# NUMERICAL METHODS FOR PARTIAL DIFFERENTIAL EQUATIONS

PROJECT 1

ABINAYA A. SAKTHIVEL

*University of Oslo*

7. OCTOBER 2019

## 1 Mathematical problem

In this project we are looking at a the two-dimensional, standard, linear wave equation, with damping

$$\frac{\partial^2 u}{\partial t^2} + b\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(q(x,y)\frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(q(x,y)\frac{\partial u}{\partial y}\right) + f(x,y,t), \tag{1}$$

$$u(x,y,0) = I(x,y), \tag{2}$$

$$u_t(x,u,0) = V(x,y), \tag{3}$$

$$\frac{\partial u}{\partial n} = 0, \tag{4}$$

which is evaluated on a rectangular spatial domain $[0,L_x]\times[0,L_y]$. Parameters $b$, $q$, $I$ and $V$ are given, while $u(x,y,t)$ is the unknown function to be estimated.

## 2 Discretization of mathematical problem

### 2.1 General scheme of interior mesh points

The wave equation (eq. 1) can be translated to following notation for discretization

$$[D_tD_tu]^n_{i,j} + b\,[D_{2t}u]^n_{i,j} = [D_xqD_xu]^n_{i,j} + [D_yqD_yu]^n_{i,j} + [f]^n_{i,j}$$

Using centered differences method, the first- and second-order derivative with respect to time can be expressed as

$$\frac{\partial u}{\partial t} \approx [D_{2t}u]^n_{i,j} = \frac{u^{n+1}_{i,j} - u^{n-1}_{i,j}}{2\Delta t},$$

$$\frac{\partial^2 u}{\partial t^2} \approx [D_tD_tu]^n_{i,j} = \frac{u^{n+1}_{i,j} - 2u^n_{i,j} + u^{n-1}_{i,j}}{\Delta t^2},$$

respectively.

The following two terms are expressed as

$$\frac{\partial}{\partial x}\left(q(x,y)\frac{\partial u}{\partial x}\right) \approx [D_xqD_xu]^n_{i,j} = \frac{1}{\Delta x^2}\left[q_{i+\frac{1}{2},j}(u^n_{i+1,j} - u^n_{i,j}) - q_{i-\frac{1}{2},j}(u^n_{i,j} - u^n_{i-1,j})\right]$$

$$\frac{\partial}{\partial y}\left(q(x,y)\frac{\partial u}{\partial y}\right) \approx [D_yqD_yu]^n_{i,j} = \frac{1}{\Delta y^2}\left[q_{i,j+\frac{1}{2}}(u^n_{i,j+1} - u^n_{i,j}) - q_{i,j-\frac{1}{2}}(u^n_{i,j} - u^n_{i,j-1})\right]$$

By replacing the derivatives in eq. 1 with these finite differences gives the general scheme of the interior mesh points

$$\begin{aligned} u^{n+1}_{i,j} = \left(1 + \frac{b\Delta t}{2}\right)^{-1} \Bigg[ & 2u^n_{i,j} + (\frac{b}{2}\Delta t - 1)u^{n-1}_{i,j} + \Delta t^2 f^n_{i,j} \\ & + C_x^2\left(q_{i+\frac{1}{2},j}(u^n_{i+1,j} - u^n_{i,j}) - q_{i-\frac{1}{2},j}(u^n_{i,j} - u^n_{i-1,j})\right) \\ & + C_y^2\left(q_{i,j+\frac{1}{2}}(u^n_{i,j+1} - u^n_{i,j}) - q_{i,j-\frac{1}{2}}(u^n_{i,j} - u^n_{i,j-1})\right)\Bigg], \end{aligned} \tag{5}$$

where $C_x^2 = \frac{\Delta t^2}{\Delta x^2}$ and $C_y^2 = \frac{\Delta t^2}{\Delta y^2}$.

## 2.2 General scheme of the first step at interior points

The initial condition (3) imply that

$$\frac{u^1_{i,j} - u^{-1}_{i,j}}{2\Delta t} = V \implies u^{-1}_{i,j} = u^1_{i,j} - 2\Delta t \tag{6}$$

along with condition (2) in (5) gives following expression for the general scheme of the first step

$$\begin{aligned} u^1_{i,j} = u^0_{i,j} + \Delta t V_{i,j} - \frac{1}{2} b \Delta t V_{i,j} + \frac{1}{2}\Big[\Delta t^2 f^0_{i,j} \\ + C_x^2 \left( q_{i+\frac{1}{2},j}(u^0_{i+\frac{1}{2},j} - u^0_{i,j}) - q_{i-\frac{1}{2},j}(u^0_{i,j} - u^0_{i-\frac{1}{2},j}) \right) \\ + C_y^2 \left( q_{i,j+\frac{1}{2}}(u^0_{i,j+\frac{1}{2}} - u^0_{i,j}) - q_{i,j-\frac{1}{2}}(u^0_{i,j} - u^0_{i,j-\frac{1}{2}}) \right) \Big]. \end{aligned} \tag{7}$$

## 2.3 Boundary conditions

We use Neumann's condition on the boundary,

$$\frac{\partial u}{\partial n} = \mathbf{n} \cdot \nabla u = n_x \frac{\partial u}{\partial x} + n_y \frac{\partial u}{\partial y} = 0.$$

Lets focus on the edges in x-direction in spatial domain. The normal vector points out of the spatial domain giving $\frac{\partial u}{\partial n} = -\frac{\partial u}{\partial x}$ for $x = 0$ and $\frac{\partial u}{\partial n} = \frac{\partial u}{\partial x}$ for $x = N_x$. Discretizing at the leftmost and rightmost mesh points gives

$$\frac{u^n_{1,j} - u^n_{-1,j}}{2\Delta x} = 0 \implies u^n_{-1,j} = u^n_{1,j} \tag{8}$$

$$\frac{u^n_{N_x+1,j} - u^n_{N_x-1,j}}{2\Delta x} = 0 \implies u^n_{N_x+1,j} = u^n_{N_x-1,j}. \tag{9}$$

Similar discretization are done for y-direction in spatial domain, which gives

$$\frac{u^n_{i,1} - u^n_{i,-1}}{2\Delta y} = 0 \implies u^n_{i,-1} = u^n_{i,1}. \tag{10}$$

$$\frac{u^n_{i,N_y+1} - u^n_{i,N_y-1}}{2\Delta y} = 0 \implies u^n_{i,N_y+1} = u^n_{i,N_y-1}. \tag{11}$$

Further, we add an addition to the Neumann condition by assuming $\partial q / \partial x = 0$, this gives following conditions

$$\frac{q^n_{i+\frac{1}{2},j} - q^n_{i-\frac{1}{2},j}}{2\Delta x} = 0 \implies q^n_{i+\frac{1}{2},j} = q^n_{i-\frac{1}{2},j} \tag{12}$$

$$\frac{q^n_{i,j+\frac{1}{2}} - q^n_{i,j-\frac{1}{2}}}{2\Delta x} = 0 \implies q^n_{i,j+\frac{1}{2}} = q^n_{i,j-\frac{1}{2}} \tag{13}$$

These boundary conditions are used in (5) and (7) at the leftmost and rightmost mesh points, $x = 0, N_x$ and $y = 0, N_y$, for all eight combinations, (i=0, j), (i=$N_x$, j), (i, j=0), (i, j=$N_y$), (i=0, j=0), (i=$N_x$, j=$N_y$), (i=$N_x$, j=0), (i=0, j=$N_y$).

## 3 Verification

### 3.1 Constant solution

For a test case we set $u(x, y, t) = 5$ as a solution. The discrete equation as expected, gives the same solution.

```
$ python tests.py

Testing constant u:

testing scalar for 4x4 mesh
[[5.  5.  5.  5.  5.]
 [5.  5.  5.  5.  5.]
 [5.  5.  5.  5.  5.]
 [5.  5.  5.  5.  5.]
 [5.  5.  5.  5.  5.]]
Largest error: 0.00e+00
testing vectorized for 4x4 mesh
[[5.  5.  5.  5.  5.]
 [5.  5.  5.  5.  5.]
 [5.  5.  5.  5.  5.]
 [5.  5.  5.  5.  5.]
 [5.  5.  5.  5.  5.]]
Largest error: 0.00e+00
```

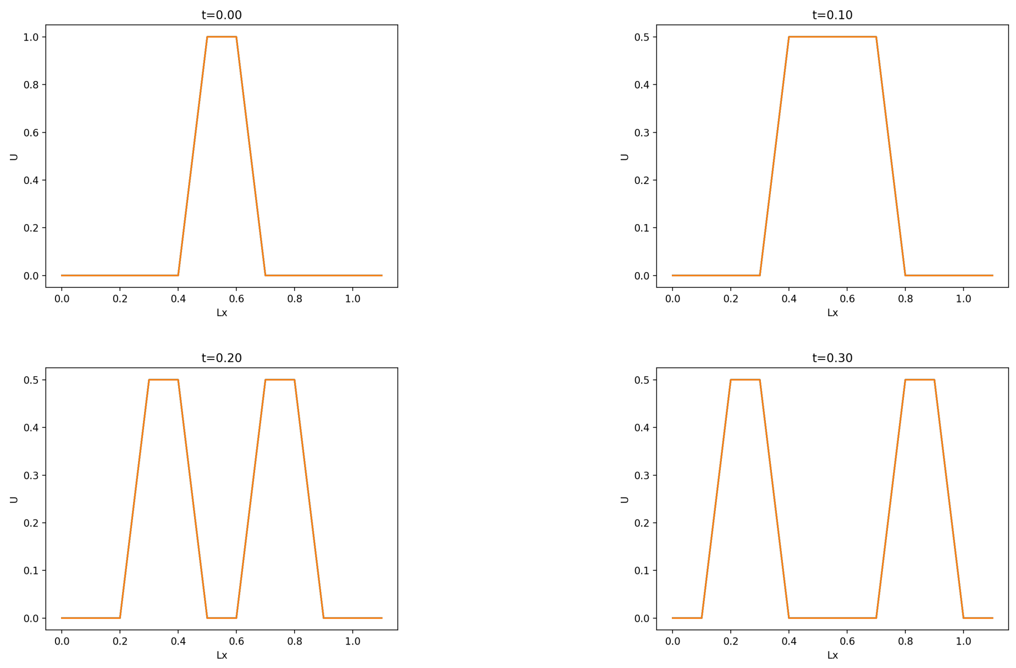
Rest of the parameters where fitted to

$$I = 5, \quad V = 0, \quad f = 0, \quad b = 1.5, \quad q = 1 \tag{14}$$

To investigate what types of error a constant solution could produce, we tested non-zero values for $V$ and $f$. This raised error. When $I$ was set to some lower or higher value than $u$, error where raised, but this time the $u$ was off by the factor that was added or subtracted in $I$. Changing the parameters $b$ and $q$ did not have a impact, thus did not raise any error.

### 3.2 Exact 1D solution

We propagate a plug wave where $I(x) = 0$ when $x - Lx/2 > 0.1$ and else 1, and similarly for $I(y)$. The initial wave splits into two waves moving in opposite directions. The figure underneath shows the wave in x-direction of the spatial domain splitting in half.



## 3.3 Standing, undamped waves

With no damping and constant wave velocity, an exact solution of a standing wave is as follows

$$u_e(x, y, t) = Acos(k_x X)cos(k_y y)cos(\omega t), \quad k_x = \frac{m_x \pi}{L_x}, k_y = \frac{m_y \pi}{L_y}. \tag{15}$$

The true error is defines as $e^n_{i,j} = u_e(x, y, t) - u$, while the error norm is calculated as E = $\left\|e^n_{i,j}\right\|$. We are interested in the convergence rate, which is calculated as

$$r = \frac{log\left(E_{i+1}/E_i\right)}{log\left(h_{i+1}/h_i\right)}, \tag{16}$$

where $h = \Delta t$.

```
python tests.py
Testing convergence rates of undamped waves:

Running Nx=Ny: 5
Running Nx=Ny: 10
Running Nx=Ny: 20
Running Nx=Ny: 40
Running Nx=Ny: 80
Running Nx=Ny: 160
Running Nx=Ny: 320

            CONVERGENCE RATES WITH DETAILS
 N(i) | N(i+1) |    dt(i)    |   dt(i+1)   |   r(i)   |

 5        10       1.414E-01   7.071E-02   2.4547
 10       20       7.071E-02   3.536E-02   2.3666
 20       40       3.536E-02   1.768E-02   1.9534
 40       80       1.768E-02   8.839E-03   2.0923
 80       160      8.839E-03   4.419E-03   1.9892
 160      320      4.419E-03   2.210E-03   2.0232
```

As $\Delta t$ gets smaller, thus also $\Delta x$ and $\Delta x$, we see that the discrete solution converges to 2, as we would hope for.

## 3.4 Manufactured solution

The source term is found by using **sympy** in **tests.py**. This time the convergence rate is far off than expected, it does not actually converge, rather jumps from positive to negative values. There is definitely some bugs (well hidden) here.

```
Testing convergence rates for manufactured solution

Running Nx=Ny: 5
Running Nx=Ny: 10
Running Nx=Ny: 20
Running Nx=Ny: 40
Running Nx=Ny: 80
Running Nx=Ny: 160
Running Nx=Ny: 320

            CONVERGENCE RATES WITH DETAILS
 N(i) | N(i+1) |    dt(i)    |   dt(i+1)   |   r(i)   |

 5        10       1.414E-01   7.071E-02   3.3174
 10       20       7.071E-02   3.536E-02   -1.4782
 20       40       3.536E-02   1.768E-02   0.8525
 40       80       1.768E-02   8.839E-03   -0.5382
 80       160      8.839E-03   4.419E-03   0.2443
 160      320      4.419E-03   2.210E-03   -0.1164
```

## 3.5 Investigate a physical problem

For the physical problem we explore what happens to a wave that enters a medium with different wave velocity. A particular physical interpretation can be wave propagation of a tsunami over a subsea hill. For each 10th step, a plots of $u$ was created and saved in a separate folder. After saving all the plots, another function collects together all the saved figures to form a movie. All movies are saved in the folder called **Movies**.

Since we are using a numerically stable $\Delta t$ the simulations are low on noise. All three physical problems we have tested and created movies of seem quite natural. The Gaussian hill in my eyes looks most natural of them all. This wave gets high on amplitude as it hits the wall, which is what we would have expected in the real world. The cosine hat has a bit less movement as it hits the wall, and might be the one that is most unlike a natural wave. The box hill has a more dramatic movements as it hits the hill, this might be a case where a more crucial situation takes place, such as tsunami.