# In-class Programming Homework, Sep. 27

Due date: Thursday, Oct. 13 at midnight. You may work with up to one other person on the entirety of the assignment and turn in a joint program if you wish; in this case, each of you gets the same grade.

## Introduction:

In this homework, you will write a C, C++ or a Fortran code that numerically integrates the linearized shallow water equations in one dimension. The procedures should be defined in separate files. You will then use the code to approximate the solution of the elevation $h$ in the shallow equations at a given time (see below) and compare it with the exact solution. The files containing the procedures that you will use as a starting point are combined in "tar" files lab2_f.tar (Fortran), lab2_c.tar (C) and lab2_cc.tar. (C++). You might practice with the three programming languages if you wish, but please select only one for the submission of your assignment.

If you are using a computer with unix or linux shell, you can transfer these tar files to Agave by using the following commands:

```
sftp username@agave.asu.edu
sftp> cd APM512
(or the name of the working directory you created for the class)
sftp> put lab2_f.tar     (or lab2_c.tar or lab2_cc.tar)
sftp> quit
```

Login to Agave and extract the files, using the commands:

```
ssh username@agave.asu.edu
$ cd APM512 (or your working directory)
$ tar -xvf lab2_f.tar     (or lab2_c.tar or lab2_cc.tar)
```

The following folder should be created be created:
**Fortran**: LAB2_FOR: contains module_precision.f, module_flux.f, module_tendency.f and main.f
**C**: LAB2_C: contains flux.c, flux.h, tendency.c, tendency.h and main.c
**C++**: LAB2_CC: contains flux.cc, flux.h, tendency.cc, tendency.h and main.cc

To compile the code connect to a compute node:

```
$ interactive
```

Then type the commands:

```
$ module load intel/2018x
$ ifort -FR -warn -o main module_precision.f module_flux.f
                    module_tendency.f main.f (for Fortran)
$ icc -std=c99 -Wall -o main flux.c tendency.c main.c (for C)
$ icpc -Wall -o main flux.cc tendency.cc main.cc (for C++)
```

**Note that the code as it stands will not compile because there are places in the program files that you need to modify for successful compilation as we discussed in the lecture. The places to modify are indicated in the program files as well as in lecture_10.**

After the required modifications, an executable file called **"main"** should be created. Type the following command to execute main

```
$ ./main
```

## Assignment:

Consider the linearized shallow water equations in one dimension

$$\frac{\partial u}{\partial t} + g\frac{\partial h}{\partial x} = 0$$
$$\frac{\partial h}{\partial t} + H\frac{\partial u}{\partial x} = 0$$

Here, $h(x,t)$ is the perturbation of elevation, $u(x,t)$ is the perturbation of velocity, $t$ represents time and $x$ is the position along the x-axis. The parameters $g$ and $H$ represent the constant of gravity, $g = 9.81ms^{-2}$, and the elevation at rest, which is selected such that $c = \sqrt{gH} = 30ms^{-1}$.

The analytical solution for the equations above is (see lecture)

$$\begin{aligned} h(x,t) &= \frac{1}{2}\Big(p(x-ct) + p(x+ct)\Big) - \frac{H}{2c}\Big(q(x+ct) - q(x-ct)\Big) \\ u(x,t) &= \frac{1}{2}\Big(q(x-ct) + q(x+ct)\Big) - \frac{g}{2c}\Big(p(x+ct) - p(x-ct)\Big) \end{aligned}$$

where, $p(x)$ and $q(x)$ are the initial conditions: $h(x,0) = p(x)$ and $u(x,0) = q(x)$.

Let the initial perturbation for evaluation $h$ be given by

$$h(x,0) = p(x) = e^{-\left(\frac{x}{d}\right)^2}$$

where $d = 1km$. If we impose that the initial perturbation propagates in the positive $x$-direction (left to right), then the initial condition for velocity must be in the form

$$u(x,0) = q(x) = \frac{c}{H}p(x) = \frac{g}{c}p(x) = \frac{g}{c}e^{-\left(\frac{x}{d}\right)^2}$$

The provided code (after modifications) integrates the shallow water equations in the domain defined by

$$-\frac{L}{2} \leq x < \frac{L}{2}$$

where $L = 16km$. The domain uses a staggered grid with periodic boundary conditions

$$h(x+L,t) = h(x,t) \;\text{ and }\; u(x+L,t) = u(x,t)$$

The code uses a total number of $M = 160$ grid points with a grid spacing of $\Delta x = 100m$

The time integration uses the third-order Runge-Kutta (RK3) time stepping scheme; and the space derivatives are approximated by fourth-order finite differences.

The numerical solver integrates the equations from $t = 0$ to $t = n_t\Delta t$, where $\Delta t$ is time step and $n_t$ is the total number of steps. The time step is determined from $\Delta t = r\Delta x/c$, where $r = 0.5$ is the Courant number. The total number of steps $n_t$ is selected such that the initial perturbation travels approximately 2.10 revolutions around the periodic domain: $n_t c\Delta t = 2.10L$.

The algorithm of the solver is:

```
• Begin the time step loop for $it = 1, 2, \cdots, n_t$
    1. Begin the RK3 substep loop for $ir = 1, 2, 3$
        (a) RK3 substep1: $ir = 1 \implies \Delta\tau = \Delta t/3,\ s = n,\quad s+1 = *$
        (b) RK3 substep2: $ir = 2 \implies \Delta\tau = \Delta t/2,\ s = *,\quad s+1 = **$
        (c) RK3 substep3: $ir = 3 \implies \Delta\tau = \Delta t,\ s = **,\quad s+1 = n+1$
        (d) Compute the tendency terms $G_p^s$ and $F_i^s$
        (e) Advance $h$ and $u$ for the substep $s+1$:
                $h_p^{s+1} = h_p^n - \Delta\tau G_p^s$
                $u_i^{s+1} = u_i^n - \Delta\tau F_i^s$
        (f) Repeat steps a, b, c, d and e for the next substep $ir = ir + 1$
    2. End the RK3 substep loop
    3. Overwrite $h^n$ and $u^n$:
            $h_p^n = h_p^{n+1}$
            $u_i^n = u_i^{n+1}$
    4. Repeat steps 1 , 2 and 3 for the next time step $it = it + 1$
• End the time step loop
```

**1)** Write procedures and calling instructions to integrate the shallow water equations described above. The files to be modified for C and C++ are main.c (.cc), tendency.c (.cc), tendency.h. The files to be modified for Fortran are main.f and module_tendency.f. The places where you will include line(s) of code are indicated in the program files and in lecture_10, where these places are highlighted in red. Use your code the integrate the equations using the parameters $(c, H, g, L, \cdots)$ described above for a total number of time steps $n_t = 2.10L/(c\Delta t)$. A file with the name "shallow.dat" should be produced after successful execution of ./main. This file should contain the position, the numerical approximation and the exact solution of $h$ for each grid point.

**2)** Use MATLAB or you favorite graphing software to plot in the same graph the numerical and the exact solutions as a function of the position $x$ using different colors (ex. exact: black; numerical: blue). Use the following ranges for the axes: x-axis [-8km, 8km] and y-axis [ -0.4, 1.2].

## Grading rubric:

- Upload the following files to Canvas for grading:
  - A tar file hw3.tar containing your program files.
  - A graph comparing the numerical and the exact solution in pdf format.
- This assignment is worth 20 points, as follows:
  - (2 points) The program structure contains the files and the functions mentioned above with the appropriate naming
  - (3 points) Your program compiles without warnings or errors when -warn (fortran) or -Wall (c/c++) option are used with the intel compiler.
  - (10 points) When compiled and executed, your program produces accurate approximation for $h$.
  - (5 points) The program is adequately (but not excessively) commented.