INFO 0939: Project 2 – Due on 20 December 2019

Updated 15/11/2019

Intermediate deadline 1 (MPI explicit code): 19 November 2019
Intermediate deadline 2 (MPI implicit code): 10 December 2019

Goals of the Project

- Solve partial differential equations.
- Experiment stability of explicit and implicit time integration schemes.
- Combine MPI and OpenMP to take advantage of both distributed and shared memory parallelisation strategies.

Statement

Wave propagation at the surface of the ocean can be modelled using the so-called "shallow water" equations, which are derived by depth-integrating the Navier-Stokes equations. This assumes that the horizontal velocity field is constant throughout the depth of the ocean, allowing to elimitate the vertical velocity from the equations. The linear shallow water model for which we neglect the Coriolis force writes:

$$\frac{\partial \eta}{\partial t} = -\nabla \cdot (h\mathbf{u}) \tag{1}$$

$$\frac{\partial \mathbf{u}}{\partial t} = -g\nabla\eta - \gamma\mathbf{u} \tag{2}$$

where the unknown fields $\eta(t, x, y)$ and $\mathbf{u}(t, x, y) = (u(t, x, y), v(t, x, y), 0)$ are respectively the free-surface elevation and the depth-averaged velocity (see Figure 1), and where

- $h(x, y)$ is the depth at rest;
- $g = 9.81\text{m/s}^2$ is the gravitational acceleration;
- γ is the dissipation coefficient.

The first equation is the continuity equation and the second is derived from the momentum equation.

The finite difference method

The domain of study is a rectangle $[0, a] \times [0, b]$ whose dimensions and bathymetric map (map of $h(x, y)$) are given by an input file. Sample input files are provided in the directory

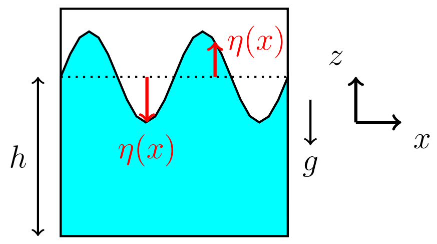


Figure 1: Schematic representation of the problem.

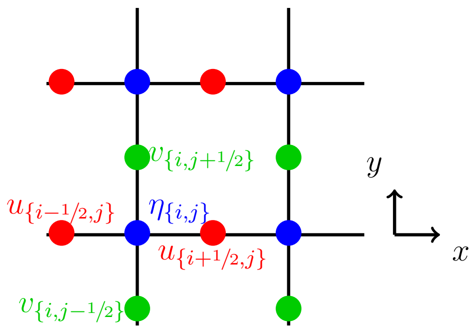


Figure 2: Spatial grid.

`/home/ulg/ace/aroyer/INFO0939/hw2` on the NIC4 cluster. You can also design your own.

Discretization in space is performed as depicted in Figure 2: Equation (1) is discretized at the cross-points of the cells (integer coordinates) while Equation (2) is discretized on the edges (half integer coordinates), both using finite differences. The time domain is discretized using an explicit or an implicit Euler scheme (see sections below), where η is computed at time $n\Delta t$ while $\mathbf{u}$ is computed at time $(n + 1/2)\Delta t$ with $n = 1, 2, \cdots, T_{\text{max}}$. The domain must be filled with "whole" cells, meaning that there are no η unknowns on the boundary.

Explicit Euler scheme

The simplest Euler method is called explicit because the unknown values at the next time step depend explicitly of the known values at the current time step. Applying this method to Equations 1 and 2, the explicit finite difference scheme you should implement is the following:

$$\begin{aligned}
\frac{\eta_{ij}^{n+1} - \eta_{ij}^{n}}{\Delta t} &= -\frac{h_{i+1/2,j}\, u_{i+1/2,j}^{n+1/2} - h_{i-1/2,j}\, u_{i-1/2,j}^{n+1/2}}{\Delta x} - \frac{h_{i,j+1/2}\, v_{i,j+1/2}^{n+1/2} - h_{i,j-1/2}\, v_{i,j-1/2}^{n+1/2}}{\Delta y}\\
\frac{u_{i+1/2,j}^{n+1+1/2} - u_{i+1/2,j}^{n+1/2}}{\Delta t} &= -g\,\frac{\eta_{i+1,j}^{n} - \eta_{i,j}^{n}}{\Delta x} - \gamma u_{i+1/2,j}^{n+1/2}\\
\frac{v_{i,j+1/2}^{n+1+1/2} - v_{i,j+1/2}^{n+1/2}}{\Delta t} &= -g\,\frac{\eta_{i,j+1}^{n} - \eta_{i,j}^{n}}{\Delta y} - \gamma v_{i,j+1/2}^{n+1/2}.
\end{aligned}$$

The initial condition is a zero free-surface elevation and zero velocity:

$$\eta_{i,j}^{0} = u_{I,j}^{1/2} = v_{i,J}^{1/2} = 0 \qquad \text{with,} \quad \begin{array}{l} i \in \left\{0, 1, \cdots, \frac{a}{\Delta x} - 1\right\},\\ j \in \left\{0, 1, \cdots, \frac{b}{\Delta y} - 1\right\},\\ I \in \left\{-1/2, 1/2, \cdots, \frac{a}{\Delta x} - 1/2\right\},\\ J \in \left\{-1/2, 1/2, \cdots, \frac{b}{\Delta y} - 1/2\right\}. \end{array} \tag{3}$$

The boundary conditions are the following: on the left, bottom and right sides of the domain, the normal velocity is fixed to zero (impermeable boundaries):

$$\begin{cases} u^{n+1/2}_{-1/2,j} = 0 \\ u^{n+1/2}_{\frac{a}{\Delta x}-1/2,j} = 0 \\ v^{n+1/2}_{i,-1/2} = 0 \end{cases} \qquad \text{with, } n \in \{1, 2, \cdots, T_{\max}/\Delta t\}\,. \tag{4}$$

On the top of the domain the v component of the velocity is imposed either as

$$v^{n+1/2}_{i,\frac{b}{\Delta y}-1/2} = A \sin(2\pi f t) \qquad \forall i, \tag{5}$$

or as

$$v^{n+1/2}_{i,\frac{b}{\Delta y}-1/2} = A \sin(2\pi f t) e^{-\frac{t}{500}} \qquad \forall i, \tag{6}$$

where A and f denote respectively the amplitude and the frequency of the wave, and where $t = (n + 1/2)\Delta t$. Condition 5 models a simple, steady-state wave source of fixed amplitude, and can be used e.g. with the bathymetry file `reflection.dat`. Condition 6 simulates the source of the 2004 Indian Ocean Tsunami, and can be used with the `sriLanka.dat` bathymetry map, which models a small area of the Indian Ocean around Sri Lanka. In this case, the time that the tsunami takes to reach the bottom of the domain is about 1h45min.

Implicit Euler scheme

For implicit Euler, the unknown values at the next time step depend both on the known values at the current time step but also of unknown values at the next time step. This dependency implies that a linear system must be solved to compute values at time step. Applying this method on Equations 1 and 2, the implicit finite difference scheme you should implement is the following:

$$\frac{\eta^{n+1}_{ij} - \eta^{n}_{ij}}{\Delta t} = -\frac{h_{i+1/2,j}\, u^{n+1+1/2}_{i+1/2,j} - h_{i-1/2,j}\, u^{n+1+1/2}_{i-1/2,j}}{\Delta x} - \frac{h_{i,j+1/2}\, v^{n+1+1/2}_{i,j+1/2} - h_{i,j-1/2}\, v^{n+1+1/2}_{i,j-1/2}}{\Delta y}$$

$$\frac{u^{n+1+1/2}_{i+1/2,j} - u^{n+1/2}_{i+1/2,j}}{\Delta t} = -g\,\frac{\eta^{n+1}_{i+1,j} - \eta^{n+1}_{i,j}}{\Delta x} - \gamma u^{n+1+1/2}_{i+1/2,j}$$

$$\frac{v^{n+1+1/2}_{i,j+1/2} - v^{n+1/2}_{i,j+1/2}}{\Delta t} = -g\,\frac{\eta^{n+1}_{i,j+1} - \eta^{n+1}_{i,j}}{\Delta y} - \gamma v^{n+1+1/2}_{i,j+1/2}$$

with the same initial and boundary condition as Equations 3, 4, 5 and 6.

This numerical scheme leads to the solution of a sparse linear system of equations, whose unknowns are all the spatial components of η^{n+1}, $u^{n+1+1/2}$ and $v^{n+1+1/2}$, stored together in a vector. To store the sparse matrix (named A), an efficient method keeps in memory only the non-zero values. In this project, we suggest to store non-zero values using 3 arrays (m, n and v) of size N_z equal to the number of non-zero values in A, defined such as:

$$\forall k \in \{0, 1, \cdots, N_z\} \quad (A)_{m(k),n(k)} = v(k). \tag{7}$$

In order to solve the linear system of equations, we ask you to choose an iterative method (and to justify your choice). A good candidate could be the conjugate gradient method, which is briefly summarized in Algorithm 1.

Algorithm 1: Conjugate gradient method.

```
$\mathbf{x}_0 = \mathbf{0}$
$\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$
$\mathbf{p}_0 = \mathbf{r}_0$
i = 0
while $\frac{\|\mathbf{r}_i\|_2}{\|\mathbf{r}_0\|_2} \geq r_{threshold}$ do
    $\alpha = \frac{\mathbf{r}_i^T \mathbf{r}_i}{\mathbf{p}_i^T A \mathbf{p}_i},$
    $\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha \mathbf{p}_i$
    $\mathbf{r}_{i+1} = \mathbf{r}_i - \alpha \mathbf{A}\mathbf{p}_i$
    $\beta = \frac{\mathbf{r}_{i+1}^T \mathbf{r}_{i+1}}{\mathbf{r}_i^T \mathbf{r}_i}$
    $\mathbf{p}_{i+1} = \mathbf{r}_{i+1} + \beta \mathbf{p}_i$
    i = i + 1
end
return $\mathbf{x}_i$.
```

Instructions

By group of **two students** (this is mandatory) you are asked to implement a code that:

1. Models the shallow water problem using **double precision** representation;
2. Uses **both** the explicit and implicit scheme;
3. Solves the linear system (for the implicit scheme);
4. Uses both MPI and OpenMP;
5. Takes from the command line a parameter file, a bathymetric map file and a flag used to chose the numerical scheme (0 for the explicit one and 1 for the implicit one). A call to your program could thus look like `./prog param.txt map.dat 0` or `./prog param.txt map.dat 1`.

The parameter file is written in ASCII and has the following structure:

g(double)
γ(double)
Δx(double)
Δy(double)
Δt(double)
$T_{\max}$(double)
A(double)
f(double)
S(unsigned int)
s(unsigned int)
$r_{threshold}$(double)

Your program should interpret the given values with the type specified between the parentheses. The parameters g, γ, Δx, Δy, Δt, $T_{\max}$, A, f are defined above; S is the sampling rate at which the results should be saved to disk ($S = 0$ stands for never saving the results, $S = 1$ corresponds to saving all time steps, $S = 2$ corresponds to saving on time step every two, ...), s is a flag used to choose the type of excitation ($s = 0$ for the sinusoidal velocity and $s = 1$ for the source simulated the tsunami) and $r_{threshold}$ is the residual threshold of the iterative linear solver.

The bathymetric map file is a binary file that has the following structure:

a	b	X	Y
$h_{0,0}$	$h_{1,0}$	$\dots$	$h_{X-1,0}$
$h_{0,1}$	$h_{1,1}$	$\dots$	$h_{X-1,1}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$
$h_{0,Y-1}$	$h_{1,Y-1}$	$\dots$	$h_{X-1,Y-1}$

where a and b are the max coordinates of the domain ($[0,a]\times[0,b]$) in double precision, where X (resp. Y) is the number of sampling points in the x (resp. y) directions, such as $h_{0,0} = h(-{}^{\Delta x}/_2, -{}^{\Delta y}/_2)$ and $h_{X-1,Y-1} = h(a - {}^{\Delta x}/_2, b - {}^{\Delta y}/_2)$. Note that the discretization of the computational grid should not necessarily be the same as the discretization of depth in this bathymetry file. To get the depth at any point (x, y) of the computational domain, you should use bilinear interpolation: you should find a couple (k,l) such that $x_k \leq x < x_{k+1}$ and $y_l \leq y < y_{l+1}$ and then applied following bilinear interpolation between the four surrounding data:

$$\begin{aligned} h(x,y) = \big[&(x_{k+1}-x)(y_{l+1}-y)h_{k,l} + (x_{k+1}-x)(y-y_l)h_{k,l+1} \\ &+(x-x_k)(y_{l+1}-y)h_{k+1,l} + (x-x_k)(y-y_l)h_{k+1,l+1}\big]/(\delta x\delta y), \end{aligned} \quad (8)$$

where $\delta x = \frac{a}{X-1}$ and $\delta y = \frac{b}{Y-1}$ are the spatial steps of the bathymetric map file;

6. Saves the solutions using the following **binary** output file format (three files per time step, `eta.dat`, `u.dat` and `v.dat`) where N (resp. M) is the number of elements in the x (resp. y) directions, written as an unsigned integer, and the $\alpha_{i,j}$ are the free-surface elevation, the velocity along x or the velocity along y written as double precision numbers:

N	M		
$\alpha_{0,0}$	$\alpha_{1,0}$	$\dots$	$\alpha_{N-1,0}$
$\alpha_{0,1}$	$\alpha_{1,1}$	$\dots$	$\alpha_{N-1,1}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$
$\alpha_{0,M-1}$	$\alpha_{1,M-1}$	$\dots$	$\alpha_{N-1,M-1}$

Submit your C code on the Montefiore submission platform `https://submit.montefiore.ulg.ac.be/`. Note that no error have to be reported during the automatic tests performed on this platform. If your last submission generates error(s), a default grade of 0/20 will be attributed.

Write a report of maximum 20 pages where you:

1. Describe your implementation;

2. Study the stability of the explicit numerical scheme for various combinations of parameters;

3. Perform a thorough scalability analysis (weak and strong) for both the explicit and implicit version of the code;

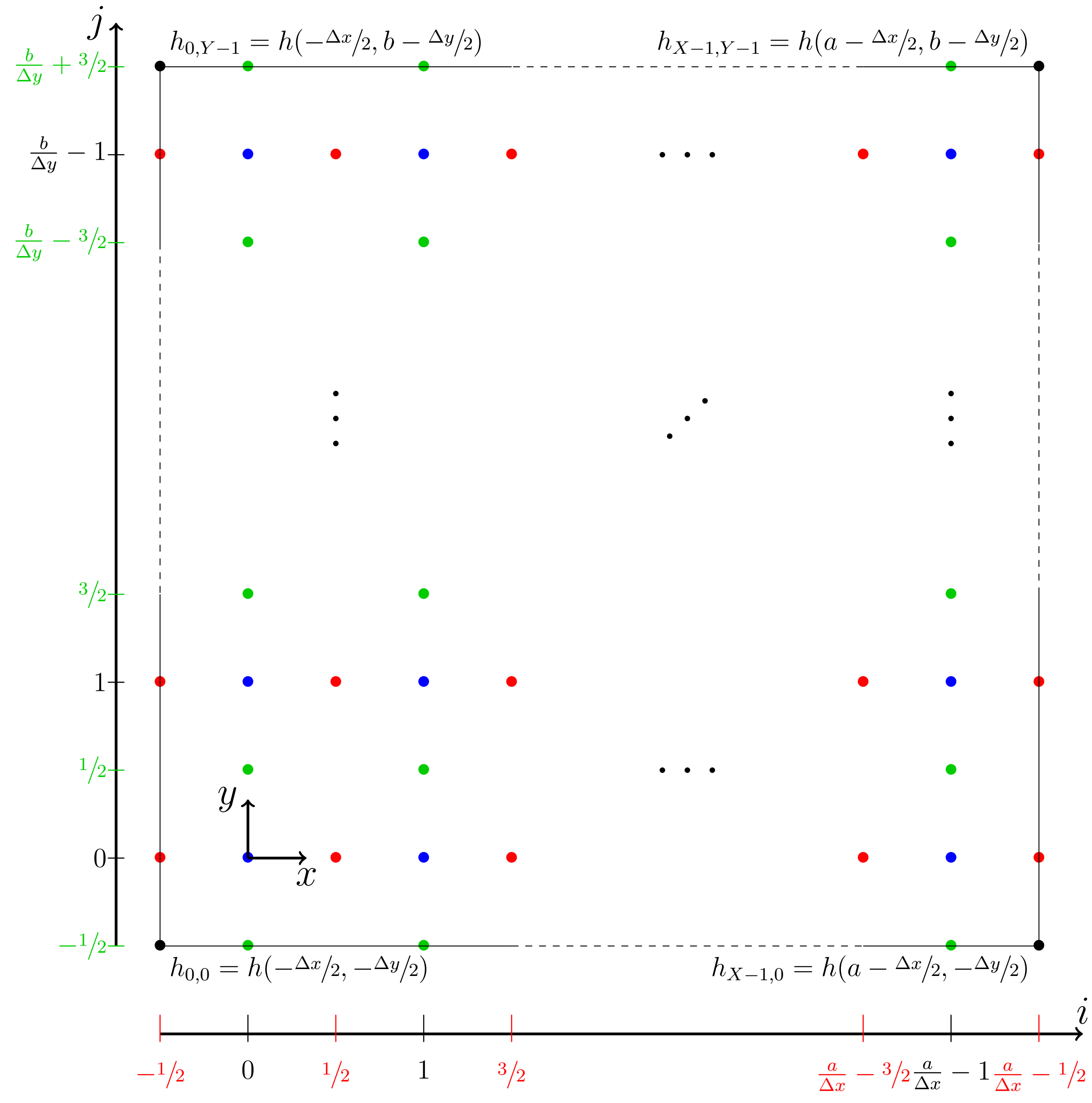
4. Compare the performance of the explicit and implicit methods for various combinations of parameters.

When your code passes on the submission platform, send your report by email to `anthony.royer@uliege.be` and in PDF format together with your C code and SLURM submission script. The files (report, C code, SLURM script) should be named

`project2_Lastname1_Lastname2.pdf`
`project2_Lastname1_Lastname2*.c`
`project2_Lastname1_Lastname2*.h` ("*" means that we can have multiple header or source files).
`project2_Lastname1_Lastname2.sh`

1 Appendix

- To clarify indices used in this project, please have a look at Figure 3.



j
$h_{0,Y-1} = h(-^{\Delta x}/_2, b - ^{\Delta y}/_2)$
$h_{X-1,Y-1} = h(a - ^{\Delta x}/_2, b - ^{\Delta y}/_2)$
$\frac{b}{\Delta y} + ^3/_2$
$\frac{b}{\Delta y} - 1$
$\frac{b}{\Delta y} - ^3/_2$
$^3/_2$
1
$^1/_2$
0
$-^1/_2$
y
x
$h_{0,0} = h(-^{\Delta x}/_2, -^{\Delta y}/_2)$
$h_{X-1,0} = h(a - ^{\Delta x}/_2, -^{\Delta y}/_2)$
i
$-^1/_2$
0
$^1/_2$
1
$^3/_2$
$\frac{a}{\Delta x} - ^3/_2$
$\frac{a}{\Delta x} - 1$
$\frac{a}{\Delta x} - ^1/_2$

Figure 3: The two finite difference grids with their indices and the Cartesian frame of reference.

- Matlab scripts are provided to help you to visualize your results (see `/home/ulg/ace/aroyer/INFO0939/hw2` on the NIC4 cluster).

- You can compare a bathymetric map that you read with your code to Figure 4 that shows an image of the `sriLanka.dat` bathymetric map.

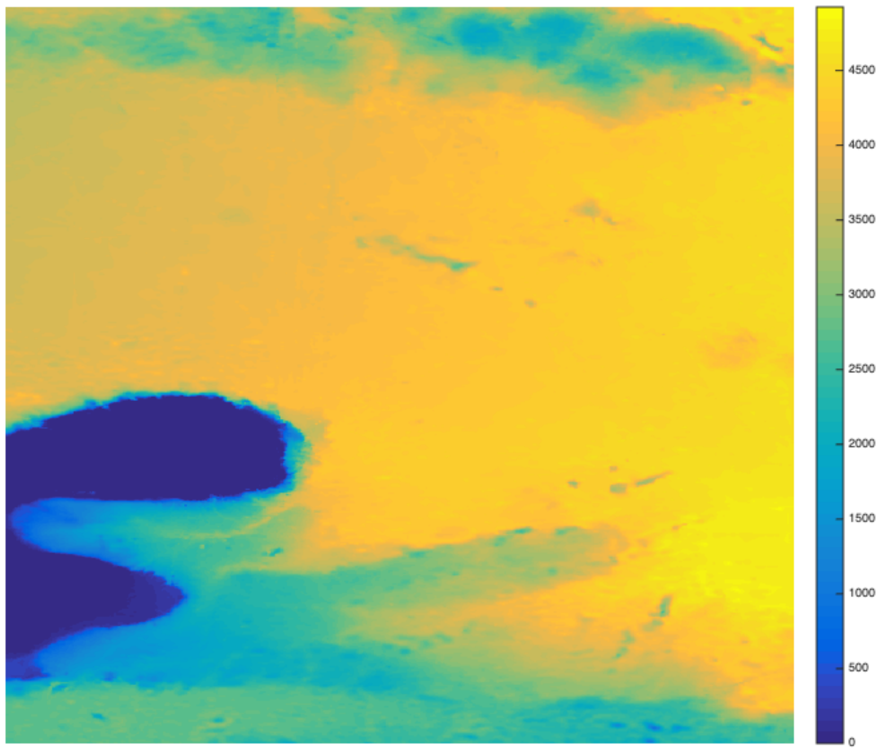


Figure 4: The Sri Lanka bathymetric map.