

## Corrigé 9

### Exercice 1

**1.a)** On pose  $u_i^0 = \sin(2\pi x_i)$ ,  $i = 0, \dots, N+1$ . Pour tout  $n \geq 0$ , étant donné  $u_i^n$ ,  $i = 0, \dots, N+1$ , le problème discrétisé revient à chercher les  $u_i^{n+1}$ ,  $i = 0, \dots, N+1$  tels que

$$(\mathcal{R}_1) \begin{cases} \frac{u_i^{n+1} - u_i^n}{\tau} - \frac{u_{i-1}^n - 2u_i^n + u_{i+1}^n}{h^2} = 0, & i = 1, \dots, N, \\ u_0^{n+1} = u_{N+1}^{n+1} = 0. \end{cases} \quad (1)$$

On a donc la formule explicite :

$$u_i^{n+1} = \left(1 - \frac{2\tau}{h^2}\right) u_i^n + \frac{\tau}{h^2} u_{i-1}^n + \frac{\tau}{h^2} u_{i+1}^n, \quad i = 1, \dots, N. \quad (2)$$

**1.b)** Le fichier MATLAB `paraprog.m` est complété de la manière suivante :

```
function [err, unew, u_ex] = paraprog(N, M, tau)
%
% Schema d'Euler progressif pour un probleme parabolique unidimensionnel
%
% parametres
%
% N      : nombre de points interieurs dans l'intervalle [0,1]
% h      : pas d'espace
% M      : nombre de pas de temps
% tau    : pas de temps
% t      : temps courant
% uold   : N-vecteur, uold(i) est une approximation de u(x_i,t_n)
% unew   : N-vecteur, unew(i) est une approximation de u(x_i,t_{n+1})
%
h = 1./(N + 1);
t = 0.;

%
% condition initiale (fct w definie ci-dessous)
%
for i = 1:N
    uold(i) = w(i*h);
end

%
% schema d'Euler progressif
%
for n = 1:M
    unew(1) = (1-2*tau/(h*h)) * uold(1) + tau/(h*h) * uold(2);
    for i = 2:N-1
        unew(i) = (1-2*tau/(h*h)) * uold(i) + tau/(h*h) * (uold(i-1) + uold(i+1));
    end
    unew(N) = (1-2*tau/(h*h)) * uold(N) + tau/(h*h) * uold(N-1);
    t = t + tau;

    %
    % reactualiser la solution et imprimer la norme euclidienne de u
    %
    norm2 = 0.;
    for i = 1:N
        uold(i) = unew(i);
        norm2 = norm2 + uold(i)*uold(i);
    end
    if (mod(n, 50) == 0)
        fprintf(' pas de temps %d temps %f norm2 %e \n',n,t,norm2)
    end
end
end
```

```

%
% imprimer l'erreur maximum au temps final
%
err = 0;
for i = 1:N
    x(i) = i*h;
    u_ex(i) = uex(i*h, t);
    erri = abs(unew(i) - uex(i*h, t));
    if (erri > err)
        err = erri;
    end
end
fprintf(' erreur maximum au temps final t = %e (M = %d, N = %d): %e \n', t, M, N, err);
plot(x, unew, '*-b', x, u_ex, '-r');
legend('solution approchee', 'solution exacte');

%
% definition de la condition initiale w(x)
%
function init = w(x)
    init = sin(2*pi*x);

%
% solution u du probleme de la chaleur (utilise pour le calcul de l'erreur)
%
function u_exact = uex(x, t)
    u_exact = w(x)*exp(-4.*pi*pi*t);

```

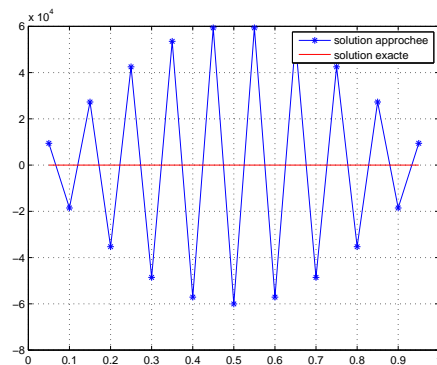
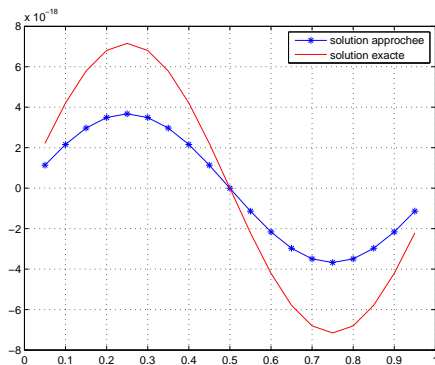
Comme  $\frac{\partial^2 u}{\partial x^2}(x, t) = -4\pi^2 \sin(2\pi x) \exp(-4\pi^2 t) = \frac{\partial u}{\partial t}(x, t)$ ,  $u(0, t) = u(1, t) = 0$  et  $u(x, 0) = \sin(2\pi x)$ ,  $u(x, t)$  est solution du problème  $(\mathcal{P})$ .

On a les résultats suivants au temps  $t = 0.1$  :

$N + 1$	$h$	$\tau$	$M$	Erreur
10	0.1	0.005	20	4.631713e-3
20	0.05	0.00125	80	1.245104e-3
40	0.025	0.0003125	320	3.127866e-4
80	0.0125	0.000078125	1280	7.828796e-5

On note que l'erreur est approximativement divisée par quatre lorsque  $h$  est divisé par deux et  $\tau$  est divisé par quatre.

- 1.c) Nous remarquons que le schéma est stable pour  $N = 19$  et  $\tau = 0.00125$  (figure de gauche), tandis qu'il est instable pour  $N = 19$  et  $\tau = 0.0013$  (figure de droite). La condition  $\tau \leq h^2/2$  est violée dans ce deuxième cas.



## Exercice 2

- 2.a) Le fichier `fd2d.m` implémente une méthode de différences finies en deux dimensions pour résoudre le problème

$$\begin{cases} -\Delta u(x, y) = 1 & \forall (x, y) \in \Omega, \\ u(x, y) = 0 & \forall (x, y) \in \partial\Omega, \end{cases}$$

avec  $\Omega = [0, 1] \times [0, 1]$ .

- 2.b) En prenant des valeurs croissantes pour le paramètre  $L$ , on vérifie que le maximum de la solution converge. Un exemple est donné par le tableau suivant :

L	$\max_{i,j}  u_{i,j} $
5	0.072115
10	0.072173
20	0.073257
40	0.073563
80	0.073643
160	0.073664
320	0.073670
640	0.073671
1280	0.073671

**2.c)** La valeur maximale du paramètre  $L$  dépend de la machine que vous utilisez, et éventuellement du logiciel que vous utilisez. Par exemple, avec une machine équipée d'un processeur Intel Xeon E5-2620@2.1Ghz et 32Go de mémoire RAM, Octave 3.8.1 retourne une erreur à partir de  $L = 4500$  environ. Ceci est dû au fait que la place mémoire nécessaire à la décomposition de Cholesky est  $O(L^3)$ , qui excède la mémoire RAM pour  $L = 4500$ .