ÉCOLE NATIONALE SUPÉRIEURE DE TECHNIQUES AVANCÉES

Finite Volume Methods for the 1D Shallow Water Equations

MF206 - Introduction to CFD 2020-2021

MOREL ROSA, Pedro

Palaiseau, April 5, 2021.

Contents

1 Introduction **3**

2 1D Shallow Water equations **3**
2.1 Introduction 3
2.2 Hyperbolicity 4

3 Dam-break **4**
3.1 Problem 4
3.2 Discretization and Methods 5
3.2.1 Rusanov's scheme 7
3.2.2 Roe's scheme 8
3.3 Results 8
3.4 Conservation of quantities 9

4 System with topography source term **10**
4.1 Problem 10
4.2 Method 11
4.3 Simulations and results 14
4.3.1 Perturbation in a lake 14
4.3.2 Oscillating Lake 16

5 References **17**

List of Figures

1 Shallow water flow configuration 3
2 Uniform finite volume grid. Two additional ghost cells are considered on the left and on the right of the interval limits. 5
3 Height for $N_C = 200$ 9
4 Velocity for $N_C = 200$ 9
5 Height for $N_C = 1000$ 9
6 Velocity for $N_C = 1000$ 9
7 Height for different times, with topography. $N_C = 200$ 14
8 Velocity for different times. $N_C = 200$ 15
9 Velocity for different times. $N_C = 200$ 15
10 Initial conditions for the oscillating lake problem 16
11 Height for the oscillating lake problem at $t_f = 19.8$ 16
12 Velocity for the oscillating lake problem at $t_f = 19.8$ 17

List of Source Codes

1 Initial treatment to obtain the flux F(U) from U. Used for both Rusanov and Roe's schemes 6
2 Flux computation for both Rusanov's and Roe's schemes 7
3 Rusanov's flux scheme. 7
4 Roe's flux scheme. 8
5 Values of the conserved discrete quantities. 10
6 Implementation of the Well-Balanced Hydrostatic Reconstruction method 13

1 Introduction

The present work aims to solve the Shallow Water equations numerically for two different scenarios. The first is a dam-break problem (section 3), in which the initial conditions will be analogous to the Riemann problem. The second will explore a problem in which the equations will be solved for boundary conditions that depend on a function $b(x)$ (section 4). In the latter, solutions to problems with dry regions will be presented as well.

The numerical framework is based on the finite volume method; particularly, solvers stemmed from Gudonov's scheme. These solvers are Rusanov's and Roe's scheme. Furthermore, the code was implemented and simulated in GNU Octave; and post-processed with Python.

2 1D Shallow Water equations

2.1 Introduction

The Shallow Water equations (or Saint-Venant equations) model a free surface incompressible inviscid flow under the assumption $\frac{L}{H} \ll 1$, where H is a characteristic flow height and L a characteristic horizontal length. The equations are obtained by depth-averaging the Navier–Stokes equations under the considered hypotheses. Here we consider a one-dimensional flow over a topography $b(x)$. The 1D shallow water equations can be written in the following conservative form:

$$\frac{\partial U}{\partial t} + \frac{\partial \mathcal{F}(U)}{\partial x} = \Psi \tag{1}$$

where:

$$U = \begin{bmatrix} h \\ hu \end{bmatrix}, \qquad \mathcal{F}(U) = \begin{bmatrix} hu \\ hu^2 + g\frac{h^2}{2} \end{bmatrix}, \qquad \Psi = \begin{bmatrix} 0 \\ -gh\frac{\partial b}{\partial x} \end{bmatrix} \tag{2}$$

Here $h(x,t)$ is the flow depth, $u(x,t)$ the flow velocity in the x direction, $b = b(x)$ denotes the bottom topography, and g is the gravity constant. Note that the free surface level is $h + b$.

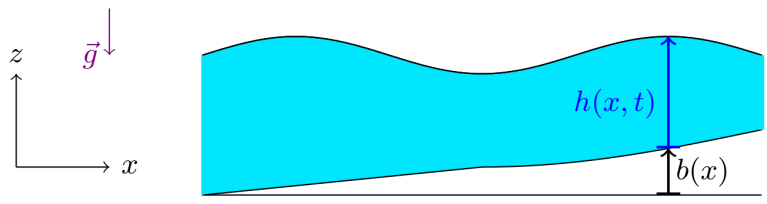


Figure 1: Shallow water flow configuration

2.2 Hyperbolicity

In order to solve the problem numerically using techniques for hyperbolic conservation laws, the problem must present itself as stricly hyperbolic. Thus, this mathematical characteristic must be proved.

Considering $b(x) = 0$, the quasi-linear form of system (1) is given by:

$$\frac{\partial U}{\partial t} + A(U)\frac{\partial U}{\partial x} = 0 \tag{3}$$

With the Jacobian matrix $A(U) = \frac{\partial \mathcal{F}(U)}{\partial U}$:

$$A(U) = \begin{bmatrix} \frac{\partial(v)}{\partial(h)} & \frac{\partial(v)}{\partial(v)} \\ \frac{\partial(\frac{v^2}{h}+g\frac{h^2}{2})}{\partial(h)} & \frac{\partial(\frac{v^2}{h}+g\frac{h^2}{2})}{\partial(v)} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ gh - u^2 & 2u \end{bmatrix} \tag{4}$$

And to find its eigenvalues and eigenvectors, one must solve $\det(A(U) - \lambda I) = 0$. This operation will have as a result :

$$(u - \lambda)^2 = gh \tag{5}$$

And this equation, in turn, will provide the eigenvalues $\lambda_1(U) = u - \sqrt{gh}$ and $\lambda_2(U) = u + \sqrt{gh}$. Additionally, we have the respective eigenvectors:

$$r_1(U) = \begin{bmatrix} 1 \\ u - \sqrt{gh} \end{bmatrix} \qquad r_2(U) = \begin{bmatrix} 1 \\ u + \sqrt{gh} \end{bmatrix} \tag{6}$$

Furthermore, we can show that the two characteristic fields are genuinely non-linear by proving that $\nabla\lambda_k(U) \cdot r_k(U) \neq 0, k = 1, 2\ \forall U$ in the set of states U with $h > 0$:

$$\nabla\lambda_1(U) \cdot r_1(U) = \frac{g(\sqrt{gh} - u)}{2\sqrt{gh}} \neq 0$$

$$\nabla\lambda_2(U) \cdot r_2(U) = \frac{g(\sqrt{gh} + u)}{2\sqrt{gh}} \neq 0$$

This information is relevant once we consider the Riemann problem. For a et of left and right states U_l and U_r, we will find constant states divided by discontinuity waves. Given the genuinely non-linear nature of these characteristic fields, the waves will consist of entities such as **rarefaction waves** and a **shock waves**, propagating with speeds between $\lambda_k(U_l)$ and $\lambda_k(U_r)$. Contrarily, if the characteristic field had a linearly degenerate nature, the waves would have exactly the speed $\lambda_k(U)$ and would consist of a **contact surface**.

3 Dam-break

3.1 Problem

A dam at $\bar{x}$ initially separates two uniform regions, which have different water heights, with the higher height in the left region. The initial velocity is assumed equal to zero

on both regions. At time $t > 0$ the dam is abruptly broken. This generates a shock wave (hydraulic jump) propagating to the right and a rarefaction wave propagating to the left.

These initial conditions correspond to a Riemann problem in terms of the variables $W = (h, u)$:

$$W(x,0) = \begin{cases} W_l = (3,0) & if x \leq \bar{x} \\ W_r = (1,0) & if x > \bar{x} \end{cases} \tag{7}$$

The problem is considered over the space interval $[4, 4]$ and the dam location is $\bar{x} = 0$.The gravity constant is $g = 1$. We will consider free flow (transmissive) boundary conditions both on the left and on the right of the considered interval.

3.2 Discretization and Methods

We consider a uniform discretization of the physical space interval $[4, 4]$ with N_C cells. The ith cell, $i \in \mathbb{Z}$, is centered at x_i and covers the interval $[x_{i-1/2}, x_{i+1/2}]$. We denote with Δx the grid spacing $\Delta x = x_i + x_{i+1} = -x_{i-1/2} + x_{i+1/2}$. Moreover, two additional ghost cells are considered outside the physical domain, to the left and to the right of the interval limits, hence the total number of cells is $N_C + 4$. The ghost cells are used to impose boundary conditions. We numerically approximate the hyperbolic system of conservation laws with $b(x) = 0$ by using three-point finite volume conservative schemes of the form:

$$U_i^{n+1} = U_i^n - \frac{\Delta t}{\Delta x}[F(U_i^n, U_{i+1}^n) - F(U_{i-1}^n, U_i^n)] \tag{8}$$

where Δt is the time step and $n \in \mathbb{N}$ denotes the time level. The numerical solution U_i^n represents an approximation of the average of the true solution at t^n over the ith cell interval $[x_{i-1/2}, x_{i+1/2}]$.

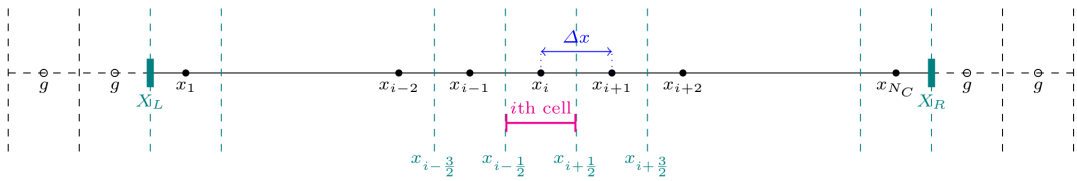


Figure 2: Uniform finite volume grid. Two additional ghost cells are considered on the left and on the right of the interval limits.

The last thing needed is a scheme to compute the numerical flux seen in equation (8). In the present work, two alternatives will be tested and explored: Rusanov's scheme and Roe's scheme.

Both schemes will be implemented as functions that have the same initial framework. Initially, they call the components of the vector U and use the information to obtain the vector $\mathcal{F}(U)$ (Listing 1).

Finally, the functions `fluxswRSn_templ` and `fluxswRoe_templ` can be called to effectively compute the flux (Listing 2).

```
function ff =fluxswRSn_templ(v,vp)

% Numerical flux F_{i+1/2} = F(u_i,u_{i+1})
% v = u_i, vp = u_{i+1}
% gravity constant
grav = 1.;
%initialization
flul = zeros(1,2);   % f(u_i)      % SWE flux evaluated at u_i
flur = zeros(1,2);   % f(u_{i+1})  % SWE flux evaluated at u_{i+1}

ff = zeros(1,2);     % numerical flux (output)

% set f(u_i)
rl = v(1);   % water height
rul = v(2);  % momentum

if(rl>0)
 ul = rul/rl;   % velocity
else
 ul=0;  % set zero velocity if dry state
end

flul(1) = rul
flul(2) = ul * rul + grav * (rl^2)/2

% set f(u_{i+1})
rr = vp(1);   % water height
rur = vp(2);  % momentum

if(rr>0)
 ur = rur/rr;   % velocity
else
 ur=0;  % set zero velocity if dry state
end

flur(1) = rur
flur(2) = ur * rur + grav * (rr^2)/2


if ((rl ==0)&&(rr==0))  % flux is zero if both left and right states are dry
return
end
```

Listing 1: Initial treatment to obtain the flux F(U) from U. Used for both Rusanov and Roe's schemes

```
Ncp2 = Nc+2;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for  istep=1:MaxStep, % starts time integration loop
     w = qv;  % store old qv

% update solution qv
   switch method

     case 'Rusanov'
     for i=3:Ncp2
     fluip = fluxswRSn_templ(w(i,:),w(i+1,:));
     flui = fluxswRSn_templ(w(i-1,:),w(i,:));
     qv(i,:) = w(i,:) - dt/h*(fluip-flui);
     end


     case 'Roe'
     for i=3:Ncp2
     fluip = fluxswRoe_templ(w(i,:),w(i+1,:));
     flui = fluxswRoe_templ(w(i-1,:),w(i,:));
     qv(i,:) = w(i,:) - dt/h*(fluip-flui);
     end

    end
```

Listing 2: Flux computation for both Rusanov's and Roe's schemes

3.2.1 Rusanov's scheme

The numerical flux for Rusanov's scheme is ruled by a regulation term $S_{i+1/2}$, based on the local maximum absolute value of characteristic speeds λ_i.

$$S_{i+1/2} = max(|u_{i+1} + \sqrt{gh_{i+1}}|, |u_{i+1} - \sqrt{gh_{i+1}}|, |u_i + \sqrt{gh_i}|, |u_i - \sqrt{gh_i}|) \quad (9)$$

And finally, this term is used to compute the flux:

$$F(U_i, U_{i+1}) = \frac{1}{2}[\mathcal{F}(U_i) + \mathcal{F}(U_{i+1}) - S_{i+1/2}(U_{i+1} - U_i)] \quad (10)$$

```
% Rusanov numerical flux

% amax = Local maximum absolute value of characteristic speeds
amax = max(max(abs(ur + sqrt(grav*rr)), abs(ur - sqrt(grav*rr))), -->
--> max(abs(ul + sqrt(grav*rl)), abs(ul - sqrt(grav*rl))))

% Finally, we compute :
ff = 0.5*((flul +flur) - amax*(vp-v));
```

Listing 3: Rusanov's flux scheme.

```
% Initialization of eigenvalues, eigenvectors and alpha coefficients :
lambdav = zeros(2); % Roe eigenvalues
Rmatv = zeros(2, 2); % Roe eigenvectors
alphav = zeros(2);  % Roe alpha coefficients, alpha = R^{-1}\Deltaq

delta = vp - v

% Roe averages :
hroe = 0.5*(rl+rr);
uroe = (sqrt(rl)*rul/rl + sqrt(rr)*rur/rr)/(sqrt(rr) + sqrt(rl));

% Roe eigenvalues :
lambdav(1)= uroe - sqrt(grav * hroe);
lambdav(2)= uroe + sqrt(grav * hroe);

% Roe eigenvectors :
Rmatv(1,1) = 1.;
Rmatv(1,2) = uroe - grav * sqrt(hroe);

Rmatv(2,1) = 1.;
Rmatv(2,2) = uroe + grav * sqrt(hroe);

% Roe alpha coefficients :
alphav(1)= 1/(2*sqrt( grav * hroe)) * ((uroe + sqrt(grav * hroe)) * delta(1) - delta(2));
alphav(2)= 1/(2*sqrt( grav * hroe)) * (-(uroe - sqrt(grav * hroe)) * delta(1) + delta(2));

%---------------------------------------------------------------------
% Roe's numerical flux

ff = (0.5*(flul+flur) - 0.5 * (abs(lambdav(1)) * alphav(1) * Rmatv(1,:)  + -->
--> + abs(lambdav(2)) * alphav(2) * Rmatv(2,:)));
```

Listing 4: Roe's flux scheme.

3.2.2 Roe's scheme

Contrarily, the numerical flux obtained from Roe's scheme is:

$$F^{Roe}(U_i, U_{i+1}) = \frac{1}{2}(\mathcal{F}(U_i) + \mathcal{F}(U_{i+1})) - \frac{1}{2}\sum_{k=1}^{2}(|\hat{\lambda_k}|\alpha_k \hat{r}_k)_{i+1/2} \tag{11}$$

Above, $\hat{\lambda}_k$ and $\hat{r}_k$, are the eigenvalues and the eigenvectors of the Roe matrix, and α_k is the coefficient of the projection of the interfaces i and $i+1$.

The present work does not aims to explain the Roe's scheme in its integrity. Therefore, we just content with the fact that the implementation of the scheme needs to take into account: Roe's eigenvalues, eigenvectors, coefficients, average height and average speed. The full implementation can be seen in the listing (4).

3.3 Results

Now that both Rusanov's and Roe's flux schemes were implemented, we can compare the results until a final time of $t_f = 1.2$, with a fixed time step Δt, with ratio $\frac{\Delta t}{\Delta x} = 0.4$.

By taking $N_C = 200$ and $N_C = 1000$, we can analyse how both methods get closer

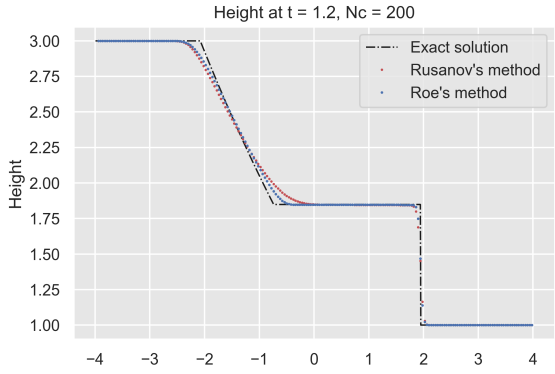


Figure 3: Height for $N_C = 200$

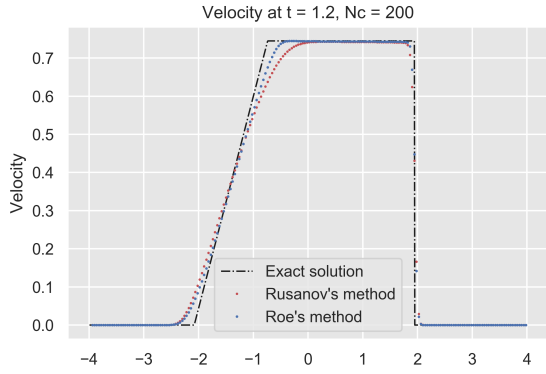


Figure 4: Velocity for $N_C = 200$

to the exact solution in the more refined scenario. It is equally important to remark the advantages of Roe's scheme when compared to Rusanov's scheme.

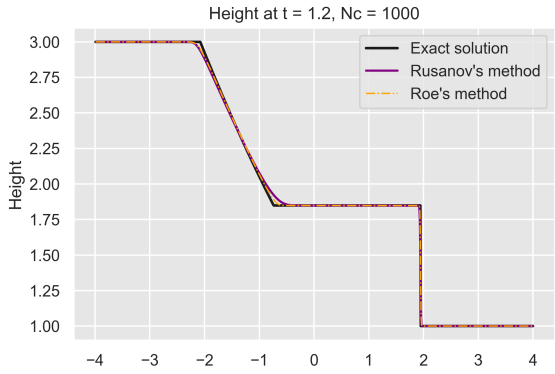


Figure 5: Height for $N_C = 1000$

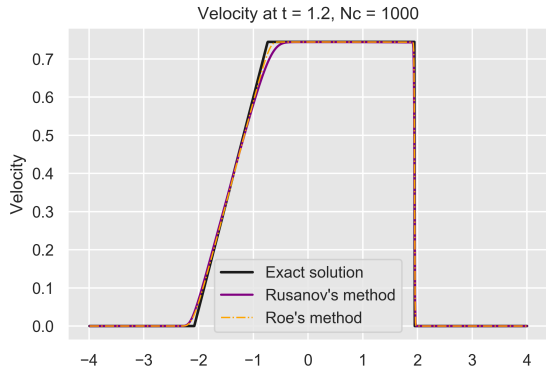


Figure 6: Velocity for $N_C = 1000$

This difference between both method's can be clearly noted in the simulations for $N_C = 200$, particularly in regions with high gradients.

3.4 Conservation of quantities

One last commentary of this problem is the importance of checking the conservation of quantities h and hu at the end of the finite volume scheme. To this aim, we consider the vectorial integral form of the system of conservation laws (1) with no source term Ψ over the considered space interval $[x_1, x_2] = [-4, 4]$ and the time interval $[0, t_f]$:

$$\int_{x_1}^{x_2} U(x, t_f)dx = \int_{x_1}^{x_2} U(x, 0)dx + \int_0^{t_f} \mathcal{F}(U(x_1, t))dt - \int_0^{t_f} \mathcal{F}(U(x_2, t))dt \qquad (12)$$

And by using $U = h$ and $\mathcal{F}(U) = hu$, we can already neglect the two last integrals, because the velocity is zero at $x_1 = -4$ and $x_2 = 4$ for the considered time period. Therefore, only the first integral needs to be developped.

$$\int_{-4}^{4} h(x, t_f)dx = \int_0^{-4} h(x, 0)dx + \int_4^0 h(x, 0)dx$$

And given the initial conditions (7), we find:

$$\int_{-4}^{4} h(x,t_f)dx = \int_{0}^{-4} 3dx + \int_{4}^{0} 1dx = 12 + 4 = 16$$

And we find, using $U = hu$ and $\mathcal{F}(U) = hu^2 + \frac{gh^2}{2}$:

$$\int_{-4}^{4} hu(x,t_f)dx = \int_{0}^{1.2} \frac{gh^2}{2}(x_1,t)dt - \int_{0}^{1.2} \frac{gh^2}{2}(x_2,t)dt$$

Once again, we used the initial conditions (7); in particular, the fact that the velocity is zero for $t = 0$. Therefore, having only the last two integrals of equation (12), we find:

$$\int_{-4}^{4} hu(x,t_f)dx = \int_{0}^{1.2} \frac{g3^2}{2}(x_1,t)dt - \int_{0}^{1.2} \frac{g1^2}{2}(x_2,t)dt = \left[\frac{8x}{2}\right]_{0}^{1.2} = 4.8$$

Now that we know the exact analytical values for the conserved quantities, we must check if this conservation is valid throughout the discrete values of the numerical solution ($\Delta x \sum_{i=1}^{N_C} h_i$ and $(\Delta x \sum_{i=1}^{N_C} (hu)_i)$. This can be done with the simple computation seen in Listing 5.

```
1 int_h = sum(qv(:,1))*h
2 int_hu = sum(qv(:,2))*h
```

Listing 5: Values of the conserved discrete quantities.

And we find precisely the values 16 and 4.8. Therefore, the finite volume method does conserve the quantities and agrees with the exact analytical values

4 System with topography source term

4.1 Problem

The main goal in this section is to study the influence of variable bottom topography $b(x) \neq 0$ in the Shallow Water equations.

Firstly, we can show the stationary state by taking specifically the Shallow Water equation with $U = hu$:

$$\frac{\partial(hu)}{\partial t} + \frac{\partial(hu^2)}{\partial x} + \frac{1}{2}\frac{\partial(gh^2)}{\partial x} = -gh\frac{\partial b}{\partial x}$$

And applying stationary conditions ($u = 0$, $\frac{\partial(\cdot)}{\partial t} = 0$):

$$\frac{1}{2}\frac{\partial(gh^2)}{\partial x} = -gh\frac{\partial b}{\partial x}$$

With the chain rule for multivariables, this leads to:

$$gh(x)\frac{dh(x)}{dx} = -gh(x)\frac{db(x)}{dx}$$

Which results in:

$$\frac{d}{dx}(h(x) + b(x)) = 0$$

And finally:

$$h + b = constant \tag{13}$$

Which expresses the fact that the water level is horizontal (so-called "lake at rest" condition).

4.2 Method

One difficulty in the numerical solution of the shallow water equations with bottom topography is the need to guarantee the preservation of the condition (13) at the discrete level, that is we need to ensure that if at time t_n we have $u_i = 0$, $h_i + b_i = constant\ \forall i$, then these conditions are maintained at time level $t_n + 1$. Numerical schemes able to guarantee this property are called well-balanced. Here we use the **Well-Balanced Hydrostatic Reconstruction scheme**. This Finite Volume scheme has the form:

$$U_i^{n+1} = U_i^n - \frac{\Delta t}{\Delta x}[\tilde{F}_l(U_i^n, U_{i+1}^n, b_i, b_{i+1}) - \tilde{F}_r(U_{i-1}^n, U_i^n, b_{i-1}, b_i)] \tag{14}$$

where b_i are discrete values of the bottom topography b, and the numerical fluxes are defined as:

$$\tilde{F}_l(U_L, U_R, b_L, b_R) = F(U_L^*, U_R^*) + \begin{bmatrix} 0 \\ \frac{1}{2}g(h_L^2 - h^{*2}_L) \end{bmatrix} \tag{15}$$

$$\tilde{F}_r(U_L, U_R, b_L, b_R) = F(U_L^*, U_R^*) + \begin{bmatrix} 0 \\ \frac{1}{2}g(h_R^2 - h^{*2}_R) \end{bmatrix} \tag{16}$$

And finally, we define the reconstructed states U_L^* and U_R^*:

$$\begin{cases} U_L^* = (h_L^*, h_L^* u_L) \\ U_R^* = (h_R^*, h_R^* u_R) \end{cases} \tag{17}$$

Where:

$$\begin{cases} h_L^* = (h_L + b_L - b^*)_+ \\ h_R^* = (h_R + b_R - b^*)_+ \end{cases} \tag{18}$$

With $b^* = max(b_L, b_R)$.

Finally, we are interested in proving that the scheme defined above preserves the stationary states with zero velocity $u = 0$. To start, we assume $u_L = u_R = 0$ and $h_L + b_L = h_R + b_R$. We can easily show that:

$$h_L^* = (h_L + b_L - b^*)_+ = (h_R + b_R - b^*)_+ = h_R^*$$

(19)

Therefore, $h_L^* = h_R^*$. Given this, and the fact that $U^* = (h^*, h^*u)$, we have:

$$U_L^* = (h_L^*, 0) = (h_R^*, 0) = U_R^* \tag{20}$$

And finally, by comparing equation (14) with equation (8), we find that $\mathcal{F}(U_L) = \tilde{F}_l$ and $\mathcal{F}(U_R) = \tilde{F}_r$

In order to take into account the Well-Balanced Hydrostatic Reconstruction method, the reconstructed variables are defined before the usual algorithm seen in Listing 2. The implementation was done in such a way that only Rusanov's flux is called.

```
Ncp2 = Nc+2;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for  istep=1:MaxStep, % starts time integration loop
     w = qv;  % store old qv

% update solution qv


     for i=3:Ncp2

     if(w(i-1,1)<0)
     ur = rur/rr;   % velocity
     else
     ur=0;  % set zero velocity if dry state
     end

     hl1 = max(w(i-1,1) + bot(i-1) - max(bot(i-1), bot(i)), 0);
     hr1 = max(w(i,1) + bot(i) - max(bot(i-1), bot(i)), 0);

     if(w(i-1,1)>0)
     Ul1 = [hl1 hl1*w(i-1,2)/w(i-1,1)];   % velocity
     else
     Ul1 = [hl1 0];  % set zero velocity if dry state
     end

     if(w(i,1)>0)
     Ur1 = [hr1 hr1*w(i,2)/w(i,1)]   % velocity
     else
     Ur1 = [hr1 0]  % set zero velocity if dry state
     end

     hl2 = max(w(i,1) + bot(i) - max(bot(i), bot(i+1)), 0);
     hr2 = max(w(i+1,1) + bot(i+1) - max(bot(i), bot(i+1)), 0);

     if(w(i,1)>0)
     Ul2 = [hl2 hl2*w(i,2)/w(i,1)];   % velocity
     else
     Ul2 = [hl2 0];  % set zero velocity if dry state
     end

     if(w(i+1,1)>0)
     Ur2 = [hr2 hr2*w(i+1,2)/w(i+1,1)]   % velocity
     else
     Ur2 = [hr2 0]  % set zero velocity if dry state
     end

     flui = fluxswRSn_templ(Ul2, Ur2) + [0 0.5*(grav*(w(i,1)**2 - hl2**2))]
     fluip = fluxswRSn_templ(Ul1, Ur1) + [0 0.5*(grav*(w(i,1)**2 - hr1**2))]

     qv(i,:) = w(i,:) - dt/h*(flui-fluip);

    end
```

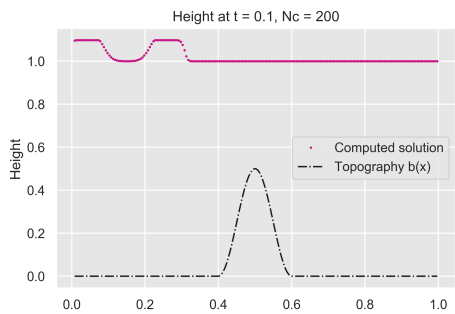
Listing 6: Implementation of the Well-Balanced Hydrostatic Reconstruction method

4.3 Simulations and results

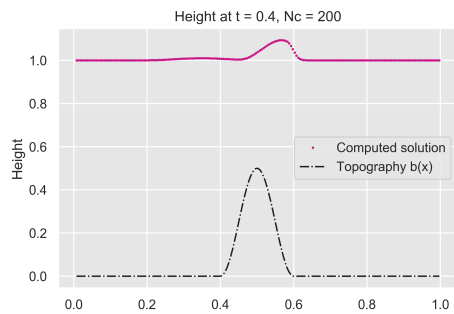
In order to test the implementation seen in Listing 6, we will simulate two different problems with bottom topography $b(x) \neq 0$. In the first one, the initial and boundary conditions are presented in such a way that the bottom topography will always be wet ($h(x) > b(x) \forall x$. In the second problem we will explore exactly the contrary situation, in which we can have dry regions.

4.3.1 Perturbation in a lake

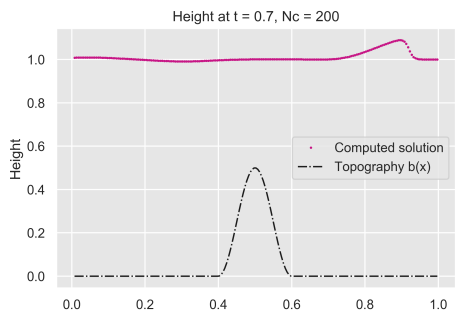
Applying the scheme over the interval $[0, 1]$ with bottom topography defined by $b(x) = 0.25(cos(\pi(x0.5)/0.1) + 1)$ if $| - 0.5 + x| < 0.1$, and $b(x) = 0$ otherwise. The initial condition consists of an initial perturbation of a steady condition defined by $h + b = 1$, $u = 0$. A perturbation of the water height $h(x, 0) = 1 + \epsilon$ is taken for $0.1 < x < 0.2$, with $\epsilon = 0.2$. The flow is initially at rest, $u(x, 0) = 0$. Using free flow (transmissive) boundary conditions on both sides of the spatial interval, we compute the solution with $N_C = 200$ grid cells and Δt with $\frac{\Delta t}{\Delta x} = 0.8$.



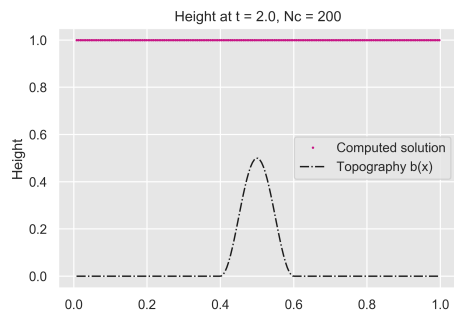
(a) Height at t = 0.1 with topography b(x)



(b) Height at t = 0.4 with topography b(x)



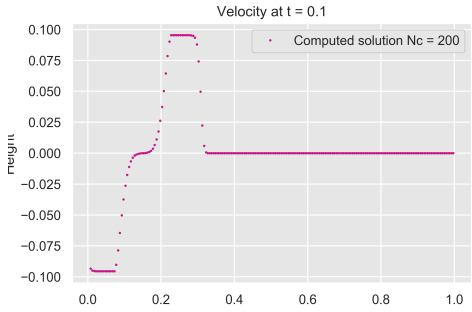
(c) Height at t = 0.7 with topography b(x)



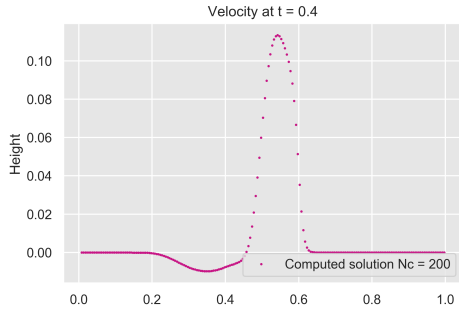
(d) Height at t = 2.0 with topography b(x)

Figure 7: Height for different times, with topography. $N_C = 200$

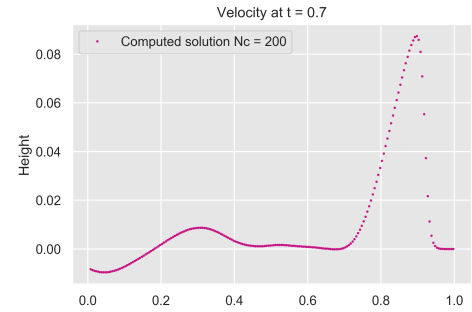
In figure (7) we can see the dynamics of the solution as time goes by. Firstly, in image (7(a)), we have the perturbation generating two waves; one going to the right and the other one to the left. In (7(b)) we have a much clearer image of the wave going right; and we can see the dispersion of its shape at (7(c)). Finally, the wave exits the field being studied and we have approximately steady solution.



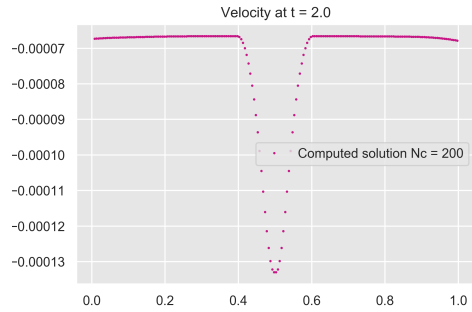
(a) Velocity at t = 0.1



(b) Velocity at t = 0.4



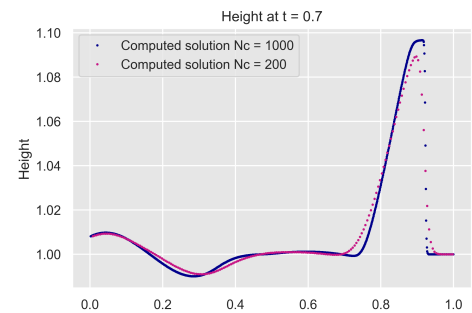
(c) Velocity at t = 0.7



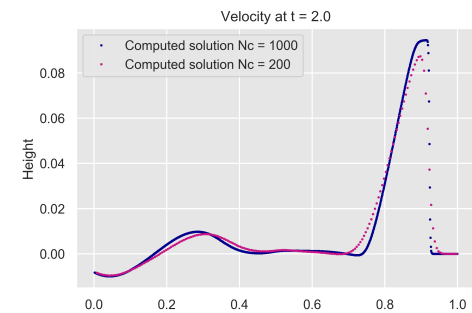
(d) Velocity at t = 2.0

Figure 8: Velocity for different times. $N_C = 200$

In figure (8) the dynamics of the simulation can be seen throught the perspective of the wave velocity. An interesting commentary that can be made about this case is the fact that we can see a much different scale in figure (8d)). In it, we can see check that at time 2.0, when the wave already exited the field of simulation, we still have some residual effect - despite being negligible. The results are in agreement with what is expected.



(a) Height comparison for different mesh refinements



(b) Velocity comparison for different mesh refinements

Figure 9: Velocity for different times. $N_C = 200$

We can also analyse, in figure (9), how a more refined mesh can change the results

observed. By taking $t = 0.7$, we can compare a simulation done with $N_C = 1000$ and another done with $N_C = 200$. And we can clearly note that the more refined solution obtains much clearer results for higher gradients and discontinuities when compared to the less refined solution.

4.3.2 Oscillating Lake

As mentioned before, now we will use the scheme with the Rusanov flux to solve the oscillating lake problem with dry regions. In this test we consider a lake over a topography defined by $b(x) = 0.5(10.5cos(\pi(x0.5)/0.5) + 1))$ over the interval $[0, 1]$. The lake is initially at rest but with a sinusoidal perturbation of the free surface $\eta(x) = 0.04sin((x0.5)/.25)$ centered at $(0, 0.4)$.

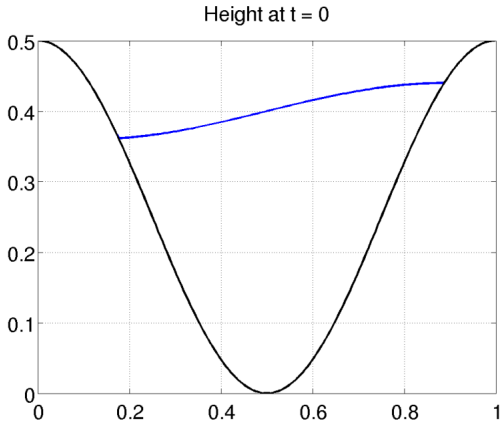


Figure 10: Initial conditions for the oscillating lake problem

The simulation reached a final time $t_f = 19.8$, at which the flow height reaches approximately a maximum level at the left shore. Additionally, the mesh consists of $N_C = 200$ cells, with limiter ratio of $\frac{\Delta t}{\Delta x} = 0.8$ for the time step.

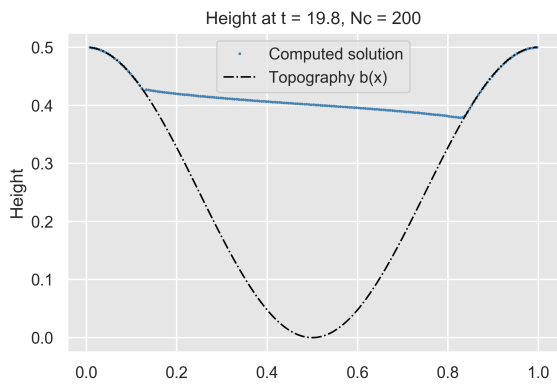


Figure 11: Height for the oscillating lake problem at $t_f = 19.8$

Note that in this problem we have dry regions with $h = 0$ (vacuum regions). For most applications of the shallow water equations it is important to be able to simulate

problems with dry regions. The Rusanov method can handle data with vacuum since it is able to preserve the non-negativity of the flow height at the discrete level. In contrast, Roe's scheme may fail in problems involving dry states, since it might compute negative values of the flow height.

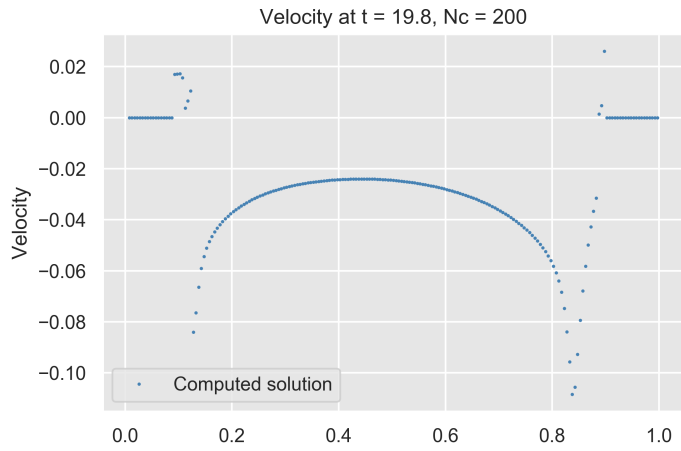


Figure 12: Velocity for the oscillating lake problem at $t_f = 19.8$

By looking figure 11 and 12, we can confirm that the regions with more velocity will certainly be those at the near ends for the oscilating lake. Finally, we can state that the results are in agreement with what is expected.

5 References

1. E. Audusse, F. Bouchut, M.-O. Bristeau, R. Klein, and B. Perthame. A fast and stable well-balanced scheme with hydrostatic reconstruction for shallow water flows. *SIAM J. Sci. Comput.*, 25(6):2050–2065, 2004.

2. F. Bouchut. *Nonlinear stability of finite volume methods for hyperbolic conservation laws, and well-balanced schemes for sources.* Birkhäuser-Verlag, 2004.

3. R. J. LeVeque. Balancing source terms and flux gradients in high-resolution Godunov methods: The quasi-steady wave-propagation algorithm. *Comput. Phys.*, 146:346–365, 1998