NUMERICAL SIMULATION OF PARTIAL DIFFERENTIAL EQUATIONS

VINCENT PEETERS AND MORITZ WOLTER

Project Report

Supervised by Stefan Vandewalle
Stefaan Poedts
December 3, 2014 – Version 1

1 HEAT EQUATION

In the first part of this report we are going to solve the heat equation:

$$\frac{\partial \phi}{\partial t} = \frac{\partial^2 \phi}{\partial x^2} \tag{1}$$

With the boundary conditions:

$$\phi(0,t) = 0,\ \phi(1,t) = 1,\ \phi(x,0) = \sin(5\pi x/2).$$

Using four different numerical solution schemes:

1.1 EXPLICIT EULER

The explicit Euler method is defined as:

$$U_j^{n+1} = U_j^n + \mu(U_{j+1}^n - 2U_j^n + U_{j-1}^n) \ \text{ with } \mu = \frac{\triangle t}{(\triangle x)^2}. \tag{2}$$

From this equation we can derive code that solves the heat equation:

```
mu = 0.3;
dX = 1/20;
dT = mu * dX^2;
%First plot:
t = 0:dT:0.5;
%second plot.
%t = 0:dT:0.05;
x = 0:dX:1;

U = zeros(length(t),length(x));
%boundary conditions:
U(:,1) = 0;
U(:,end) = 1;
U(1,:) = sin(5*pi*x/2);

%compute the solution:
for n = 1:1:(length(t)-1)
    for j = 2:1:(length(x)-1)
        U(n+1,j) = U(n,j) + mu*( U(n,j+1) - 2*U(n,j) + U(n,j-1));
    end
end
mesh(x,t,U)
```

Listing 1: explicit Euler

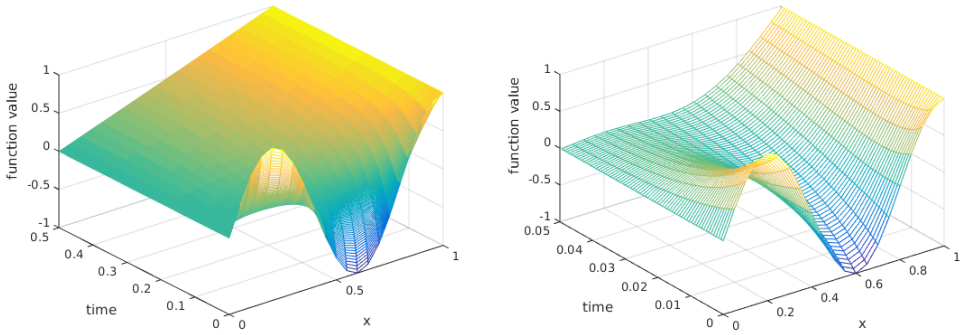

Figure 1: Solution of the heat equation with the explicid Euler method. Until time $t = 0.5$ (left) and until $t = 0.05$ (right). The boundary conditions are: 0, 1, $\sin(5\pi x/2)$.

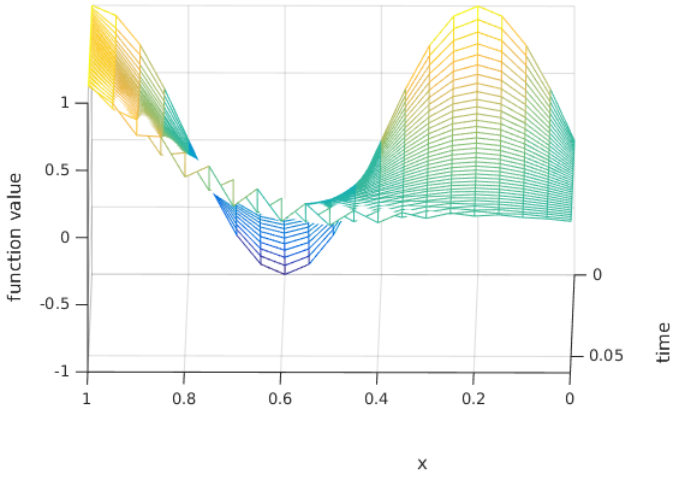

Figure 2: Instabilities forming with $\mu = 0.55$ at time $t = 0.05$.

Running this code leads to the images in figure 1. The computations are done using a mesh ratio $\mu = 0.3$ and $\triangle x = \frac{1}{20}$. Therefore we have time steps of size $\triangle t = 0.00075 = 7.5 * 10^{-4}$. This scheme is stable for mesh ratios $\mu \leqslant 0.5$. Therefore if we increase the time step to ≈ 0.0013 we are expecting to see instability. A plot of forming instabilities is given in figure 2

1.2 A SLIGHT VARIATION OF THE PROBLEM

Next we are going to consider a small variation of the problem. In fact the boundary conditions are going to change to:

$$\phi(0,t) = 0,\ \phi(1,t) = 0,\ \phi(x,0) = \sin(\pi x).$$

For this set of boundary conditions we know the exact solution:

$$\pi(x,t) = \exp(-\pi^2 t)\sin(\pi x). \tag{3}$$

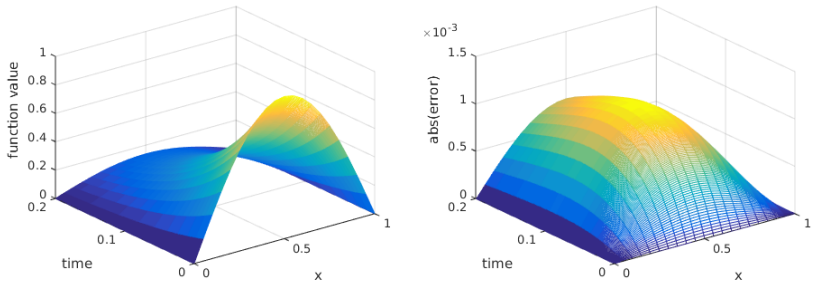

Figure 3: Numerical solution of the heat equation with the second boundary value set (left). Absolute value of the numerical solution (right).

Knowledge of the exact solution enables us to check the code we provided earlier. We are now able to compute the error in every grid point. The solution should not deviate too much from the exact solution. A plot of the numerical solution and it's error is given in figure 3. As the biggest error in any grid point is equal to 0.0011 we conclude our implementation is probably correct.

TODO: more math here...

1.3 EULER, CRANK-NICOLSON AND THE θ-METHOD

In this section we are going to use the more general theta-method-scheme to compare the errors of the explicit-Euler, implicit-Euler and Crank-Nicolson methods. Θ-type methods are defined as :

∂_x^2 *denotes double application of a central difference*

$$U_j^{n+1} - U_j^n = \mu[\theta\partial_x^2 U_j^{n+1} + (1-\theta)\partial_x^2 U_j^n]. \tag{4}$$

With $\theta = 0$ this we have the explicit Euler method, $\theta = 1$ leads to the implicit Euler method and finally $\theta = 0.5$ leads to the Crank-Nicolson method. In order do be able to implement a function in matlab that takes care of finding the error for each of these methods we have to derive two essential matrices. This is done by expanding the central differences from equation 4 and rearranging:

$$-\mu\theta U_{j-1}^{n+1} + U_j^{n+1}(1+2\mu\theta) - \mu\theta U_{j+1}^{n+1}$$

$$= \mu(1-\theta)U_{j-1}^n + (1-2\mu(1-\theta)U_j^n + \mu(1-\theta)U_{j+1}^n. \tag{5}$$

Here we are looking at an equation of the form $A\mathbf{U}^{n+1} = B\mathbf{U}^n$. Therefor the matrices A and B must be:

$$A = \begin{pmatrix} (1+2\mu\theta) & -\mu\theta & & \\ -\mu\theta & (1+2\mu\theta) & -\mu\theta & \\ & -\mu\theta & (1+2\mu\theta) & -\mu\theta \\ & \ddots & \ddots & \ddots \end{pmatrix}$$

$$B = \begin{pmatrix} 1-2\mu(1-\theta) & \mu(1-\theta) & & & \\ \mu(1-\theta) & 1-2\mu(1-\theta) & \mu(1-\theta) & & \\ & \mu(1-\theta) & 1-2\mu(1-\theta) & \mu(1-\theta) & \\ & & \ddots & \ddots & \ddots \end{pmatrix}. \quad (6)$$

Now we are able to write the following function, which when executed with different thetas θ and grid parameters $\triangle x, \triangle t$ will allow us to learn more about the error.

```
function error  = thetaMethod( dT,dX,tEnd,xEnd,leftBound,...
                                    rightBound,xBound,theta)
mu = dT/(dX^2);
t = 0:dT:tEnd;
x = 0:dX:xEnd;
U = meshgrid(x,t);
exSol = zeros(length(t),length(x));
error = zeros(length(t),length(x));
%Time boundary conditions.
U(:,1) = leftBound;
U(:,end) = rightBound;
%x boundary condition
U(1,:) = xBound;

%construct the step Matrix:
leftMat = toeplitz([(1 + 2*mu*theta) -mu*theta, ...
                    zeros(1,length(x)-4)],...
                   [(1 + 2*mu*theta) -mu*theta,...
                    zeros(1,length(x)-4)]);

rightMat = toeplitz([(1 - 2*mu*(1-theta)) mu*(1-theta), ...
                     zeros(1,length(x)-4)],...
                    [(1 - 2*mu*(1-theta)) mu*(1-theta), ...
                     zeros(1,length(x)-4)]);

for n = 1:1:(length(t)-1)
    %compute the values for the next time step.
    U(n+1,2:(end-1)) = leftMat\(rightMat*U(n,2:end-1)')...
                       + mu*[U(n+1,1); zeros(length(x)-4,1); ...
                         U(n+1,end)];
    %Exact solution:
    exSol(n,:) = exp(-pi^2*t(n))*sin(pi*x);
    %Error:
    error(n,:) = U(n,:) - exSol(n,:);
end
end
```

Listing 2: generic theta Method

In the following section we will describe and interpret the results we obtained.

	$\triangle x = 1/20$	$\triangle x = 1/40$	$\triangle x = 1/80$	$\triangle x = 1/160$	$\triangle x = 1/320$
$\triangle t = 1/10$	0.8919	0.6577	0.4729	0.3368	0.2390
$\triangle t = 1/20$	0.1427	0.1119	0.0817	0.0584	0.0415
$\triangle t = 1/40$	0.0196	0.0214	0.0166	0.0121	0.0086
$\triangle t = 1/80$	0.0047	0.0032	0.0035	0.0027	0.0019

Table 1: Error values of the Crank-Nicolson-scheme for different grid values. **Every entry has to be multiplied by 1.0e-03**.

	$\triangle x = 1/20$	$\triangle x = 1/40$	$\triangle x = 1/80$	$\triangle x = 1/160$	$\triangle x = 1/320$
$\triangle t = 1/10$	0.0068	0.0049	0.0035	0.0025	0.0018
$\triangle t = 1/20$	0.0023	0.0017	0.0012	0.0008	0.0006
$\triangle t = 1/40$	0.0009	0.0007	0.0005	0.0003	0.0002
$\triangle t = 1/80$	0.0004	0.0003	0.0002	0.0001	0.0001

Table 2: Error of the implicit Euler-scheme for different grid values.

1.3.1 *Results*

We proceeded to do a little more simulations where we took a little more space variables to look at the behaviour of the error in this direction. We choose to include two grid distances: $\Delta t = \frac{1}{10}, \frac{1}{20}$. We then used 11 different grid spacings starting from $\frac{1}{40}$ and increasing by multiplying by 2. This was done for the crank-nicholson scheme and the implicit Euler. Results are shown in figure 4.

We expect the error the behave like the consistency error for the general θ-scheme. This is given by

$$T_j^{n+1/2} = \left[(\frac{1}{2} - \theta)\Delta t u_{xxt} - \frac{1}{12}(\Delta x)^2 u_{xxxx}\right] + \left[\frac{1}{24}(\Delta t)^2 u_{ttt} - \frac{1}{8}(\Delta t)^2 u_{xxtt}\right] \quad (7)$$

$$+ \left[\frac{1}{12}(\frac{1}{2} - \theta)\Delta t(\Delta x)^2 u_{xxxxt} - \frac{2}{6!}(\Delta x)^4 u_{xxxxxx}\right] \quad (8)$$

So for the Crank Nicholson scheme where $\theta = 0$ we can easily find from 8 that $(T_j^{n+1/2})_{CN} = \mathcal{O}(\Delta(x)^2 + \Delta(t^2))$. Since we have held the time step constant in each experiment however, this means that as $\Delta x \to 0$ the error will bump into a constant term arising from the remaining finite time step size. We thus expect to see the quadratic behaviour as a constant slope in the part of the figure where Δx is large enough in comparison to Δt and more constant behaviour when Δx is really small. In figure 4 we only see the constant slope, so we assume that the error arising from the time step size is still to small to notice. For the implicit Euler we expect to have the same order in terms of Δx but there is a difference in the time step dependent term. $(T_j^{n+1/2})_{IE} = \mathcal{O}(\Delta(x)^2 + \Delta(t))$. We will not notice this in the general

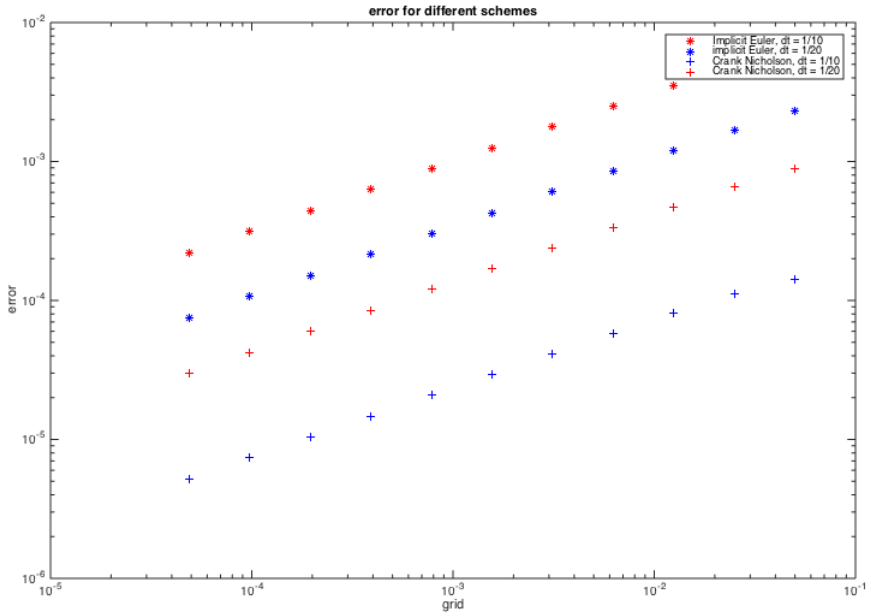

Figure 4: Plots show the behaviour of the error with grid spacing for Implicit euler and Crank Nicholson

behaviour in this plot, because only space step size is plotted, which should have the same slope and this is what we see. We also notice that the error is generally larger than the Crank Nicholson scheme which could be attributed to the poor time step size dependency.

We have then repeated the experiment but instead let time step size vary. We used time steps starting from $\frac{1}{10}$ where every next one was half the size of the previous until we had 14 step sizes. We used these parameters for different grid sizes using again the Crank-Nicholson and the Implicit Euler. What do we expect to see? For Cranck Nicholson we had $(T_j^{n+1/2})_{CN} = \mathcal{O}(\Delta(x)^2 + \Delta(t^2))$. Which again means we expect to see a constant slope corresponding to the $(\Delta t)^2$ there where the time step sizes are large enough until we bump into the contant term arising from the constant grid size where grid spacing is so small its error is negatable in comparison. For Implicit Euler we have $(T_j^{n+1/2})_{IE} = \mathcal{O}(\Delta(x)^2 + \Delta(t))$. which is only first order in Δt so we expect that this slope will be lower then the slope from crank nicholson. We can see all of this behaviour in figure 5.

Note that it would also be interesting to include a similar plot using a constant μ instead of contant Δt or Δx for than we would not have to care about bumping in constant terms which would mean cleaner slopes. We could then more easily include explicit euler as we could just give it a sufficient μ. But this has already been done and explained in the book corresponding to the course, so that would be uninteresting work.

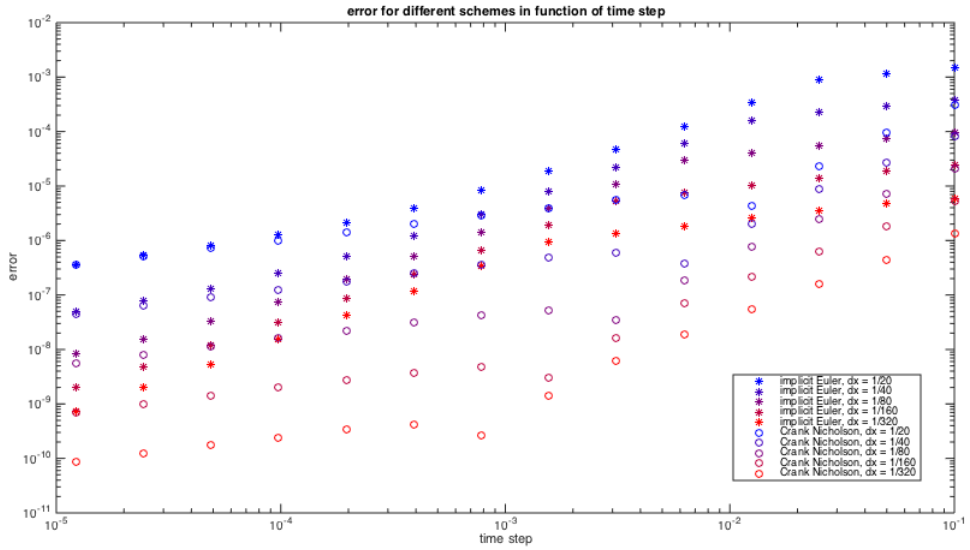

Figure 5: Plots show the behaviour of the error with time step spacing for Implicit euler and Crank Nicholson

2

HEAT EQUATION WITH CONVECTION

In this part a convection term in introduced into the equation. We are now solving a different problem of the form:

$$\frac{\partial \phi}{\partial t} = a\frac{\partial \phi}{\partial x} + \frac{\partial^2 \phi}{\partial x^2} \tag{9}$$

To solve this new problem we are forced to approximate the first derivative. We have three choices for these approximations central, forward and backward differences. The central difference approximates the derivative as:

$$a\frac{\partial \phi}{\partial x} \approx \frac{a_j^{n+1}}{2\triangle x}(U_{j+1}^{n+1} - U_{j-1}^{n+1}) \tag{10}$$

The forward difference approximation uses:

$$a\frac{\partial \phi}{\partial x} \approx \frac{a_j^{n+1}}{2\triangle x}(U_{j+1}^{n+1} - U_{j}^{n+1}) \tag{11}$$

And finally the backward difference approximation:

$$a\frac{\partial \phi}{\partial x} \approx \frac{a_j^{n+1}}{2\triangle x}(U_{j}^{n+1} - U_{j-1}^{n+1}) \tag{12}$$

These approximations allow us to solve the problem. Results are shown in figure 6. To do the computations we used $\triangle t = 0.0001$ and $\triangle x = 0.1$ which leads to $\mu = 0.01$.

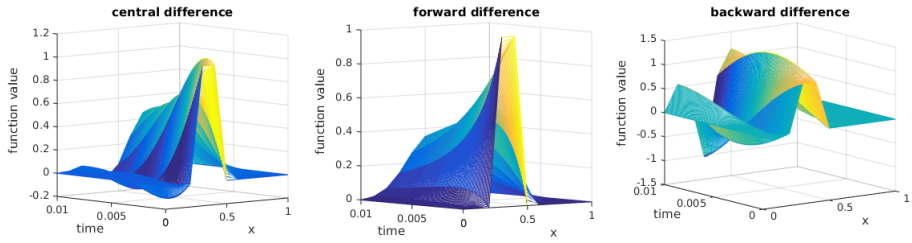

Figure 6: Solution of the problem with convection-term using different approximations for the first derivative term.