A multilevel method for finite volume discretization of the two dimensional nonlinear Shallow-Water equations

Arthur Bousquet

September 16, 2009

Supervisors : Roger Temam, Sylvain Faure

Contents

1 Introduction

I did my internship at Indiana University, in the Institute for Scientific Computing and Applied Mathematics. The institute belongs to the department of Mathematics of Indiana University. The institute is composed of the Director Professor Roger Temam, postdoctoral fellows, graduate students. I was also collaborating with Sylvain Faure from the Orsay university.

The subject of my internship was to carry on the numerical part of the thesis of K. Adamy. The program for this thesis was only in Fortran and was focusing on testing method of multilevel applied on all the mesh. The goal was to modified this code to apply a benchmark of the physics. I had to solve the shallow-water equations to obtain a Rossby soliton.

The main improvement of my program are :

- Python and Fortran, instead of only Fortran
- Adding the Coriolis force
- localized multilevel method, instead of a global one

I will first explain the multilevel method in one dimension, then I will go on about the equations of shallow-water with the multilevel method in two dimensions. At the end I will explain my programming and show the validations.

2 System in 1D

In this section, I present the multilevel for finite volumes in one dimension.

2.1 System

The example system we are solving in 1 dimension is, on $\Omega = [0, L_x]$:

$$\partial_t u + \partial_x \left(\frac{u^2}{2} \right) = 0, \ \forall x \in \Omega \ and \ t > 0$$

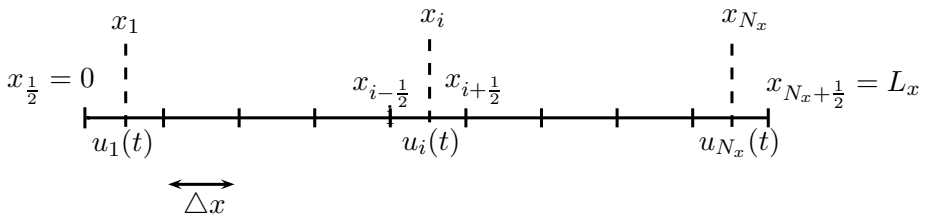
Then we apply a finite volume discretization : we divide Ω in N_x cells $K_i = \left[x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}} \right]$ of length $\triangle x$, for $i \in \{1, ..., N_x\}$:

$$x_{i+\frac{1}{2}} = i \triangle x$$

We define x_i the center of each cell K_i :

$$x_i = i \triangle x + \frac{\triangle x}{2}$$

Then we define $u(x,t)$ on each cells, that we denote $u_i(t)$ the average value of u on K_i (on the spatial component). The new unknowns become the $\{u_i \mid \ for \ i = 1...N_x\}$



$$u_i(t) = \frac{1}{\triangle x} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} u(x,t)dx, \ \forall i \in \{1, ..., N_x\}$$

Then we approximate with the flux :

$$\partial_x \left(\frac{u_i^2(t)}{2} \right) \simeq \frac{F_{i+\frac{1}{2}}(t) - F_{i-\frac{1}{2}}(t)}{\triangle x}, \ \forall i \in \{1, .., N_x\}$$

With F_i is the flux which will depends of the scheme we use.

Then the system to solve is :

$$\partial_t \left(u_i(t) \right) = -\frac{F_{i+\frac{1}{2}}(t) - F_{i-\frac{1}{2}}(t)}{\triangle x}, \ \forall i \in \{1, .., N_x\} \ \ and \ t > 0 \tag{2.1}$$

2.2 Times discretization

For $t \in [0, T]$, we divided $[0, T]$ in N_t cells $[t_n, t_{n+1}]$ of length $\triangle t$ for $n \in \{0, ..., N_t\}$, with $t_n = n \triangle t$.

We denote $u_i^n := u_i(t_n)$,for $i \in \{1, ..., N_x\}$ and $n \in \{0, ..., N_t\}$ which are the new unknowns.

We approximate the derivative in times using the Heun method. First we rewrite the system:

$$\partial_t \left(u_i(t) \right)(t_n) \simeq R(u_i^n, t_n) = -\frac{F_{i+\frac{1}{2}}(t_n) - F_{i-\frac{1}{2}}(t_n)}{\triangle x}, \ \forall i \in \{1, .., N_x\} \ \ and \ n \in \{0, ..., N_t\}$$

Then we apply the following discretization :

$$\begin{cases} k_i^1 = R(u_i^n, t_n) \\ k_i^2 = R(u_i^n + \triangle t k_i^1, t_n + \triangle t) \\ u_i^{n+1} = u_i^n + \frac{\triangle t}{2}(k_i^1 + k_i^2) \end{cases} \tag{2.2}$$

2.3 Example of fluxes

For the finite volume method we can use many different fluxes, here is an example of some of them.

2.3.1 Centered:

$$F_{i+\frac{1}{2}}(t) = \frac{1}{4}\left(u_{i+1}^2(t) + u_i^2(t)\right),\ \forall i \in \{1, ..., N_x - 1\}$$

To define $F_{\frac{1}{2}}$ and $F_{N_x+\frac{1}{2}}$, it will depends of the boundary condition.

Dirichlet with $u(0,t) = u_0(t)$ and $u(L_x,t) = u_{N_x+1}(t)$ given :
$F_{\frac{1}{2}} = \frac{1}{4}u_0^2$
$F_{N_x+\frac{1}{2}} = \frac{1}{4}u_{N_x+1}^2$
Periodic uses $u_0(t) = u_{N_x}(t)$ and $u_{N_x+1}(t) = u_0(t)$:
$F_{\frac{1}{2}} = \frac{1}{4}(u_{N_x}^2 + u_1^2)$
$F_{N_x+\frac{1}{2}} = \frac{1}{4}(u_{N_x}^2 + u_1^2)$
Neumann homogeneous uses $u_0(t) = u_1(t)$ and $u_{N_x+1}(t) = u_{N_x}(t)$:
$F_{\frac{1}{2}} = \frac{1}{4}(u_1^2 + u_1^2)$
$F_{N_x+\frac{1}{2}} = \frac{1}{4}(u_{N_x}^2 + u_{N_x}^2)$

Then we solve :

$$\partial_t\left(u_i(t)\right) = -\frac{1}{4\triangle x}\left((u_{i+1}^n(t))^2 - (u_{i-1}^n(t))^2\right),\ \forall i \in \{2, .., N_x - 1\}$$

For $i = 1$ and $i = N_x$ it will depend on the boundary condition.

Then we add the discretization in times :

$$\begin{aligned} k_i^1 &= -\tfrac{1}{4\triangle x}\left((u_{i+1}^n)^2 - (u_{i-1}^n)^2\right) \\ k_i^2 &= -\tfrac{1}{4\triangle x}\left((u_{i+1}^n + \triangle t k_i^1)^2 - (u_{i-1}^n + \triangle t k_i^1)^2\right) \\ u_i^{n+1} &= u_i^n + \tfrac{\triangle t}{2}(k_i^1 + k_i^2) \end{aligned}$$

2.3.2 Lax-Friedrichs :

$$F_{i+\frac{1}{2}}(t_n) = \frac{1}{4}\left(u_{i+1}^2(t) + u_i^2(t)\right) - \frac{\triangle x}{2\triangle t}(u_{i+1}^n - u_i^n),\ \forall i \in \{1, ..., N_x - 1\}$$

To define $F_{\frac{1}{2}}$ and $F_{N_x+\frac{1}{2}}$, it will depends of the boundary condition.

Dirichlet with $u(0,t) = u_0(t)$ and $u(L_x,t) = u_{N_x+1}(t)$ given :
$F_{\frac{1}{2}} = \frac{1}{4}u_0^2$
$F_{N_x+\frac{1}{2}} = \frac{1}{4}u_{N_x+1}^2$
Periodic uses $u_0(t) = u_{N_x}(t)$ and $u_{N_x+1}(t) = u_0(t)$:
$F_{\frac{1}{2}}(t_n) = \frac{1}{4}(u_{N_x}^2 + u_1^2) - \frac{\triangle x}{2\triangle t}(u_1^n - u_{N_x}^n)$
$F_{N_x+\frac{1}{2}}(t_n) = \frac{1}{4}(u_{N_x}^2 + u_1^2) - \frac{\triangle x}{2\triangle t}(u_1^n - u_{N_x}^n)$
Neumann homogeneous uses $u_0(t) = u_1(t)$ and $u_{N_x+1}(t) = u_{N_x}(t)$:
$F_{\frac{1}{2}} = \frac{1}{4}(u_1^2 + u_1^2)$
$F_{N_x+\frac{1}{2}} = \frac{1}{4}(u_{N_x}^2 + u_{N_x}^2)$

Then we solve :

$$\partial_t\left(u_i(t)\right)(t_n) = -\frac{1}{4\triangle x}\left((u_{i+1}^n(t))^2 - (u_{i-1}^n(t))^2\right) - \frac{\triangle x}{2\triangle t}(u_{i+1}^n + u_{i-1}^n),\ \forall i \in \{2, .., N_x - 1\}\ `and\ n \in \{0, ..., N_t\}$$

For $i = 1$ and $i = N_x$ it will depend on the boundary condition.

Then we add the discretization in times :

$$k_i^1 = -\frac{1}{4\triangle x}\left((u_{i+1}^n)^2 - (u_{i-1}^n)^2\right) - \frac{\triangle x}{2\triangle t}(u_{i+1}^n + u_{i-1}^n)$$
$$k_i^2 = -\frac{1}{4\triangle x}\left((u_{i+1}^n + \triangle t k_i^1)^2 - (u_{i-1}^n + \triangle t k_i^1)^2\right) - \frac{\triangle x}{2\triangle t}(u_{i+1}^n + \triangle t k_i^1 + u_{i-1}^n + \triangle t k_i^1)$$
$$u_i^{n+1} = u_i^n + \frac{\triangle t}{2}(k_i^1 + k_i^2)$$

2.3.3 Central-upwind :

This schema is supposed to be precise and to handle chocks with a little of diffusion. It takes data from "left" as many as from "right", but it also considers the sens of propagation.

$$F_{i+\frac{1}{2}}(t_n) = \frac{1}{2}\left(\frac{a_{i+\frac{1}{2}}^+(u_{i+\frac{1}{2}}^-)^2 - a_{i+\frac{1}{2}}^-(u_{i+\frac{1}{2}}^+)^2}{a_{i+\frac{1}{2}}^+ - a_{i+\frac{1}{2}}^-}\right) - \frac{a_{i+\frac{1}{2}}^+ a_{i+\frac{1}{2}}^-}{a_{i+\frac{1}{2}}^+ - a_{i+\frac{1}{2}}^-}(u_{i+\frac{1}{2}}^+ - u_{i+\frac{1}{2}}^-),\ \forall i \in \{1, ..., N_x - 1\}$$

With :

$$u_{i+\frac{1}{2}}^+ = p_{i+1}(x_{i+\frac{1}{2}})\ \left(\text{polynomial of reconstruction of u on } \left[x_{i+\frac{1}{2}}, x_{i+\frac{3}{2}}\right]\right)$$
$$u_{i+\frac{1}{2}}^- = p_i(x_{i+\frac{1}{2}})\ \left(\text{polynomial of reconstruction of u on } \left[x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}\right]\right)$$
$$a_{i+\frac{1}{2}}^+ = max\left\{\lambda_N\left(\frac{\partial f}{\partial u}(u_{i+\frac{1}{2}}^-)\right), \lambda_N\left(\frac{\partial f}{\partial u}(u_{i+\frac{1}{2}}^+)\right), 0\right\} = max\left\{u_{i+\frac{1}{2}}^-, u_{i+\frac{1}{2}}^+, 0\right\}$$
$$a_{i+\frac{1}{2}}^- = min\left\{\lambda_1\left(\frac{\partial f}{\partial u}(u_{i+\frac{1}{2}}^-)\right), \lambda_1\left(\frac{\partial f}{\partial u}(u_{i+\frac{1}{2}}^+)\right), 0\right\} = min\left\{u_{i+\frac{1}{2}}^-, u_{i+\frac{1}{2}}^+, 0\right\}$$

Where :

$\frac{\partial f}{\partial u}(w) = w$ (Jacobian)
$\lambda_1\left(\frac{\partial f}{\partial u}(w)\right) < ... < \lambda_N\left(\frac{\partial f}{\partial u}(w)\right)$ being the N eigenvalue of the Jacobian $\frac{\partial f}{\partial u}$ at w
$(u_x)_i^n = minmod\left(\frac{1}{\triangle x}(u_i^n - u_{i+1}^n); \frac{1}{\triangle x}(u_{i+1}^n - u_i^n)\right)$
$minmod(a, b) = \frac{1}{2}\left(sgn(a) + sgn(b)\right) min(\mid a \mid, \mid b \mid)$

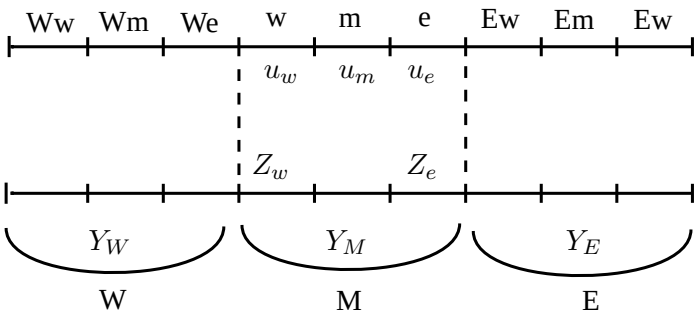
To define $F_{\frac{1}{2}}$ and $F_{N_x+\frac{1}{2}}$, it will depends of the boundary condition. Then we use the same method for the boundary conditions and the time resolution as the two previous fluxes.

2.4 Multilevel

We define also $Y \simeq u$ on each cell of length $3\triangle x$, which the coarse mesh.

$$u \ \simeq \ Y + Z$$

Where Z is define on the fine mesh.



We define the variables :

$$\begin{cases} Y_M = \frac{1}{3}(u_w + u_m + u_e) \\ Z_e = u_e - \frac{1}{3}(2Y_M + Y_E) = O(\triangle x^2) \\ Z_w = u_w - \frac{1}{3}(2Y_M + Y_W) = O(\triangle x^2) \end{cases} \tag{2.3}$$

So :

$$\begin{cases} u_m = 3Y_M - Z_e - Z_w - \frac{1}{3}(4Y_M + Y_E + Y_W) \\ u_e = Z_e + \frac{1}{3}(2Y_M + Y_E) \\ u_w = Z_w + \frac{1}{3}(2Y_M + Y_W) \end{cases} \tag{2.4}$$

Proof of the $O(\triangle x^2)$ using Taylor expansion :

$$\begin{aligned} Y_M &= \tfrac{1}{3}(u_w + u_m + u_e) \\ &= \tfrac{1}{3}(\underbrace{u_e - 2\triangle x\partial_x u_e + O(\triangle x^2)}_{u_w} + \underbrace{u_e - \triangle x\partial_x u_e + O(\triangle x^2)}_{u_m} + u_e) \\ &= u_e - \triangle x\partial_x u_e + O(\triangle x^2) \end{aligned}$$

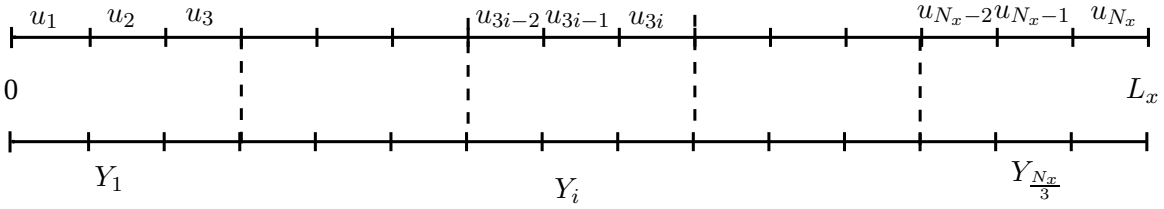
$$\begin{aligned} Y_E &= \tfrac{1}{3}(u_{Ew} + u_{Em} + u_{Ee}) \\ &= \tfrac{1}{3}(\underbrace{u_e + \triangle x\partial_x u_e + O(\triangle x^2)}_{u_{Ew}} + \underbrace{u_e + 2\triangle x\partial_x u_e + O(\triangle x^2)}_{u_{Em}} + \underbrace{u_e + 3\triangle x\partial_x u_e + O(\triangle x^2)}_{u_{Ee}}) \\ &= u_e + 2\triangle x\partial_x u_e + O(\triangle x^2) \end{aligned}$$

$Then$:

$$\begin{aligned} Z_e &= u_e - \tfrac{1}{3}(2u_e - 2\triangle x\partial_x u_e + u_e + 2\triangle x\partial_x u_e + O(\triangle x^2)) \\ &= O(\triangle x^2) \end{aligned}$$

It is the same with Z_w.

Then the scheme for Y_i:



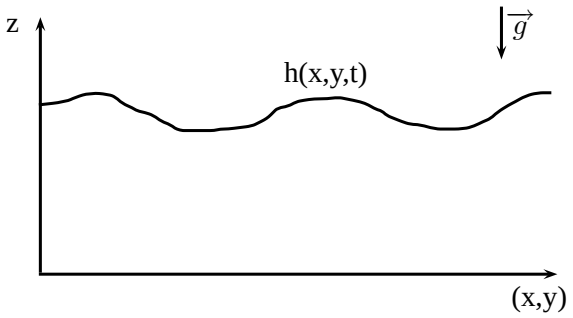
With (2.1) and (2.3), we have for $i \in \{1, ..., \frac{N_x}{3}\}$:

$$\begin{cases} \partial_t Y_i &= \frac{1}{3}(\partial_t u_{3i} + \partial_t u_{3i-1} + \partial_t u_{3i-2}) \\ &= \frac{1}{3} \times \frac{1}{\triangle x} \times (F_{3i+\frac{1}{2}} - F_{3i-\frac{1}{2}} + F_{3i-\frac{1}{2}} - F_{3i-\frac{3}{2}} + F_{3i-\frac{3}{2}} - F_{3i-\frac{5}{2}}) \\ &= \frac{1}{12\triangle x}(F_{3i+\frac{1}{2}} - F_{3i-\frac{5}{2}}) \end{cases} \tag{2.5}$$

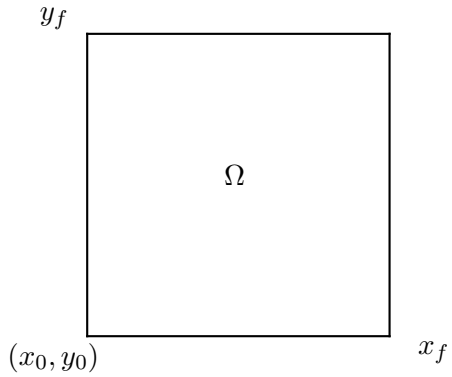
3 Shallow Water

The equations that my program solves are the shallow-water equations. In this section I present the equations with the multilevel method for finite volume applied to them.

3.1 The equations



On $\{\Omega = [x_0, x_f] \times [y_0, y_f]\} \times \{0 < t < T\}$, with Dirichlet boundary conditions.



$$\begin{cases} \dfrac{\partial h}{\partial t} + \dfrac{\partial uh}{\partial x} + \dfrac{\partial vh}{\partial y} = 0, \\ \dfrac{\partial uh}{\partial t} + \dfrac{\partial hu^2}{\partial x} + \dfrac{\partial huv}{\partial y} + \dfrac{g}{2}\dfrac{\partial h^2}{\partial x} - fvh = 0, \\ \dfrac{\partial vh}{\partial t} + \dfrac{\partial huv}{\partial x} + \dfrac{\partial hv^2}{\partial y} + \dfrac{g}{2}\dfrac{\partial h^2}{\partial y} + fuh = 0. \end{cases} \tag{3.1}$$

Here :

- h is the fluid depth above the bottom which is supposed flat
- u and v are the x and y components of the velocity
- g denotes the gravity
- f is the Coriolis force, which is equals to $f_0 + \beta y$

The boundary conditions can be of two types. We use the periodic boundary conditions to validate the program, because we know an analytic solution which is periodic. We use the Dirichlet boundary conditions to solve the equations for the Equatorial Rossby Soliton, in this case the conditions are :

$$\begin{cases} u(x=x_0,y,t) &= u_{x=x_0}(y,t), \\ u(x=x_f,y,t) &= u_{x=x_f}(y,t), \\ u(x,y=y_f,t) &= u_{y=y_f}(x,t), \\ u(x,y=f_0,t) &= u_{y=y_0}(x,t), \end{cases} \tag{3.2}$$

$$\begin{cases} v(x=x_0,y,t) &= v_{x=x_0}(y,t), \\ v(x=x_f,y,t) &= v_{x=x_f}(y,t), \\ v(x,y=y_f,t) &= v_{y=y_f}(x,t), \\ v(x,y=y_0,t) &= v_{y=y_0}(x,t), \end{cases} \tag{3.3}$$

$$\begin{cases} h(x=x_0,y,t) &= h_{x=x_0}(y,t), \\ h(x=x_f,y,t) &= h_{x=x_f}(y,t), \\ h(x,y=y_f,t) &= h_{y=y_f}(x,t), \\ h(x,y=y_0,t) &= h_{y=y_0}(x,t). \end{cases} \tag{3.4}$$

We denote the initial solution as follow :

$$\begin{cases} u(x,y,0) = u_0(t), \\ v(x,y,0) = v_0(t), \\ h(x,y,0) = h_0(t), \end{cases} \tag{3.5}$$

then we write :

$$U = uh \text{ and } V = vh$$

$$Q = \begin{pmatrix} h \\ U \\ V \end{pmatrix},\ F(Q) = \begin{pmatrix} U \\ \frac{1}{h}U^2 + \frac{1}{2}gh^2 \\ \frac{1}{h}UV \end{pmatrix},\ G(Q) = \begin{pmatrix} V \\ \frac{1}{h}UV \\ \frac{1}{h}V^2 + \frac{1}{2}gh^2 \end{pmatrix},\ \Phi(Q) = \begin{pmatrix} 0 \\ -fV \\ fU \end{pmatrix},$$

which permits us to write the system in the conservative form

$$\begin{cases} \dfrac{\partial h}{\partial t} + \nabla.(U,V) = 0 \\ \dfrac{\partial U}{\partial t} + \nabla.(uU,uV) + \dfrac{g}{2}\nabla_x(h^2) - fV = 0 \\ \dfrac{\partial V}{\partial t} + \nabla.(vU,vV) + \dfrac{g}{2}\nabla_y(h^2) + fU = 0 \end{cases} \tag{3.6}$$

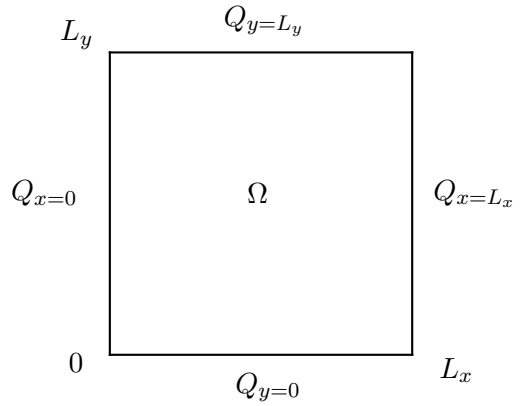
or equivalently:

$$\frac{\partial Q}{\partial t} + \frac{\partial F(Q)}{\partial x} + \frac{\partial G(Q)}{\partial y} + \Phi(Q) = 0. \tag{3.7}$$

In the case of the boundary condition of Dirichlet, we have :

$$Q_{x=0} = \begin{pmatrix} h_{x=0} \\ U_{x=0} \\ V_{x=0} \end{pmatrix},\ Q_{x=L_x} = \begin{pmatrix} h_{x=L_x} \\ U_{x=L_x} \\ V_{x=L_x} \end{pmatrix}$$

$$Q_{y=0} = \begin{pmatrix} h_{y=0} \\ U_{y=0} \\ V_{y=0} \end{pmatrix}, \; Q_{y=L_y} = \begin{pmatrix} h_{y=L_y} \\ U_{y=L_y} \\ V_{y=L_y} \end{pmatrix}$$

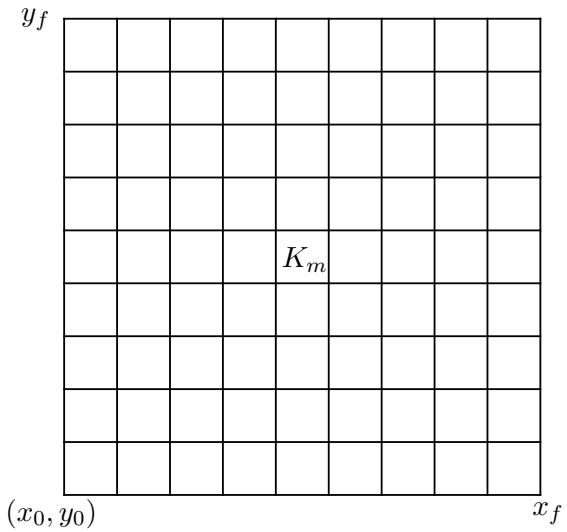


Then we have to discretize the equations in space and in time. First we discretize the equations in space, using the finite volume methods with the "central-upwind" approximation.

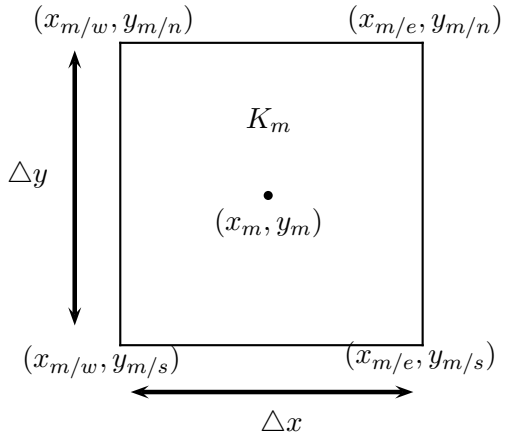
3.2 Discretization in space

3.2.1 Discretization of the mesh

The discretization of Ω is done using rectangular finite volumes :



$K_m = [x_{m/w}, x_{m/e}] \times [y_{m/s}, y_{m/n}]$ of centers (x_m, y_m) and of dimensions $\Delta x \times \Delta y$, with $N_x \Delta x = L_x = (x_f - x_0)$ and $N_y \Delta y = L_y = (y_f - y_0)$:



3.2.2 Discretization of the variables

We use a NSWE (North-South-West-East) stencil to identify the unknowns :

K_{nw}	K_n	K_{ne}
K_w	K_m	K_e
K_{sw}	K_s	K_{se}

The unknowns will be approximations of the cell averages:

$$Q_m(t) = \frac{1}{\Delta x \Delta y} \int_{K_m} Q(t,x,y) dxdy,$$

where $Q_m(t) = (h_m(t), U_m(t), V_m(t))^T$ with:

$$h_m(t) = \frac{1}{\Delta x \Delta y} \int_{K_m} h(t,x,y) dxdy,$$

$$U_m(t) = \frac{1}{\Delta x \Delta y} \int_{K_m} U(t,x,y) dxdy,$$

$$V_m(t) = \frac{1}{\Delta x \Delta y} \int_{K_m} V(t,x,y) dxdy,$$

For the space discretization, we integrate the system (3.7) on each cell K_m, divide by its area $\Delta x \Delta y$:

$$\frac{\partial}{\partial t} \frac{1}{\Delta x \Delta y} \int_{k_m} Q + \frac{1}{\Delta x \Delta y} \int_{k_m} \frac{\partial F(Q)}{\partial x} + \frac{1}{\Delta x \Delta y} \int_{k_m} \frac{\partial G(Q)}{\partial y} + \frac{1}{\Delta x \Delta y} \int_{k_m} \Phi(Q) = 0. \quad (3.8)$$

For $\Phi(Q)$ which is :

$$\Phi(Q) = \begin{pmatrix} 0 \\ -fV \\ fU \end{pmatrix}$$

With $f = f_0 + \beta y$ depends of the cell K_m. The integration will give us :

$$\begin{aligned} fU &\simeq \tfrac{1}{\triangle x \triangle y} \int_{K_m} fU \\ &\simeq U_m \tfrac{1}{\triangle x \triangle y} \int_{K_m} f_0 + \beta y \\ &= U_m \tfrac{1}{\triangle x \triangle y} \left(f_0 \triangle y \triangle x + \beta \triangle x (y_{m/n}^2 - y_{m/s}^2) \right) \\ &= U_m f_0 + U_m \beta \tfrac{y_{m/n} + y_{m/s}}{2} \\ &= (f_0 + \beta y_m) U_m \end{aligned}$$

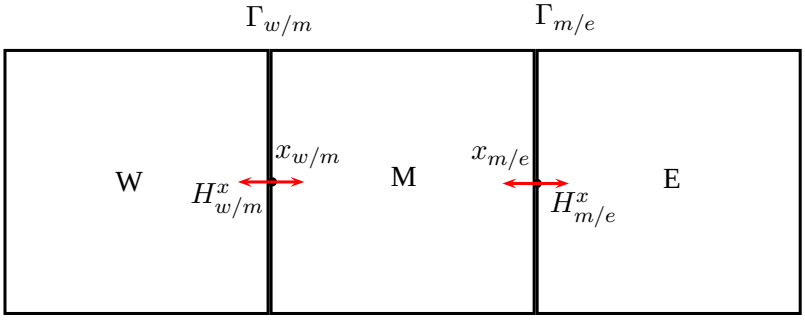
And for V :

$$fV \simeq (f_0 + \beta y_m) V_m$$

So :

$$\frac{1}{\Delta x \Delta y} \int_{k_m} \Phi(Q) \cong \Phi(Q_m) = \begin{pmatrix} 0 \\ -(f_0 + \beta y_m) V_m \\ (f_0 + \beta y_m) U_m \end{pmatrix}$$

For $\frac{\partial F(Q)}{\partial x}$:



$$\frac{1}{\Delta x \Delta y} \int_{k_m} \frac{\partial F(Q)}{\partial x} = \frac{1}{\Delta x \Delta y} \int_{\Gamma_{m/e}} F(Q(x_{m/e}, y, t) dy - \frac{1}{\Delta x \Delta y} \int_{\Gamma_{w/m}} F(Q(x_{w/m}, y, t) dy$$

Here $H^x_{m/e}(t)$ and $H^y_{w/m}(t)$ are respectively the horizontal fluxes on the edges between K_m / K_e and between K_m / K_w.

Then :

$$H^x_{m/e}(t) = \frac{1}{\Delta y} \int_{\Gamma_{m/e}} F(Q(x_{m/e}, y, t)) dy$$

Which give us :

$$\frac{1}{\Delta x \Delta y} \int_{k_m} \frac{\partial F(Q)}{\partial x} = \frac{H^x_{m/e}(t) - H^x_{w/m}(t)}{\Delta x}$$

We apply the same notation for $\frac{\partial G(Q)}{\partial y}$. Then for the all system we obtain:

$$\frac{d}{dt} Q_m(t) \cong - \frac{H^x_{m/e}(t) - H^x_{w/m}(t)}{\Delta x} - \frac{H^y_{m/n}(t) - H^y_{s/m}(t)}{\Delta y} - \Phi(Q_m) \tag{3.9}$$

These fluxes depend on the method employed; we will consider central-upwind fluxes, but *the multilevel method can also be based on other fluxes.*

3.3 Scheme in space

The space discretization is done using a semi-discrete central-upwind scheme; we describe here in detail the expression of these central-upwind fluxes. These types of schemes have the advantage of being perfectly adapted to the discretization of hyperbolic systems of conservation laws due to their upwind nature while being robust and simple since they do not require to solve any Riemann problem; moreover they are non staggered schemes. The starting point of the construction of this type of schemes is the equivalent integral formulation of the system. They are based on integration over Riemann fans using the one-sided local speeds of propagation.

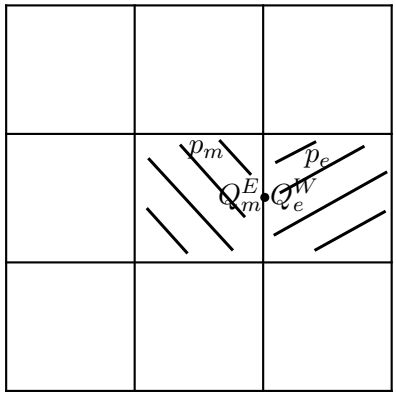
3.3.1 Computation of the fluxes

Recall that the semi-discrete form of the scheme reads:

$$\frac{d}{dt}Q_m(t) \cong -\frac{H^x_{m/e}(t) - H^x_{w/m}(t)}{\Delta x} - \frac{H^y_{m/n}(t) - H^y_{s/m}(t)}{\Delta y} - \Phi(Q_m),$$

We use a second order version, and the corresponding numerical fluxes are:

$$H^x_{m/e} \cong \frac{a^+_{m/e}F(Q^E_m) - a^-_{m/e}F(Q^W_e)}{a^+_{m/e} - a^-_{m/e}} + \frac{a^+_{m/e}a^-_{m/e}(Q^W_e - Q^E_m)}{a^+_{m/e} - a^-_{m/e}} \tag{3.10}$$



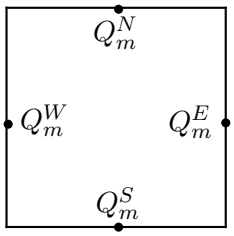
and :

$$\begin{aligned}
H^x_{w/m} &\cong \frac{a^+_{w/m}F(Q^E_w) - a^-_{w/m}F(Q^W_m)}{a^+_{w/m} - a^-_{w/m}} + \frac{a^+_{w/m}a^-_{w/m}(Q^W_m - Q^E_w)}{a^+_{w/m} - a^-_{w/m}} \\
H^y_{m/n} &\cong \frac{b^+_{m/n}G(Q^N_m) - b^-_{m/n}G(Q^S_n)}{b^+_{m/n} - b^-_{m/n}} + \frac{b^+_{m/n}b^-_{m/n}(Q^S_n - Q^N_m)}{b^+_{m/n} - b^-_{m/n}} \\
H^y_{s/m} &\cong \frac{b^+_{s/m}G(Q^N_s) - b^-_{s/m}G(Q^S_m)}{b^+_{s/m} - b^-_{s/m}} + \frac{b^+_{s/m}b^-_{s/m}(Q^S_m - Q^N_s)}{b^+_{s/m} - b^-_{s/m}}
\end{aligned} \tag{3.11}$$

Here, we use a non-oscillatory linear polynomial reconstruction to evaluate the following point values which are present in (3.10), (3.11):

$$Q^E_m = p_m(t, x_{m/e}, y_m),\ Q^W_m = p_m(t, x_{m/w}, y_m),$$

$$Q_m^N = p_m(t, x_m, y_{m/n}),\ Q_m^S = p_m(t, x_m, y_{m/s}).$$



where $p_m(t,x,y) = Q_m(t) + s_m^x(t)(x - x_m) + s_m^y(t)(y - y_m)$. is a piecewise polynomial of reconstruction on K_m

We use a piecewise linear reconstruction in order to obtain a second order scheme. The order of the scheme also relates to the order of the quadrature formula used to approximate the flux integrals coming from the integral formulation.

The slopes of this linear approximation are calculated using a minmod limiter:

$$s_m^x(t) = \text{minmod}\left((\theta\frac{Q_m(t) - Q_w(t)}{\Delta x}; \frac{Q_e(t) - Q_w(t)}{2\Delta x}; \theta\frac{Q_e(t) - Q_m(t)}{\Delta x})\right),$$

$$s_m^y(t) = \text{minmod}\left((\theta\frac{Q_m(t) - Q_s(t)}{\Delta y}; \frac{Q_n(t) - Q_s(t)}{2\Delta y}; \theta\frac{Q_n(t) - Q_m(t)}{\Delta y}\right)),$$

with

$$\text{minmod}(x_1, x_2, ..) := \begin{cases} \min(x_i),\ \text{if } x_i > 0\ \forall i \\ \max(x_i),\ \text{if } x_i < 0\ \forall i \\ 0,\ \text{otherwise.} \end{cases}$$

where $\theta \in [1, 2]$.

Then the one-sided local speeds of propagation are given by:

$$a_{m/e}^+ = \max[\lambda_{max}(\frac{\partial F}{\partial Q}(Q_e^W)), \lambda_{max}(\frac{\partial F}{\partial Q}(Q_m^E)), 0]$$

$$a_{m/e}^- = \min[\lambda_{min}(\frac{\partial F}{\partial Q}(Q_e^W)), \lambda_{min}(\frac{\partial F}{\partial Q}(Q_m^E)), 0]$$

$$a_{w/m}^+ = \max[\lambda_{max}(\frac{\partial F}{\partial Q}(Q_m^W)), \lambda_{max}(\frac{\partial F}{\partial Q}(Q_w^E)), 0]$$

$$a_{w/m}^- = \min[\lambda_{min}(\frac{\partial F}{\partial Q}(Q_m^W)), \lambda_{min}(\frac{\partial F}{\partial Q}(Q_w^E)), 0]$$

$$b_{m/n}^+ = \max[\lambda_{max}(\frac{\partial G}{\partial Q}(Q_n^S)), \lambda_{max}(\frac{\partial G}{\partial Q}(Q_m^N)), 0]$$

$$b_{m/n}^- = \min[\lambda_{min}(\frac{\partial G}{\partial Q}(Q_n^S)), \lambda_{min}(\frac{\partial G}{\partial Q}(Q_m^N)), 0]$$

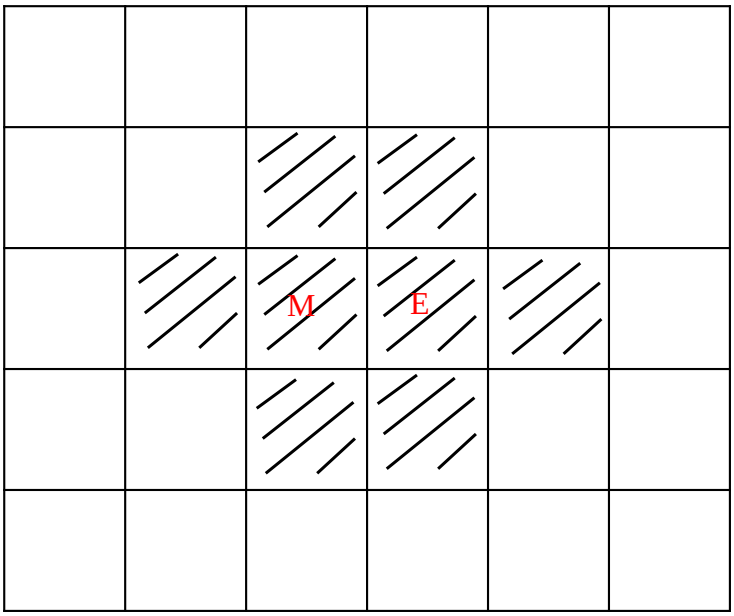
$$b_{s/m}^+ = \max[\lambda_{max}(\frac{\partial G}{\partial Q}(Q_m^S)), \lambda_{max}(\frac{\partial G}{\partial Q}(Q_s^N)), 0]$$

$$b_{s/m}^- = \min[\lambda_{min}(\frac{\partial G}{\partial Q}(Q_m^S)), \lambda_{min}(\frac{\partial G}{\partial Q}(Q_s^N)), 0]$$

where $\lambda_{max}(\frac{\partial F}{\partial Q}(\tilde{Q}))$ and $\lambda_{min}(\frac{\partial F}{\partial Q}(\tilde{Q}))$ (resp. $\lambda_{max}(\frac{\partial G}{\partial Q}(\tilde{Q}))$ and $\lambda_{min}(\frac{\partial G}{\partial Q}(\tilde{W})))$ are respectively the largest and the smallest eigenvalue of the Jacobian matrix of F, $\frac{\partial F}{\partial Q}$ (resp. of G, $\frac{\partial G}{\partial Q}$) at the point $\tilde{Q}$.
This gives a central-upwind scheme of second order.

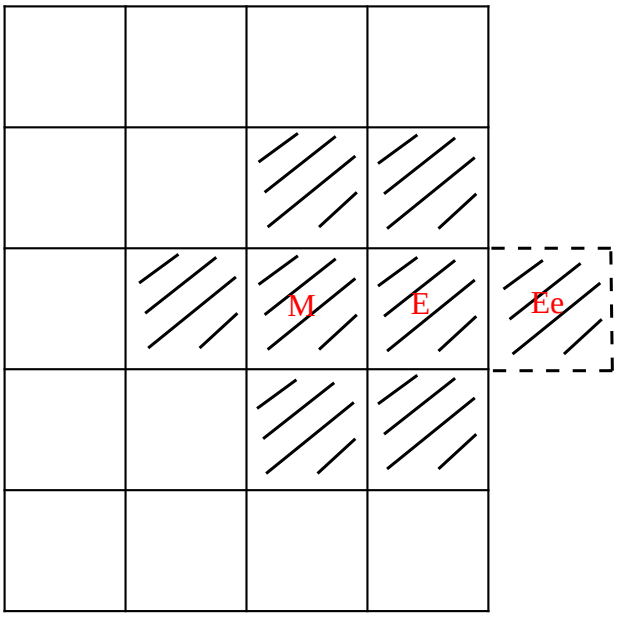
3.3.2 Issue on the boundary

We will talk about the Dirichlet conditions, because with the periodic conditions we just have to apply the fluxes as usual. The issue is near the boundary, because when we compute one flux we need the value of Q on 9 squares :



For the horizontal fluxes between the cells M and E

So if E is on the boundary we have to fide a value for the fictive cell on the east :



In this case we have to compute the value of Q_{Ee} knowing Q_E and $Q_{x=x_f}$.

$$Q_E = Q_{Ee} + \triangle x Q'_{Ee} + O(\triangle x^2)$$
$$Q_{x=x_f} = Q_{Ee} + \frac{\triangle x}{2} Q'_{Ee} + O(\frac{\triangle x^2}{2})$$

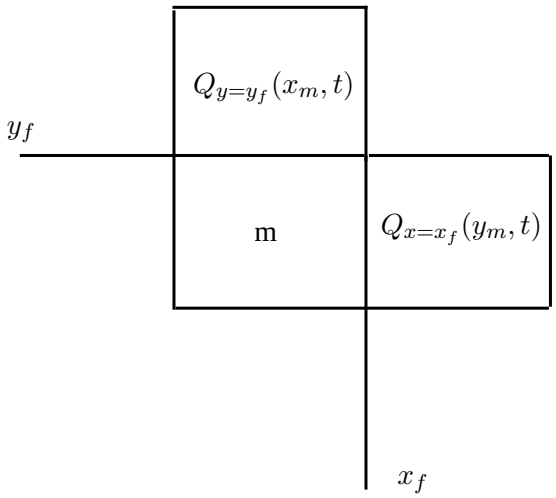
$$\Longrightarrow Q_{Ee} \simeq 2Q_{x=x_f} - Q_E$$

We can apply this method to all the fictive cells for the reconstruction.

3.3.3 Boundary conditions

When we consider periodic boundary conditions, then the quantities on the boundaries are evaluated using periodicity.

For the Dirichlet boundary conditions, every times the cell "m" touches the border we use a fictitious exterior cell; see Figure below showing the fictitious cells when "m" is at the upright corner .
For example if "**n**" and "**e**" do not exist we create two fictitious cells :



So for each case we do :

- If "**n**" does not exist :
$$H^y_{m/n} = G(Q_{y=y_f})$$
- If "**s**" does not exist :
$$H^y_{s/m} = G(Q_{y=y_0})$$
- If "**e**" does not exist :
$$H^x_{m/e} = F(Q_{x=x_f})$$
- If "**w**" does not exist :
$$H^x_{w/m} = F(Q_{x=x_0})$$

3.4 Scheme in time

First we discretize in time $[0,T]$, with N_t cells $[t_n, t_{n+1}]$ of length $\triangle t$. With :

$$t_n = n\triangle t,\ \forall i \in \{0,...,N_t\}$$

If we rewrite (3.9) as

$$\frac{d}{dt}Q_m = R(Q_m(t),t),$$

For Runge-Kutta order 2 (Heun), we apply the following time discretization:

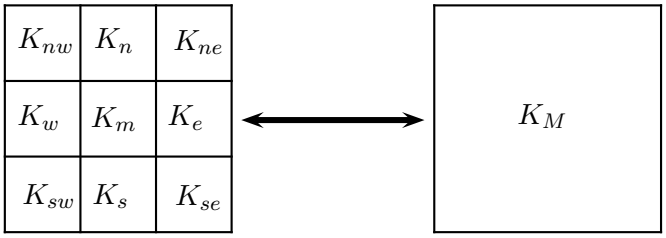
$$\begin{cases} k^n_{1,m} = R(Q^n_m, t_n), \\ k^n_{2,m} = R(Q^n_m + \Delta t k^n_{1,m}, t_n + \Delta t), \\ Q^{n+1}_m = Q^n_m + \frac{\Delta t}{2}(k^n_{1,m} + k^n_{2,m}). \end{cases} \tag{3.12}$$

For Runge-Kutta order 4, we apply the following time discretization:

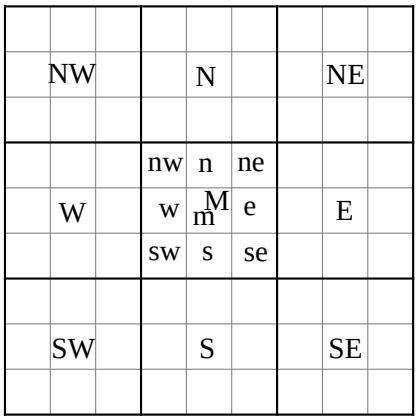
$$\begin{cases} k^n_{1,m} = R(Q^n_m, t_n), \\ k^n_{2,m} = R(Q^n_m + \frac{\Delta t}{k^n_{1,m}}, t_n + \frac{\Delta t}{2}), \\ k^n_{3,m} = R(Q^n_m + \frac{\Delta t}{k^n_{2,m}}, t_n + \frac{\Delta t}{2}), \\ k^n_{4,m} = R(Q^n_m + \Delta t k^n_{3,m}, t_n + \Delta t), \\ Q^{n+1}_m = Q^n_m + \frac{\Delta t}{6}(k^n_{1,m} + 2k^n_{2,m} + 2k^n_{3,m} + k^n_{4,m}). \end{cases} \tag{3.13}$$

3.5 Multilevel

The domain is discretized by two levels of rectangular finite volume meshes: the fine mesh $\mathcal{M}_1$ counts $N_x \times N_y$ control volumes of dimensions $\Delta x \times \Delta y$, with $N_x \Delta x = L_x$, $N_y \Delta y = L_y$; and the coarse mesh $\mathcal{M}_2$ has $\dfrac{N_x N_y}{9}$ control volumes of dimensions $3\Delta x \times 3\Delta y$.
Here we use small letters for the fine mesh and capital letters for the coarse mesh: we denote by K_m a control volume of the fine mesh and by K_M a control volume of the coarse mesh.

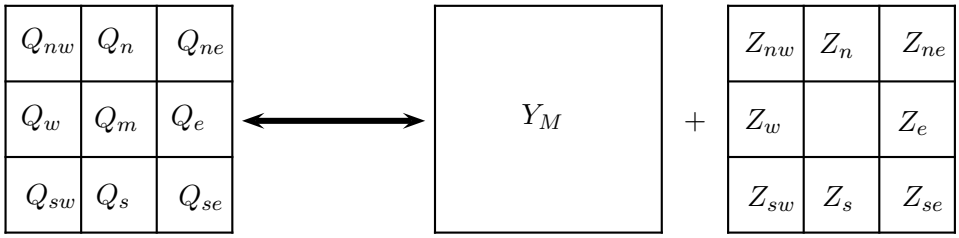


On a larger scale :



We define the incremental unknowns for the conservative variables of the Shallow Water system, that is the three components of Q. We split each of the unknowns in a large-scale component Y and a small-scale component Z, which is meant to be frozen during a certain number of time steps. By large-scale and small scale, we mean that Y contains the major information on the solution and that Z represents a correcting term which is comparatively small, as explained in Lemma 2.1.

Definition 3.1. *Suppose that $Q = (h, U, V)^T$ is known on the fine mesh $\mathcal{M}_1$.*
Then on the control volume K_M, the large-scale component $Y_M = (y_h, y_U, y_V)^T$ and the small-scale components Z_e, Z_w, Z_n, Z_s, Z_{ne}, Z_{se}, Z_{nw}, Z_{sw} are defined as follows:



$$
\begin{aligned}
&Y_M = \frac{1}{9}(Q_m + Q_e + Q_w + Q_n + Q_s + Q_{ne} + Q_{nw} + Q_{se} + Q_{sw}),\\
&Z_e = Q_e - \frac{1}{3}(Y_E + 2Y_M), \quad Z_w = Q_w - \frac{1}{3}(Y_W + 2Y_M),\\
&Z_n = Q_n - \frac{1}{3}(Y_N + 2Y_M), \quad Z_s = Q_s - \frac{1}{3}(Y_S + 2Y_M),\\
&Z_{ne} = Q_{ne} - \frac{1}{3}(Y_E + Y_M + Y_N), \quad Z_{se} = Q_{se} - \frac{1}{3}(Y_S + Y_M + Y_E),\\
&Z_{nw} = Q_{nw} - \frac{1}{3}(Y_W + Y_M + Y_N), \quad Z_{sw} = Q_{sw} - \frac{1}{3}(Y_S + Y_M + Y_W),
\end{aligned}
$$

Remark 3.1. The correspondence between the solution on the fine mesh Q and its large-scale and small-scale components is 1-to-1; that is knowing the component Y on the coarse mesh and the components Z on the fine mesh, we can compute Q on the fine mesh; here are the recomposition formulas:

$$
\begin{aligned}
Q_m &= 5Y_M - Y_E - Y_W - Y_N - Y_S\\
&- (Z_e + Z_w + Z_n + Z_s + Z_{ne} + Z_{se} + Z_{nw} + Z_{sw}),\\
Q_e &= Z_e + \frac{1}{3}(Y_E + 2Y_M), \qquad Q_w = Z_w + \frac{1}{3}(Y_W + 2Y_M),\\
Q_n &= Z_n + \frac{1}{3}(Y_N + 2Y_M), \qquad Q_s = Z_s + \frac{1}{3}(Y_S + 2Y_M),\\
Q_{ne} &= Z_{ne} + \frac{1}{3}(Y_E + Y_M + Y_N), \quad Q_{se} = Z_{se} + \frac{1}{3}(Y_S + Y_M + Y_E),\\
Q_{nw} &= Z_{nw} + \frac{1}{3}(Y_W + Y_M + Y_N), \quad Q_{sw} = Z_{sw} + \frac{1}{3}(Y_S + Y_M + Y_W).
\end{aligned}
$$

Lemma 3.1. *The small-scale components Z_e, Z_w, Z_s, Z_n, Z_{ne}, Z_{se}, Z_{nw}, Z_{sw} are of order $\Delta x^2 + \Delta y^2$.*

Proof. Using Taylor's formula, we have for example for Z_s:

$$
\begin{aligned}
Z_s &= Q_s - \frac{1}{3}(Y_S + 2Y_M)\\
&= Q_s - \frac{1}{27}\Big[Q_{Sm} + Q_{Se} + Q_{Sw} + Q_{Sn} + Q_{Ss} + Q_{Sne} + Q_{Snw} + Q_{Sse}\\
&+ Q_{Ssw} + 2(Q_m + Q_e + Q_w + Q_n + Q_s + Q_{ne} + Q_{nw} + Q_{se} + Q_{sw})\Big]\\
&= \frac{1}{27}\Big[25Q_s - (Q_s - 2\Delta y\partial_y Q_s) - (Q_s - \Delta x\partial_x Q_s - 2\Delta y\partial_y Q_s)\\
&- (Q_s + \Delta x\partial_x Q_s - 2\Delta y\partial_y Q_s) - (Q_s - \Delta y\partial_y Q_s) - (Q_s - 3\Delta y\partial_y Q_s)\\
&- (Q_s - \Delta x\partial_x Q_s - \Delta y\partial_y Q_s) - (Q_s + \Delta x\partial_x Q_s - \Delta y\partial_y Q_s)\\
&- (Q_s - \Delta x\partial_x Q_s - 3\Delta y\partial_y Q_s) - (Q_s + \Delta x\partial_x Q_s - 3\Delta y\partial_y Q_s)\\
&- 2(Q_s + \Delta y\partial_y Q_s) - 2(Q_s + 2\Delta y\partial_y Q_s) - 2(Q_s + \Delta x\partial_x Q_s + \Delta y\partial_y Q_s)\\
&- 2(Q_s - \Delta x\partial_x Q_s + \Delta y\partial_y Q_s) - 2(Q_s + \Delta x\partial_x Q_s) - 2(Q_s + 2\Delta x\partial_x Q_s)\\
&- 2(Q_s - \Delta x\partial_x Q_s + 2\Delta y\partial_y Q_s) - 2(Q_s + \Delta x\partial_x Q_s + 2\Delta y\partial_y Q_s)\Big]\\
&+ \mathcal{O}(\Delta x^2 + \Delta y^2)\\
&= \mathcal{O}(\Delta x^2 + \Delta y^2)
\end{aligned}
$$

It works similarly for Z_e, Z_w, Z_n, Z_{ne}, Z_{se}, Z_{sw}, Z_{nw}. ∎

We split each component of $Q = (h, U, V)^T$ into its large-scale component $Y = (Y^h, Y^U, Y^V)^T$ and its small-scale component $(Z^h, Z^U, Z^V)^T$. To obtain the scheme on the coarse grid, we write (3.9) on each fine cell $K_m, K_e, K_w, K_n, K_s, K_{ne}, K_{se}, K_{nw}, K_{sw}$ of the coarse cell K_M, and we take the mean value by summing all these equations and dividing by 9. This results in:

$$\begin{aligned}\frac{d}{dt}Y_M(t) = \frac{1}{9\Delta x}\Big[&H^x_{m/w} - H^x_{m/e} + H^x_{n/nw}) - H^x_{n/ne} + H^x_{s/sw} - H^x_{s/se}\\ &+ H^x_{We/w} - H^x_{m/w} + H^x_{nw/Wne} - H^x_{nw/n} + H^x_{sw/Wse} - H^x_{sw/s}\\ &+ H^x_{n/ne} - H^x_{Enw/ne} + H^x_{m/e} - H^x_{e/Ew} - H^x_{se/Esw} + H^x_{s/se}\Big]\\ &+ \frac{1}{9\Delta y}\Big[H^y_{m/n} - H^y_{n/Ns} + H^y_{s/m} - H^y_{n/m} + H^y_{s/Sn} - H^y_{s/m}\\ &+ H^y_{ne/e} - H^y_{ne/Nse} + H^y_{e/se} - H^y_{ne/e} + H^y_{se/Snw} - H^y_{se/e}\\ &+ H^y_{nw/w} - H^y_{nw/Nsw} + H^y_{sw/w} - H^y_{w/nw} + H^y_{sw/Sne} - H^y_{sw/w}\Big] - \Phi(Y_M),\end{aligned} \tag{3.14}$$

which gives after simplifications the following semi-discrete scheme to be applied on the coarse grid.

$$\begin{aligned}\frac{d}{dt}Y_M(t) = \frac{1}{9\Delta x}\Big[&(H^x_{nw/Wne} + H^x_{We/w} + H^x_{sw/Wse}) - (H^x_{Enw/ne} + H^x_{e/Ew}\\ &+ H^x_{se/Esw})\Big] + \frac{1}{9\Delta y}\Big[(H^y_{sw/Sne} + H^y_{s/Sn} + H^y_{se/Snw}) - (H^y_{nw/Nsw}\\ &+ H^y_{n/Ns} + H^y_{ne/Nse})\Big] - \Phi(Y_M).\end{aligned} \tag{3.15}$$

We can thus use different schemes depending on the definition of the fluxes and the resolution on the coarse level can thus be done locally in certain parts of the domain.
During the iterations on the coarse grid, while the large-scale components Y are computed through this scheme, the Z components are frozen.

Remark 3.2. Formula (3.15) being equivalent to (3.9), this process is completely recursive and can be repeated for simulations on three or more levels of grids.

Remark 3.3. For example for a simulation on two levels, as considered here we chose to repeat cycles of the form: 111122221111, where 1 corresponds to the fine grid and 2 to the coarse one. Of course alternate choices of the sequence of levels are possible and, in future work, we intend to develop adaptative procedures for changing the levels. Therefore at the n^{th} iteration, we compute :

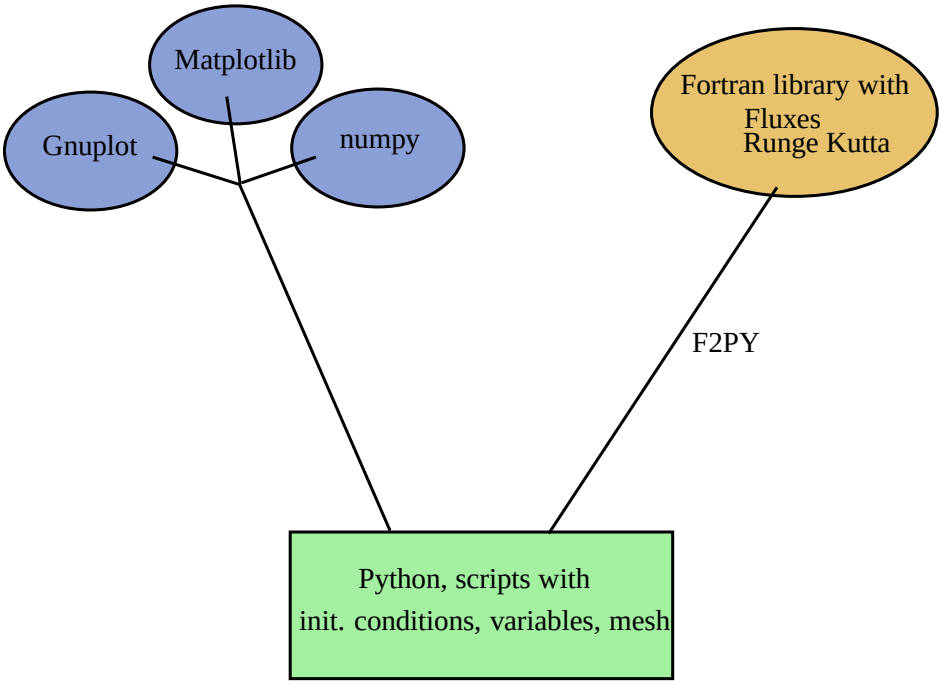
- At level 1 we work on the fine mesh $\mathcal{F}_1$ and compute Q^{n+1} with the classical scheme (described in our case in (3.10) and (3.11) below).
- At level 2
  - we calculate explicitly the fluxes needed by the scheme (3.15),
  - we split Q^n into its large-scale Y^n and small-scale Z^n components,
  - we compute Y^{n+1} with (3.15),
  - we recompose Q^{n+1} from Y^{n+1} and Z^n.

We freeze the small-scale components Z during each iteration at level 2. This induces an error on Z of the order $\Delta t \times$ magnitude of $Z = \Delta t(\Delta x^2 + \Delta y^2)$.

4 Numerics

4.1 The program

The program is made in Python and Fortran. It uses matplotlib and gnuplot to plot the figures, and it also uses numpy for some scientific operations. I have made a Fortran library to do the computation of the fluxes and the iteration of Runge Kutta which are the longest of the computation, that is why I need a fast programming language for this. Then I called these functions in Python using F2PY. This way I use Python to declare all the initials variable, creating the mesh and plotting the solution, which is easier than doing it in Fortran.



4.2 Validations

4.2.1 Source term

To validate the computation we add a source term $\mathbf{S}$:

$$\frac{\partial Q}{\partial t} + \frac{\partial F(Q)}{\partial x} + \frac{\partial G(Q)}{\partial y} + \Phi(Q) = S(t). \tag{4.1}$$

We approximate the source by its average value as we did for Q, this give us :

$$\frac{d}{dt}Q_m(t) = -\frac{H^x_{m/e}(t) - H^x_{w/m}(t)}{\Delta x} - \frac{H^y_{m/n}(t) - H^y_{s/m}(t)}{\Delta y} + S_m(t) \tag{4.2}$$

Then we solve these new equations for analytic functions such as sinus and co-sinus.

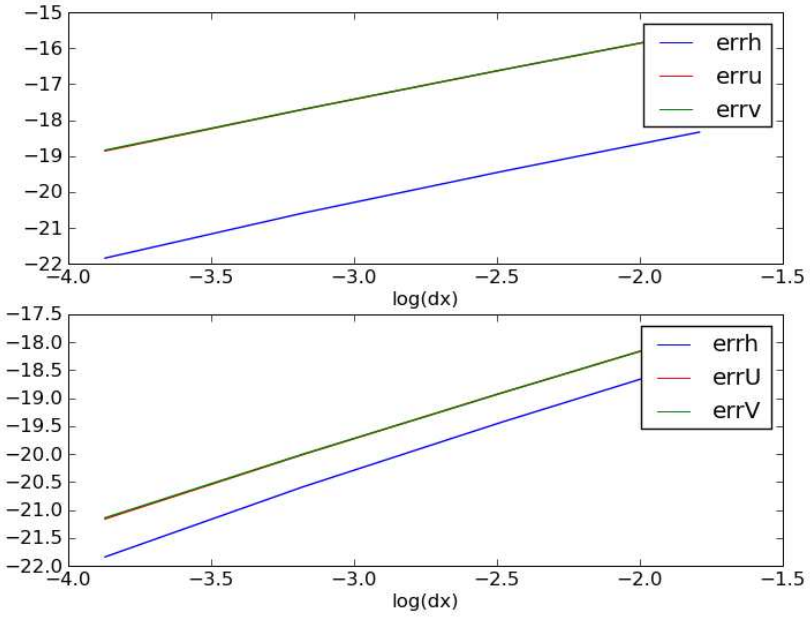
4.2.2 Convergence

To verify the convergence we solve the previous equations for these functions :

$$h(t,x,y) = h_0(1 + \epsilon \sin(\frac{2\pi t}{T}) \cos(\frac{4\pi x}{L}) \sin(\frac{4\pi y}{L})),$$
$$u(t,x,y) = u_0(1 + \epsilon \sin(\frac{2\pi t}{T}) \cos(\frac{4\pi x}{L}) \cos(\frac{4\pi y}{L})),$$
$$v(t,x,y) = u_0(1 + \epsilon \sin(\frac{2\pi t}{T}) \sin(\frac{4\pi x}{L}) \cos(\frac{4\pi y}{L})),$$

The Coriolis force is set to zeros. And the source terms are computed to correspond to these exact solutions. And for the boundary conditions we use the periodic ones, because the Dirichlet conditions which are not constant do not work well with the Runge Kutta methods.

We compute for initial time 0 to final time 0.01 with a step in time of 10^{-4}, then we solve for : 60x60, 120x120, 240x240, 480x480 squares in the mesh. This give us the following errors in norm L^2:



Error in space

We can see that the error in space L^2 is around the order 1.6. For the last two errors, 240x240 squares and 480x480 squares, in L^2 at t=0.01, we have the following slopes :

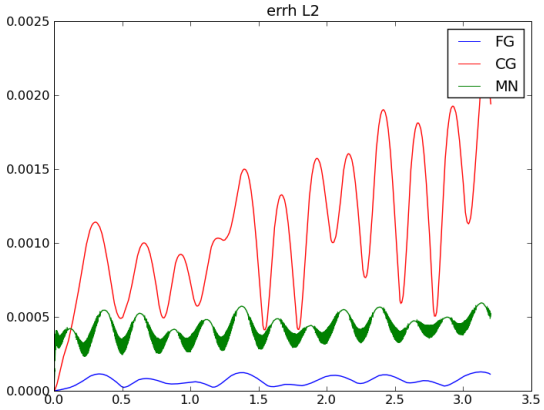
```
error h = 1.80950773091
error u = 1.66281430689
error v = 1.64344686762
error h*u = 1.66283100601
error h*v = 1.64346622874
```

4.2.3 Multilevel

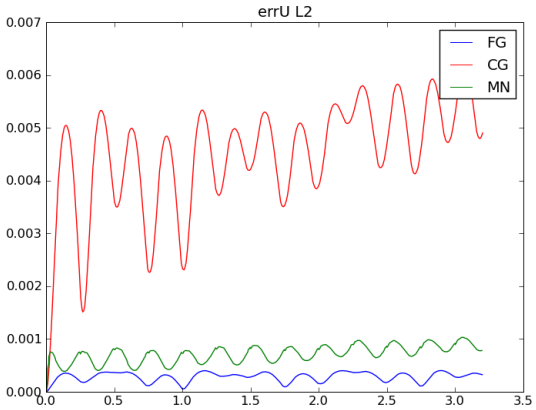
We will validate the global multilevel method, i.e. we use alternatively a fine square mesh and a coarse square mesh. We use the same exact functions introduced in the convergence. We compute for initial time 0 to final time 3.2 with a step in time of 10^{-4}. For the fine mesh we have 300x300 squares in the mesh, for the coarse we have 100x100. For the multilevel we alternate between 300x300 and 100x100, the multilevel method use a cycle of [1,1,1,1,1,2,2,2,2,2,2,1,1,1,1,1] where 1 is a computation of the fine mesh and 2 is a computation on the coarse mesh.

Remark 4.1. *The more time we spend on the coarse grid, the faster the computation will be. However if during one cycle we spend too many iterations on the coarse grid, the method loses its accuracy and it can become less accurate than the one-level computation on the coarse grid.*

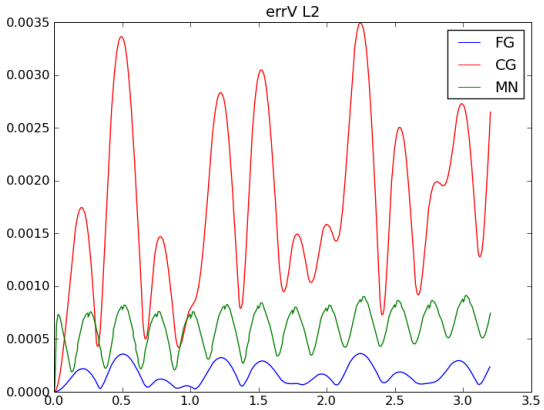
Following are the figures representing the error in L^2 (discrete) along the time :



$$\| h - h_{exact} \|_{L^2}$$



$$\| U - U_{exact} \|_{L^2}$$



$$\| V - V_{exact} \|_{L^2}$$

- FG : Fine Grid, error on the fine mesh
- CG : Coarse Grid, error on the coarse mesh
- MN : Multilevel, error on the multilevel method

We can compare the time elapsed to do the computation for 16 iterations in times :

- On 1-level : 322.07732296
- On 2-levels : 306.576307058 seconds

4.3 Resolution for the Rossby Soliton

For the Rossby Soliton we use the Dirichlet boundary conditions, and we use Runge Kutta of order 4 because the conditions on the boundary are constants. We use the model from *http://marine.rutgers.edu/po/tests/rossby/index.html*. The equations and all quantities being non-dimensional we take $\{\Omega = (-24, 24) \times (-8, 8)\} \times \{0 < t < T\}$, and :

- $g = 1$
- $f_0 = 0$ and $\beta = 1$

Boundary conditions are Dirichlet :

$$\begin{cases} h_{\partial\Omega} &= 1 \\ u_{\partial\Omega} &= 0 \\ u_{\partial\Omega} &= 0 \end{cases} \tag{4.3}$$

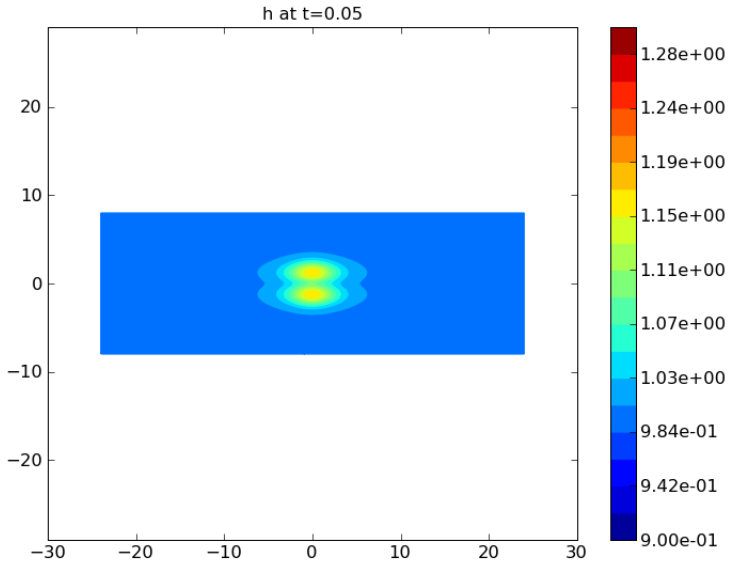
Initial solution :

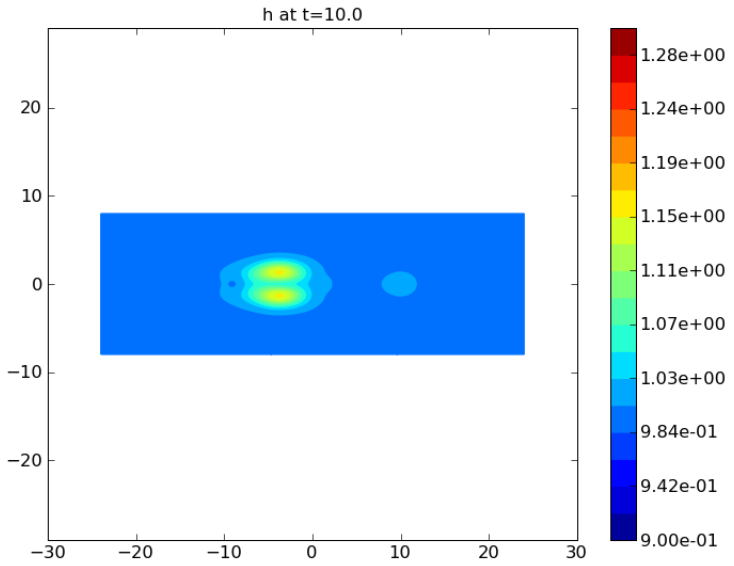
$$\begin{cases} u(x,y,0) = \phi(x)\dfrac{(-9+6y^2)}{4}e^{-\frac{y^2}{2}} \\ v(x,y,0) = \dfrac{\partial\phi(x)}{\partial x}(2y)e^{-\frac{y^2}{2}} \\ h(x,y,0) = \phi(x)\dfrac{(3+6y^2)}{4}e^{-\frac{y^2}{2}} + 1 \end{cases} \tag{4.4}$$

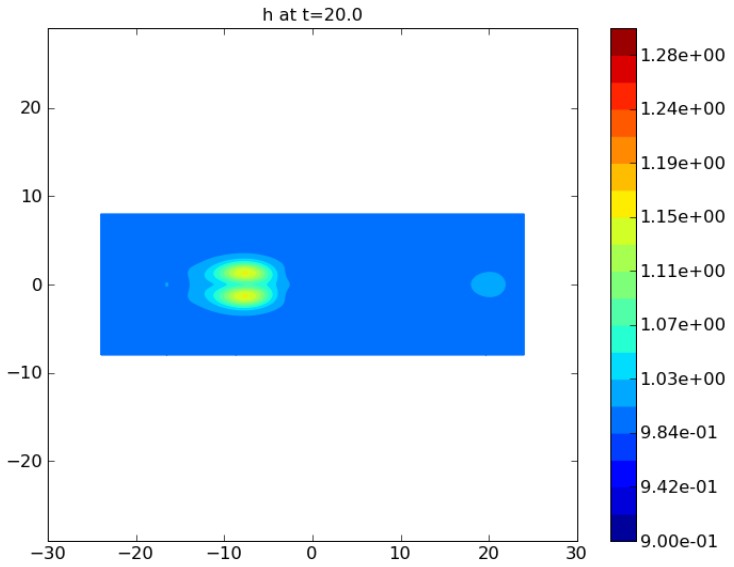
with :

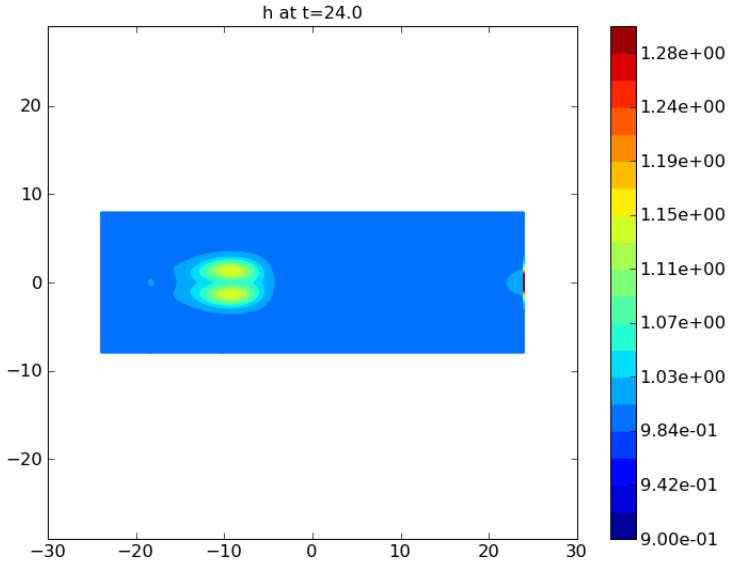
$$\begin{aligned} B &= 0.395 \\ A &= 0.7771B^2 \\ \phi(x) &= Asech^2Bx \\ \frac{\partial\phi(x)}{\partial x} &= -2Btanh(Bx)\phi \end{aligned} \quad .$$

The solution that we obtained by our computation is the following (Equatorial Rossby Soliton) :



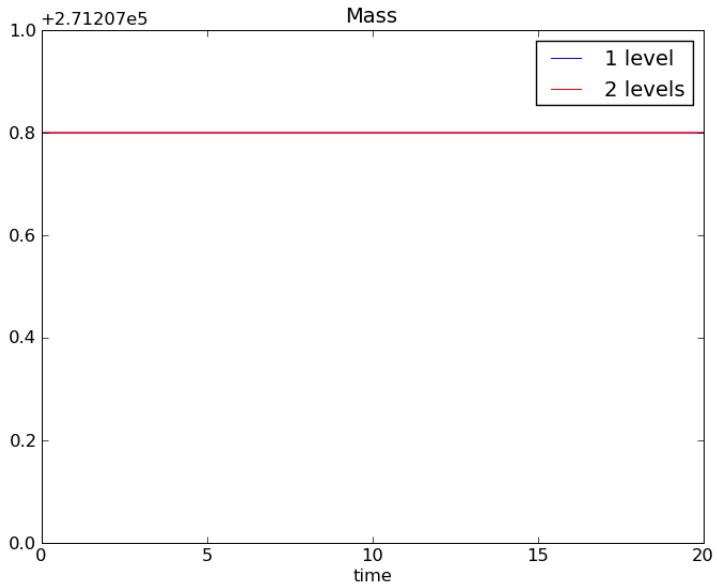






We are going to compare using the method of Runge Kutta 4, a time step of 10^{-3}, a fine mesh of 900x300 squares, and a coarse mesh of 300x100 squares. The multilevel method will use a cycle of [1,1,1,1,1,2,2,2,2,2,2,1,1,1,1,1], where 1 is the computation on the fine mesh and 2 on the coarse mesh.

First the conservation of the mass :



Computation of $\int_{\partial\Omega} h dx dy$

We can see that for both, the computation on the fine mesh and the one on 2-levels (fine mesh /coarse mesh) the mass in constant. Which is validated by :

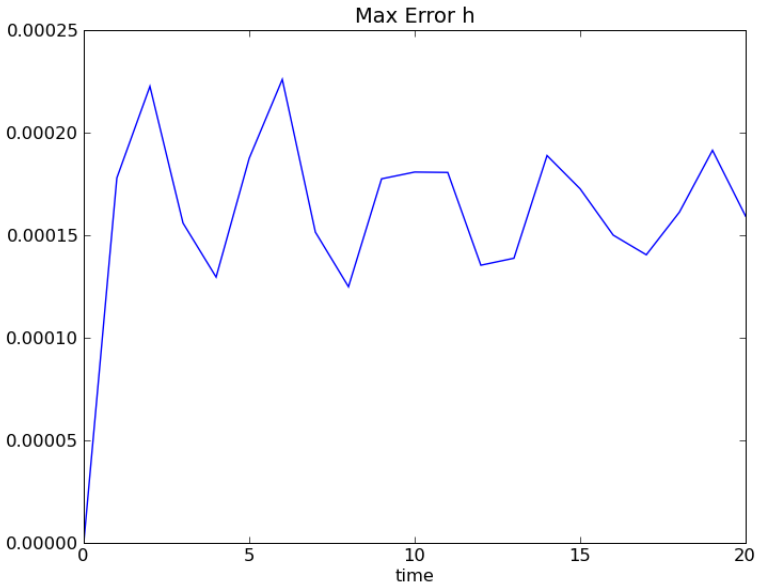
$$m * \frac{\partial}{\partial t} \int_{\partial\Omega} h dx dy = 0$$

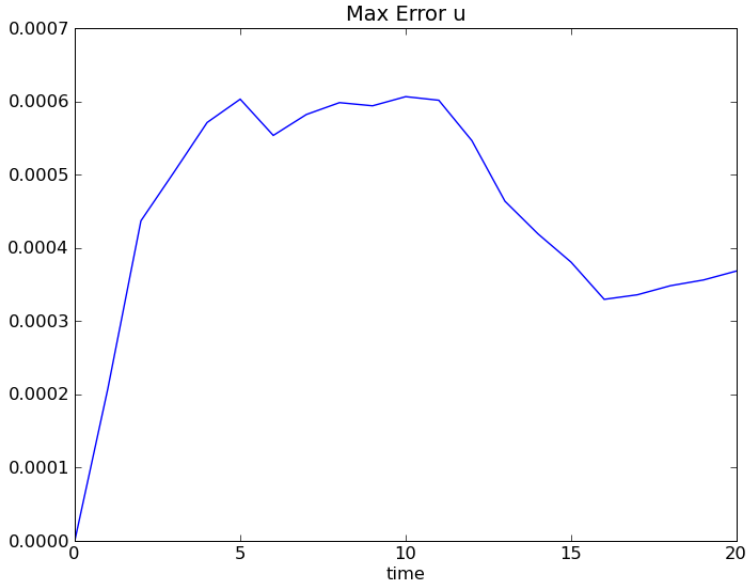
Proof, the fist line of the system give us :

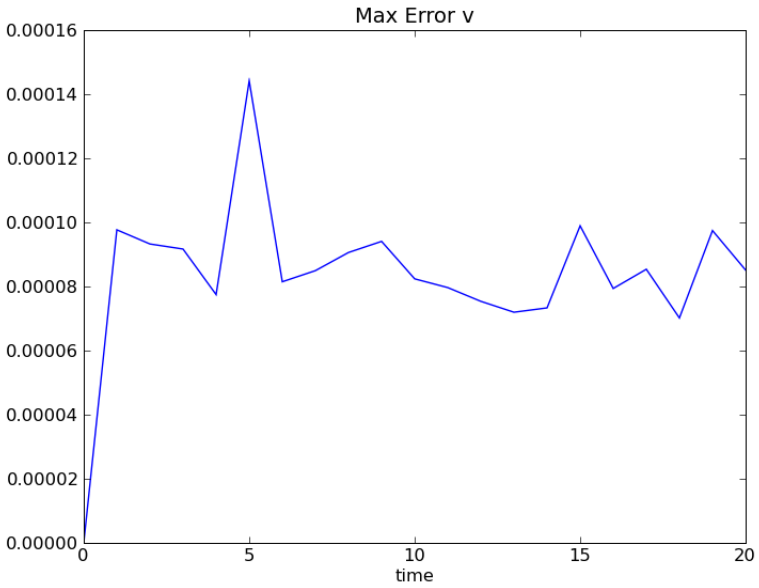
$$\frac{\partial}{\partial t} \int_{\partial\Omega} h dx dy + \frac{\partial}{\partial t} \int_{\partial\Omega} uh dx dy + \frac{\partial}{\partial t} \int_{\partial\Omega} vh dx dy = 0,$$

because u and v = 0 on the boundaries.

Then we compare the difference maximal between the h,u and v of the two computations (one on the coarse and one using multilevel method).


Max Error h
0.00025
0.00020
0.00015
0.00010
0.00005
0.00000
0
5
10
15
20
time


Max Error u
0.0007
0.0006
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
0
5
10
15
20
time



We can see that h,u,v are the same at 10^{-3} until the wave touch the boundary on x=0, where there is a condition of Dirichlet.

For the time saving, we compare the 1-level method on the fine mesh and the one using 2-levels for 16 iterations in time which are the length of a cycle for the multilevel one. It takes 55 seconds for the 1-level method and 40 seconds for the 2-levels method. We save 15 seconds for only 16 iterations in time. This means we gain 16.6% of CPU time.

References

[1] K.Adamy, S. Faure, J. Lamini, R. Temam,
A multilevel method for finite volume discretization of the two dimensional nonlinear Shallow-Water equations, 2008.

[2] R. J. leveque,
Finite Volume Methods for Hyperbolic Problems, *Cambridge*.

[3] B. Cusman-Roisin,
Introduction to Geophysical Fluid Dynamics, *Prentice Hall*.

[4] A. Kurganov, E. Tadmor,
New high-resolution central schemes for nonlinear conservation laws and convection-diffusion equations, *J. Comput. Phys. 160*, 2000.

[5] A. Kurganov, G. Petrova,
A third-order semi-discrete genuinely multidimensionnal central scheme for hyperbolic conservation laws and related problems *Numer. Math. 88*, 2001.

[6] A. Kurganov, S. Noelle, G. Petrova,
Semidiscrete central upwind schemes for hyperbolic conservation laws and Hamilton-Jacobi equations, *SIAM J. Sci. Comput. 23, no3*, 2001.

[7] A. Kurganov, D. Levy,
Central-upwind schemes for the Saint-Venant system,*M2AN, Vol.36, no3*, 2002.