INF5620 - EXERCISES

KRISTER STRÆTE KARLSEN

28TH SEPTEMBER, 2015

Exercise 13 - Implementations of Neumann boundary conditions

The full boundary value problem:

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial}{\partial x}\left(q(x)\frac{\partial u}{\partial x}\right) + f(x,t), \quad \left.\frac{\partial u}{\partial x}\right|_{x=0,L} = 0, \quad u(x,0) = I(x), \quad \left.\frac{\partial u}{\partial t}\right|_{t=0} = V(x)$$

a)
We are using a manufactured solution $u = cos(\frac{\pi}{L}x)cos(t)$ and $q(x) = c(x)^2 = 1 + (x - L/2)^4$. Then using the program `Ex13a.py` to adjust the source term, and finally using the scheme below to get a numerical solution.

For inner points:

$$u_i^{n+1} = -u_i^{n-1} + 2u_i^n + \frac{K}{2}\left((q_i + q_{i+1})(u_{i+1}^n - u_i^n) - (q_i + q_{i-1})(u_i^n - u_{i-1}^n)\right) + \Delta t^2 f_i^n$$

First step formulas:

$$u_i^1 = \Delta t V + u_i^0 + \frac{K}{4}\left((q_i + q_{i+1})(u_{i+1}^0 - u_i^0) - (q_i + q_{i-1})(u_i^0 - u_{i-1}^0)\right) + \Delta t^2 f_i^0$$
$$u_0^1 = \Delta t V + u_0^0 + \frac{K}{4}\left((q_0 + q_1)(u_1^0 - u_0^0) - (q_0 + q_1)(u_0^0 - u_1^0)\right) + \Delta t^2 f_0^0$$
$$u_{N_x}^1 = \Delta t V + u_{N_x}^0 + \frac{K}{4}\left((q_{N_x} + q_{N_x-1})(u_{N_x-1}^0 - u_{N_x}^0) - (q_{N_x} + q_{N_x-1})(u_{N_x}^0 - u_{N_x-1}^0)\right) + \Delta t^2 f_{N_x}^0$$

Next step boundary formulas:

$$u_0^{n+1} = -u_0^{n-1} + 2u_0^n + \frac{K}{2}\left((q_0 + q_1)(u_1^n - u_0^n) - (q_0 + q_1)(u_0^n - u_1^n)\right) + \Delta t^2 f_0^n$$
$$u_{N_x}^{n+1} = -u_{N_x}^{n-1} + 2u_{N_x}^n + \frac{K}{2}\left((q_{N_x} + q_{N_x-1})(u_{N_x-1}^n - u_{N_x}^n) - (q_{N_x} + q_{N_x-1})(u_{N_x}^n - u_{N_x-1}^n)\right) + \Delta t^2 f_{N_x}^n$$

An implementation of this scheme can be found in `neumann_solvers.py` on github, along with an animation of a simulation.

Experimenting with different values of Δt and comparing with the exact solution gave the following result:

Δt_i	$E(\Delta t_i)$
0.2	1.38666154499
0.1	0.347812820981
0.05	0.155472257865
0.025	0.0636079519987
0.0125	0.0327684639893

The error, E_i, was computed the following way:

$$\frac{1}{(N_x+1)(N_t+1)}\sum_{i=0}^{N_x}\sum_{n=0}^{N_t}|u_{Ei}^n - u_i^n|$$

This is the average error of all cells in time and space. u_E is the exact solution.

b)
Using the same scheme and manufactured solution as above, just changing the wave velocity to $q(x) = c(x)^2 = 1 + cos(x - L/2)$ we get the following results:

Δt_i	$E(\Delta t_i)$
0.2	0.18655678091
0.1	0.0908278889876
0.05	0.0436440639965
0.025	0.0226928418887
0.0125	0.010969919377

This results can be obtained by running `Ex13b.py`.
We see that when $q(x)$ is symmetric the error is less than in a).

c)
Now using $u_1 - u_0 = 0$ and $u_{N_x} - u_{N_x-1} = 0$ the formulas for the boundaries simplefy to:

$$\begin{aligned} u_0^1 &= u_1^1 \\ u_{N_x}^1 &= u_{Nx-1}^1 \\ u_0^{n+1} &= u_1^{N+1} \\ u_{N_x}^{n+1} &= u_{Nx-1}^{n+1} \end{aligned}$$

An implementation of this is found in `neumann_solvers.py`, and the solver function is called *solver2(...)*.

Δt_i	$E(\Delta t_i)$
0.2	2.55237866851
0.1	1.06424090816
0.05	0.457826612961
0.025	0.233779449801
0.0125	0.11577157663

This leads to even bigger errors as we can see from the results in the table above. The same is illustrated in *animation_ c.gif*.

d)
Using the given simplifications i end up with the same scheme as in c) and thus the same results.