Ecole d'ingénieurs Sup Galilée
Energétique - Informatique - Instrumentation
Mathématiques Appliquées et Calcul Scientifique
Télécommunications et Réseaux

Université Sorbonne Paris Nord

Analyse Numérique I*

Sup'Galilée, Ingénieurs Energétique, 1ère année

François Cuvelier

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

2025/04/03

*Compilé le 2025/04/03 à 06:22:59

Plan du cours

Plan

EDP modèle instationnaire en dimension 1 : équation de la chaleur

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

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

$$u(0,x) = u_0(x), \qquad \forall x \in [a,b] \tag{2}$$

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

$$u(t,b) = \beta(t), \qquad \forall t \in [0,T] \tag{4}$$

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 :

$$u_0(b) = \beta(0). \tag{5}$$

$$x_i = a + ih_x, \qquad \forall i \in [\![0, N_x]\!], \qquad \text{avec } h_x = (b-a)/N_x$$
$$t^n = nh_t, \qquad \forall n \in [\![0, N_t]\!], \qquad \text{avec } h_t = T/N_t.$$

Objectif: Trouver $\boldsymbol{u}_i^n \approx u(t^n, x_i)$, $\forall n \in [\![0, N_t]\!]$, $\forall i \in [\![0, N_x]\!]$

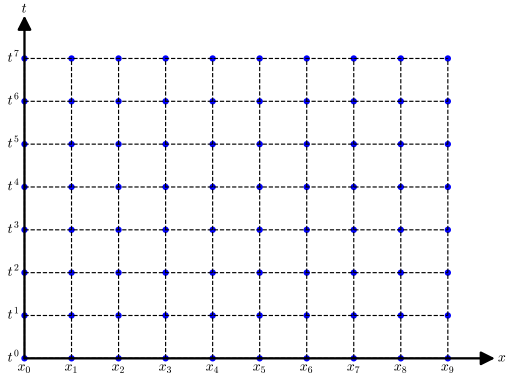


Figure: Grille espace-temps avec $N_t = 7$ et $N_x = 9$.

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

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

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

$$u(t^0, x_i) = u_0(x_i), \qquad \forall i \in [\![0, N_x]\!], \tag{7}$$

$$-D\frac{\partial u}{\partial x}(t^n, x_0) = \alpha(t^n), \qquad \forall n \in [\![0, N_t]\!] \tag{8}$$

$$u(t^n, x_{N_x}) = \beta(t^n), \qquad \forall n \in [\![0, N_t]\!] \tag{9}$$

$\Longrightarrow$ il nous faut maintenant discrétiser les opérateurs de dérivation

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

Taylor dimension $n > 1$

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 $\boldsymbol{x} \in U$, $i \in [\![1, n]\!]$, et $h \in \mathbb{R}^*$ vérifiant $\forall t \in [0, 1]$, $\boldsymbol{x} + th\boldsymbol{e}^{[i]} \in U$ où $\boldsymbol{e}^{[i]}$ est le i-ème vecteur de la base canonique de $\mathbb{R}^n$.
Alors il existe $\theta \in]0, 1[$ tel quel

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

où $\boldsymbol{e}^{[i]}$ 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 $\boldsymbol{x}$ dans la direction $\boldsymbol{e}^{[i]}$

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

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

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

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

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

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

On déduit des développements de Taylor :

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

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

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

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

On rappelle (6)

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

qui devient avec (13) et (12)

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

avec $n \in [\![0, N_t[\![$ et $i \in]\!]0, N_x[\![$.
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): $\forall n \in [\![0, N_t[\![,\ \forall i \in]\!]0, N_x[\![$

$$\frac{\boldsymbol{u}_i^{n+1} - \boldsymbol{u}_i^n}{h_t} - D\frac{\boldsymbol{u}_{i+1}^n - 2\boldsymbol{u}_i^n + \boldsymbol{u}_{i-1}^n}{h_x^2} = \boldsymbol{f}_i^n \tag{17}$$

avec $\boldsymbol{f}_i^n = f(t^n, x_i)$ et (en espérant) $\boldsymbol{u}_i^n \approx u(t^n, x_i)$.
(17) est équivalent à

$$\boldsymbol{u}_i^{n+1} = \boldsymbol{u}_i^n + D\frac{h_t}{h_x^2}\left(\boldsymbol{u}_{i+1}^n - 2\boldsymbol{u}_i^n + \boldsymbol{u}_{i-1}^n\right) + h_t\boldsymbol{f}_i^n \tag{18}$$

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

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

On rappelle respectivement (7) et (9):

$$u(t^0, x_i) = u_0(x_i), \qquad \forall i \in [\![0, N_x]\!]$$
$$u(t^n, x_{N_x}) = \beta(t^n), \qquad \forall n \in [\![0, N_t]\!]$$

qui donne immédiatement (sans approximation)

$$\boldsymbol{u}_i^0 = u_0(x_i), \qquad \forall i \in [\![0, N_x]\!]$$
$$\boldsymbol{u}_{N_x}^n = \beta(t^n), \qquad \forall n \in [\![0, N_t]\!] \tag{19}$$

Et (8)?

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

On rappelle (8) $\forall n \in [\![0, N_t]\!]$

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

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):

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

ou encore

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

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

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

$$\boldsymbol{u}_i^{n+1} = C\boldsymbol{u}_i^n + E(\boldsymbol{u}_{i+1}^n + \boldsymbol{u}_{i-1}^n) + h_t \boldsymbol{f}_i^n, \qquad \begin{cases} \forall n \in [\![0, N_t[\![, \\ \forall i \in]\!]0, N_x[\![\end{cases} \tag{18}$$

$$\boldsymbol{u}_i^0 = u_0(x_i), \qquad \forall i \in [\![0, N_x]\!] \tag{21}$$

$$\boldsymbol{u}_0^n = \frac{1}{3}\left(\frac{2h_x}{D}\alpha(t^n) + 4\boldsymbol{u}_1^n - \boldsymbol{u}_2^n\right), \qquad \forall n \in [\![0, N_t]\!] \tag{20}$$

$$\boldsymbol{u}_{N_x}^n = \beta(t^n), \qquad \forall n \in [\![0, N_t]\!] \tag{19}$$

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

$$\boldsymbol{u}_i^{n+1} = C\boldsymbol{u}_i^n + E(\boldsymbol{u}_{i+1}^n + \boldsymbol{u}_{i-1}^n) + h_t \boldsymbol{f}_i^n, \ \forall i \in]\!]0, N_x[\![, \tag{22}$$

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

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

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

L'algorithme formel est donc le suivant :

1: $\boldsymbol{u}_i^0 \leftarrow u_0(x_i),\ \forall i \in [\![0, N_x]\!]$
2: **Pour** $n \leftarrow 0$ **à** $N_t - 1$ **faire**
3: $\quad \boldsymbol{u}_i^{n+1} \leftarrow C\boldsymbol{u}_i^n + E(\boldsymbol{u}_{i+1}^n + \boldsymbol{u}_{i-1}^n) + h_t \boldsymbol{f}_i^n,\ \forall i \in]\!]0, N_x[\!]$
4: $\quad \boldsymbol{u}_{N_x}^{n+1} \leftarrow \beta(t^{n+1})$
5: $\quad \boldsymbol{u}_0^{n+1} \leftarrow \frac{1}{3}\left(\frac{2h_x}{D}\alpha(t^{n+1}) + 4\boldsymbol{u}_1^{n+1} - \boldsymbol{u}_2^{n+1}\right)$
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* $\boldsymbol{u}_i^n$, $i \in [\![0, N_x]\!]$, $n \in [\![1, N_t]\!]$ *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 connait la solution exacte.*

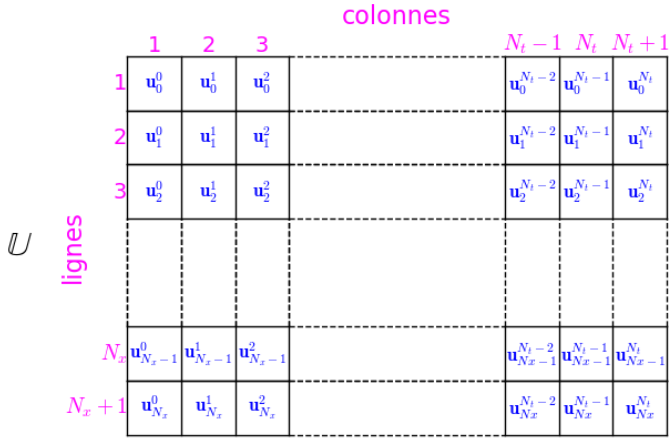
Correction: les pages qui suivent ...

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.

Résultat : $\mathbb{U}$: tableau 2D/matrice $(N_x+1)\times(N_t+1)$ de réels tel que $\mathbb{U}(i+1,n+1)=\boldsymbol{u}_i^n$, $\forall i\in[\![0,N_x]\!]$, $\forall n\in[\![0,N_t]\!]$.



Algorithme Fonction Heat1Dex (version non vectorisée)

1: **Fonction** $[\boldsymbol{t}, \boldsymbol{x}, \mathbb{U}] \leftarrow$ Heat1Dex($a, b, T, D, f, u_0, \alpha, \beta, N_x, N_t$)
2: $\quad \boldsymbol{t} \leftarrow \mathrm{DisReg}(0, T, N_t)$
3: $\quad h_t \leftarrow T/N_t$
4: $\quad \boldsymbol{x} \leftarrow \mathrm{DisReg}(a, b, N_x)$
5: $\quad h_x \leftarrow (b-a)/N_x$
6: $\quad E \leftarrow D\frac{h_t}{h_x^2}$, $C \leftarrow 1 - 2E$
7: $\quad$ **Pour** $i \leftarrow 1$ **à** $N_x + 1$ **faire** ▷ Condition initiale
8: $\quad\quad \mathbb{U}(i, 1) \leftarrow u_0(\boldsymbol{x}(i))$
9: $\quad$ **Fin**
10: $\quad$ **Pour** $n \leftarrow 1$ **à** N_t **faire** ▷ Boucle en temps
11: $\quad\quad$ **Pour** $i \leftarrow 2$ **à** N_x **faire** ▷ Schéma
12: $\quad\quad\quad \mathbb{U}(i, n+1) \leftarrow C * \mathbb{U}(i, n) + E * (\mathbb{U}(i+1, n) + \mathbb{U}(i-1, n)) + h_t * f(\boldsymbol{t}(n), \boldsymbol{x}(i))$
13: $\quad\quad$ **Fin**
14: $\quad\quad \mathbb{U}(N_x, n+1) \leftarrow \beta(\boldsymbol{t}(n+1))$
15: $\quad\quad \mathbb{U}(1, n+1) \leftarrow (\frac{h_x}{D} * \alpha(\boldsymbol{t}(n+1)) + 4\mathbb{U}(2, n+1) - \mathbb{U}(3, n+1))/3$
16: $\quad$ **Fin**
17: **Fin**

Algorithme Fonction Heat1Dex (version vectorisée)

Fonction $[\boldsymbol{t}, \boldsymbol{x}, \mathbb{U}] \leftarrow \text{Heat1Dex}(\ a, b, T, D, f, u_0, \alpha, \beta, N_x, N_t\ )$
  $\boldsymbol{t} \leftarrow \text{DisReg}(0, T, N_t),\ h_t \leftarrow T/N_t$
  $\boldsymbol{x} \leftarrow \text{DisReg}(a, b, N_x),\ h_x \leftarrow (b-a)/N_x$
  $E \leftarrow D\frac{h_t}{h_x^2},\ C \leftarrow 1 - 2E$
  $\mathbb{U}(:, 1) \leftarrow u_0(\boldsymbol{x})$ ▷ Condition initiale
  $\boldsymbol{I} \leftarrow [2 : N_x]$ ▷ Indices des points intérieurs
  Pour $n \leftarrow 1$ **à** N_t **faire** ▷ Boucle en temps
    $\mathbb{U}(\boldsymbol{I}, n+1) \leftarrow C * \mathbb{U}(\boldsymbol{I}, n) - E * (\mathbb{U}(\boldsymbol{I}+1, n) + \mathbb{U}(\boldsymbol{I}-1, n)) + h_t * f(\boldsymbol{t}(n), \boldsymbol{x}(\boldsymbol{I}))$
    $\mathbb{U}(N_x, n+1) \leftarrow \beta(\boldsymbol{t}(n+1))$
    $\mathbb{U}(1, n+1) \leftarrow (\frac{h_x}{D} * \alpha(\boldsymbol{t}(n+1)) + 4\mathbb{U}(2, n+1) - \mathbb{U}(3, n+1))/3$
  Fin
Fin

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

$$\begin{aligned} f(t,x) &\stackrel{\text{def}}{=} D\cos(t)\cos(x) - \sin(t)\cos(x), \\ u_0(x) &\stackrel{\text{def}}{=} \cos(x), \\ \alpha(t) &\stackrel{\text{def}}{=} D\cos(t)\sin(a), \\ \beta(t) &\stackrel{\text{def}}{=} \cos(t)\cos(b). \end{aligned}$$

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 $N_x = 50$

```
a=0;b=2*pi;Nx=50;
T=10;Nt=5000;
D=1;
u=@(t,x) cos(t).*cos(x); % solution exacte
f=@(t,x) D*cos(t).*cos(x) - sin(t).*cos(x);
beta=@(t) u(t,b);
alpha=@(t)  D*cos(t).*sin(a);
u0=@(x) u(0,x);

[t,x,U]=Heat1DexLight(a,b,T,D,f,u0,alpha,beta,Nx,Nt);
```

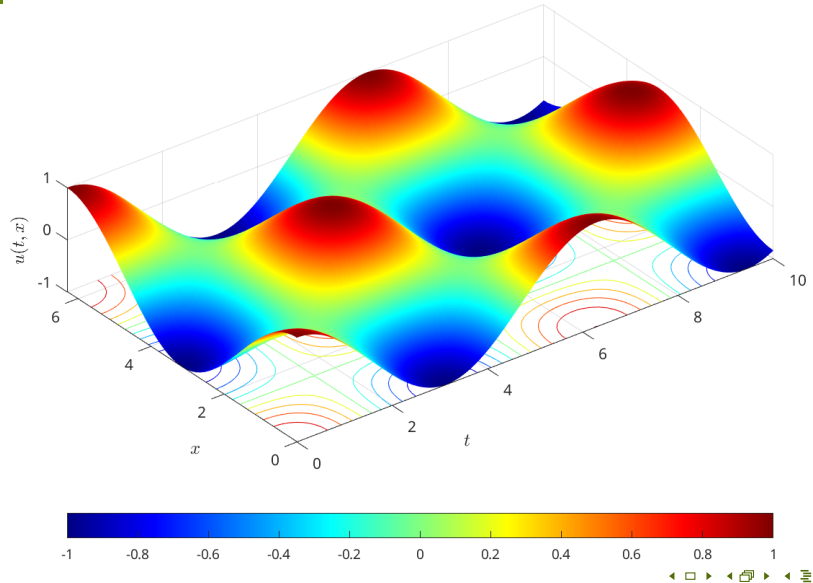
Listing: Equation de la chaleur 1D avec solution exacte. Calcul par schéma explicite : code Matlab, fichier `sampleHeat1Dex01.m`

```
sampleHeat1Dex01

surfc(t,x,U)
axis equal;shading interp;colormap jet;
xlabel('$t$','interpreter','latex','fontsize',12)
ylabel('$x$','interpreter','latex','fontsize',12)
zlabel('$u(t,x)$','interpreter','latex','fontsize',12)
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 $N_x = 50$



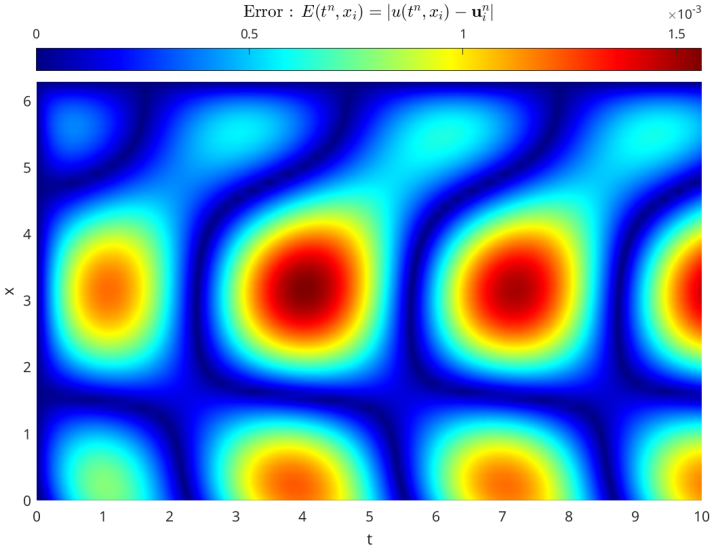
Application avec solution exacte: error $N_x = 50$

```
sampleHeat1Dex01
[T,X]=meshgrid(t,x);
Uex=u(T,X);

pcolor(t,x,abs(U-Uex))
xlabel('t'),ylabel('x')
shading interp, colormap(jet)
axis image;
h=colorbar('Location','northoutside');
set(get(h,'title'),'string','Error␣:␣$E(t^n,x_i)=|u(t^n,x_i)-\mathbf{u}_i^n|$', ...
    'interpreter','latex','fontsize',12)
```

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

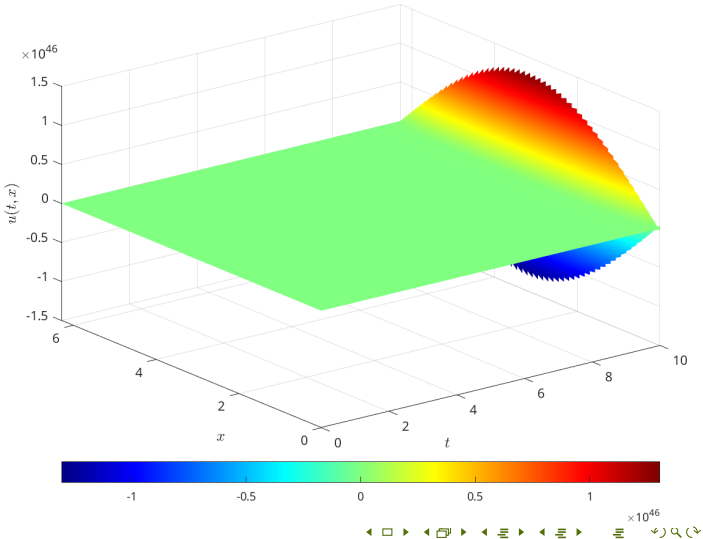
Application avec solution exacte: error $N_x = 50$



Application avec solution exacte: numerical solution $N_x = 100$

On change uniquement N_x de 50 à 100.

☞ A quoi peut-on s'attendre?



Application avec solution exacte: Phénomène d'instabilité

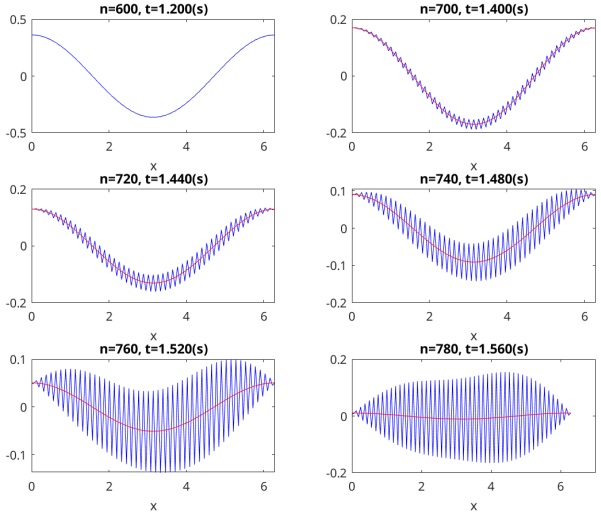


Figure: Equation de la chaleur 1D avec solution exacte (en rouge). Debut du phénomène d'instabilité.

Application avec solution exacte: 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

$$D\frac{h_t}{h_x^2} \leqslant \frac{1}{2}. \tag{25}$$

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

$$E(h_x, h_t) = \max_{i\in[\![0,N_x]\!], n\in[\![0,N_t]\!]} |\boldsymbol{u}_i^n - u_{\text{ex}}(t^n, x_i)|$$

où u_{ex} est la solution exacte.

Application avec solution exacte: Phénomène d'instabilité

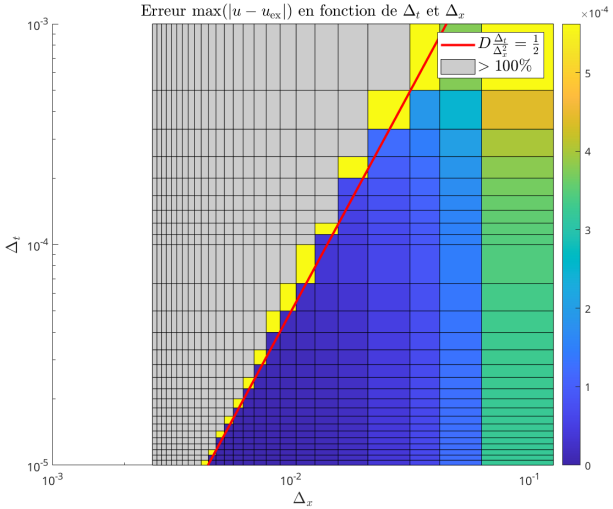


Figure: Equation de la chaleur 1D avec solution exacte. Condition de CFL : $\Delta_x = h_x$, $\Delta_t = h_t$

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

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

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

$$u(t^0, x_i) = u_0(x_i), \qquad \forall i \in [\![0, N_x]\!], \tag{7}$$

$$-D\frac{\partial u}{\partial x}(t^n, x_0) = \alpha(t^n), \qquad \forall n \in [\![0, N_t]\!] \tag{8}$$

$$u(t^n, x_{N_x}) = \beta(t^n), \qquad \forall n \in [\![0, N_t]\!] \tag{9}$$

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

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

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): $\forall n \in]\!]0, N_t]\!], \ \forall i \in]\!]0, N_x[\!]$

$$\frac{\boldsymbol{u}_i^n - \boldsymbol{u}_i^{n-1}}{h_t} - D\frac{\boldsymbol{u}_{i+1}^n - 2\boldsymbol{u}_i^n + \boldsymbol{u}_{i-1}^n}{h_x^2} = \boldsymbol{f}_i^n, \tag{26}$$

avec $\boldsymbol{f}_i^n = f(t^n, x_i)$ et (en espérant) $\boldsymbol{u}_i^n \approx u(t^n, x_i)$.
(29) est équivalent à

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

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

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

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

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

$$C\boldsymbol{u}_i^n - E(\boldsymbol{u}_{i+1}^n + \boldsymbol{u}_{i-1}^n) = \boldsymbol{u}_i^{n-1} + h_t\boldsymbol{f}_i^n, \ \forall i \in]\!]0, N_x[\![\tag{29}$$

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

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

$$\mathbb{A}\boldsymbol{U}^n = \boldsymbol{b}^n \tag{31}$$

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

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

$$\mathbb{A}\boldsymbol{U}^n = \boldsymbol{b}^n \iff$$

$$\left(\begin{array}{c|c|ccccc|c} 3 & -4 & 1 & 0 & \cdots & 0 & 0 \\ \hline -E & C & -E & 0 & \cdots & 0 & 0 \\ 0 & -E & C & -E & \ddots & \vdots & \vdots \\ \vdots & 0 & \ddots & \ddots & \ddots & 0 & \vdots \\ \vdots & \vdots & \ddots & -E & C & -E & 0 \\ 0 & 0 & \dots & 0 & -E & C & -E \\ \hline 0 & 0 & \dots & 0 & 0 & 0 & 1 \end{array}\right) \left(\begin{array}{c} \boldsymbol{u}_0^n \\ \hline \boldsymbol{u}_1^n \\ \boldsymbol{u}_2^n \\ \vdots \\ \boldsymbol{u}_{N_x-2}^n \\ \boldsymbol{u}_{N_x-1}^n \\ \hline \boldsymbol{u}_{N_x}^n \end{array}\right) = \left(\begin{array}{c} 2\frac{h_x}{D}\alpha(t^n) \\ \hline \boldsymbol{u}_1^{n-1} + h_t \boldsymbol{f}_1^n \\ \boldsymbol{u}_2^{n-1} + h_t \boldsymbol{f}_2^n \\ \vdots \\ \boldsymbol{u}_{N_x-2}^{n-1} + h_t \boldsymbol{f}_{N_x-2}^n \\ \boldsymbol{u}_{N_x-1}^{n-1} + h_t \boldsymbol{f}_{N_x-1}^n \\ \hline \beta(t^n) \end{array}\right) \tag{32}$$

L'algorithme formel est donc le suivant :

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

Exercice 2

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

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

où α, β, 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* $\boldsymbol{B} \in \mathbb{R}^d$ *défini par*

$$\boldsymbol{B} = \begin{pmatrix} \alpha \\ c_1 \\ \vdots \\ c_{d-2} \\ \beta \end{pmatrix} \tag{2}$$

où α, β, $c_1, \dots, c_{d-2}$ *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* $\boldsymbol{u}_i^n$, $i \in [\![0, N_x]\!]$, $n \in [\![1, N_t]\!]$ *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 connait la solution exacte.*

Algorithme Fonction Heat1Dim (version non vectorisée)

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

Algorithme Fonction Heat1Dim (version vectorisée)

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

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