Numerical Methods in ESP

Numerical Methods II

Graziano Giuliani

International Centre for Theoretical Physics

Second Semester 2023-24

/afs/ictp.it/public/g/ggiulian/WORLD/num2_lesson8.pdf

Gravity Wave 1D 1

- Consider the one-dimensional shallow water equations, with Coriolis forces neglected, and linearized about a state of rest $u = 0$, $\Phi = gH$.

$$
\begin{aligned}
\frac{\partial u}{\partial t} + \frac{\partial \phi}{\partial x} &= 0 \quad momentum \\
\frac{\partial \phi}{\partial t} + \Phi \frac{\partial u}{\partial x} &= 0 \quad mass
\end{aligned}
\tag{1}
$$

- This system of equations support solutions of the form

$$
\begin{aligned}
u(x,t) &= \hat{u} e^{i(kx - \omega t)} \\
\phi(x,t) &= \hat{\phi} e^{i(kx - \omega t)}
\end{aligned}
\tag{2}
$$

with dispersion relation $\omega^2 = k^2 \Phi$.

- Solution are non-dispersive waves all with same phase speed: $\omega / k = \sqrt{\Phi}$.

Gravity Wave 1D 2

- If we consider over a periodic domain initial conditions of the form:

$$\begin{aligned} \phi(x,0) &= F(x) \\ u(x,0) &= 0 \end{aligned} \tag{3}$$

- the solution is:

$$\phi(x,t) = \frac{1}{2}F(x-at) + \frac{1}{2}F(x+at) \tag{4}$$

Gravity Wave Discretization - M1

- We can discretize the shallow water equations [1] using second-order centered differences in space and time:

$$\begin{aligned}
\frac{u_j^{n+1} - u_j^{n-1}}{2\Delta t} + \frac{\phi_{j+1}^{n} - \phi_{j-1}^{n}}{2\Delta x} &= 0 \\
\frac{\phi_j^{n+1} - \phi_j^{n-1}}{2\Delta t} + \Phi \frac{u_{j+1}^{n} - u_{j-1}^{n}}{2\Delta x} &= 0
\end{aligned} \tag{5}$$

- Rearranging for the unknowns we have:

$$\begin{aligned}
u_j^{n+1} &= u_j^{n-1} - \frac{\Delta t}{\Delta x}\left(\phi_{j+1}^{n} - \phi_{j-1}^{n}\right) \\
\phi_j^{n+1} &= \phi_j^{n-1} - \Phi \frac{\Delta t}{\Delta x}\left(u_{j+1}^{n} - u_{j-1}^{n}\right)
\end{aligned} \tag{6}$$

Gravity Wave Discretization - M2

- We can discretize the shallow water equations [1] using center in space and forward/backward in time:

$$\begin{aligned} \frac{u_j^{n+1} - u_j^n}{\Delta t} + \frac{\phi_{j+1}^n - \phi_{j-1}^n}{2\Delta x} &= 0 \\ \frac{\phi_j^{n+1} - \phi_j^n}{\Delta t} + \Phi \frac{u_{j+1}^{n+1} - u_{j-1}^{n+1}}{2\Delta x} &= 0 \end{aligned} \tag{7}$$

- Rearranging for the unknowns we have:

$$\begin{aligned} u_j^{n+1} &= u_j^n - \frac{\Delta t}{2\Delta x}\left(\phi_{j+1}^n - \phi_{j-1}^n\right) \\ \phi_j^{n+1} &= \phi_j^n - \Phi \frac{\Delta t}{2\Delta x}\left(u_{j+1}^{n+1} - u_{j-1}^{n+1}\right) \end{aligned} \tag{8}$$

Gravity Wave solution stability M1-1

- Looking for solutions of the form:

$$\begin{aligned} u_j^n &= A^n e^{ikj\Delta x} \\ \phi_j^n &= BA^n e^{ikj\Delta x} \end{aligned} \tag{9}$$

- we obtain by substituting in [5]:

$$\begin{aligned} A^2 &= 1 - \frac{\Delta t}{\Delta x} BA \left(e^{ik\Delta x} - e^{-ik\Delta x} \right) \\ BA^2 &= B - \Phi \frac{\Delta t}{\Delta x} A \left(e^{ik\Delta x} - e^{-ik\Delta x} \right) \end{aligned} \tag{10}$$

Gravity Wave solution stability M1-2

- Eliminate B and rearrange, and let $c^2 = \Phi(\Delta t/\Delta x)^2$ (c is the Courant number for this problem)

$$(A^2)^2 + A^2(4(c\ sin(k\Delta x)^2) - 2) + 1 = 0 \tag{11}$$

- We have 4 solutions for A. Two give physical modes, corresponding to left and right propagating gravity waves. The other two give computational modes.

$$A^2 = 1 - 2(c\ sin(k\Delta x)) \pm \sqrt{[-4(c\ sin(k\Delta x))^2 + 4(c\ sin(k\Delta x))^4]} \tag{12}$$

  - If $|c| \leq 1$ then the square root is purely imaginary and $|A^2| = 1$; the scheme is stable.
  - If $|c| > 1$ then for some value of k the square root is real, $|A^2| > 1$ for at least one of the roots, and the scheme is unstable.

Gravity Wave solution stability M2-1

- Looking for solutions of the form:

$$
\begin{aligned}
u_j^n &= \mathbf{H} A^n e^{ikj\Delta x} \\
\phi_j^n &= \mathbf{U} A^n e^{ikj\Delta x}
\end{aligned}
\tag{13}
$$

- we obtain by substituting in [5]:

$$
A = 1 - \frac{c^2}{2} sin^2(k\Delta x) \pm \frac{ic}{2} sin(k\Delta x)\sqrt{4 - c^2 sin^2(k\Delta x)} \tag{14}
$$

Gravity Wave solution stability M2-2

- The $\pm$ here comes from the two phisical modes solutions for the two waves travelling in opposite directions as above
- No computational mode is present
- For $|c| < 2$ we obtain $|A|^2 = 1$, and thus the scheme is stable and we do not have damping.
- For $|c| > 2$, for some k the amplification factor can be greater than one, and thus the scheme is unstable.

Exercise on Gravity Waves

- Write a Fortran program to integrate the gravity wave equations in [1] using discretization in [5] and [7] in the domain $0 \leq x \leq 1000m$ with mean height of the system such that $\Phi = gH = 1m^2/s^2$. Let $\Delta x = 0.5m$ and assume periodic boundary conditions. Assume the initial shape to be:

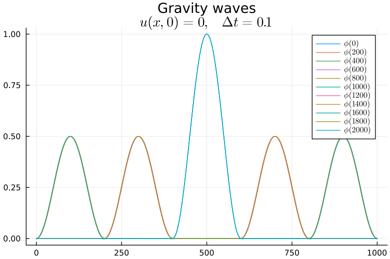
$$\phi(x,0) = \begin{cases} 0.0 & for \quad x < 400 \\ sin(((x-400)/200)*\pi)^2 & for \quad 400 \leq x < 600 \\ 0.0 & for \quad x > 600 \end{cases} \tag{15}$$

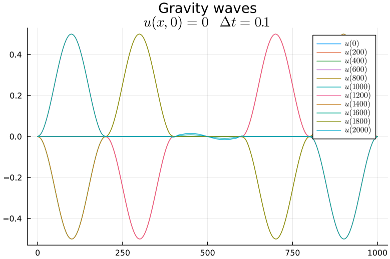
  and initial velocity $u(x,0) = 0$.

- Integrate forward and show solutions from $t = 0s$ up to $t = 2000s$ every $200s$ and explain the characteristics of the solution.

Attention: Select Δt to have a stable scheme remembering the CFL condition!

Expected result - Method 1

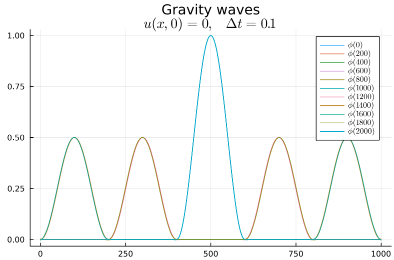


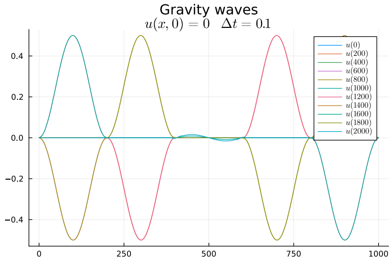


Julia Code

```
dtdx = dt/dx;
c = sqrt(P) * dtdx;
function ueq_ft(u_now,p_now)
  u_now - 0.5 * dtdx * (circshift(p_now,-1) - circshift(p_now,1))
end;
function peq_ft(p_now,u_now)
  p_now - 0.5 * c * (circshift(u_now,-1) - circshift(u_now,1))
end;
function ueq(u_old,p_now)
  u_old - dtdx * (circshift(p_now,-1) - circshift(p_now,1))
end;
function peq(p_old,u_now)
  p_old - c * (circshift(u_now,-1) - circshift(u_now,1))
end;
p_now = peq_ft(p_old,u_old);
u_now = ueq_ft(u_old,p_old);
t = t0+dt;
while (t < t1)
  u_new = ueq(u_old,p_now);
  p_new = peq(p_old,u_now);
  global p_old[:] = p_now;
  global p_now[:] = p_new;
  global u_old[:] = u_now;
  global u_now[:] = u_new;
  global t = t + dt;
  if mod(t,tp) < dt
    it = Int(round(t))
    plot!(...)
  end;
end;
```

Expected result - Method 2





Julia Code

```
dtdx = dt/dx;
c = sqrt(P) * dtdx;
function ueq_fo(u_now,p_now)
  u_now - 0.5 * dtdx * (circshift(p_now,-1) - circshift(p_now,1))
end;
function peq_fo(p_now,u_fut)
  p_now - c * 0.5 * (circshift(u_fut,-1) - circshift(u_fut,1))
end;

t = t0;
while (t < t1)
  global t = t + dt;
  u_new = ueq_fo(u_now,p_now);
  p_new = peq_fo(p_now,u_new);
  global p_now[:] = p_new;
  global u_now[:] = u_new;
  if mod(t,tp) < dt
    it = Int(round(t))
    plot!(...)
  end;
end;
```