Computational Fluid Dynamics
Homework 5

Francisco Jose Castillo Carrasco

February 27, 2018

Introduction

In this assignment I will be using the Crank-Nicholson method to solve the following PDE

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x^2} + q(x,t),$$

defined on the domain $x \in [-1,1]$ with boundary conditions

$$T(-1,t) = 2 - \sin\left(\frac{3\pi}{2}t\right),$$
$$\frac{\partial T}{\partial x}(1,t) = 0,$$

and initial condition $T(x,0) = 2$, with $\alpha = 0.1$. The source term q for $t > 0$ is given by

$$q(x,t) = \frac{3\pi}{2\sqrt{t}} \sin\left(\frac{\pi}{2}\sqrt{t}\right) \cos\left(\frac{\pi}{2}\sqrt{t}\right) \left(x^3 - x^2 - x + 1\right) + \frac{3\pi}{2} \cos\left(\frac{3\pi}{2}t\right) \sin\left(\frac{\pi}{2}x\right)$$
$$+ \alpha \left(\sin^2\left(\frac{\pi}{2}\sqrt{t}\right)(6 - 18x) + \frac{\pi^2}{4} \sin\left(\frac{3\pi}{2}t\right) \sin\left(\frac{\pi}{2}x\right) \right),$$

and $q(x,0) = 0$.

This first part of the assignment (the one typed) includes the formulation of the problem using the Crank Nicholson method for a node based and a cell based mesh.

Node Based Mesh

For simplicity, since I am using MATLAB, I will discretize the domain starting at node 1. Since we are using N elements, a node based mesh has $N+1$ nodes in the coordinate x. The number of nodes in time will depend on the size of our time step and for how long we want to run the simulation. The PDE discretized is

$$\left.\frac{\partial T}{\partial t}\right|_i^n = \alpha \left.\frac{\partial^2 T}{\partial x^2}\right|_i^n + q_i^n,$$

where n is a superscript and not a power. Note that, to follow MATLAB indices, $i = 1, 2, ..., N+1$. The boundary conditions in index form are expressed as

$$T_1^n = 2 - \sin\left(\frac{3\pi}{2}t_n\right) = g(t_n) = g^n,$$
$$\left.\frac{\partial T}{\partial x}\right|_{N+1}^n = 0 \quad \Rightarrow \quad T_{N+1}^n = T_N^n,$$

and the initial condition as $T_i^0 = 2$. In the time coordinate we don't follow MATLAB indices since we will be overwriting the solution, therefore there is no problem with the index $n = 0$. Now we can take the discretized PDE and apply forward finite differences in time and central in space, keeping in mind that the Crank-Nicholson method average the right hand side between the present time step n and the next one $n+1$. Thus, the PDE turns into

$$\frac{T_i^{n+1} - T_i^n}{\Delta t} = \frac{\alpha}{2}\left(\frac{T_{i+1}^{n+1} - 2T_i^{n+1} + T_{i-1}^{n+1}}{h^2} + \frac{T_{i+1}^n - 2T_i^n + T_{i-1}^n}{h^2}\right) + \frac{1}{2}\left(q_i^n + q_i^{n+1}\right).$$

Multiplying by the time step and taking $1/h^2$ common factor,

$$T_i^{n+1} - T_i^n = \frac{\alpha\Delta t}{2h^2}\left(T_{i+1}^{n+1} - 2T_i^{n+1} + T_{i-1}^{n+1} + T_{i+1}^n - 2T_i^n + T_{i-1}^n\right) + \frac{\Delta t}{2}\left(q_i^n + q_i^{n+1}\right).$$

Defining

$$B = \frac{\alpha\Delta t}{2h^2},$$

and gathering all the $n+1$ terms on the left hand side and the n terms on the right hand side we obtain

$$-BT_{i-1}^{n+1} + (1+2B)T_i - BT_{i+1}^{n+1} = T_i^n + B\left(T_{i+1}^n - 2T_i^n + T_{i-1}^n\right) + \frac{\Delta t}{2}\left(q_i^n + q_i^{n+1}\right),$$

which has the tridiagonal form that we were looking form

$$a_i T_{i-1}^{n+1} + b_i T_i + c_i T_{i+1}^{n+1} = d_i, i = 1, ..., N+1.$$

We only need to solve this tridiagonal system for the interior nodes, $i = 2, ..., N$, since we have the boundary conditions at $i = 1$ and $i = N+1$. Therefore the vectors a, b, c, d have dimension $N-1$. To account for this difference in size we modify the previous equation with a $+1$ shift in the i index and the previous equations can be written as

$$-BT_i^{n+1} + (1+2B)T_{i+1} - BT_{i+2}^{n+1} = T_{i+1}^n + B\left(T_{i+2}^n - 2T_{i+1}^n + T_i^n\right) + \frac{\Delta t}{2}\left(q_{i+1}^n + q_{i+1}^{n+1}\right),$$

$$a_i T_i^{n+1} + b_i T_{i+1} + c_i T_{i+1}^{n+1} = d_i, \quad i = 1, ..., N-1$$

Now there is a direct correspondance between the values of the temperature vector and the vectors a, b, c, d needed to solve the tridiagonal system in the interior. Thus, we have

$$\begin{aligned} a_i &= -B, \\ b_i &= 1 + 2B, \\ c_i &= -B, \\ d_i &= T_{i+1}^n + B\left(T_{i+2}^n - 2T_{i+1}^n + T_i^n\right) + \frac{\Delta t}{2}\left(q_{i+1}^n + q_{i+1}^{n+1}\right), \end{aligned}$$

where $i = 1, 2, ..., N-1$. However, the previous values of the tridiagonal vectors are only acceptable for the *interior of the interior*, $i = 2, ..., N-2$. For the last nodes of the interior we need to impose the boundary conditions:

- Solve for $i = 1$.

$$-BT_1^{n+1} + (1+2B)T_2^{n+1} - BT_3^{n+1} = T_2^n + B\left(T_3^n - 2T_2^n + T_1^n\right) + \frac{\Delta t}{2}\left(q_2^n + q_2^{n+1}\right),$$

$$(1+2B)T_2^{n+1} - BT_3^{n+1} = T_2^n + B\left(T_3^n - 2T_2^n + g^n + g^{n+1}\right) + \frac{\Delta t}{2}\left(q_2^n + q_2^{n+1}\right),$$

where we have used that $T_1^n = g^n$ and $T_1^{n+1} = g^{n+1}$. Hence,

$$\begin{aligned} a_1 &= 0, \\ b_1 &= 1 + 2B, \\ c_1 &= -B, \\ d_1 &= T_2^n + B\left(T_3^n - 2T_2^n + g^n + g^{n+1}\right) + \frac{\Delta t}{2}\left(q_2^n + q_2^{n+1}\right). \end{aligned}$$

However, since a_1 does not affect the solution, there is no need to change its value.

- Solve for $i = N - 1$.

$$\begin{aligned} -BT_{N-1}^{n+1} + (1+2B)T_N^{n+1} - BT_{N+1}^{n+1} &= T_N^n + B\left(T_{N+1}^n - 2T_N^n + T_{N-1}^n\right) + \frac{\Delta t}{2}\left(q_N^n + q_N^{n+1}\right), \\ -BT_{N-1}^{n+1} + (1+2B)T_N^{n+1} - BT_N^{n+1} &= T_N^n + B\left(T_N^n - 2T_N^n + T_{N-1}^n\right) + \frac{\Delta t}{2}\left(q_N^n + q_N^{n+1}\right), \\ -BT_{N-1}^{n+1} + (1+B)T_N^{n+1} &= T_N^n + B\left(-T_N^n + T_{N-1}^n\right) + \frac{\Delta t}{2}\left(q_N^n + q_N^{n+1}\right), \end{aligned}$$

where we have used that $T_{N+1}^{n+1} = T_N^{n+1}$ and $T_{N+1}^n = T_N^n$. Hence,

$$\begin{aligned} a_{N-1} &= -B, \\ b_{N-1} &= 1 + B, \\ c_{N-1} &= 0, \\ d_{N-1} &= T_N^n + B\left(-T_N^n + T_{N-1}^n\right) + \frac{\Delta t}{2}\left(q_N^n + q_N^{n+1}\right). \end{aligned}$$

However, since c_{N-1} does not affect the solution, there is no need to change its value.

Cell Based Mesh

In the case of a cell based mesh we use a similar approach. Since the points of study are the centers of the cells, on the domain $-1 \leq x \leq 1$, the points are located at $-\frac{h}{2} - 1, -1 + \frac{h}{2}, -1 + \frac{3h}{2}, ..., 1 - \frac{h}{2}, 1 + \frac{h}{2}$, where the first and last points correspond to the ghost cells. Therefore, having N elements we will have $N + 2$ cell centers. To follow MATLAB indices, $x_1 = -1 - \frac{h}{2}$, $x_2 = -1 + \frac{h}{2}$, and so on. The temperature values T_i are the ones corresponding to the cell centered positions x_i. Hence, for N elements we have T_i values of tempereature, $i = 1, 2, ..., T_{N+2}$, where T_1 and T_{N+2} correspond to ghost cells. Like before we only need to solve for the interior of our vector T. The discretized PDE is the same as before

$$\left.\frac{\partial T}{\partial t}\right|_i^n = \alpha \left.\frac{\partial^2 T}{\partial x^2}\right|_i^n + q_i^n,$$

where n is a superscript and not a power. Note that, to follow MATLAB indices, $i = 1, 2, ..., N + 2$. The boundary conditions in index form are expressed as

$$\begin{aligned} T_{1+1/2}^n = \frac{T_1^n + T_2^n}{2} = 2 - \sin\left(\frac{3\pi}{2} t_n\right) = g^n \quad &\Rightarrow \quad T_1^n = 2g^n - T_2^n, \\ \left.\frac{\partial T}{\partial x}\right|_{N+1+1/2}^n = 0 \quad &\Rightarrow \quad T_{N+2}^n = T_{N+1}^n, \end{aligned}$$

and the initial condition as $T_i^0 = 2$. In the time coordinate we don't follow MATLAB indices since we are overwriting the solution, therefore there is no problem with the index $n = 0$. Note that the point $T_{1+1/2}$ corresponds to the temperature value between the cells 1 and 2, between the ghost cell and the first cell of

the domainat the boundary, i.e. at the boundary. Same thing for the other boundary. The code is going to work following this process: calculate tridiagonal vectors for the present time step, solve the tridiagonal system for the interior cells and obtain the solution for the interior in the next time step, update the ghost cells, recalculate the tridiagonal vectors and repeat. The Crank-Nicholson index equation, thanks to the index formulation chosen in both cases, is the same as in the previous section

$$-BT_{i-1}^{n+1} + (1+2B)T_i - BT_{i+1}^{n+1} = T_i^n + B\left(T_{i+1}^n - 2T_i^n + T_{i-1}^n\right) + \frac{\Delta t}{2}\left(q_i^n + q_i^{n+1}\right),$$

where B is defined as above

$$B = \frac{\alpha \Delta t}{2h^2}.$$

The index equation has the tridiagonal form that we were looking form

$$a_i T_{i-1}^{n+1} + b_i T_i + c_i T_{i+1}^{n+1} = d_i, i = 1, ..., N+2.$$

We only need to solve this tridiagonal system for the interior nodes, $i = 2, ..., N+1$, since we have the ghost cells at $i = 1$ and $i = N+2$. Therefore the vectors a, b, c, d have dimension N, and will start at the first node of the interior. Hence, there is a $+1$ shift in the space index to account for this difference in size and the previous equation can be written as

$$-BT_i^{n+1} + (1+2B)T_{i+1} - BT_{i+2}^{n+1} = T_{i+1}^n + B\left(T_{i+2}^n - 2T_{i+1}^n + T_i^n\right) + \frac{\Delta t}{2}\left(q_{i+1}^n + q_{i+1}^{n+1}\right),$$

$$a_i T_i^{n+1} + b_i T_{i+1} + c_i T_{i+2}^{n+1} = d_i, \quad i = 1, ..., N.$$

The vectors a, b, c, d have dimension N (whereas they had dimension $N-1$ in the node based mesh). Hence,

$$\begin{aligned} a_i &= -B, \\ b_i &= 1 + 2B, \\ c_i &= -B, \\ d_i = T_{i+1}^n &+ B\left(T_{i+2}^n - 2T_{i+1}^n + T_i^n\right) + \frac{\Delta t}{2}\left(q_{i+1}^n + q_{i+1}^{n+1}\right). \end{aligned}$$

for $i = 2, ..., N-1$. To obtain the values of the tridiagonal vector at the first and last cell we solve the equation at those locations.

- Solve for $i = 1$.

$$\begin{aligned} -BT_1^{n+1} + (1+2B)T_2^{n+1} - BT_3^{n+1} &= T_2^n + B\left(T_3^n - 2T_2^n + T_1^n\right) + \frac{\Delta t}{2}\left(q_2^n + q_2^{n+1}\right), \\ -2Bg^{n+1} + BT_2^{n+1} + (1+2B)T_2^{n+1} - BT_3^{n+1} &= T_2^n + B\left(T_3^n - 2T_2^n + 2g^n - T_2^n\right) + \frac{\Delta t}{2}\left(q_2^n + q_2^{n+1}\right), \\ (1+3B)T_2^{n+1} - BT_3^{n+1} &= T_2^n + B\left(T_3^n - 3T_2^n + 2g^n + 2g^{n+1}\right) + \frac{\Delta t}{2}\left(q_2^n + q_2^{n+1}\right), \end{aligned}$$

  where we have used that $T_1^n = 2g^n - T_2^n$ and $T_1^{n+1} = 2g^{n+1} - T_2 n + 1$. Hence,

$$\begin{aligned} a_1 &= 0, \\ b_1 &= 1 + 3B, \\ c_1 &= -B, \\ d_1 = T_2^n &+ B\left(T_3^n - 3T_2^n + 2g^n + 2g^{n+1}\right) + \frac{\Delta t}{2}\left(q_2^n + q_2^{n+1}\right). \end{aligned}$$

  However, since a_1 does not affect the solution, there is no need to change its value.

- Solve for $i = N$.

$$-BT_N^{n+1} + (1+2B)T_{N+1}^{n+1} - BT_{N+2}^{n+1} = T_{N+1}^n + B\left(T_{N+2}^n - 2T_{N+1}^n + T_N^n\right) + \frac{\Delta t}{2}\left(q_{N+1}^n + q_{N+1}^{n+1}\right),$$
$$-BT_N^{n+1} + (1+2B)T_{N+1}^{n+1} - BT_{N+1}^{n+1} = T_{N+1}^n + B\left(T_{N+1}^n - 2T_{N+1}^n + T_N^n\right) + \frac{\Delta t}{2}\left(q_{N+1}^n + q_{N+1}^{n+1}\right),$$
$$-BT_N^{n+1} + (1+B)T_{N+1}^{n+1} = T_{N+1}^n + B\left(-T_{N+1}^n + T_N^n\right) + \frac{\Delta t}{2}\left(q_{N+1}^n + q_{N+1}^{n+1}\right),$$

  where we have used that $T_{N+2} = T_{N+1}$. Hence,

$$\begin{aligned} a_N &= -B, \\ b_N &= 1 + B, \\ c_N &= 0, \\ d_N &= T_{N+1}^n + B\left(-T_{N+1}^n + T_N^n\right) + \frac{\Delta t}{2}\left(q_{N+1}^n + q_{N+1}^{n+1}\right). \end{aligned}$$

  However, since c_N does not affect the solution, there is no need to change its value.

The values specified in both the node base mesh and the cell based mesh will be implemented in MATLAB to obtain the solutions.

Maximum stable time step for the FTCS

As we saw in class, the maximum stable time step size for the FTCS is

$$\Delta t = \frac{h^2}{2\alpha},$$

and the time step used in the code is four time this,

$$\Delta t = 2\frac{h^2}{\alpha}.$$

There is no index form for this.

HOMEWORK 5 - FRANCISCO CASTILLO

Contents

Defined functions

```
function d=GaussTriSol(a,b,c,d)
    N=length(a);
    for i=2:N
        b(i)=b(i)-c(i-1)*a(i)/b(i-1);
        d(i)=d(i)-d(i-1)*a(i)/b(i-1);
    end
    d(N)=d(N)/b(N);
    for i=N-1:-1:1
        d(i)=(d(i)-c(i)*d(i+1))/b(i);
    end
end
```

Problem 1

In this problem we implement the code for the node based mesh fromulated previously. I start by initializing the variables and ploting the initial condition, an horizontal profile $T = 2$.

```
format long
clear all; close all; clc
labelfontsize=14;
linewidth=1.5;
axisfontsize=15;

M=256;
L=2;
alpha = 0.1;
h=L/M;
dt=4*h^2/(2*alpha);
x=linspace(-1,1,M+1)';
g = @(t) 2-sin(3*pi*t/2);
q = @(x,t) 3*pi*sin(pi*sqrt(t)/2).*cos(pi*sqrt(t)/2).*(x.^3-x.^2-x+1)./(2*sqrt(t))...
    +3*pi*cos(3*pi*t/2).*sin(pi*x/2)/2+...
    alpha*(sin(pi*sqrt(t)/2).^2.*(6-18*x)+pi^2*sin(3*pi*t/2).*sin(pi*x/2)/4);

T=2*ones(M+1,1);
qnew=zeros(M+1,1);
time=0;
outputTime=[1 2 3];
endtime=outputTime(end);
j=1;
```

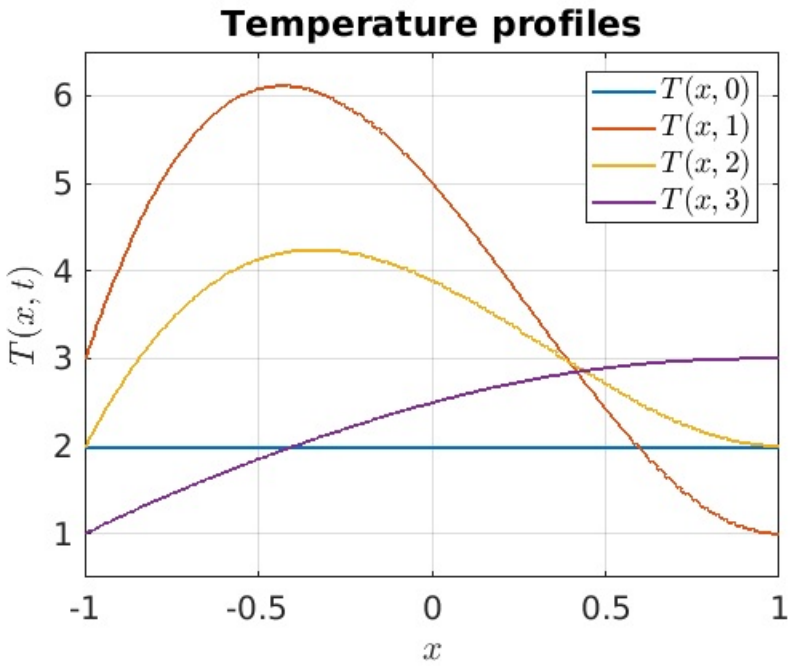
At the beginning of the loop I check that the next time step is not going to be after the output time desired. I adjust the time step if that was the case and for the next iteration I reset the time step to the specified by the problem

$$\Delta t = 2\frac{h^2}{\alpha}$$

Since the value of dt might be different, so it might the value of B and therefore the values of the tridiagonal vectors are recalculated before the tridiagonal solver function. In the case of the right hand side vector, it needs to be recalculated since its value depends on the Temperature values and the values of q.

```
figure(1)
plot(x,T,'linewidth',linewidth)
hold on
axis([-1 1 0.5 6.5])
grid on
xlabel('$x$','fontsize',labelfontsize,'interpreter','latex')
ylabel('$T(x,t)$','fontsize',labelfontsize,'interpreter','latex')
set(gca,'fontsize',axisfontsize)
title('Temperature profiles')

while time < endtime
    if (time < outputTime(j) && time+dt >= outputTime(j))
        dt=outputTime(j)-time;
    else
        dt=4*h^2/(2*alpha);
    end
    qold=qnew;
```

```
    qnew=q(x,time+dt);

    B=alpha*dt/(2*h^2);
    a=-B*ones(M-1,1);
    b=(1+2*B)*ones(M-1,1);
    b(end)=1+B;
    c=-B*ones(M-1,1);
    d=zeros(M-1,1);

    for i=2:M-1 %interior of the interior for d
        d(i)=T(i+1)+B*(T(i+2)-2*T(i+1)+T(i))+...
           dt*0.5*(qold(i+1)+qnew(i+1));
    end
    d(1)=T(2)+B*(T(3)-2*T(2)+g(time)+g(time+dt))+...
        +dt*0.5*(qold(2)+qnew(2));
    d(end)=T(M)+B*(-T(M)+T(M-1))+...
        +dt*0.5*(qold(M)+qnew(M));
    %
    T(2:end-1)=GaussTriSol(a,b,c,d);
    T(1)=g(time+dt);
    T(end)=T(end-1);
    if time+dt == outputTime(j)
        j=j+1;
        figure(1)
        plot(x,T,'linewidth',linewidth)
        axis([-1 1 0.5 6.5])
    end
    time=time+dt;
end
lh = legend({'$T(x,0)$','$T(x,1)$','$T(x,2)$','$T(x,3)$'},...
    'Interpreter','latex','Fontsize',14);
```



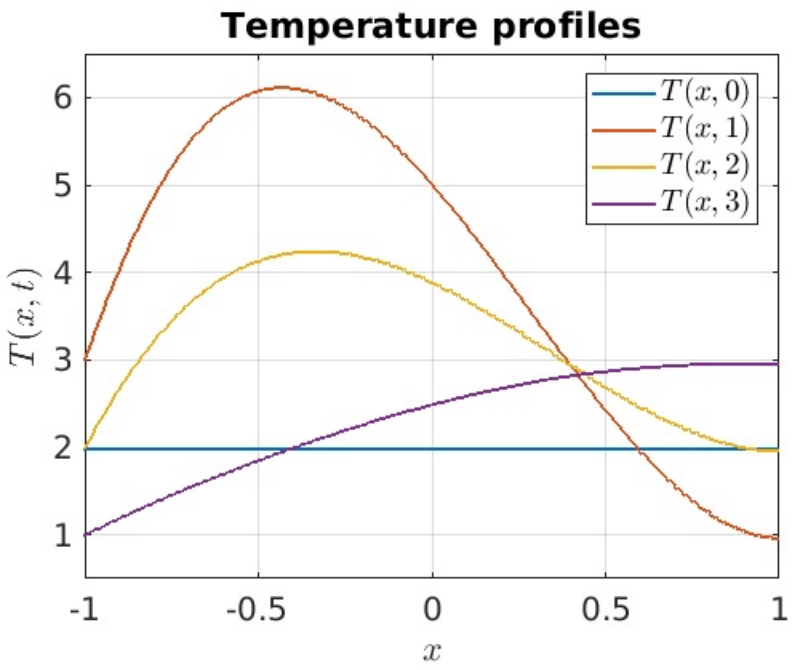
In the previous image we can see the temperature profiles with time. The temperature profiles have obviously oscillatory behaviour. This is easier seen by plotting all the timesteps into a plot and making a movie. I simulated up to 40 seconds and never got to a stabilization.

Problem 2

In this problem we repeat the same proccess but using cell based mesh. The code follows the same logic, only the equations for a, b, c, d and the boundary conditions change.

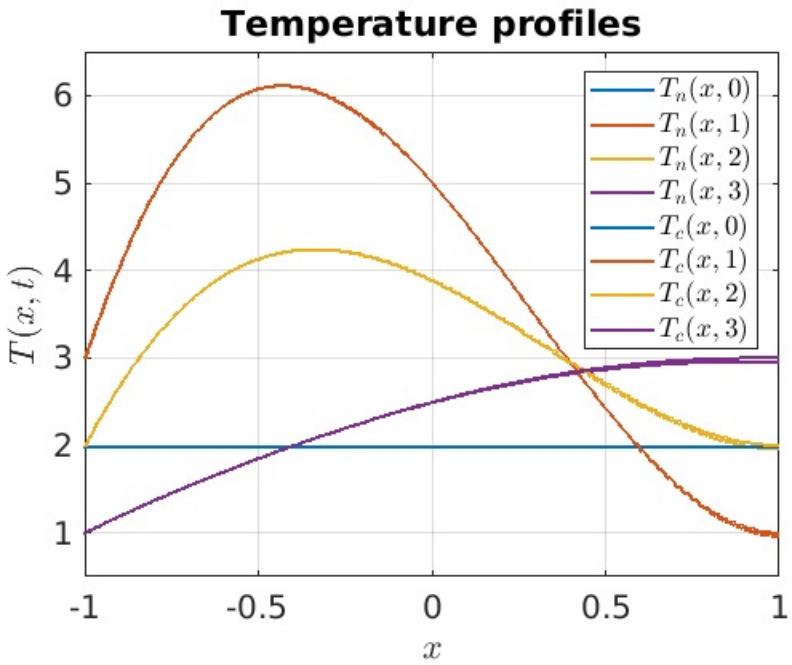
```
M=256;
L=2;
alpha = 0.1;
h=L/M;
dt=4*h^2/(2*alpha);
x=[-1-h/2:h:1+h/2]';
g = @(t) 2-sin(3*pi*t/2);
q = @(x,t) 3*pi*sin(pi*sqrt(t)/2).*cos(pi*sqrt(t)/2).*(x.^3-x.^2-x+1)./(2*sqrt(t))...
    +3*pi*cos(3*pi*t/2).*sin(pi*x/2)/2+...
    alpha*(sin(pi*sqrt(t)/2).^2.*(6-18*x)+pi^2*sin(3*pi*t/2).*sin(pi*x/2)/4);

T=2*ones(M+2,1);
qnew=zeros(M+2,1);
time=0;
outputTime=[1 2 3];
endtime=outputTime(end);
```

```
j=1;

figure(2)
plot(x,T,'linewidth',linewidth)
hold on
axis([-1 1 0.5 6.5])
grid on
xlabel('$x$','fontsize',labelfontsize,'interpreter','latex')
ylabel('$T(x,t)$','fontsize',labelfontsize,'interpreter','latex')
set(gca,'fontsize',axisfontsize)
title('Temperature profiles')

while time < endtime
    if (time < outputTime(j) && time+dt >= outputTime(j))
        dt=outputTime(j)-time;
    else
        dt=4*h^2/(2*alpha);
    end
    qold=qnew;
    qnew=q(x,time+dt);

    B=alpha*dt/(2*h^2);
    a=-B*ones(M,1);
    b=(1+2*B)*ones(M,1);
    b(1)=1+3*B;
    b(end)=1+B;
    c=-B*ones(M,1);
    c(end)=0;
    d=zeros(M,1);

    for i=2:M-1 %interior of the interior for d
        d(i)=T(i+1)+B*(T(i+2)-2*T(i+1)+T(i))+...
           dt*0.5*(qold(i+1)+qnew(i+1));
    end
    d(1)=T(2)+B*(T(3)-3*T(2)+2*g(time)+2*g(time+dt))+...
        +dt*0.5*(qold(2)+qnew(2));
    d(end)=T(M+1)+B*(-T(M+1)+T(M+1))+...
        +dt*0.5*(qold(M+1)+qnew(M+1));

    Tint(:)=GaussTriSol(a,b,c,d);
    T(2:end-1)=Tint(:);
    T(1)=2*g(time+dt)-T(2);
    T(end)=T(end-1);
    if time+dt == outputTime(j)
        j=j+1;
        figure(2)
        plot(x,T,'linewidth',linewidth)
        axis([-1 1 0.5 6.5])
    end
    time=time+dt;
end
lh = legend({'$T(x,0)$','$T(x,1)$','$T(x,2)$','$T(x,3)$'},...
    'Interpreter','latex','Fontsize',14);
```



In the previous image we can see the temperature profiles with time. The temperature profiles are clearly equal to the ones obtained by the node based method. However, I will superpose them to make sure.

```
L = findobj(1,'type','line');
copyobj(L,findobj(2,'type','axes'));
lh = legend({'$T_n(x,0)$','$T_n(x,1)$','$T_n(x,2)$','$T_n(x,3)$','$T_c(x,0)$','$T_c(x,1)$','$T_c(x,2)$','$T_c(x,3)$'},...
    'Interpreter','latex','Fontsize',13);
```



In the previous image the n subindex stands for node based and the c stands for cell based mesh. We can observe how the matching of the two types of meshing is good, as expected. It gets a bit worse close to the right boundary condition, the Neumann boundary condition, and for higher values of time.

Published with MATLAB® R2017a