Effects of Hill Shapes
on Tsunami Simulations

Kristian Pedersen and Gustav Baardsen

October 15, 2012

Abstract

In this project we study how the shape of hills at the sea bottom affects numerical and physical properties of simulations of earthquake-generated tsunamis. The tsunamis are modelled using two-dimensional wave equations, and a finite difference scheme is used to solve the partial differential equations.

Contents

1 Introduction

In this text we will addresses the two-dimensional, standard, linear wave equation, with damping and reflecting boundaries. We will develop a scheme for solving it and apply it to the problem of a tsunami over an uneven seabed. Especial focus will be put on the question of which kinds of seabed causes numerical instability.

2   Mathematical model

For x,y in the domain $D = [0, L_y] \times [0, L_x]$ and $t \in [0, T]$ we will study the following set of equations:

$$u_t t + b u_t = (q(x, y) u_x)_x + (q(x, y) u_y)_y + f(x, y, t), \quad x, y \in D, t \in [0, T] \quad (1)$$

$$\frac{\delta u}{\delta n} = 0 \qquad\qquad\qquad x, y \in \delta D, t \in [0, T] \quad (2)$$

$$u(x, y, 0) = I(x, y) \qquad\qquad\qquad x, y \in \delta D \quad (3)$$

$$u_t(x, y, 0) = V(x, y) \qquad\qquad\qquad x, y \in \delta D \quad (4)$$

3   Numerical scheme

3.1   Numerical scheme for the inner points

In operator notation we want to use the following scheme for inner points:

$$[D_t D_t u + b D_2 t u = D_x(q D_x u) + D_y(q D_y u) + f]_{ij}^n \quad (5)$$

Calculating each summand we find:

$$[D_t D_t u]_{ij}^n = \frac{u_{i,j}^{n+1} - 2u_{i,j}^n + u_{i,j}^{n-1}}{\Delta t^2} \quad (6)$$

$$[b D_2 t u]_{ij}^n = b \frac{u_{i,j}^{n+1} - u_{i,j}^{n-1}}{2\Delta t} \quad (7)$$

$$[D_x q D_x u]_{ij}^n = \frac{q_{i+\frac{1}{2},j}(u_{i+1,j}^n - u_{i,j}^n) - q_{i-\frac{1}{2},j}(u_{i,j}^n - u_{i-1,j}^n)}{\Delta x^2} \quad (8)$$

$$[D_y q D_y u]_{ij}^n = \frac{q_{i,j+\frac{1}{2}}(u_{i,j+1}^n - u_{i,j}^n) - q_{i,j-\frac{1}{2}}(u_{i,j}^n - u_{i,j-1}^n)}{\Delta x^2} \quad (9)$$

$$[f]_{ij}^n = f(x_i, y_i, t_i) \quad (10)$$

Inserting these expressions in (5) and solving for $u_{i,j}^{n+1}$ we get the following scheme for inner points:

$$u_{i,j}^{n+1} = \frac{2u_{i,j}^n + (\frac{1}{2}b\Delta t - 1.0)u_{i,j}^{n-1}}{1 + \frac{1}{2}b\Delta t}$$

$$+ \frac{\Delta t^2}{\Delta x^2(\frac{1}{2}b\Delta t - 1.0)}(q_{i+\frac{1}{2},j}(u_{i+1,j}^n - u_{i,j}^n) - q_{i-\frac{1}{2},j}(u_{i,j}^n - u_{i-1,j}^n))$$

$$+ (\frac{\Delta t^2}{\Delta y^2(\frac{1}{2}b\Delta t - 1.0)}(q_{i,j+\frac{1}{2}}(u_{i,j+1}^n - u_{i,j}^n) - q_{i,j-\frac{1}{2}}(u_{i,j}^n - u_{i,j-1}^n))$$

$$+ \frac{\Delta t^2}{\frac{1}{2}b\Delta t - 1.0}f(x_i, y_i, t_i)$$

3.2 Numerical scheme for the first time step

For the first time step $u_{i,j}^{n-1}$ isn't a part of the grid. To circumvent this problem we have to apply the intial condition $u_t(x, y, 0) = V(x, y)$ for $x, y \in \delta D$. We discretize it and get:

$$[D_{2t}u = V]_{ij}^{0} \implies \frac{u_{ij}^{1} - u_{ij}^{-1}}{2\Delta t} = V(x_i, y_j) \implies u_{ij}^{-1} = u_{ij}^{1} - 2\Delta t V(x_i, y_j)$$

Inserting this in equation (6) and (12) we get:

$$[D_t D_t u]_{ij}^{n} = 2\left(\frac{u_{i,j}^{n+1} - u_{i,j}^{n}}{\Delta t^2} - \frac{V(x_i, y_i)}{\Delta t}\right) \tag{11}$$

$$[bD_2 tu]_{ij}^{n} = b\frac{V(x_i, y_i)}{\Delta t} \tag{12}$$

Inserting these expressions in (5) and solving for $u_{i,j}^{n+1}$ we get the following scheme for inner points in the first time step:

$$u_{i,j}^{n+1} = u_{i,j}^{n} + (1 - \frac{1}{2}b\Delta t) * \Delta t V_{i,j}^{n-1}$$
$$+ \frac{\Delta t^2}{2\Delta x^2}(q_{i+\frac{1}{2},j}(u_{i+1,j}^{n} - u_{i,j}^{n}) - q_{i-\frac{1}{2},j}(u_{i,j}^{n} - u_{i-1,j}^{n}))$$
$$+ (\frac{\Delta t^2}{\Delta 2y^2}(q_{i,j+\frac{1}{2}}(u_{i,j+1}^{n} - u_{i,j}^{n}) - q_{i,j-\frac{1}{2}}(u_{i,j}^{n} - u_{i,j-1}^{n}))$$
$$+ \frac{\Delta t^2}{2}f(x_i, y_i, t_i)$$

3.3 Numerical scheme for the boundary

On the boundary this scheme won't work as one, when adding or subtracting one from one of the indices, risks leaving grid. We discretize the Neumann condition to solve this problem:

$$[D_{2x}u]_{0j}^{n} = [D_{2x}u]_{Nxj}^{n} = 0 \quad \text{for j} = 0,1..N_y, \text{ n} = 0,1,..N_t \tag{13}$$

$$[D_{2y}u]_{i0}^{n} = [D_{2y}u]_{0,Ny}^{n} = 0 \quad \text{for i} = 0,1..N_y, \text{ n} = 0,1,..N_t \tag{14}$$

Which leads to the equations:

$$u_{1,j}^{n} = u_{-1,j}^{n}, u_{N_x-1,j}^{n} = u_{Nx+1,j}^{n} \quad \text{for j} = 0,1..N_y, \text{ n} = 0,1,..N_t \tag{15}$$

$$u_{i,1}^{n} = u_{i,-1}^{n}, u_{i,Ny+1}^{n} = u_{i,Nx+1}^{n} \quad \text{for i} = 0,1..N_y, \text{ n} = 0,1,..N_t \tag{16}$$

A scheme for the borders can now be found by applying each equality to the inner point scheme for its boundary. The corners are found by applying two equalities, one for each of the adjacent border.

3

3.4 Approximating q(x,y) outside the grid

To approximate the q function when evaluated outside the grid we will apply the arithmetic and harmonic mean:

$$q_{i+\frac{1}{2},j} = \frac{q_{i,j} + q_{i+1,j}}{2} \tag{17}$$

$$q_{i+\frac{1}{2},j} = 2(\frac{1}{q_{i,j}} + \frac{1}{q_{i+1,j}})^{-1} \tag{18}$$

4 Implementation

We implemented the code in pure python. See the wave2D_du0.py for the full code.

TODO: Show the core of the program in minted enviroment

5 Numerical experiments

6 Conclusions