NUMERICAL SIMULATION OF PARTIAL DIFFERENTIAL EUQATIONS IN TWO DIMENSIONS.

Vincent Peeters & Moritz Wolter

December 21, 2014

1 Implementation of explicit methods

In this report we are going to implement explicit methods to solve three different partial differential equations in two dimensions.

1.1 Heat Equation

We will begin with the numerical solution of the heat equation:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}. \tag{1}$$

From the lecture we know that this problem may be solved by extension of the one dimensional explicit scheme:

$$\frac{U^{n+1} - U^n}{\triangle t} = b[\frac{\delta_x^2 U^n}{(\triangle x)^2} + \frac{\delta_y^2 U^n}{(\triangle y)^2}]. \tag{2}$$

with b = 1 in our case. By expanding the central differences we arrive at:

$$U_{r,x}^{n+1} = U_{r,s}^n(1 - 2\mu_x - 2\mu_y) + \mu_x U_{r+1,s}^n + \mu_x U_{r-1,s}^n + \mu_y U_{r,s+1}^n + \mu_y U_{r,s-1}^n. \tag{3}$$

Equation 3 my be implemented in matlab. As we are using a symmetric grid we have $\mu_x = \mu_y$ which leads to the implementation in listing 1:

```
tend = 3;
dt = 0.0001;
J = 30;
dx = 1/J;
dy = 1/J;
mu = dt/dx^2;

%Set up a mesh.
[x,y] = meshgrid(linspace(0,1,J));
%Initial solution.
```

```
U = sin(pi*x).*sin(pi*y);
U1 = zeros(J);
U2 = zeros(J);
for t = 1:(tend/dt)
    elements = 2:J-1;
    for i = 1:1:J
        %compute the columns where x is const.
        U1(elements,i) = mu*U(elements+1,i) + mu*U(elements-1,i);
        %compute the columns where y is const.
        U2(i,elements) = mu*U(i,elements+1) + mu*U(i,elements-1);
    end
    Unew = (1 - 4*mu) .* U + U1 + U2;
    U = Unew;
end
surf(x,y,U); axis([0 1 0 1 -1 1 -1 1]);
```

Listing 1: Explicit solution of the heat equation in two dimensions.

1.2 Wave equation

Next we are going to implement an explicit scheme to solve the wave equation:

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}. \tag{4}$$

Approximating the second derivatives with central differences and solving for $U_{r,s}^{n+1}$ we obtain the scheme:

$$\begin{aligned} U_{r,s}^{n+1} = U_{r,s}(2 - 2\mu_x - 2\mu_y) - U_{r,s}^{n-1} \\ +\mu_x U_{r-1,s}^n + \mu_x U_{r+1,s}^n \\ +\mu_y U_{r,s-1}^n + \mu_y U_{r,s+1}. \end{aligned}$$

In comparison to listing 1 we only have to change the for loop to implement this scheme. We got:

```
for t = 1:(tend/dt)
    elements = 2:J-1;
    for i = 1:1:J
        %compute the columns where x is const.
        U1(elements,i) = mu*U(elements+1,i) + mu*U(elements-1,i);
        %compute the columns where y is const.
        U2(i,elements) = mu*U(i,elements+1) + mu*U(i,elements-1);
    end
    Unew = (2 - 4*mu) .* U - Uold + U1 + U2;
    Uold = U;
    U = Unew;
end
```

Listing 2: Code for solving the wave equation.

1.3 Transport equation

Before we are going to consider the numerical stability and accuracy of these methods we will implement a final scheme to solve the transport equation:

$$\frac{\partial u}{\partial t} = \frac{\partial u}{\partial x} + \frac{\partial u}{\partial y}. \tag{5}$$

Using exclusively forward differences to approximate the first order differentials and solving for $U_{r,s}^{n+1}$ once more we obtain:

$$U_{r,s}^{n+1} = U_{r,s}^{n}(1 - \mu_x - \mu_y) + \mu_x U_{r+1,s} + \mu_y U_{r,s+1}. \tag{6}$$

Which leads to the modified for-loop for the transport case:

```
elements = 2:J-1;
for i = 1:1:J
    %compute the columns where x is const.
    U1(elements,i) = mu*U(elements+1,i);
    %compute the columns where y is const.
    U2(i,elements) = mu*U(i,elements+1);
end
Unew = (1 - 2*mu) .* U + U1 + U2;
U = Unew;
```

Listing 3: Code for solving the two dimesional transport equation.

2 Analysis

We will proceed with taking a close look at the numerical properties of the methods we described so far. Figure 1 shows stable and unstable solutions for the three equations.

2.1 Transport Equation

2.1.1 Exact Solution of Transport Equation

As a solution for the transport equation of form

$$\frac{\delta u}{\delta t} = \frac{\delta u}{\delta x} + \frac{\delta u}{\delta y} \tag{7}$$

with a given initial conditions, $u_0(x, y)$, and homogenous dirichet boundary conditions we propose a solution of the following form:

$$u(x, y, t) = u_0(x - vt, y - vt) \tag{8}$$

Calculating the partial derivatives found in the transport equation we get

$$\frac{\delta u}{\delta t} = (-v)\frac{\delta u_0(x, y)}{\delta x} + (-v)\frac{\delta u_0(x, y)}{\delta y} \tag{9}$$

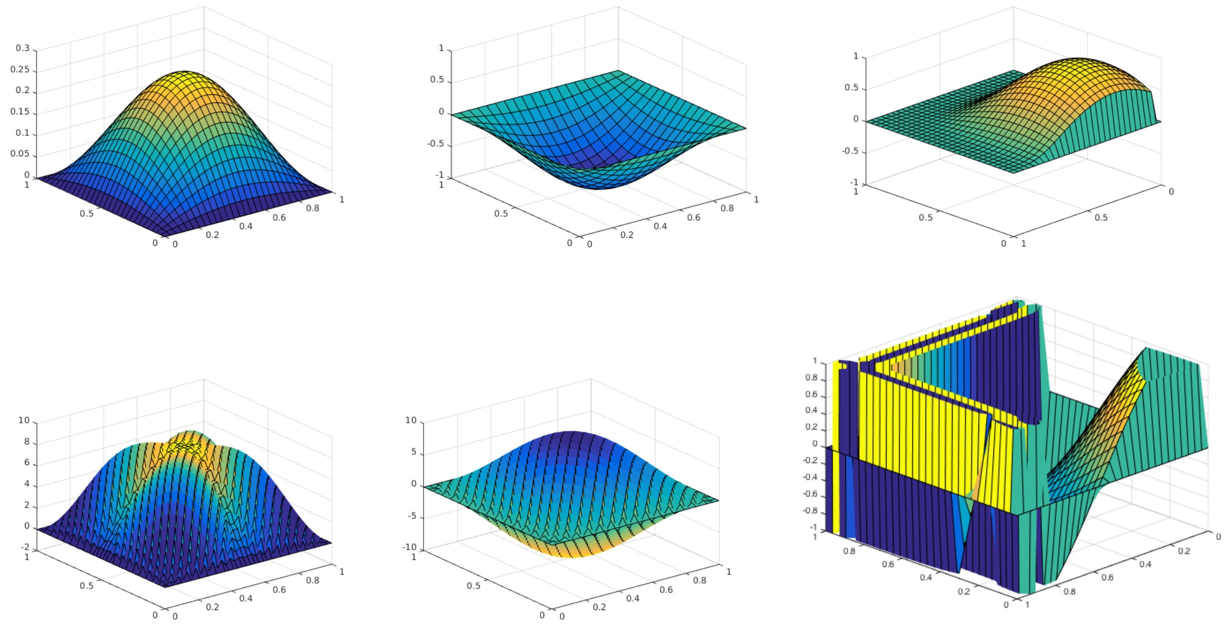

Figure 1: Numerical solutions computed using the schemes described above for stable (top row) and unstable (bottom row) grid ratios.

$$\frac{\delta u}{\delta x} = \frac{\delta u_0(x,y)}{\delta x}, \frac{\delta u}{\delta y} = \frac{\delta u_0(x,y)}{\delta y} \tag{10}$$

Filling this then in the transport equation gives

$$-v\frac{\delta u_0}{\delta x} - v\frac{\delta u_0}{\delta y} = \frac{\delta u_0}{\delta x} + \frac{\delta u_0}{\delta y} \tag{11}$$

which fits when $v = -1$ and so our solution is

$$u(x,y,t) = u_0(x+t, y+t) \tag{12}$$

2.1.2 Error analysis of the transport problem

The truncation error for the upwind scheme with $a = 1$ can be found to be

$$T_j^n = -\frac{1}{2}(1-\nu)\Delta x u_{xx} + ... \tag{13}$$

Which is first order in Δx, and therefore, under constant ν, also first order in Δt.

We then now that the maximum error E^n at a point in time n is bound by a function of the same order. And this is what we see in figure 2

3 Another initial solution

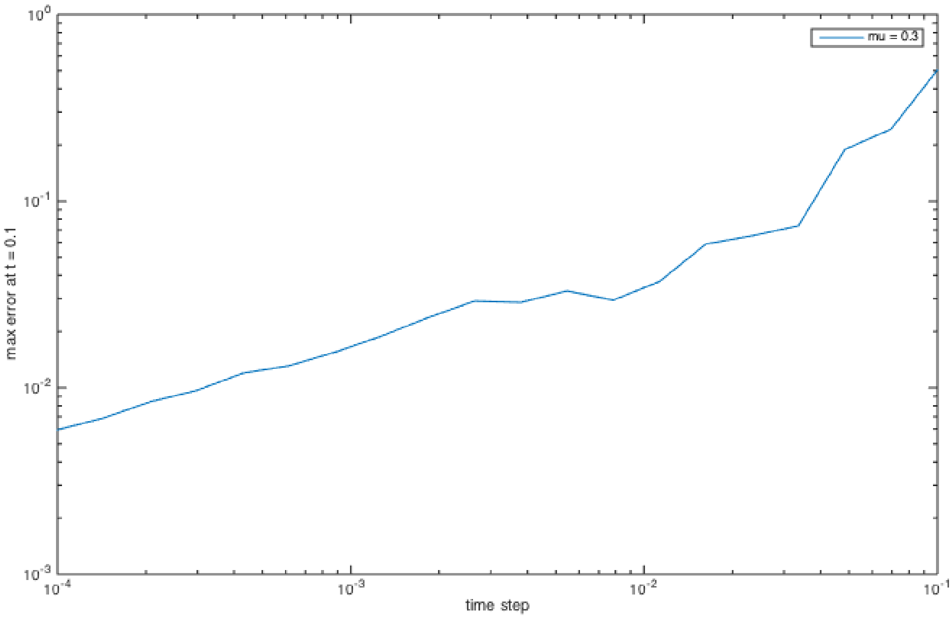

Figure 2: The maximum error at time = 0.1 in function of the tilmestep at a constant ν of 0.3