IN5270 Mandatory exercise 2

Soran Hussein Mohmmed / soranhm

September 2018

Exercise 13: Compare discretizations of Neumann condition

1D wave equation with variable wave velocity: $u_{tt} = (q(x)u_x)_x + f(x,t)$, And Neumann condition u_x at x=0,L can be discredited as shown in (54) and (57). I have used *wave1D_dn_vc.py* as a guide and added/changed some of it.

a)

We hav been given q and a exact u, that we use to calculate f with a function i python.

$$q(x) = 10(x - \frac{L}{2})^4$$
$$u(x,t) = cos(\frac{\pi x}{L})cos(\omega t)$$

$\omega = 1$ is given and i chose L = 1 (to make it easyer). Performing numerical experiments. Find the convergence rate of the error with:

$$E = C_t \Delta t^r + C_x \Delta x^p$$

$C_t, C_x, randpareconstants$, intreducing a single discretization paramater $h = \Delta t = \widetilde{c}\Delta x$ Δt and Δx is related true Courant number, gives h = $C\Delta x/c$. c in our case is $\sqrt{q}$. All this and some more expressions gives us:

$$e_i = u_e(x_i, t_n) - u_i^n$$
$$E = ||e_i^n||_{l\infty} =^{max}_{0\leq i\leq N_x} |e_i^n|$$

We have E_{i+1} and E_i who measure error in two diffrent time, and same with h_i and h_{i+1}:

$$E_{i+1} = Dh_{i+1}^r$$
$$E_i = Dh_i r$$
$$r = \frac{ln \quad E_{i+1}/E_i}{ln \quad h_{i+1}/h_i}$$

i = 0,...,m-2 : $(h_0, E_0), ..., (h_{m-1}, E_{m-1})$. We expect r = 2 in the wave equation, since the error terms are of order Δt^2 and Δx^2. Using (54) with u_t = V(x):

$$\frac{u_i^{n+1} - u_i^{n-1}}{2\Delta t} = V(x) \quad \longrightarrow \quad u_i^{n-1} = u_i^{n+1} - V(x)2\Delta t$$

V(x) = 0 when t ≠ 0, than this gives the u in the program for (54).

b)

Now we have:

$$q = 1 + cos(\pi x/L)$$

Doing the same as i did for (54) and adding it to the python file.

c)

This is a more simple one-sided difference, using given boundary conditions on Nx and 0. This one gives us r = 1 , because we 'lose' one part in each b.c.

d)

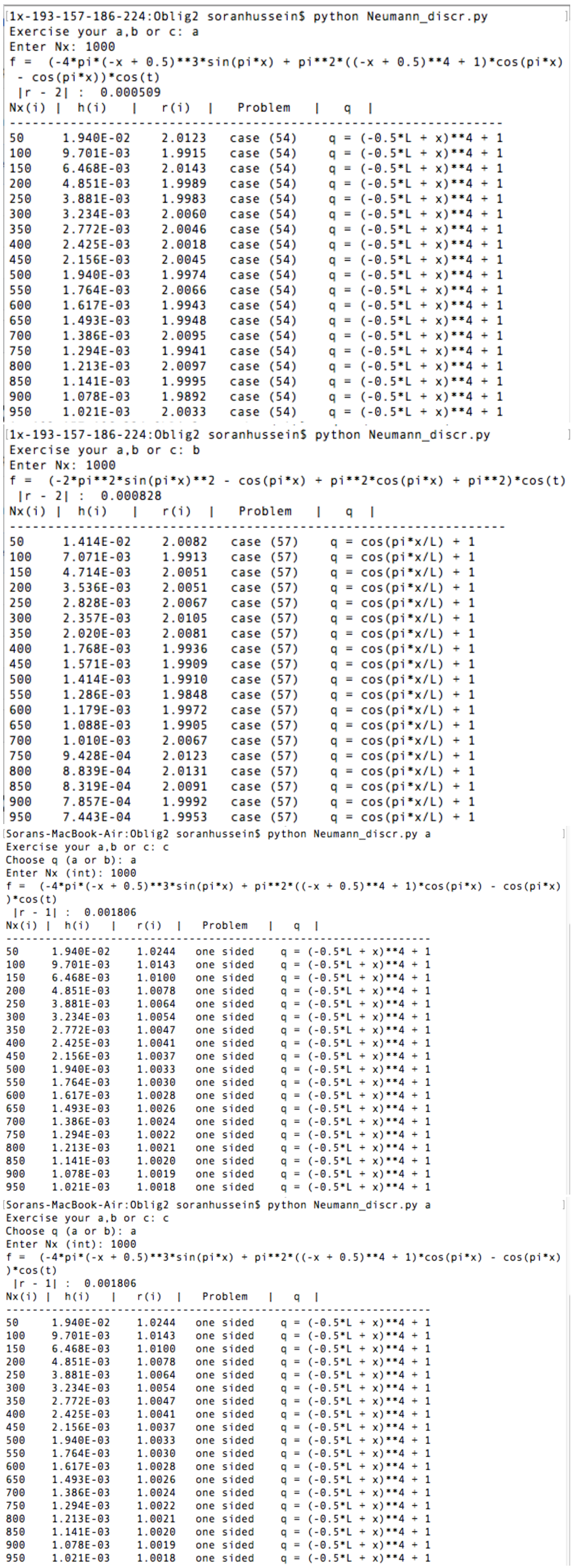
A Fourth technique that views the whole scheme:

$$[D_t D_t u]_i^n = \frac{1}{\Delta x}([qD_x u]_{i+\frac{1}{2}}^n - [qD_x u]_{i+-\frac{1}{2}}^n) + [f]_i^n$$

While the boundary at $x_{i+\frac{1}{2}}$ is the end, instead of exactly at the physical boundary (we move the boundary 1/2 "back"). I have not implemented this to my file, but need to change move back the initial value and change a bit in the other values

Conclusion

I have tested all the data with a chosen $N_x = 1000$, we can se that the convergence rate in a is much smaller then in b, means (54) is better approximation. In c we can se that when we use q in b we get a much smaller convergence ratet han q in s. It looks like this in each cases:



```
1x-193-157-186-224:Oblig2 soranhussein$ python Neumann_discr.py
Exercise your a,b or c: a
Enter Nx: 1000
f =  (-4*pi*(-x + 0.5)**3*sin(pi*x) + pi**2*((-x + 0.5)**4 + 1)*cos(pi*x)
 - cos(pi*x))*cos(t)
 |r - 2| :  0.000509
Nx(i) |  h(i)     |   r(i)   |    Problem    |   q  |
-----------------------------------------------------------------------
50      1.940E-02     2.0123    case (54)      q = (-0.5*L + x)**4 + 1
100     9.701E-03     1.9915    case (54)      q = (-0.5*L + x)**4 + 1
150     6.468E-03     2.0143    case (54)      q = (-0.5*L + x)**4 + 1
200     4.851E-03     1.9989    case (54)      q = (-0.5*L + x)**4 + 1
250     3.881E-03     1.9983    case (54)      q = (-0.5*L + x)**4 + 1
300     3.234E-03     2.0060    case (54)      q = (-0.5*L + x)**4 + 1
350     2.772E-03     2.0046    case (54)      q = (-0.5*L + x)**4 + 1
400     2.425E-03     2.0018    case (54)      q = (-0.5*L + x)**4 + 1
450     2.156E-03     2.0045    case (54)      q = (-0.5*L + x)**4 + 1
500     1.940E-03     1.9974    case (54)      q = (-0.5*L + x)**4 + 1
550     1.764E-03     2.0066    case (54)      q = (-0.5*L + x)**4 + 1
600     1.617E-03     1.9943    case (54)      q = (-0.5*L + x)**4 + 1
650     1.493E-03     1.9948    case (54)      q = (-0.5*L + x)**4 + 1
700     1.386E-03     2.0095    case (54)      q = (-0.5*L + x)**4 + 1
750     1.294E-03     1.9941    case (54)      q = (-0.5*L + x)**4 + 1
800     1.213E-03     2.0097    case (54)      q = (-0.5*L + x)**4 + 1
850     1.141E-03     1.9995    case (54)      q = (-0.5*L + x)**4 + 1
900     1.078E-03     1.9892    case (54)      q = (-0.5*L + x)**4 + 1
950     1.021E-03     2.0033    case (54)      q = (-0.5*L + x)**4 + 1
1x-193-157-186-224:Oblig2 soranhussein$ python Neumann_discr.py
Exercise your a,b or c: b
Enter Nx: 1000
f =  (-2*pi**2*sin(pi*x)**2 - cos(pi*x) + pi**2*cos(pi*x) + pi**2)*cos(t)
 |r - 2| :  0.000828
Nx(i) |  h(i)     |   r(i)   |    Problem    |   q  |
-----------------------------------------------------------------------
50      1.414E-02     2.0082    case (57)      q = cos(pi*x/L) + 1
100     7.071E-03     1.9913    case (57)      q = cos(pi*x/L) + 1
150     4.714E-03     2.0051    case (57)      q = cos(pi*x/L) + 1
200     3.536E-03     2.0051    case (57)      q = cos(pi*x/L) + 1
250     2.828E-03     2.0067    case (57)      q = cos(pi*x/L) + 1
300     2.357E-03     2.0105    case (57)      q = cos(pi*x/L) + 1
350     2.020E-03     2.0081    case (57)      q = cos(pi*x/L) + 1
400     1.768E-03     1.9936    case (57)      q = cos(pi*x/L) + 1
450     1.571E-03     1.9909    case (57)      q = cos(pi*x/L) + 1
500     1.414E-03     1.9910    case (57)      q = cos(pi*x/L) + 1
550     1.286E-03     1.9848    case (57)      q = cos(pi*x/L) + 1
600     1.179E-03     1.9972    case (57)      q = cos(pi*x/L) + 1
650     1.088E-03     1.9905    case (57)      q = cos(pi*x/L) + 1
700     1.010E-03     2.0067    case (57)      q = cos(pi*x/L) + 1
750     9.428E-04     2.0123    case (57)      q = cos(pi*x/L) + 1
800     8.839E-04     2.0131    case (57)      q = cos(pi*x/L) + 1
850     8.319E-04     2.0091    case (57)      q = cos(pi*x/L) + 1
900     7.857E-04     1.9992    case (57)      q = cos(pi*x/L) + 1
950     7.443E-04     1.9953    case (57)      q = cos(pi*x/L) + 1
Sorans-MacBook-Air:Oblig2 soranhussein$ python Neumann_discr.py a
Exercise your a,b or c: c
Choose q (a or b): a
Enter Nx (int): 1000
f =  (-4*pi*(-x + 0.5)**3*sin(pi*x) + pi**2*((-x + 0.5)**4 + 1)*cos(pi*x) - cos(pi*x)
)*cos(t)
 |r - 1| :  0.001806
Nx(i) |  h(i)     |   r(i)   |    Problem    |   q  |
-----------------------------------------------------------------------
50      1.940E-02     1.0244    one sided      q = (-0.5*L + x)**4 + 1
100     9.701E-03     1.0143    one sided      q = (-0.5*L + x)**4 + 1
150     6.468E-03     1.0100    one sided      q = (-0.5*L + x)**4 + 1
200     4.851E-03     1.0078    one sided      q = (-0.5*L + x)**4 + 1
250     3.881E-03     1.0064    one sided      q = (-0.5*L + x)**4 + 1
300     3.234E-03     1.0054    one sided      q = (-0.5*L + x)**4 + 1
350     2.772E-03     1.0047    one sided      q = (-0.5*L + x)**4 + 1
400     2.425E-03     1.0041    one sided      q = (-0.5*L + x)**4 + 1
450     2.156E-03     1.0037    one sided      q = (-0.5*L + x)**4 + 1
500     1.940E-03     1.0033    one sided      q = (-0.5*L + x)**4 + 1
550     1.764E-03     1.0030    one sided      q = (-0.5*L + x)**4 + 1
600     1.617E-03     1.0028    one sided      q = (-0.5*L + x)**4 + 1
650     1.493E-03     1.0026    one sided      q = (-0.5*L + x)**4 + 1
700     1.386E-03     1.0024    one sided      q = (-0.5*L + x)**4 + 1
750     1.294E-03     1.0022    one sided      q = (-0.5*L + x)**4 + 1
800     1.213E-03     1.0021    one sided      q = (-0.5*L + x)**4 + 1
850     1.141E-03     1.0020    one sided      q = (-0.5*L + x)**4 + 1
900     1.078E-03     1.0019    one sided      q = (-0.5*L + x)**4 + 1
950     1.021E-03     1.0018    one sided      q = (-0.5*L + x)**4 + 1
Sorans-MacBook-Air:Oblig2 soranhussein$ python Neumann_discr.py a
Exercise your a,b or c: c
Choose q (a or b): a
Enter Nx (int): 1000
f =  (-4*pi*(-x + 0.5)**3*sin(pi*x) + pi**2*((-x + 0.5)**4 + 1)*cos(pi*x) - cos(pi*x)
)*cos(t)
 |r - 1| :  0.001806
Nx(i) |  h(i)     |   r(i)   |    Problem    |   q  |
-----------------------------------------------------------------------
50      1.940E-02     1.0244    one sided      q = (-0.5*L + x)**4 + 1
100     9.701E-03     1.0143    one sided      q = (-0.5*L + x)**4 + 1
150     6.468E-03     1.0100    one sided      q = (-0.5*L + x)**4 + 1
200     4.851E-03     1.0078    one sided      q = (-0.5*L + x)**4 + 1
250     3.881E-03     1.0064    one sided      q = (-0.5*L + x)**4 + 1
300     3.234E-03     1.0054    one sided      q = (-0.5*L + x)**4 + 1
350     2.772E-03     1.0047    one sided      q = (-0.5*L + x)**4 + 1
400     2.425E-03     1.0041    one sided      q = (-0.5*L + x)**4 + 1
450     2.156E-03     1.0037    one sided      q = (-0.5*L + x)**4 + 1
500     1.940E-03     1.0033    one sided      q = (-0.5*L + x)**4 + 1
550     1.764E-03     1.0030    one sided      q = (-0.5*L + x)**4 + 1
600     1.617E-03     1.0028    one sided      q = (-0.5*L + x)**4 + 1
650     1.493E-03     1.0026    one sided      q = (-0.5*L + x)**4 + 1
700     1.386E-03     1.0024    one sided      q = (-0.5*L + x)**4 + 1
750     1.294E-03     1.0022    one sided      q = (-0.5*L + x)**4 + 1
800     1.213E-03     1.0021    one sided      q = (-0.5*L + x)**4 + 1
850     1.141E-03     1.0020    one sided      q = (-0.5*L + x)**4 + 1
900     1.078E-03     1.0019    one sided      q = (-0.5*L + x)**4 + 1
950     1.021E-03     1.0018    one sided      q = (-0.5*L + x)**4 + 1
```


Figure 1: All the cases