Institutionen för datavetenskap
Umeå universitet
Eddie Wadbro, Martin Berggren

17 september 2019
Teknisk beräkningsvetenskap II

A sluice gate opens

In this lab, we will simulate the hydrodynamic effects (the nonlinear wave motion) of opening a sluice gate in a simplified model of the flow dynamics. Section 1 derives the governing equation, the shallow-water equations in one space dimension, and section 2 describes the particular case that we will study. Section 3 contains the studies you should perform, and section 4 a few implementation hints.

1 Modeling shallow channel flow

Shallow-water flow is, like the name suggests, a fluid-flow case with a small vertical length scale. More precisely, the horizontal length scales are assumed to be much larger than the vertical length scale. When using the appropriate approximations for this case, the water surface level h as a function of space and time enters as an unknown function that is solved for together with the horizontal velocity. The shallow-water equations can be derived by depth-averaging a more general set of governing equations. However, we will derive them from scratch under the following assumptions.

Assumptions.
1. The fluid has the constant density ρ (in kg/m^3).
2. The channel is narrow and shallow with a planar bottom.
3. Inertial forces dominate over all other forces. In particular, the fluid's friction forces against the bottom and sides of the channel are ignored, as is the air pressure on the surface.

We assume that the bottom of the channel is located at $z = 0$ and that its width is δ (figure 1). The narrowness of the channel (Assumption 2) together with the dominance of inertial forces (Assumption 3) means that the x-direction dynamics dominates, and we therefore assume that the water level is constant in the y direction. Thus, the area of each vertical cross section is given by

$$A(x,t) = h(x,t)\delta, \tag{1}$$

where $h(x,t)$ is the surface level. The shallowness (Assumption 2) means that we also ignore the vertical dynamics, which implies that we may assume that the velocity field is

$$\boldsymbol{v}(x,t) = (v(x,t), 0, 0). \tag{2}$$

That is, we ignore the vertical velocity component and assume that the horizontal velocity component v do not depend on the z and y coordinates. Alternatively, and more accurately, we can say that $\boldsymbol{v}(x,t)$ represents the actual, more complicated horizontal velocity averaged over a vertical cross section with area $A(x,t)$. A second consequence of ignoring the vertical dynamics is that we may assume a static pressure distribution in the vertical direction:

$$p(x,z,t) = \rho g(h(x,t) - z) \qquad \text{for } z \in [0, h(x,t)]; \tag{3}$$

that is, the pressure at vertical level z equals the pressure given by the column of water above z. (The pressure in expression (3) is the *atmospheric overpressure*; that is, $p = 0$ at the water surface.)

We will derive equations for the water surface level h and the horizontal velocity v as a function of x and t. The equations will follow from the laws of mass conservation and momentum balance together with assumption 3.

Now consider an arbitrary interval (a,b) on the x axis (Figure 1). The mass of fluid between vertical planes at the end points of the interval is

$$m(t) = \int_a^b \rho A(x,t)\, dx. \tag{4}$$

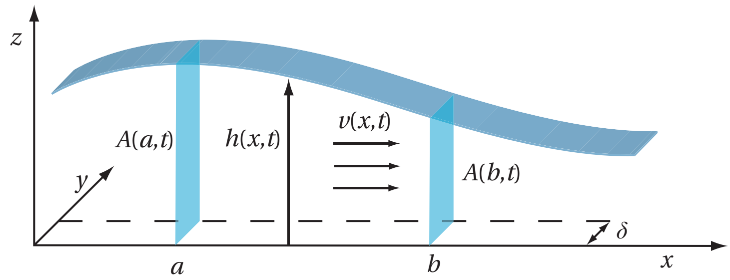

FIGURE 1. Shallow-water flow in a narrow channel of width δ. The water surface level $h(x,t)$ and the horizontal velocity $v(x,t)$ are the unknown functions.

The mass flux (in kg/s) in the positive x direction through a vertical cross section with area $A(x,t)$ at point x and time t is

$$f^{(m)}(x,t) = \rho v(x,t) A(x,t). \tag{5}$$

The law of mass conservation requires that the increase of mass in the region equals the influx through the cross section at $x = a$ minus the outflux through the cross section at $x = b$,

$$\frac{dm}{dt} = f^{(m)}(a,t) - f^{(m)}(b,t) = -\int_a^b (f^{(m)})_x \, dx, \tag{6}$$

which by expressions (4) and (5) yields

$$\int_a^b (\rho A)_t \, dx = -\int_a^b (\rho v A)_x \, dx, \tag{7}$$

and furthermore, by substituting expression (1), that

$$\int_a^b (\rho h \delta)_t \, dx = -\int_a^b (\rho v h \delta)_x \, dx. \tag{8}$$

Dividing expression (8) with $\delta\rho$, using that δ and ρ are constant and that the interval (a,b) is arbitrary, we obtain the equation of mass conservation under the current assumptions,

$$h_t + (vh)_x = 0. \tag{9}$$

The momentum (sv. *rörelsemängd*) of a particle is its mass times its velocity. The momentum density of a fluid, that is, the fluid momentum per unit volume is its density times its velocity, $\rho \boldsymbol{v}$, so the total momentum of the fluid in a control volume V is

$$\boldsymbol{r}_V(t) = \int_V \rho \boldsymbol{v} \, dV. \tag{10}$$

As opposed to mass, momentum is a vector quantity with direction. However, due to assumption 2, we can in this case describe the momentum of the water as a signed (that is, it can be positive as well as negative) scalar r, positive in the x-axis direction. The momentum of the region between vertical cross sections at $x = a$ and $x = b$ is

$$r(t) = \int_a^b \rho v(x,t) A(x,t) \, dx. \tag{11}$$

Momentum is a conserved quantity, like mass, if there are no forces acting on the system. (This is Newton's first law). Otherwise, for a fixed amount of material subject to forces from the exterior, the change in momentum equals the applied forces. (For solid bodies, this is Newton's second law). If the amount of material is not fixed, as in our case for the material in the region in (a, b), the *momentum balance law*, a generalization of Newton's second law to the case of fluids, says that the change of momentum equals the net influx of momentum plus the applied forces,

$$\frac{dr}{dt} = f^{(r)}(a,t) - f^{(r)}(b,t) + f^{(p)}(a,t) - f^{(p)}(b,t) = -\int_a^b (f^{(r)} + f^{(p)})_x \, dx, \tag{12}$$

where $f^{(r)}(x,t)$ is the flux of momentum in the positive x-direction through a vertical plane at x and at time t, and $f^{(p)}(x,t)$ the force applied in the positive x-direction on a vertical plane at x and at time t.

The flux of momentum in the positive x-direction through a vertical cross section at point x with area $A(x,t)$ is

$$f^{(r)}(x,t) = \rho v(x,t)^2 A(x,t), \tag{13}$$

that is, the product of ρv, the momentum/m^3, and vA, the volume flux in m^3/s. Integrating the pressure (3) over $A(x,t)$ yields the force in the positive x-direction on a vertical cross section at point x,

$$\begin{aligned} f^{(p)}(x,t) &= \int_0^h \int_0^\delta p(x,z,t) \, dy \, dz = \int_0^h \int_0^\delta \rho g \, (h-z) \; dy \, dz. \\ &= -\delta \rho g \frac{(h-z)^2}{2} \Bigg|_{z=0}^{h} = -\delta \rho g \frac{h^2}{2}. \end{aligned} \tag{14}$$

Substituting the momentum expression (11), and the flux expressions (13), (14) into the momentum balance law (12) yields

$$\int_a^b (\rho v A)_t \, dx = -\int_a^b \left(\rho v^2 A + \delta \rho g h^2/2\right)_x \, dx, \tag{15}$$

which after substituting expression (1) yields that

$$\int_a^b (\rho v h \delta)_t \, dx = -\int_a^b \left(\rho v^2 h \delta + \delta \rho g h^2/2\right)_x \, dx, \tag{16}$$

Again dividing by the constant $\delta\rho$ and using that interval (a, b) is arbitrary, we obtain the equation of momentum balance under the current assumptions,

$$(vh)_t + (v^2 h)_x + g(h^2/2)_x = 0. \tag{17}$$

Defining $q = vh$ ($q\delta$ is the volume flux in m^3/s), we may write equations (9) and (17) as the following system of conservation laws, the one-dimensional *shallow-water equations*,

$$\frac{\partial}{\partial t}\begin{pmatrix} h \\ q \end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} q \\ q^2/h + gh^2/2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{18}$$

Defining $u = (h, q)^T$ and

$$f = \begin{pmatrix} q \\ q^2/h + gh^2/2 \end{pmatrix}, \tag{19}$$

the shallow-water equations in vector form assumes the standard form

$$u_t + f_x = 0. \tag{20}$$

for conservation laws in one space dimension. Note, however that equation (20), as opposed to the equations studied in the first lab, is a *system* of conservation laws.

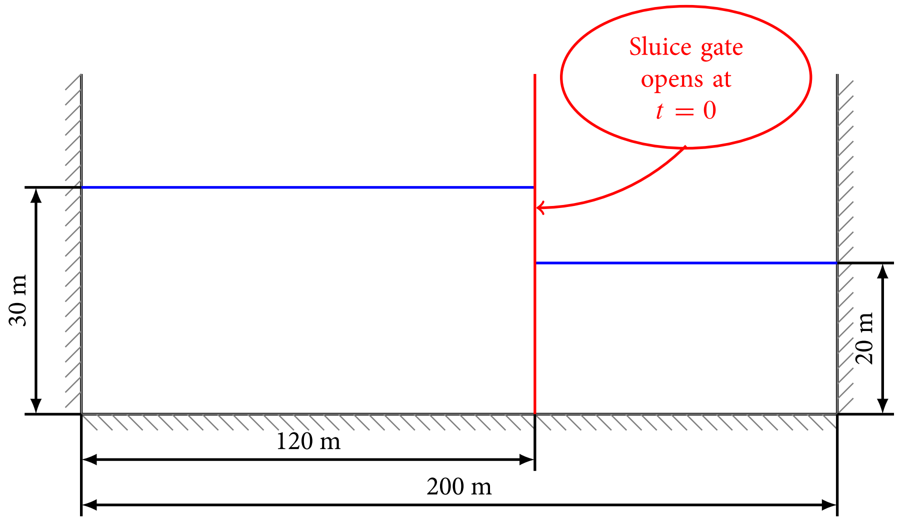

FIGURE 2. Sketch of the sluice setup before the gate opens.

2 Opening of a sluice gate

Now we are ready to look at the particular features of the problem we want to study. The sluice portion (the lock) is depicted in Figure 2. Here we are considering a lock consisting of two chambers of length 120 and 80 meters, respectively. Initially, the water height is 30 m in the large lock and 20 m in the smaller lock. At time $t = 0$, the gate between the two locks opens and water is free to flow between the two chambers. We will simply model the opening of the lock by suddenly removing the complete gate. In reality, the gate opening is typically more controlled in that the gate is only partially opened at the bottom, but a simulation of this case would require a more complete model. We assume that the water is completely still at the instant the gate is removed, that is velocity $v \equiv 0$ and thus $q \equiv 0$ at $t = 0$. Hence, the initial condition for system (18) is

$$h(0,x) = \begin{cases} 30 & \text{if} \quad 0 \le x \le 120, \\ 20 & \text{if } 120 < x \le 200, \end{cases} \qquad \text{and} \qquad q(0,x) = 0 \text{ for } x \in [0,200]. \tag{21}$$

To close the system, we need boundary conditions at $x = 0$ and $x = 200$. There is no flow of water across the chamber walls, so $u = 0$ on these boundaries and hence

$$q(t,0) = q(t,200) = 0 \quad \forall t \ge 0. \tag{22}$$

Thus, the complete initial–boundary-value problem will be

$$\begin{aligned} \frac{\partial}{\partial t}\begin{pmatrix} h \\ q \end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} q \\ q^2/h + gh^2/2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, & \qquad \text{for } x \in (0,200),\, t > 0, \\ q(t,0) = q(t,200) = 0 \quad t \ge 0. & \\ h(0,x) = \begin{cases} 30 & 0 \le x \le 120, \\ 20 & 120 < x \le 200, \end{cases} & \\ q(0,x) = 0 \qquad x \in [0,200]. & \end{aligned} \tag{23}$$

3 Computer Exercises

1. Numerically solve initial–boundary-value problem (23) on a uniform grid using Lax–Friedrichs' method. Solve the problem for a sufficiently large time so that the effect of the walls at the sides

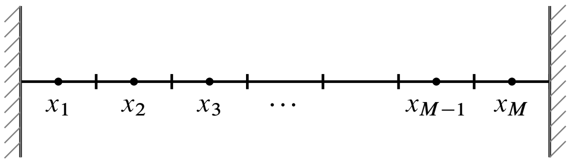

FIGURE 3. The positioning of the cell centers $x_i = i\,\Delta x/2$, $i = 1, \ldots, M$, and the cell interfaces $x_{i-1/2} = i\,\Delta x$, $i = 0, \ldots, M$.

of the sluice is clear. Observe the *two* kinds of nonlinear wave phenomena that appear in the solution!

2. Estimate the speed of the shock wave traveling from the gate ($x = 120$) towards the right wall ($x = 200$). Here, we are only interested in the initial speed of the shock. It is thus sufficient to solve the problem until the shock wave has moved a certain distance, but not yet reached the right wall.
3. Vary the time and space discretization. Does such changes affect the estimated shock speed?
4. Vary the initial condition, that is, change the water height in the two basins. Try for example, 20 m (left basin) and 10 m (right basin) as well as 30 m (left basin) and 15 m (right basin). Does this change affect the estimated shock speed?

4 Implementation instructions and hints

4.1 General instructions and hints for all exercises

- Let the number of cells M and the time–space step ratio $\lambda = \Delta t/\Delta x$ be input parameters.
- For this problem, we need to set $\lambda \lesssim 0.05$ to avoid stability problems.
- Remember that, as illustrated in Figure 3, x_i, $i = 1, 2, \ldots, M$ are the cell *centers* located at $\Delta x/2, 3\Delta x/2, \ldots, (2M-1)\Delta x/2$ and that u_i^n, $i = 1, 2, \ldots, M$ are corresponding cell averages. The cell *interfaces* are located at $x_{i+1/2} = i\ \Delta x$, $i = 0, 1, \ldots, M$.
- Plot the evolution of the water height with time, which is easily accomplished by placing the plotting routine inside the time stepping loop. To be able to see anything, you will likely need to insert a pause after each plot (see `help pause` for documentation on how to do this). For the plotting you can simply plot the water height against the cell-center coordinates, as discussed in the last lab.
- The family of finite volume schemes that we consider in this course have the general form

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{\Delta x}\Big(F_{i+1/2}^n - F_{i-1/2}^n\Big), \tag{24}$$

  where u_i^n denotes the cell average in cell i at time t_n. Note that in this case, we are considering the *system* of conservation laws (18), so all four terms in equation (24) are vectors with two components!
- Recall that the numerical flux function for the Lax–Friedrichs method is:

$$F_{i+1/2}^n = \frac{1}{2}\left(f(u_{i+1}^n) + f(u_i^n)\right) - \frac{\Delta x}{2\Delta t}\left(u_{i+1}^n - u_i^n\right). \tag{25}$$

  By inserting flux (25) in the general scheme (24), we obtain

$$u_i^{n+1} = \frac{1}{2}\left(u_{i+1}^n + u_{i-1}^n\right) - \frac{\Delta t}{2\Delta x}\left(f(u_{i+1}^n) - f(u_{i-1}^n)\right). \tag{26}$$

- The boundary conditions at $x = 0$ and $x = 200$ can be imposed using *ghost cells*, as in lab 1. An alternative to ghost cells is to use fluxes different than (25) for the boundary fluxes $F^n_{1/2}$ and $F^n_{M+1/2}$ in order to enforce the boundary conditions, as will now be described. Consider update equation (24) for $i = 1$,

$$u_1^{n+1} = u_1^n - \frac{\Delta t}{\Delta x}\left(F^n_{3/2} - F^n_{1/2}\right) \tag{27}$$

  Here, we use the normal Lax–Friedrichs flux (25) to compute the right flux $F^n_{3/2}$. For the left flux $F^n_{1/2}$, instead of the Lax–Friedrichs flux, we consider the exact flux (19) in which we impose $q = 0$ and insert the value $h = h_1^n$, that is

$$F^n_{1/2} = \begin{pmatrix} 0 \\ 0 + g(h_1^n)^2/2 \end{pmatrix}. \tag{28}$$

  Update (27) then becomes

$$\begin{aligned}
u_1^{n+1} &= u_1^n - \frac{\Delta t}{\Delta x}\left(F^n_{3/2} - F^n_{1/2}\right) \\
&= u_1^n - \frac{\Delta t}{\Delta x}\left(\frac{1}{2}\left(f(u_2^n) + f(u_1^n)\right) - \frac{\Delta x}{2\Delta t}\left(u_2^n - u_1^n\right) - \begin{bmatrix} 0 \\ g(h_1^n)^2/2 \end{bmatrix}\right) \\
&= \frac{1}{2}\left(u_2^n + u_1^n\right) - \frac{\Delta t}{\Delta x}\left(\frac{1}{2} f(u_2^n) + \frac{1}{2}\begin{bmatrix} q_1^n \\ (q_1^n)^2/h_1^n + g(h_1^n)^2/2 \end{bmatrix} - \begin{bmatrix} 0 \\ g(h_1^n)^2/2 \end{bmatrix}\right) \\
&= \frac{1}{2}\left(u_2^n + u_1^n\right) - \frac{\Delta t}{2\Delta x}\left(f(u_2^n) - \begin{bmatrix} -q_1^n \\ -(q_1^n)^2/h_1^n + g(h_1^n)^2/2 \end{bmatrix}\right) \\
&= \frac{1}{2}\left(u_2^n + u_1^n\right) - \frac{\Delta t}{2\Delta x}\left(f(u_2^n) - f(-u_1^n)\right),
\end{aligned}$$

  where expression (19) has been used in the third equality. By a similar argument at $x = 200$, we obtain the following update for the averages in the last cell

$$u_M^{n+1} = \frac{1}{2}\left(u_M^n + u_{M-1}^n\right) - \frac{\Delta t}{2\Delta x}\left(f(-u_M^n) - f(u_{M-1}^n)\right).$$

- Define the initial water height in the right basin as well as the difference in height between the two basins to be variables in your code or input parameters to your function.
- Let `base` denote the initial water height in the right basin and `increase` denote the height difference. Given the vector `xm` of the cell center positions, the initial condition for the water height can in Matlab be written as

```
base+increase*(xm<=120);
```

- With `base` and `increase` as above, the shock position can be found by

```
delta_x*find(h>base+increase/4,1,'last');
```

  where `h` is the vector with the cell averaged water heights and `delta_x` is the space step. Read the documentation (`doc find`) for more information of the very useful command `find`!
- A small warning. Matlab statements like `xm<=120` and `h>base+increase/4` above return not floating point arrays but *logical arrays* of the same size as `xm` and `h`, respectively. These arrays indicate for each component whether the condition is true or false. (See `help logical`.) If these arrays are directly used as if they were floating point numbers, unexpected surprises can appear.

However when a logical array is multiplied with a floating-point number (such as above, when the logical array `xm<=120` is multiplied with the floating-point number `increase`), the product is converted to a floating point number. Thus, if you want to construct an array of floating point numbers with the help of logical operations, as done above, remember to multiply the logical array with a floating point number (could for instance be `1.`).