Ciências ULisboa

Modelação Numérica 2022
Aula 11

Modelo de ondas gravíticas em águas pouco profundas (shallow-water)

O modelo shallow water

Velocidade de fase das ondas:

$$c = \sqrt{gH} \gg u, v$$

Independente do comprimento de onda (**ondas não dispersivas**).

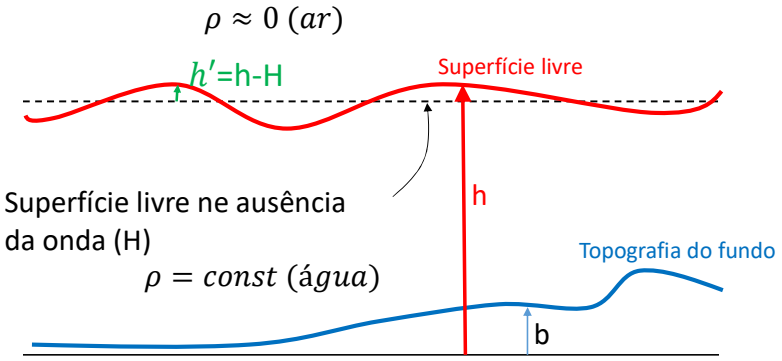
A partir da equação de Navier-Stokes para um **fluido incompressível**, pode obter-se esta equação relacionando a altura da superfície livre do fluido (h) com a velocidade horizontal media vertical (u, v) , no caso de ondas **com comprimento de onda muito maior que a profundidade**

$$\frac{\partial u}{\partial t} = -\frac{\partial}{\partial x}\left(u^2\right) - \frac{\partial}{\partial y}(uv) - g\frac{\partial h}{\partial x}$$

$$\frac{\partial v}{\partial t} = -\frac{\partial}{\partial x}(uv) - \frac{\partial}{\partial y}\left(v^2\right) - g\frac{\partial h}{\partial y}$$

$$\frac{\partial h}{\partial t} = -\frac{\partial}{\partial x}((h-b)u) - \frac{\partial}{\partial y}\left((h-b)v\right)$$

- ✓ 3 equações
- ✓ 3 incógnitas (u, v, h)
- ✓ Não linear



No oceano aberto $H \approx 5000\ m$:

$$c = \sqrt{gH} \approx 223\ ms^{-1} \approx 800\ km\ h^{-1}$$

Modelo para ondas longas ($\lambda \gg 5000m$), e.g. maré, tsunamis

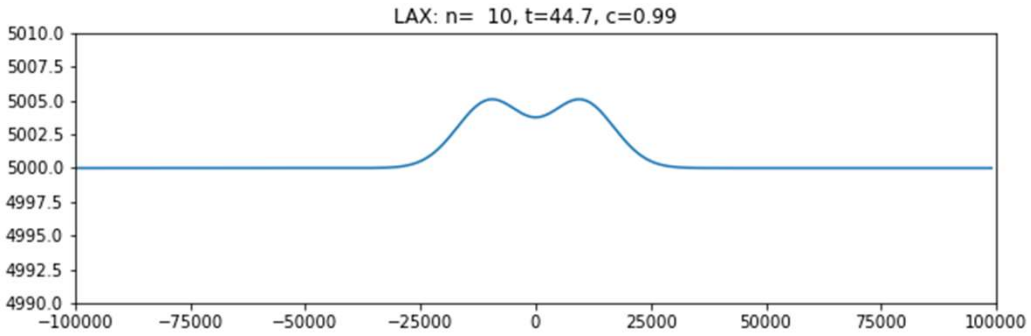
O modelo shallow water

Condições fronteira cíclicas (ou periódicas)

A 1 dimensão:

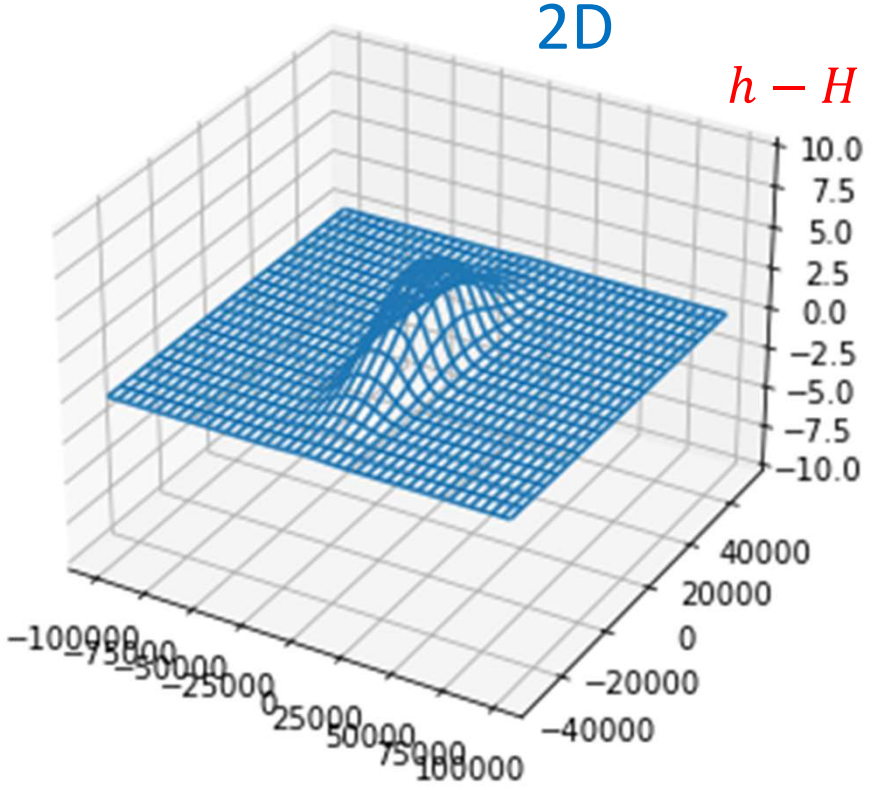
$$\frac{\partial u}{\partial t} = -\frac{\partial}{\partial x}(u^2) - g\frac{\partial h}{\partial x}$$

$$\frac{\partial h}{\partial t} = -\frac{\partial}{\partial x}((h-b)u)$$

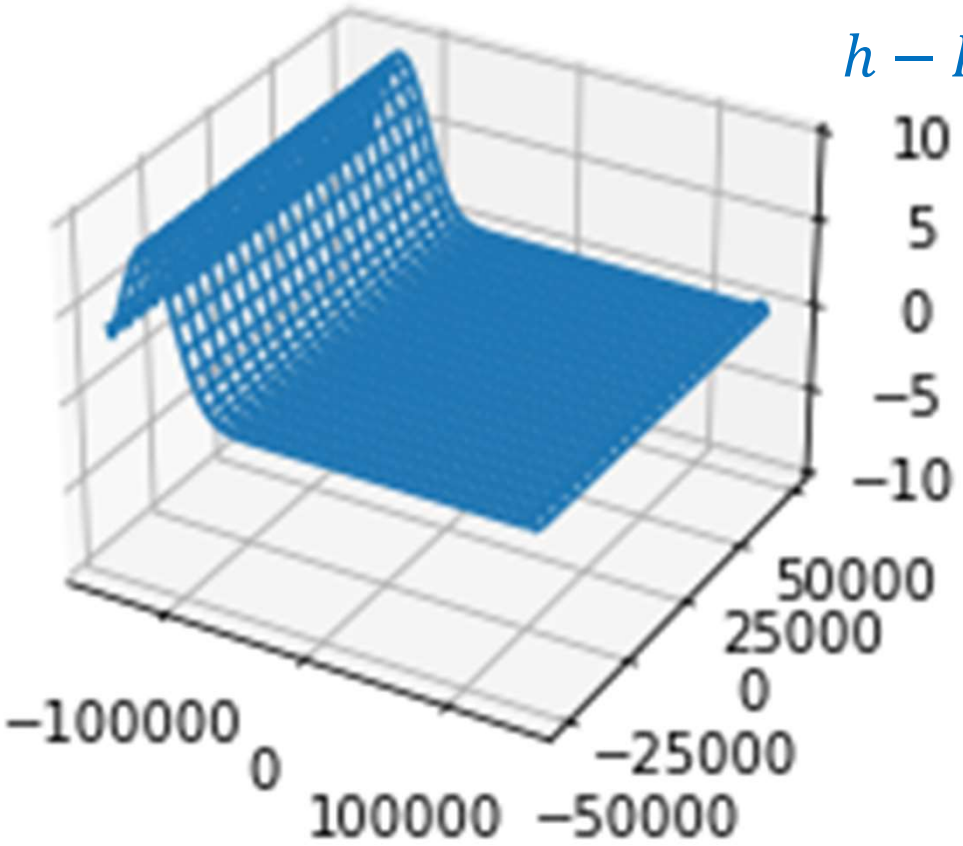


2D

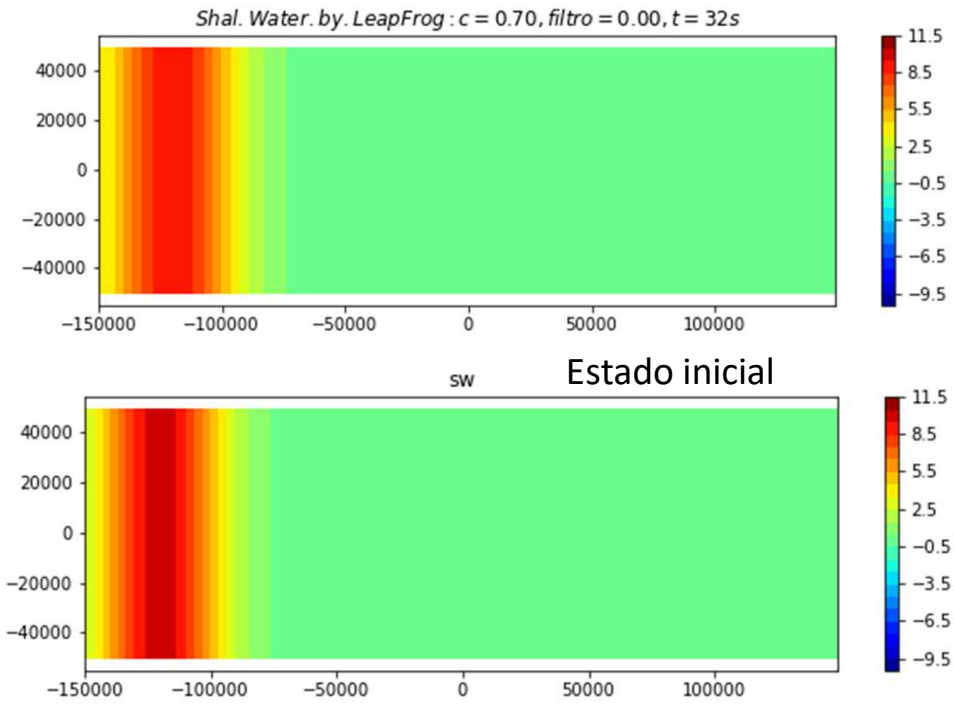
$h - H$



Condição fronteira fechada (refletora) em x_{min}, x_{max}

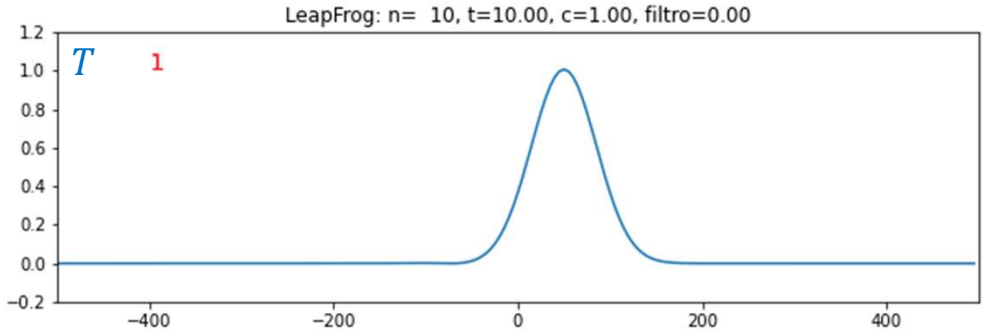


$em\ x = 0, L_x: u = 0, \frac{\partial v}{\partial x} = 0, \frac{\partial h}{\partial x} = 0$



Recapitulando

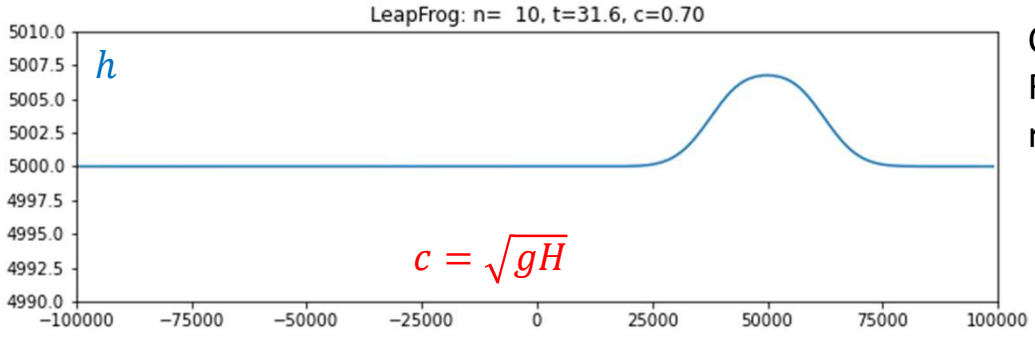
$$\frac{\partial T}{\partial t} = -u\frac{\partial T}{\partial x}$$



Cond.f Fronteira cíclicas

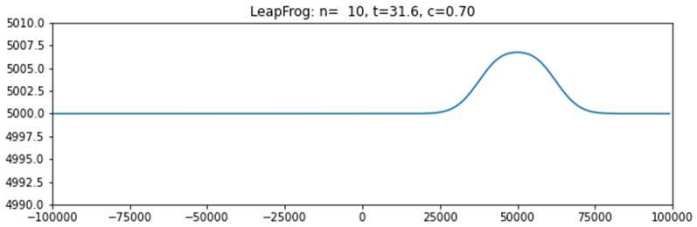
$$\frac{\partial u}{\partial t} = -\frac{\partial}{\partial x}(u^2) - g\frac{\partial h}{\partial x}$$

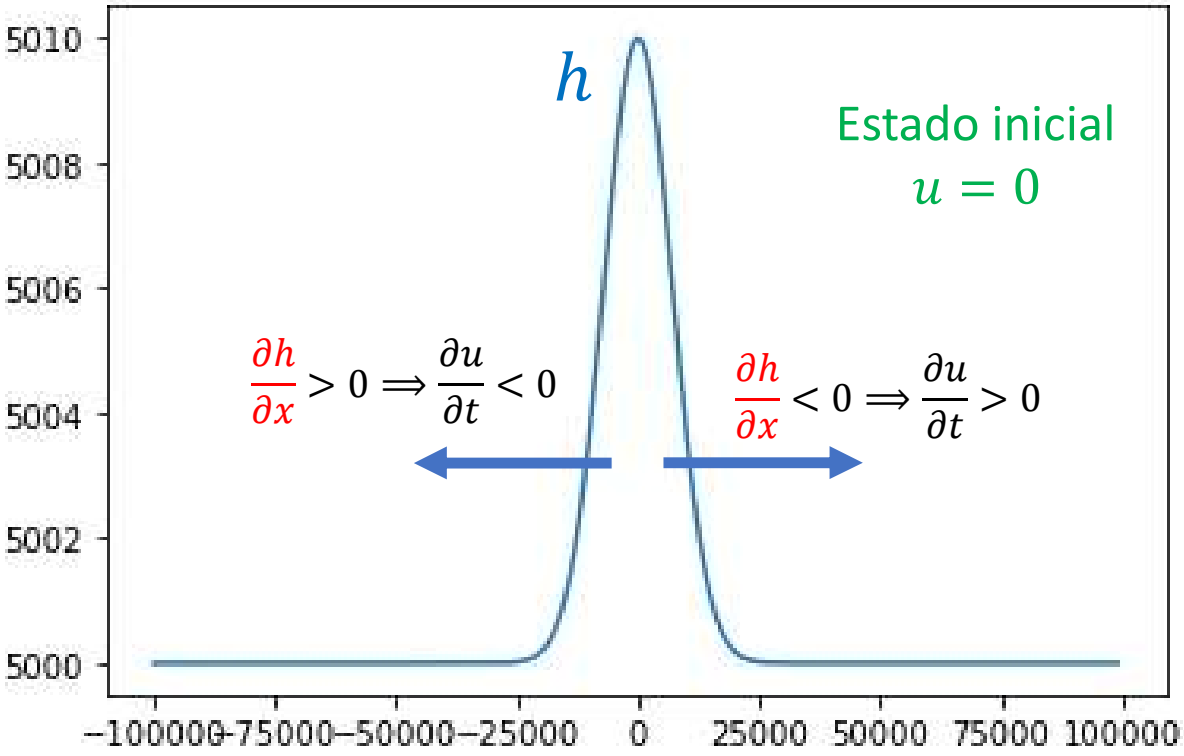
$$\frac{\partial h}{\partial t} = -\frac{\partial}{\partial x}\big((h-b)u\big)$$



Cond.f Fronteira reflexivas

Shallow-water 1D: $\frac{\partial u}{\partial t} = -\frac{\partial}{\partial x}\left(u^2\right) - g\frac{\partial h}{\partial x}$





1D shallow water, equação para U (leapfrog)

$$\frac{\partial u}{\partial t} = -\frac{\partial}{\partial x}(u^2) - g\frac{\partial h}{\partial x}$$

$$uu = u^2$$

$$\frac{(u_i^{n+1} - u_i^{n-1})}{2\Delta t} = -\frac{uu_{i+1}^n - uu_{i-1}^n}{2\Delta x} - g\frac{h_{i+1}^n - h_{i-1}^n}{2\Delta x}$$

$\mathtt{U} \equiv u_i^n$; $\mathtt{UM} \equiv u_i^{n-1}$; $\mathtt{UP} \equiv u_i^{n+1}$; $\mathtt{UU=U**2}$

```
UP[i]=UM[i]-dt/dx*(UU[i+1]-UU[i-1])-g*dt/dx*(H[i+1]-H[i-1])
```

o resto é semelhante à equação de adveção linear, mas atenção à condição fronteira.

Shallow-water 1D

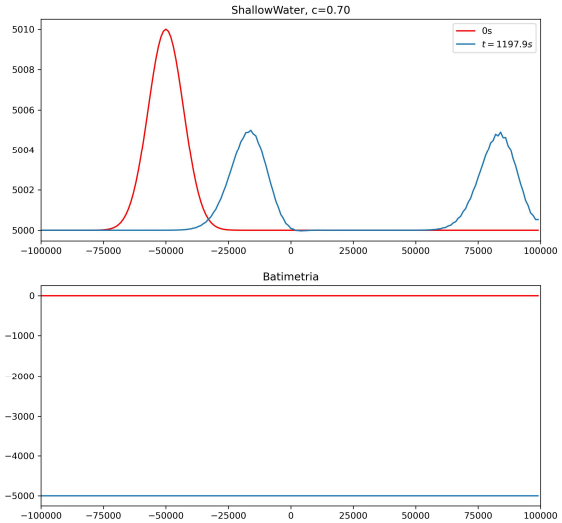
```
import numpy as np;import matplotlib.pyplot as plt
import os;import imageio
path=r'd:/'
g=9.81;h0=5000
nx=200;dx=1000;nt=2000;passo=20;snap=1200
closex=True
wavespeed=np.sqrt(g*h0)
courant=0.7; dt=courant*dx/wavespeed
Lx=nx/2*dx;x=np.arange(-Lx,Lx,dx)
h=np.ones(x.shape)*h0
hJUMP=10;xJUMP=-50000;LxJUMP=10000
h=h+hJUMP*np.exp(-((x-xJUMP)/LxJUMP)**2) #nivel inicial
b=np.zeros(h.shape)
b=0*np.exp(-((x-50000)/20000)**2) #batimetria
plt.figure(1,figsize=(10,10))
plt.subplot(2,1,1);plt.plot(x,h,color='red',label='0s');
plt.title('ShallowWater, c=%3.2f' % courant);plt.xlim(-Lx,Lx)
plt.subplot(2,1,2);plt.plot(x,b-h0);
plt.plot(x,np.ones(x.size)*0,color='red');plt.title('Batimetria')
plt.xlim(-Lx,Lx)
```

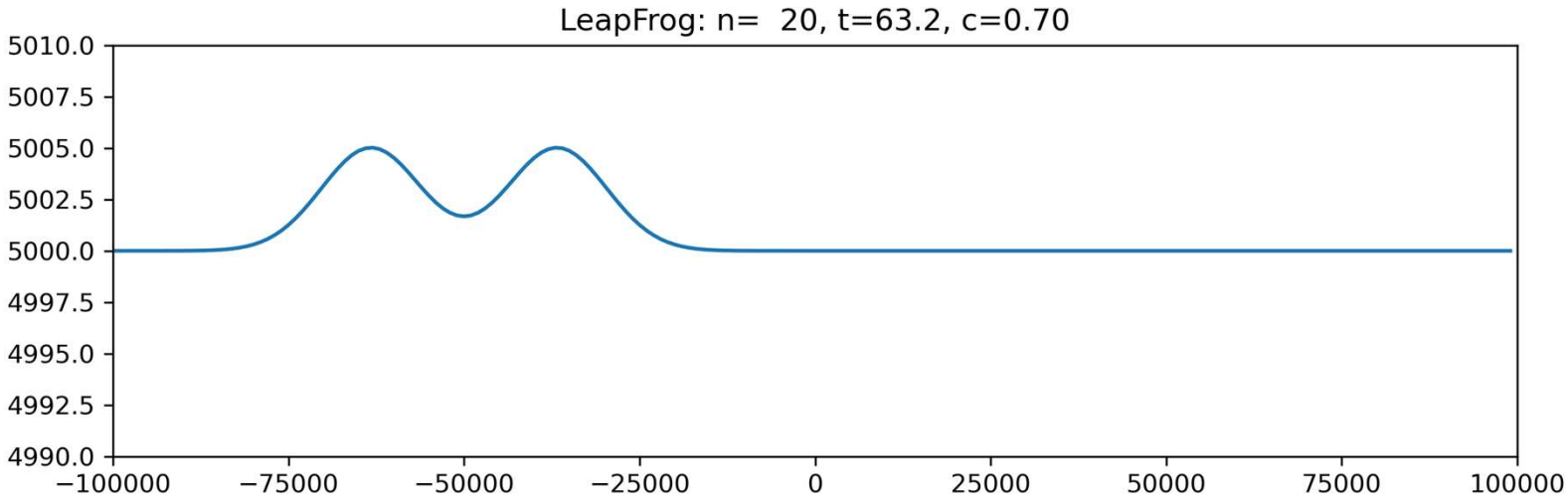
```
t=np.arange(0,dt*nt,dt)
frames=[]
u=np.zeros(h.shape)
uP=np.copy(u);uM=np.copy(u)
hP=np.copy(h);hM=np.copy(h)
dtdx2=0.5*dt/dx
dtdx=dt/dx
#1st step Euler
hmb=h-b;hmu=hmb*u;uu=u*u
for ix in range(nx):
    ixm=ix-1;ixp=ix+1
    if ix==0:  #cíclico
        ixm=nx-1
    elif ix==nx-1:
        ixp=0
    uP[ix]=u[ix]-dtdx2*(uu[ixp]-uu[ixm])-dtdx2*g*(h[ixp]-h[ixm])
    hP[ix]=h[ix]-dtdx2*(hmu[ixp]-hmu[ixm])
if closex: #fronteira fechada
    uP[0]=0
    uP[nx-1]=0
    hP[0]=hP[1]
    hP[nx-1]=hP[nx-1]
```

```
#LeapFrog
for it in range(2,nt):
    print(it,nt)
    hmb=h-b;hmu=hmb*u;uu=u*u
    for ix in range(nx):
        ixm=ix-1;ixp=ix+1
        if ix==0:  #cíclico
            ixm=nx-1
        elif ix==nx-1:
            ixp=0
        uP[ix]=uM[ix]-dtdx*(uu[ixp]-uu[ixm])-dtdx*g*(h[ixp]-h[ixm])
        hP[ix]=hM[ix]-dtdx*(hmu[ixp]-hmu[ixm])
    if closex: #fronteira fechada
        uP[0]=0
        uP[nx-1]=0
        hP[0]=hP[1]
        hP[nx-1]=hP[nx-2]
    uM=np.copy(u);u=np.copy(uP);hM=np.copy(h);h=np.copy(hP)
```

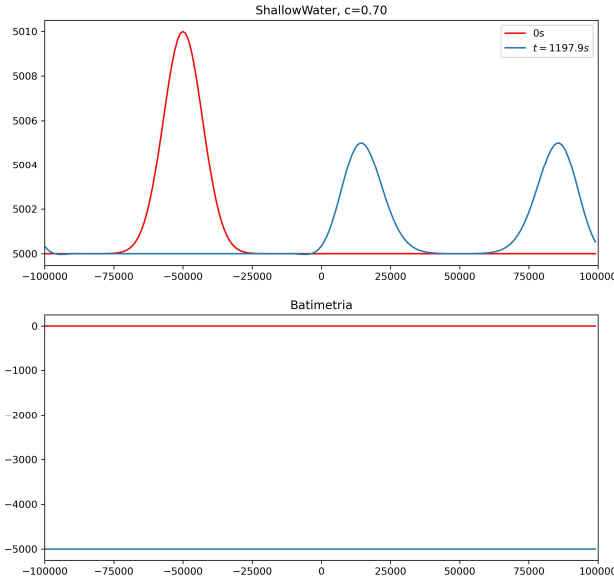
```
    if it%passo==0:
        plt.figure(2,figsize=(10,3))
        plt.plot(x,h)
        plt.plot(x,b,color='black')
        plt.axis([-Lx,Lx,h0-hJUMP,h0+hJUMP])
        plt.title('LeapFrog: n=%4i, t=%4.1f, c=%3.2f' %(it,dt*it,courant))
        fn=path+'mov'+str(it)+'.png'
        plt.savefig(fn)
        plt.clf() #close figure
        frames.append(fn)
    if snap>=it*dt and snap<=(it+1)*dt:
        plt.figure(1)
        plt.subplot(2,1,1);plt.plot(x,h,label=r'$t=%4.1f s$' % (it*dt))
plt.figure(1);plt.legend()

images=[] #frames para filme
for frame in frames:
    images.append(imageio.imread(frame))
    os.remove(frame)
imageio.mimsave(path+'SHA1dLFR'+'.gif', images,duration=0.1)
```

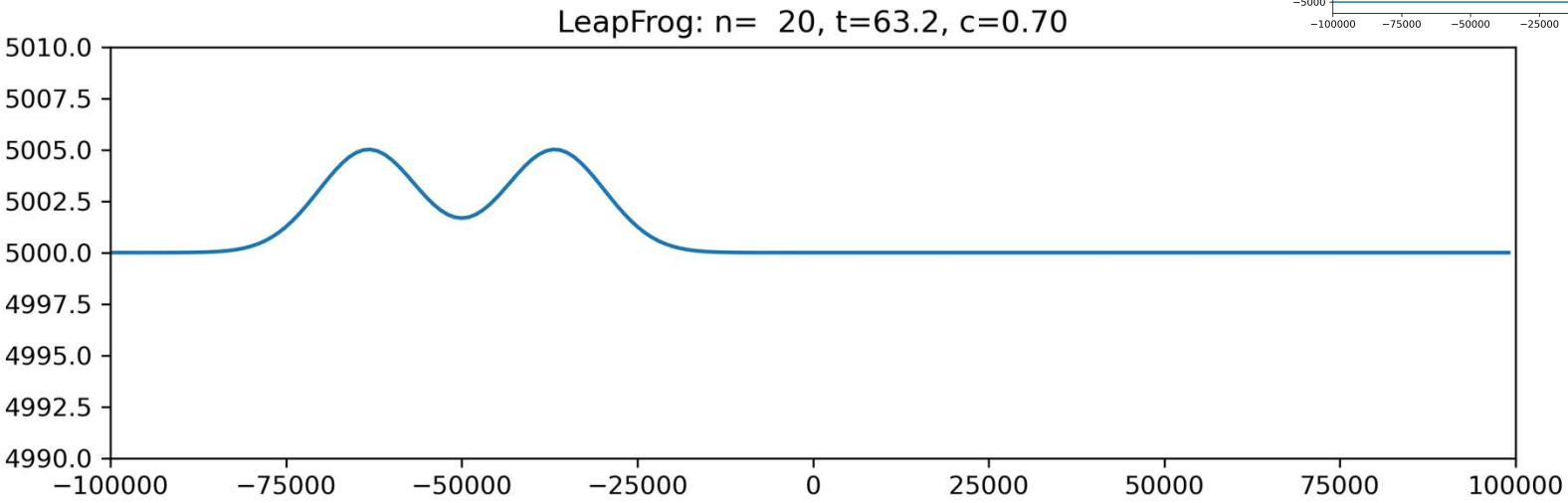
Batimetria FLAT, closeX=True





Batimetria FLAT, closeX=False





Fundo com montanha submarina

```
b=3000*np.exp(-((x-50000)/20000)**2)
```