forall i \in \llbracket 0, N_x \rrbracket, \quad (7)$$

$$-D \frac{\partial u}{\partial x}(t^n, x_0) = \alpha(t^n), \quad \forall n \in \llbracket 0, N_t \rrbracket \quad (8)$$

$$u(t^n, x_{N_x}) = \beta(t^n), \quad \forall n \in \llbracket 0, N_t \rrbracket \quad (9)$$

⇒ il nous faut maintenant discrétiser les opérateurs de dérivation

$$\frac{\partial u}{\partial t}, \quad \frac{\partial^2 u}{\partial x^2} \quad \text{et} \quad \frac{\partial u}{\partial x}$$


Proposition 4.1 : (admis)

Soit U un ouvert non vide de \mathbb{R}^n et f une application $f : U \subset \mathbb{R}^n \longrightarrow \mathbb{R}$ avec $f \in \mathcal{C}^{r+1}(U)$. Soient $\mathbf{x} \in U$, $i \in \llbracket 1, n \rrbracket$, et $h \in \mathbb{R}^*$ vérifiant $\forall t \in [0, 1]$, $\mathbf{x} + t h \mathbf{e}^{[i]} \in U$ où $\mathbf{e}^{[i]}$ est le i -ème vecteur de la base canonique de \mathbb{R}^n .

Alors il existe $\theta \in]0, 1[$ tel quel

$$f(\mathbf{x} + h \mathbf{e}^{[i]}) = f(\mathbf{x}) + \sum_{k=1}^r \frac{h^k}{k!} \frac{\partial^k f}{\partial x_i^k}(\mathbf{x}) + \frac{h^{r+1}}{(r+1)!} \frac{\partial^{r+1} f}{\partial x_i^{r+1}}(\mathbf{x} + \theta h \mathbf{e}^{[i]}) \quad (10)$$

où $\mathbf{e}^{[i]}$ est le i -ème vecteur de la base canonique de \mathbb{R}^n . Cette formule est le développement limité de f à l'ordre r en \mathbf{x} dans la direction $\mathbf{e}^{[i]}$

$$f(\mathbf{x} + h \mathbf{e}^{[i]}) = f(\mathbf{x}) + \sum_{k=1}^r \frac{h^k}{k!} \frac{\partial^k f}{\partial x_i^k}(\mathbf{x}) + \mathcal{O}(h^{r+1}) \quad (11)$$

On déduit des développements de Taylor :

$$\frac{\partial^2 u}{\partial x^2}(t, x) = \frac{u(t, x + h) - 2u(t, x) + u(t, x - h)}{h^2} + \mathcal{O}(h^2)$$

Avec $h = h_x$ on obtient

$$\frac{\partial^2 u}{\partial x^2}(t^n, x_i) = \frac{u(t^n, x_{i+1}) - 2u(t^n, x_i) + u(t^n, x_{i-1}))}{h_x^2} + \mathcal{O}(h_x^2) \quad (12)$$

On déduit des développements de Taylor :

$$\begin{aligned}\frac{\partial u}{\partial t}(t, x) &= \frac{u(t+h, x) - u(t, x)}{h} + \mathcal{O}(h) \\ \frac{\partial u}{\partial t}(t, x) &= \frac{u(t, x) - u(t-h, x)}{h} + \mathcal{O}(h).\end{aligned}$$

Avec $h = h_t$ on obtient

$$\frac{\partial u}{\partial t}(t^n, x_i) = \frac{u(t^{n+1}, x_i) - u(t^n, x_i)}{h_t} + \mathcal{O}(h_t) \quad (13)$$

$$\frac{\partial u}{\partial t}(t^n, x_i) = \frac{u(t^n, x_i) - u(t^{n-1}, x_i)}{h_t} + \mathcal{O}(h_t). \quad (14)$$

On déduit des développements de Taylor :

$$\frac{\partial u}{\partial x}(t, x) = \frac{u(t, x+h) - u(t, x)}{h} + \mathcal{O}(h)$$

$$\frac{\partial u}{\partial x}(t, x) = \frac{u(t, x) - u(t, x-h)}{h} + \mathcal{O}(h)$$

$$\frac{\partial u}{\partial x}(t, x) = \frac{u(t, x+h) - u(t, x-h)}{2h} + \mathcal{O}(h^2)$$

$$\frac{\partial u}{\partial x}(t, x) = \frac{-3u(t, x) + 4u(t, x+h) - u(t, x+2h)}{2h} + \mathcal{O}(h^2)$$

$$\frac{\partial u}{\partial x}(t, x) = \frac{3u(t, x) - 4u(t, x-h) + u(t, x-2h)}{2h} + \mathcal{O}(h^2)$$

On veut approcher $\frac{\partial u}{\partial x}(t^n, x_0)$ à l'ordre 2! \Rightarrow 4ème approximation.

Avec $h = h_x$ on obtient

$$\frac{\partial u}{\partial x}(t^n, x_0) = \frac{-3u(t^n, x_0) + 4u(t^n, x_1) - u(t^n, x_2)}{2h_x} + \mathcal{O}(h_x^2) \quad (15)$$

- 1 Exemples d'E.D.P.
 - Equation de Laplace/Poisson
 - Equation de la chaleur
 - Equation des ondes
- 2 Méthodes de résolution numérique d'EDP

- 3 Méthode des différences finies 1D
 - EDP stationnaire 1D + Dirichlet
 - EDP stationnaire + CL mixtes
- 4 **Problème modèle évolutif**
 - Schéma explicite
 - Schéma implicite

Schéma explicite en temps pour l'EDP (6) à (9)

On rappelle (6)

$$\frac{\partial u}{\partial t}(t^n, x_i) - D \frac{\partial^2 u}{\partial x^2}(t^n, x_i) = f(t^n, x_i), \quad \forall n \in \llbracket 0, N_t \rrbracket, \quad \forall i \in \llbracket 0, N_x \rrbracket,$$

qui devient avec (13) et (12)

$$\begin{aligned} & \frac{u(t^{n+1}, x_i) - u(t^n, x_i)}{h_t} + \mathcal{O}(h_t) \\ & - D \frac{u(t^n, x_{i+1}) - 2u(t^n, x_i) + u(t^n, x_{i-1}))}{h_x^2} + \mathcal{O}(h_x^2) = f(t^n, x_i) \end{aligned} \quad (16)$$

avec $n \in \llbracket 0, N_t \rrbracket$ et $i \in \llbracket 0, N_x \rrbracket$.

En utilisant (14) en lieu et place de (13) on obtient un schéma implicite...

Schéma explicite en temps pour l'EDP (6) à (9)

Un schéma numérique d'ordre 1 en temps et d'ordre 2 en espace pour (6): $\forall n \in \llbracket 0, N_t \llbracket, \forall i \in \llbracket 0, N_x \llbracket$

$$\frac{\mathbf{u}_i^{n+1} - \mathbf{u}_i^n}{h_t} - D \frac{\mathbf{u}_{i+1}^n - 2\mathbf{u}_i^n + \mathbf{u}_{i-1}^n}{h_x^2} = \mathbf{f}_i^n \quad (17)$$

avec $\mathbf{f}_i^n = f(t^n, x_i)$ et (en espérant) $\mathbf{u}_i^n \approx u(t^n, x_i)$.

(17) est équivalent à

$$\mathbf{u}_i^{n+1} = \mathbf{u}_i^n + D \frac{h_t}{h_x^2} (\mathbf{u}_{i+1}^n - 2\mathbf{u}_i^n + \mathbf{u}_{i-1}^n) + h_t \mathbf{f}_i^n \quad (18)$$

Et (7), (8), (9)?

Schéma explicite en temps pour l'EDP (6) à (9)

On rappelle respectivement (7) et (9):

$$\begin{aligned}u(t^0, x_i) &= u_0(x_i), & \forall i \in \llbracket 0, N_x \rrbracket \\u(t^n, x_{N_x}) &= \beta(t^n), & \forall n \in \llbracket 0, N_t \rrbracket\end{aligned}$$

qui donne immédiatement (sans approximation)

$$\begin{aligned}\mathbf{u}_i^0 &= u_0(x_i), & \forall i \in \llbracket 0, N_x \rrbracket \\ \mathbf{u}_{N_x}^n &= \beta(t^n), & \forall n \in \llbracket 0, N_t \rrbracket\end{aligned} \tag{19}$$

Et (8)?

Schéma explicite en temps pour l'EDP (6) à (9)

On rappelle (8) $\forall n \in \llbracket 0, N_t \rrbracket$

$$-D \frac{\partial u}{\partial x}(t^n, x_0) = \alpha(t^n)$$

qui donne avec (15)

$$-D \frac{-3u(t^n, x_0) + 4u(t^n, x_1) - u(t^n, x_2)}{2h_x} + \mathcal{O}(h_x^2) = \alpha(t^n)$$

Un schéma numérique d'ordre 2 en espace pour (8):

$$-D \frac{-3\mathbf{u}_0^n + 4\mathbf{u}_1^n - \mathbf{u}_2^n}{h_x} = \alpha(t^n)$$

ou encore

$$\mathbf{u}_0^n = \frac{1}{3} \left(\frac{2h_x}{D} \alpha(t^n) + 4\mathbf{u}_1^n - \mathbf{u}_2^n \right). \quad (20)$$

Schéma explicite en temps pour l'EDP (6) à (9)

En résumé, avec $E = D \frac{h_t}{h_x^2}$ et $C = 1 - 2E$

$$\mathbf{u}_i^{n+1} = C\mathbf{u}_i^n + E(\mathbf{u}_{i+1}^n + \mathbf{u}_{i-1}^n) + h_t \mathbf{f}_i^n, \quad \left\{ \begin{array}{l} \forall n \in \llbracket 0, N_t \llbracket, \\ \forall i \in \llbracket 0, N_x \llbracket \end{array} \right. \quad (18)$$

$$\mathbf{u}_i^0 = u_0(x_i), \quad \forall i \in \llbracket 0, N_x \llbracket \quad (21)$$

$$\mathbf{u}_0^n = \frac{1}{3} \left(\frac{2h_x}{D} \alpha(t^n) + 4\mathbf{u}_1^n - \mathbf{u}_2^n \right), \quad \forall n \in \llbracket 0, N_t \llbracket \quad (20)$$

$$\mathbf{u}_{N_x}^n = \beta(t^n), \quad \forall n \in \llbracket 0, N_t \llbracket \quad (19)$$

Peut-on calculer $(\mathbf{u}_i^{n+1})_{i=0}^{N_x}$ connaissant $(\mathbf{u}_i^n)_{i=0}^{N_x}$?

$$\mathbf{u}_i^{n+1} = C\mathbf{u}_i^n + E(\mathbf{u}_{i+1}^n + \mathbf{u}_{i-1}^n) + h_t \mathbf{f}_i^n, \quad \forall i \in \llbracket 0, N_x \llbracket, \quad (22)$$

$$\mathbf{u}_{N_x}^{n+1} = \beta(t^{n+1}), \quad (23)$$

$$\mathbf{u}_0^{n+1} = \frac{1}{3} \left(\frac{2h_x}{D} \alpha(t^{n+1}) + 4\mathbf{u}_1^{n+1} - \mathbf{u}_2^{n+1} \right). \quad (24)$$

Schéma explicite en temps pour l'EDP (6) à (9)

L'algorithme formel est donc le suivant :

- 1: $\mathbf{u}_i^0 \leftarrow u_0(x_i), \forall i \in \llbracket 0, N_x \rrbracket$
- 2: **Pour** $n \leftarrow 0$ à $N_t - 1$ **faire**
- 3: $\mathbf{u}_i^{n+1} \leftarrow C\mathbf{u}_i^n + E(\mathbf{u}_{i+1}^n + \mathbf{u}_{i-1}^n) + h_t \mathbf{f}_i^n, \forall i \in \llbracket 0, N_x \rrbracket$
- 4: $\mathbf{u}_{N_x}^{n+1} \leftarrow \beta(t^{n+1})$
- 5: $\mathbf{u}_0^{n+1} \leftarrow \frac{1}{3} \left(\frac{2h_x}{D} \alpha(t^{n+1}) + 4\mathbf{u}_1^{n+1} - \mathbf{u}_2^{n+1} \right)$
- 6: **Fin**

Exercice 1

Q. 1

*Ecrire une fonction algorithmique **Heat1Dex** permettant de retourner la discrétisation en temps, la discrétisation en espace et l'ensemble des \mathbf{u}_i^n , $i \in \llbracket 0, N_x \rrbracket$, $n \in \llbracket 1, N_t \rrbracket$ calculés par le schéma explicite en temps pour l'EDP (6) à (9).*

Q. 2

Ecrire un programme utilisant cette fonction et permettant de calculer la solution numérique d'un problème dont on connaît la solution exacte.

Correction: les pages qui suivent ...

Schéma explicite en temps pour l'EDP (6) à (9)

- Données :**
- a, b : deux réels $a < b$,
 - T : $T > 0$,
 - D : D réel strictement positif, (coefficient de diffusivité)
 - f : $f : [0, T] \times [a, b] \longrightarrow \mathbb{R}$,
 - u_0 : $u_0 : [a, b] \longrightarrow \mathbb{R}$,
 - α : $\alpha : [0, T] \longrightarrow \mathbb{R}$,
 - β : $\beta : [0, T] \longrightarrow \mathbb{R}$, tel que $\beta(0) = u_0(b)$,
 - N_x : $N_x \in \mathbb{N}^*$, nombre de discrétisation en espace,
 - N_t : $N_t \in \mathbb{N}^*$, nombre de discrétisation en temps.

Algorithme Fonction Heat1Dex (version non vectorisée)

```
1: Fonction [t, x, U] ← Heat1Dex( a, b, T, D, f, u0, α, β, Nx, Nt )
2:   t ← DisReg(0, T, Nt)
3:   ht ← T/Nt
4:   x ← DisReg(a, b, Nx)
5:   hx ← (b - a)/Nx
6:   E ← D * ht / hx2, C ← 1 - 2E
7:   Pour i ← 1 à Nx + 1 faire
8:     U(i, 1) ← u0(x(i))
9:   Fin
10:  Pour n ← 1 à Nt faire
11:    Pour i ← 2 à Nx faire
12:      U(i, n + 1) ← C * U(i, n) + E * (U(i + 1, n) + U(i - 1, n)) + ht * f(t(n), x(i))
13:    Fin
14:    U(Nx, n + 1) ← β(t(n + 1))
15:    U(1, n + 1) ← (hx / D * α(t(n + 1)) + 4U(2, n + 1) - U(3, n + 1))/3
16:  Fin
17: Fin
```

▷ Condition initiale

▷ Boucle en temps

▷ Schéma

Algorithme Fonction Heat1Dex (version vectorisée)

```
1: Fonction [ $\mathbf{t}, \mathbf{x}, \mathbb{U}$ ]  $\leftarrow$  Heat1Dex(  $a, b, T, D, f, u_0, \alpha, \beta, N_x, N_t$  )
2:    $\mathbf{t} \leftarrow$  DisReg(0,  $T, N_t$ ),  $h_t \leftarrow T/N_t$ 
3:    $\mathbf{x} \leftarrow$  DisReg( $a, b, N_x$ ),  $h_x \leftarrow (b - a)/N_x$ 
4:    $E \leftarrow D \frac{h_t}{h_x^2}$ ,  $C \leftarrow 1 - 2E$ 
5:    $\mathbb{U}(:, 1) \leftarrow u_0(\mathbf{x})$ 
6:    $I \leftarrow [2 : N_x]$ 
7:   Pour  $n \leftarrow 1$  à  $N_t$  faire
8:      $\mathbb{U}(I, n + 1) \leftarrow C * \mathbb{U}(I, n) - E * (\mathbb{U}(I + 1, n) + \mathbb{U}(I - 1, n)) + h_t * f(\mathbf{t}(n), \mathbf{x}(I))$ 
9:      $\mathbb{U}(N_x, n + 1) \leftarrow \beta(\mathbf{t}(n + 1))$ 
10:     $\mathbb{U}(1, n + 1) \leftarrow (\frac{h_x}{D} * \alpha(\mathbf{t}(n + 1)) + 4\mathbb{U}(2, n + 1) - \mathbb{U}(3, n + 1))/3$ 
11:   Fin
12: Fin
```

- ▷ Condition initiale
- ▷ Indices des points intérieurs
- ▷ Boucle en temps

```

1 function [t,x,U]=Heat1DexLight(a,b,T,D,f,u0,alpha,beta,Nx,Nt)
2     dt=T/Nt;      t=0:dt:T;
3     dx=(b-a)/Nx; x=[a:dx:b]';
4     U=zeros(Nx+1,Nt+1);
5     E=D*dt/dx^2; C=1-2*E;
6     I=2:Nx;
7     U(:,1)=u0(x);
8     for n=1:Nt
9         U(I,n+1)=C*U(I,n) + E*(U(I+1,n)+U(I-1,n)) + dt*f(t(n),x(I));
10        U(Nx+1,n+1)=beta(t(n+1));
11        U(1,n+1)=((2*dx/D)*alpha(t(n+1)) + 4*U(2,n+1) - U(3,n+1))/3;
12    end
13 end

```

Listing: fonction pour la résolution de l'EDP de la chaleur (6)-(9) par le schéma explicite (22)-(24)

Application avec solution exacte

On choisi

$$u(t, x) \stackrel{\text{def}}{=} \cos(t) \cos(x)$$

et en injectant dans l'EDP on obtient

$$f(t, x) \stackrel{\text{def}}{=} D \cos(t) \cos(x) - \sin(t) \cos(x),$$

$$u_0(x) \stackrel{\text{def}}{=} \cos(x),$$

$$\alpha(t) \stackrel{\text{def}}{=} D \cos(t) \sin(a),$$

$$\beta(t) \stackrel{\text{def}}{=} \cos(t) \cos(b).$$

Calcul numérique avec

$$D = 1, a = 0, b = 2\pi, T = 10, N_x = 50 \text{ et } N_t = 5000$$

Application avec solution exacte: numerical solution $N_x = 50$

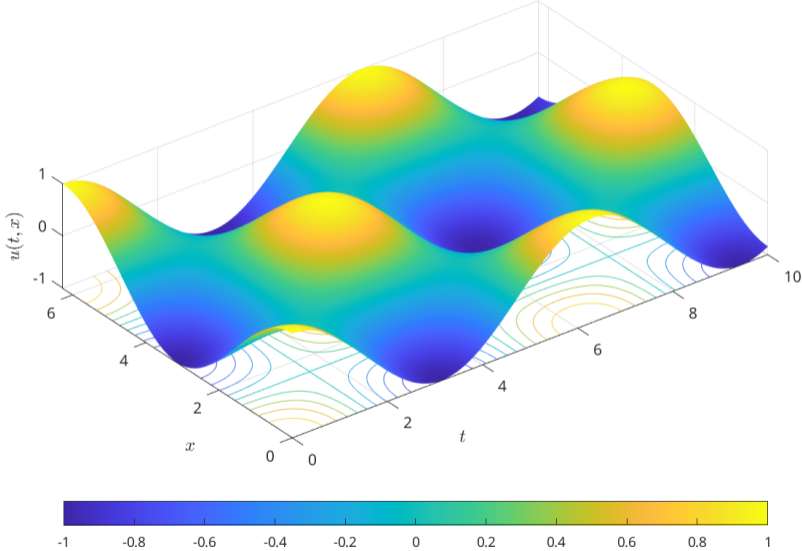
```
1 a=0;b=2*pi;Nx=50;
2 T=10;Nt=5000;
3 D=1;
4 u=@(t,x) cos(t).*cos(x); % solution exacte
5 f=@(t,x) D*cos(t).*cos(x) - sin(t).*cos(x);
6 beta=@(t) u(t,b);
7 alpha=@(t) D*cos(t).*sin(a);
8 u0=@(x) u(0,x);
9
10 [t,x,U]=Heat1DexLight(a,b,T,D,f,u0,alpha,beta,Nx,Nt);
```

Listing: Equation de la chaleur 1D avec solution exacte. Calcul par schéma explicite : code Matlab, fichier sampleHeat1Dex01.m

```
1 sampleHeat1Dex01
2
3 surfc(t,x,U)
4 axis equal;shading interp;colormap parula;
5 xlabel('$t$','interpreter','latex','fontsize',12)
6 ylabel('$x$','interpreter','latex','fontsize',12)
7 zlabel('$u(t,x)$','interpreter','latex','fontsize',12)
8 colorbar('location','SouthOutside')
```

Listing: Equation de la chaleur 1D avec solution exacte. Représentation de la solution calculée : code Matlab

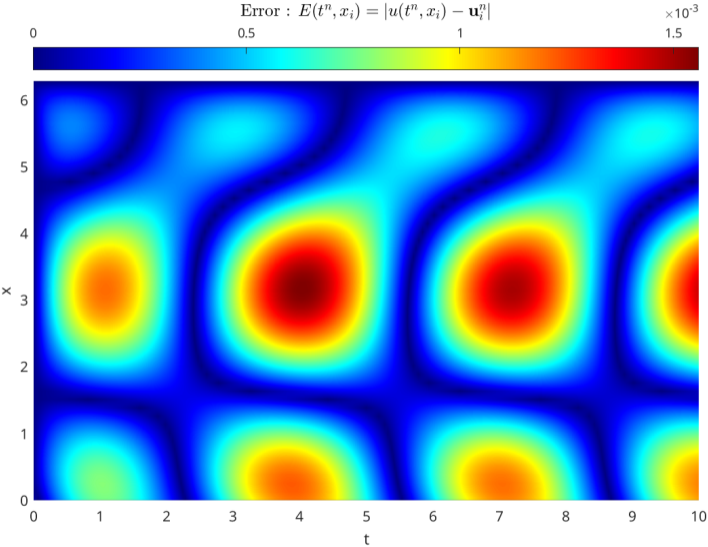
Application avec solution exacte: numerical solution $N_x = 50$



```
1 sampleHeat1Dex01
2 [T,X]=meshgrid(t,x);
3 Uex=u(T,X);
4
5 pcolor(t,x,abs(U-Uex))
6 xlabel('t'),ylabel('x')
7 shading interp, colormap(jet)
8 axis image;
9 h=colorbar('Location','northoutside');
10 set(get(h,'title'),'string','Error:  $E(t^n, x_i) = |u(t^n, x_i) - \mathbf{u}_i^n|$ ', ...
11     'interpreter','latex','fontsize',12)
```

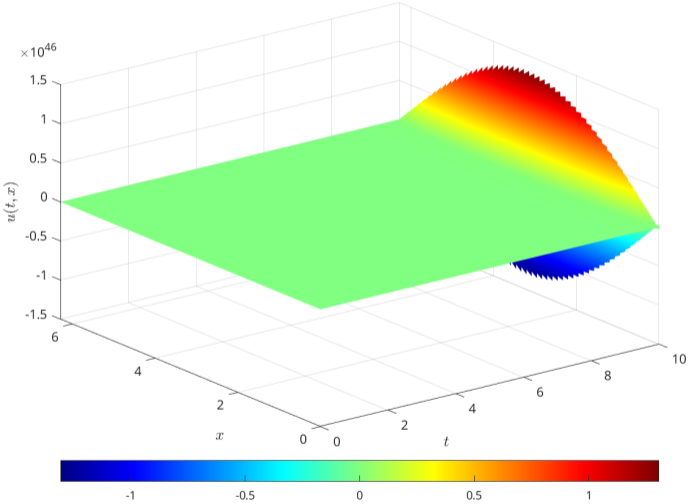
Listing: Equation de la chaleur 1D avec solution exacte. Représentation de l'erreur : code Matlab

Application avec solution exacte: error $N_x = 50$



On change uniquement N_x de 50 à 100.

👉 A quoi peut-on s'attendre?



Application avec solution exacte: Phénomène d'instabilité

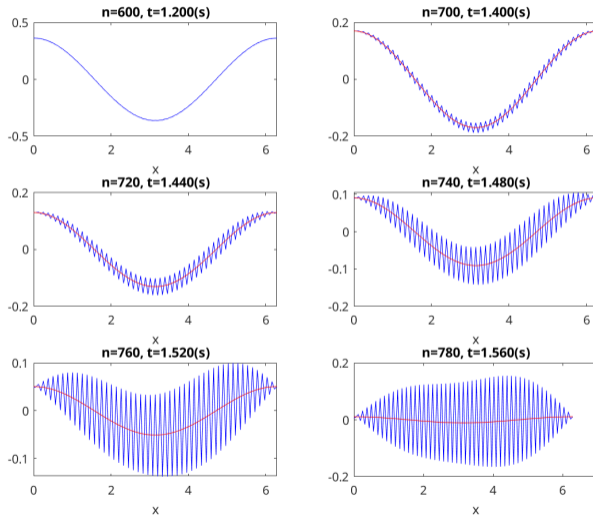


Figure: Equation de la chaleur 1D avec solution exacte (en rouge). Debut du phénomène d'instabilité.

Etude de la stabilité au sens de Von Neumann du schéma explicite donne la **condition de C.F.L.** (R. Courant, K. Friedrichs, and H. Lewy en 1928):
le schéma est stable si

$$D \frac{h_t}{h_x^2} \leq \frac{1}{2}. \quad (25)$$

Illustration: représenter en échelle logarithmique l'erreur commise en fonction de h_x et h_t calculée par

$$E(h_x, h_t) = \max_{i \in \llbracket 0, N_x \rrbracket, n \in \llbracket 0, N_t \rrbracket} |u_i^n - u_{\text{ex}}(t^n, x_i)|$$

où u_{ex} est la solution exacte.

Application avec solution exacte: Phénomène d'instabilité

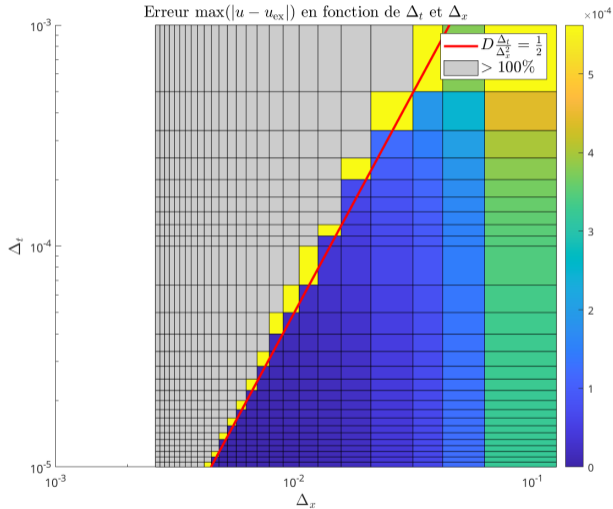


Figure: Equation de la chaleur 1D avec solution exacte. Condition de CFL $\Delta_x \leq \sqrt{2 D \Delta_t}$

- 1 Exemples d'E.D.P.
 - Equation de Laplace/Poisson
 - Equation de la chaleur
 - Equation des ondes
- 2 Méthodes de résolution numérique d'EDP

- 3 Méthode des différences finies 1D
 - EDP stationnaire 1D + Dirichlet
 - EDP stationnaire + CL mixtes
- 4 **Problème modèle évolutif**
 - Schéma explicite
 - **Schéma implicite**



EDP modèle d'évolution en dimension 1 : équation de la chaleur, formulation aux points de discrétisation

Trouver $u(t^n, x_i) \in \mathbb{R}$, $\forall n \in \llbracket 0, N_t \rrbracket$, $\forall i \in \llbracket 0, N_x \rrbracket$, tels que

$$\frac{\partial u}{\partial t}(t^n, x_i) - D \frac{\partial^2 u}{\partial x^2}(t^n, x_i) = f(t^n, x_i), \quad (6)$$

$$u(t^0, x_i) = u_0(x_i), \quad \forall i \in \llbracket 0, N_x \rrbracket, \quad (7)$$

$$-D \frac{\partial u}{\partial x}(t^n, x_0) = \alpha(t^n), \quad \forall n \in \llbracket 0, N_t \rrbracket \quad (8)$$

$$u(t^n, x_{N_x}) = \beta(t^n), \quad \forall n \in \llbracket 0, N_t \rrbracket \quad (9)$$

Pour approcher $\frac{\partial u}{\partial t}(t^n, x_i)$, on utilise cette fois

$$\frac{\partial u}{\partial t}(t^n, x_i) = \frac{u(t^n, x_i) - u(t^{n-1}, x_i)}{h_t} + \mathcal{O}(h_t). \quad (14)$$

Schéma implicite en temps pour l'EDP (6) à (9)

Un schéma numérique d'ordre 1 en temps et d'ordre 2 en espace pour (6): $\forall n \in \llbracket 0, N_t \rrbracket, \forall i \in \llbracket 0, N_x \rrbracket$

$$\frac{\mathbf{u}_i^n - \mathbf{u}_i^{n-1}}{h_t} - D \frac{\mathbf{u}_{i+1}^n - 2\mathbf{u}_i^n + \mathbf{u}_{i-1}^n}{h_x^2} = \mathbf{f}_i^n, \quad (26)$$

avec $\mathbf{f}_i^n = f(t^n, x_i)$ et (en espérant) $\mathbf{u}_i^n \approx u(t^n, x_i)$.

(29) est équivalent à

$$\mathbf{u}_i^n - D \frac{h_t}{h_x^2} (\mathbf{u}_{i+1}^n - 2\mathbf{u}_i^n + \mathbf{u}_{i-1}^n) = \mathbf{u}_i^{n-1} + h_t \mathbf{f}_i^n. \quad (27)$$

Ce schéma est **implicite en temps** : il n'est pas possible de calculer explicitement \mathbf{u}_i^n en fonction des \mathbf{u}_i^{n-1} (au temps précédent).

Schéma implicite en temps pour l'EDP (6) à (9)

Le calcul des \mathbf{u}_i^n , $\forall i \in \llbracket 0, N_x \rrbracket$, est possible en résolvant les $N_x + 1$ équations **linéaires** suivantes où l'on note $E = D \frac{h_t}{h_x^2}$ et $C = 1 + 2E$:

$$3\mathbf{u}_0^n - 4\mathbf{u}_1^n + \mathbf{u}_2^n = 2 \frac{h_x}{D} \alpha(t^n). \quad (28)$$

$$C\mathbf{u}_i^n - E(\mathbf{u}_{i+1}^n + \mathbf{u}_{i-1}^n) = \mathbf{u}_i^{n-1} + h_t \mathbf{f}_i^n, \quad \forall i \in \llbracket 0, N_x \rrbracket \quad (29)$$

$$\mathbf{u}_{N_x}^n = \beta(t^n), \quad (30)$$

Schéma implicite en temps pour l'EDP (6) à (9)

$$\mathbb{A} \mathbf{U}^n = \mathbf{b}^n \quad (31)$$

avec $\mathbb{A} \in \mathcal{M}_{N_x+1}(\mathbb{R})$, $\mathbf{b}^n \in \mathbb{R}^{N_x+1}$ et

$$\mathbf{U}^n = \begin{pmatrix} \mathbf{u}_0^n \\ \mathbf{u}_1^n \\ \vdots \\ \mathbf{u}_{N_x-1}^n \\ \mathbf{u}_{N_x}^n \end{pmatrix} \in \mathbb{R}^{N_x+1}, \quad \text{i.e. } \mathbf{U}^n(i) = \mathbf{u}_{i-1}^n, \quad \forall i \in \llbracket 1, N_x + 1 \rrbracket$$

$$\mathbb{A}U^n = b^n \iff \left(\begin{array}{cccc|cccc} 3 & -4 & 1 & 0 & \cdots & 0 & 0 & 0 \\ -E & C & -E & 0 & \cdots & 0 & 0 & 0 \\ \hline 0 & -E & C & -E & \ddots & \vdots & \vdots & \vdots \\ \vdots & 0 & \ddots & \ddots & \ddots & 0 & \vdots & \vdots \\ \vdots & \vdots & \ddots & -E & C & -E & 0 & \vdots \\ 0 & 0 & \cdots & 0 & -E & C & -E & 0 \\ \hline 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 1 \end{array} \right) \begin{pmatrix} u_0^n \\ u_1^n \\ u_2^n \\ \vdots \\ u_{N_x-2}^n \\ u_{N_x-1}^n \\ u_{N_x}^n \end{pmatrix} = \begin{pmatrix} 2\frac{h_x}{D}\alpha(t^n) \\ u_1^{n-1} + h_t f_1^n \\ u_2^{n-1} + h_t f_2^n \\ \vdots \\ u_{N_x-2}^{n-1} + h_t f_{N_x-2}^n \\ u_{N_x-1}^{n-1} + h_t f_{N_x-1}^n \\ \beta(t^n) \end{pmatrix} \quad (32)$$

L'algorithme formel est donc le suivant :

- 1: $U_i^0 \leftarrow u_0(x_i), \forall i \in \llbracket 0, N_x \rrbracket$
- 2: **Pour** $n \leftarrow 1$ à N_t **faire**
- 3: Calcul de b^n
- 4: Résoudre le système linéaire $\mathbb{A}U^n = b^n$
- 5: **Fin**

Exercice 2

Q. 1 Ecrire la fonction `AssembleMatGen1D` retournant la matrice $M \in \mathcal{M}_d(\mathbb{R})$ définie par

$$M = \begin{pmatrix} a_1 & a_2 & a_3 & 0 & \dots & \dots & 0 \\ \beta & \alpha & \beta & 0 & \dots & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & \dots & 0 & \beta & \alpha & \beta \\ 0 & \dots & \dots & 0 & b_3 & b_2 & b_1 \end{pmatrix} \quad (1)$$

où $\alpha, \beta, a_1, a_2, a_3, b_1, b_2$ et b_3 sont des réels donnés.

Q. 2 Ecrire la fonction `SndMbrGen1D` retournant le vecteur $B \in \mathbb{R}^d$ défini par

$$B = \begin{pmatrix} \alpha \\ c_1 \\ \vdots \\ c_{d-2} \\ \beta \end{pmatrix} \quad (2)$$

où $\alpha, \beta, c_1, \dots, c_{d-2}$ sont des réels donnés.

Exercice 3

Q. 1

*Ecrire une fonction algorithmique `Heat1Dim` permettant de retourner la discrétisation en temps, la discrétisation en espace et l'ensemble des \mathbf{u}_i^n , $i \in \llbracket 0, N_x \rrbracket$, $n \in \llbracket 1, N_t \rrbracket$ calculés par le schéma **implicite** en temps pour l'EDP (6) à (9).*

Q. 2

Ecrire un programme utilisant cette fonction et permettant de calculer la solution numérique d'un problème dont on connaît la solution exacte.

Algorithme Fonction Heat1Dim (version non vectorisée)

```
1: Fonction [ $\mathbf{t}, \mathbf{x}, \mathbb{U}$ ]  $\leftarrow$  Heat1Dim(  $a, b, T, D, f, u_0, \alpha, \beta, N_x, N_t$  )
2:    $\mathbf{t} \leftarrow$  DisReg(0,  $T, N_t$ )
3:    $h_t \leftarrow T/N_t$ 
4:    $\mathbf{x} \leftarrow$  DisReg( $a, b, N_x$ )
5:    $h_x \leftarrow (b - a)/N_x$ 
6:    $E \leftarrow D * h_t / (h_x * h_x), C \leftarrow 1 + 2 * E$ 
7:   Pour  $i \leftarrow 1$  à  $N_x + 1$  faire
8:      $\mathbb{U}(i, 1) \leftarrow u_0(\mathbf{x}(i))$ 
9:   Fin
10:   $\mathbb{A} \leftarrow$  AssembleMatGen1D( $d, (3, -4, 1), (1, 0, 0), C, -E$ )
11:  Pour  $n \leftarrow 1$  à  $N_t$  faire
12:    Pour  $i \leftarrow 1$  à  $N_x - 2$  faire
13:       $\mathbf{v}(i) \leftarrow \mathbb{U}(i + 1, n) + h_t * f(\mathbf{t}(n + 1), \mathbf{x}i)$ 
14:    Fin
15:     $L \leftarrow 2 * (h_x/D) * \alpha(\mathbf{t}(n + 1))$ 
16:     $\mathbf{B} \leftarrow$  SndMbrGen1D( $d, \mathbf{v}, L, \beta(\mathbf{t}(n + 1))$ )
17:     $\mathbb{U}(:, n + 1) \leftarrow$  Solve( $\mathbb{A}, \mathbf{B}$ )
18:  Fin
19: Fin
```

▷ Condition initiale

▷ Boucle en temps
▷ Schéma

Algorithme Fonction Heat1Dim (version vectorisée)

```
1: Fonction [ $\mathbf{t}, \mathbf{x}, \mathbb{U}$ ]  $\leftarrow$  Heat1Dim(  $a, b, T, D, f, u_0, \alpha, \beta, N_x, N_t$  )
2:    $\mathbf{t} \leftarrow$  DisReg(0,  $T, N_t$ )
3:    $h_t \leftarrow T/N_t$ 
4:    $\mathbf{x} \leftarrow$  DisReg( $a, b, N_x$ )
5:    $h_x \leftarrow (b - a)/N_x$ 
6:    $E \leftarrow D * h_t / (h_x * h_x), C \leftarrow 1 + 2 * E$ 
7:    $\mathbb{U}(:, 1) \leftarrow u_0(\mathbf{x})$ 
8:    $\mathbb{A} \leftarrow$  AssembleMatGen1D( $d, (3, -4, 1), (1, 0, 0), C, E$ )
9:   Pour  $n \leftarrow 1$  à  $N_t$  faire
10:      $\mathbf{v} \leftarrow \mathbb{U}([2 : N_x], n) + h_t * f(\mathbf{t}(n + 1), \mathbf{x}([2 : N_x]))$ 
11:      $L \leftarrow 2 * (h_x/D) * \alpha(\mathbf{t}(n + 1))$ 
12:      $\mathbf{B} \leftarrow$  SndMbrGen1D( $d, \mathbf{v}, L, \beta(\mathbf{t}(n + 1))$ )
13:      $\mathbb{U}(:, n + 1) \leftarrow$  Solve( $\mathbb{A}, \mathbf{B}$ )
14:   Fin
15: Fin
```

▷ Condition initiale

▷ Boucle en temps

Une étude de stabilité au sens de Von Neumann de ce schéma **implicite** permet de montrer qu'il est **inconditionnellement stable**!