CERI- 8104/CIVL-8126

Data Analysis in Geophysics

Finite Difference Approximation of 1D Wave Equation (50 points)

Submit a short report with 4 figures documenting your results! Submit your MATLAB code along with a PDF document. The latter needs to be generated with latex.

Required submission files:

1) PDF of report (include 4 figures: 1. colormap of wave amplitudes 2. three amplitude profiles at first, 50^{th} and last time step)
2) MATLAB code used to solve the problem set

I. Theoretical Background

Consider the homogeneous 1D wave equation:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2} \tag{1}$$

The equation describes the second order spatial and time derivatives of particle displacements, u, and speed, c of a planar wave propagating in the x-direction.

The general solution to Eq. 1 has the form:

$$u(x,t) = f(x-ct) + g(x+ct).$$

Eq. 1 can be factorized so that:

$$\left(\frac{\partial}{\partial t} + c\,\frac{\partial}{\partial x}\right)\left(\frac{\partial}{\partial t} - c\,\frac{\partial}{\partial x}\right)u = 0$$

It follows that either:

$$\left(\frac{\partial}{\partial t} + c\,\frac{\partial}{\partial x}\right)u = u_t + cu_x = 0 \tag{2}$$

Or

$$\left(\frac{\partial}{\partial t} - c\,\frac{\partial}{\partial x}\right)u = u_t - cu_x = 0 \tag{3}$$

In the first case, the solution is $u = f(x-ct)$ (wave travelling to the right) and in the second case $u = f(x+ct)$ (wave traveling to the left). We will examine numerical solutions to Eq. 2 in more detail.

We can rewrite Eq. 2 as:

$$\frac{\partial u}{\partial t} = -c\frac{\partial u}{\partial x} \tag{4}$$

We will solve this equation using an explicit finite difference approach.

More specifically, we will rewrite Eq. 4 using forward differences in time and backward differences for the spatial derivative:

$$\frac{u_i^{n+1}-u_i^n}{\Delta t} = -c\,\frac{u_i^n-u_{i-1}^n}{\Delta x} \tag{5}$$

where *i* denote indices with respect to space, and *n* indices with respect to time. Δx and Δt are the respective spatial and temporal increments. We can now solve Eq. 5 for the amplitude profile u_i, at any future time step *n*+1:

$$u_i^{n+1} = u_i^n - c\,\frac{\Delta t}{\Delta x}\left(u_i^n - u_{i-1}^n\right) \tag{6}$$

Equation 6 is the key equation that we will implement in MATLAB to numerically approximate a solution to Eq. (4).
The numerical approximation requires initial (IC) and boundary conditions (BC):

IC: $u(x, t=0) = g(x)$
BC: $u(0,t) = 0, u(N,t) = 0$, with *i*=1,…,*N*

II. Problem set:

Solve the unidirectional wave equation in 1D over the spatial domain $\Omega = [0,1]$ km given the initial conditions:

IC: $g(x) = A_0 \exp\left(-\frac{(x-\mu)^2}{\sigma^2}\right)$. and

BC: $u(0,t) = 0, u(N,t) = 0,$ with *i*=1,…,*N*

The initial conditions describe a Gaussian amplitude profile with peak amplitude A_0, location, μ and scale, σ. You can use the following numerical values for these parameters:

A_0=1.5, μ= 0.3 and σ=0.005.

The wave speed, c is 0.5 km/s.

Use a grid spacing of Δx = 0.01 ($n_x = x_{max}/dx + 1$) and a time step of $\Delta t = 0.5\Delta x / c$. Note that the explicit solution becomes unstable for large time steps. The stability criterion for the numerical solution is given by: $c\Delta t/\Delta x \leq 0.5$.

To solve Eq. 6, you have to implement a nested *for-loop*. Create an equally spaced time vector between 0 and t_{max}=1 in Δt increments. Solve for the amplitude profile at each new time step, with the amplitudes at the first time-step being equal to the initial conditions.

a) Plot the amplitude profile at every time step and describe your observations!
b) In which direction does the wave propagate?
c) What happens to the amplitude of the signal as it propagates? (This is an artifact of the numerical method, not a property of the solution.)

d) Create a matrix that holds the amplitude profiles at each time step (i.e. dim= $nx \times nt$). Store the amplitude vector at each time step in this matrix. We will now use this matrix to compute the slowness: $s = 1/c$.

To visualize the propagating matrix, you can use:

```
[m_x, m_t] = meshgrid( a_x, a_t);
plot3 = pcolor(ax3, m_x, m_t, flipud(m_u'));
shading interp;
axis on
```

Use the resulting figure to graphically estimate the slowness. Discuss your results!