T.P. 2 : Equation de la chaleur effet régularisant

On s'intéresse à la simualation de l'évolution de la température dans un barreau métallique de longueur L. La variable d'état, $u(x,t)$ est donc la température du barreau au point d'abscisse $x \in [0, L]$ au temps t. L'équation d'évolution la régissant est l'équation de la chaleur, que nous écrivons sous la forme :

$$\partial_t u(x,t) - \nu \Delta u(x,t) = v(x,t).$$

Le symbole Δ désigne l'opérateur Laplacien, défini par :

$$\Delta u(x,t) = \partial_{xx} u(x,t),$$

le coefficient ν est une constante dépendant des caractéristiques du métal.

1. Proposer une discrétisation de cette équation d'évolution et l'implémenter.
2. Transformer le code précédent de tel sorte qu'il ne contienne plus qu'une seule boucle "*for*".
3. Illustrer par un test l'effet régularisant de l'équation de la chaleur.
4. Comment obtenir simplement un code pour la simulation de l'évolution de la chaleur dans une plaque métallique ? (i.e. comment passer en dimension 2 ?)
5. Matlab propose en fait déjà des solveurs de l'équation de la chaleur. Voici un code qui les utilise. Essayer de comprendre à quoi correspond chaque ligne de ce code.

```
%PDEDEMO5 Animation demo for heat conduction problem.

%       Copyright 1994-2001 The MathWorks, Inc.
%       $Revision: 1.8 $  $Date: 2001/02/09 17:03:14 $

echo on
clc

%       We solve the standard heat equation with a source term
%        du/dt-div(grad(u))=1
%       on a square with a discontinuous initial condition.
pause % Strike any key to continue.
clc

%       Problem definition
g='squareg'; % The unit square
b='squareb1'; % 0 on the boundary
c=1;
a=0;
f=1;
d=1;

%       Mesh
[p,e,t]=initmesh(g);

%       Initial condition: 1 inside the circle with radius 0.4.
%       0 otherwise.
u0=zeros(size(p,2),1);
ix=find(sqrt(p(1,:).^2+p(2,:).^2)<0.4);
u0(ix)=ones(size(ix));
pause % Strike any key to continue.
clc

%       We want the solution at 20 points in time between 0 and 0.1.
nframes=20;
tlist=linspace(0,0.05,nframes);

%       Solve parabolic problem
u1=parabolic(u0,tlist,b,p,e,t,c,a,f,d);
pause % Strike any key to continue.
clc

%       To speed up the plotting, we interpolate to a rectangular grid.
```

```
%       Make the animation
newplot;
Mv = moviein(nframes);
umax=max(max(u1));
umin=min(min(u1));
for j=1:nframes,...
  u=tri2grid(p,t,u1(:,j),tn,a2,a3);i=find(isnan(u));u(i)=zeros(size(i));...
  surf(x,y,u);caxis([umin umax]);colormap(cool),...
  axis([-1 1 -1 1 0 1]);...
  Mv(:,j) = getframe;...
end

% Show movie
movie(Mv,2)
pause % Strike any key to end.

echo off
```