Project 6: 2D Finite-Volume Solver

Changkun Li

1 Problem Description

1.1 Equation to solve

$$\frac{\partial}{\partial t}\begin{bmatrix} h \\ hu \\ hv \end{bmatrix} + \frac{\partial}{\partial x}\begin{bmatrix} hu \\ hu^2 + gh^2/2 \\ hvu \end{bmatrix} + \frac{\partial}{\partial y}\begin{bmatrix} hv \\ huv \\ hv^2 + gh^2/2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

1.2 Initial condition

$$h^0(x,y) = 1.0 + 0.3e^{-50(x-1.5)^2 - 50(y-0.7)^2}, \quad \vec{v}^0(x,y) = \vec{0}$$

1.3 Mesh

city0.gri, city1.gri, city2.gri from previous project.

1.4 Boundary Condition

Wall boundary condition should be used for all boundary groups, boundary flux is just continuous flux dotted with the normal vector, computed with the boundary state (zero normal velocity).

1.5 Time Stepping

Forward Euler will be used.

$$\Delta t = \min_i \left(\frac{2A_i \,\mathrm{CFL}}{\sum_{e=1}^{3} |s|_{i,e} \Delta l_{i,e}} \right), \quad \mathbf{u}_i^{n+1} = \mathbf{u}_i^n - \frac{\Delta t}{A_i} \mathbf{R}_i^n$$

1.6 Output Calculation

We are interested in the forces on the building. In the shallow water equations, the force exerted on a building is

$$\vec{F} = F_x \hat{\mathbf{x}} + F_y \hat{\mathbf{y}} = \int_{\text{building}} \frac{1}{2} \rho g h^2 \vec{n} \, dl$$

2 Tasks

2.1 Roe flux and test

2.1.1 Roe state

$$F = \vec{\mathbf{F}} \cdot \vec{n} = n_x \begin{bmatrix} hu \\ hu^2 + gh^2/2 \\ hvu \end{bmatrix} + n_y \begin{bmatrix} hv \\ huv \\ hv^2 + gh^2/2 \end{bmatrix} = \begin{bmatrix} n_x hu + n_y hv \\ (n_x (hu)^2 + n_y (hu)(hv))/h + n_x gh^2/2 \\ (n_x (hu)(hv) + n_y (hv)^2)/h + n_y gh^2/2 \end{bmatrix}$$

$$\frac{\partial F}{\partial \mathbf{u}} = \begin{bmatrix} 0 & n_x & n_y \\ -u(n_x u + n_y v) + n_x gh & 2n_x u + n_y v & n_y u \\ -v(n_x u + n_y v) + n_y gh & n_x v & 2n_y v + n_x u \end{bmatrix}$$

The set of equations we need to solve are given below:

$$\begin{aligned} F_R - F_L &= \begin{bmatrix} n_x(h_R u_R - h_L u_L) + n_y(h_R v_R - h_L v_L) \\ n_x(h_R u_R^2 - h_L u_L^2) + n_y(h_R u_R v_R - h_L u_L v_L) + n_x g(h_R^2 - h_L^2)/2 \\ n_x(h_R u_R v_R - h_L u_L v_L) + n_y(h_R v_R^2 - h_L v_L^2) + n_y g(h_R^2 - h_L^2)/2 \end{bmatrix} \\ &= \frac{\partial F}{\partial \mathbf{u}}(\mathbf{u}_R - \mathbf{u}_L) \\ &= \begin{bmatrix} 0 & n_x & n_y \\ -u(n_x u + n_y v) + n_x gh & 2n_x u + n_y v & n_y u \\ -v(n_x u + n_y v) + n_y gh & n_x v & 2n_y v + n_x u \end{bmatrix} \begin{bmatrix} h_R - h_L \\ h_R u_R - h_L u_L \\ h_R v_R - h_L v_L \end{bmatrix} \\ &= \begin{bmatrix} n_x(h_R u_R - h_L u_L) + n_y(h_R v_R - h_L v_L) \\ \\ n_x(-u^2 h_R + u^2 h_L + 2uh_R u_R - 2uh_L u_L) + n_x gh(h_R - h_L) + \dots \\ n_y(-uvh_R + uvh_L + vh_R u_R - vh_L u_L + uh_R v_R - uh_L v_L) \\ \\ n_x(-vuh_R + vuh_L + vh_R u_R - vh_L u_L + uh_R v_R - uh_L v_L) + \dots \\ n_y(-v^2 h_R + v^2 h_L + 2vh_R v_R - 2vh_L v_L) + n_y gh(h_R - h_L) \end{bmatrix} \end{aligned} \tag{1}$$

We can rewrite the above equations as

$$\begin{cases} (1) & h = (h_R + h_L)/2 \\ (2) & -u^2 h_R + u^2 h_L + 2uh_R u_R - 2uh_L u_L = h_R u_R^2 - h_L u_L^2 \\ (3) & -uvh_R + uvh_L + vh_R u_R - vh_L u_L + uh_R v_R - uh_L v_L = h_R u_R v_R - h_L u_L v_L \\ (4) & -v^2 h_R + v^2 h_L + 2vh_R v_R - 2vh_L v_L = h_R v_R^2 - h_L v_L^2 \end{cases} \tag{2}$$

Equation (2) and (4) are quadratic equation of u and v, respectively. The solution to these equations are

$$u = \frac{h_R u_R - h_L u_L \pm \sqrt{h_R h_L}(u_R - u_L)}{h_R - h_L}$$

$$v = \frac{h_R v_R - h_L v_L \pm \sqrt{h_R h_L}(v_R - v_L)}{h_R - h_L}$$

or equivalently

$$u = \frac{\sqrt{h_R}u_R - \sqrt{h_L}u_L}{\sqrt{h_R} - \sqrt{h_L}} \quad \text{or} \quad \frac{\sqrt{h_R}u_R + \sqrt{h_L}u_L}{\sqrt{h_R} + \sqrt{h_L}}$$
$$v = \frac{\sqrt{h_R}v_R - \sqrt{h_L}v_L}{\sqrt{h_R} - \sqrt{h_L}} \quad \text{or} \quad \frac{\sqrt{h_R}v_R + \sqrt{h_L}v_L}{\sqrt{h_R} + \sqrt{h_L}}$$

also notice that u, v need to satisfy constrains (3), by substituting four possible combinations of u, v into (3), we found that only one of them is valid:

$$u = \frac{\sqrt{h_R}u_R + \sqrt{h_L}u_L}{\sqrt{h_R} + \sqrt{h_L}}, \quad v = \frac{\sqrt{h_R}v_R + \sqrt{h_L}v_L}{\sqrt{h_R} + \sqrt{h_L}}$$

alternatively, we can say that because u, v have to be finite when $h_R = h_L$, then the only possible solution is the one shown above.

As a summary, the Roe state of conservational 2D SWEs is

$$h = \frac{h_R + h_L}{2}, \quad u = \frac{\sqrt{h_R}u_R + \sqrt{h_L}u_L}{\sqrt{h_R} + \sqrt{h_L}}, \quad v = \frac{\sqrt{h_R}v_R + \sqrt{h_L}v_L}{\sqrt{h_R} + \sqrt{h_L}}$$

2.1.2 Roe flux

$$\det\left(\frac{\partial F}{\partial \mathbf{u}} - \lambda \mathbf{I}\right) = 0 \quad \Longrightarrow \quad \lambda_1 = \vec{v}\cdot\vec{n} + \sqrt{gh}, \quad \lambda_2 = \vec{v}\cdot\vec{n} - \sqrt{gh}, \quad \lambda_3 = \vec{v}\cdot\vec{n}$$

with the eigenvalues, we can obtain corresponding right eigenvectors $\vec{x} = [x_1, x_2, x_3]^T$. For conciseness, we define $v_n = n_x u + n_y v$. The equations are listed below

$$\begin{cases} n_x x_2 + n_y x_3 = \lambda x_1 \\ (-u v_n + n_x g h)x_1 + (n_x u + v_n)x_2 + n_y u x_3 = \lambda x_2 \\ (-v v_n + n_y g h)x_1 + n_x v x_2 + (n_y v + v_n)x_3 = \lambda x_3 \end{cases} \tag{3}$$

when $\lambda = \lambda_3 = v_n$, we have

$$\begin{cases} x_1 = 0 \\ x_2 = -n_y \\ x_3 = n_x \end{cases}$$

otherwise, we will have

$$\begin{cases} x_2 = \{\lambda u + \frac{n_y g h}{\lambda - v_n}(n_x v - n_y u)\}\frac{x_1}{v_n} \\ x_3 = \{\lambda v - \frac{n_x g h}{\lambda - v_n}(n_x v - n_y u)\}\frac{x_1}{v_n} \end{cases}$$

by setting $x_1 = v_n$, we get that

$$\lambda = \lambda_1: \quad \begin{cases} x_1 = v_n \\ x_2 = v_n(u + n_x\sqrt{gh}) \\ x_3 = v_n(v + n_y\sqrt{gh}) \end{cases}, \quad \lambda = \lambda_2: \quad \begin{cases} x_1 = v_n \\ x_2 = v_n(u - n_x\sqrt{gh}) \\ x_3 = v_n(v - n_y\sqrt{gh}) \end{cases}$$

For the sake of condition of right eigenvector matrix, it's a good practice to scale our eigenvectors. After scaling, we get the following eigenvalue-eigenvector pairs:

$$\lambda_1 = v_n + \sqrt{gh}, \quad \vec{r}_1 = \begin{bmatrix} 1 \\ u + n_x\sqrt{gh} \\ v + n_y\sqrt{gh} \end{bmatrix} = \begin{bmatrix} 1 \\ u + n_x c \\ v + n_y c \end{bmatrix}$$

$$\lambda_2 = v_n - \sqrt{gh}, \quad \vec{r}_2 = \begin{bmatrix} 1 \\ u - n_x\sqrt{gh} \\ v - n_y\sqrt{gh} \end{bmatrix} = \begin{bmatrix} 1 \\ u - n_x c \\ v - n_y c \end{bmatrix}$$

$$\lambda_3 = v_n, \quad \vec{r}_3 = \begin{bmatrix} 0 \\ -n_y\sqrt{gh} \\ n_x\sqrt{gh} \end{bmatrix} = \begin{bmatrix} 0 \\ -n_y c \\ n_x c \end{bmatrix}$$

So the eigenvalue decomposition of flux Jacobian is

$$\begin{aligned} \frac{\partial F}{\partial \mathbf{u}} &= \begin{bmatrix} 1 & 1 & 0 \\ u + n_x c & u - n_x c & -n_y c \\ v + n_y c & v - n_y c & n_x c \end{bmatrix} \begin{bmatrix} v_n + c & 0 & 0 \\ 0 & v_n - c & 0 \\ 0 & 0 & v_n \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 \\ u + n_x c & u - n_x c & -n_y c \\ v + n_y c & v - n_y c & n_x c \end{bmatrix}^{-1} \\ &= \begin{bmatrix} 1 & 1 & 0 \\ u + n_x c & u - n_x c & -n_y c \\ v + n_y c & v - n_y c & n_x c \end{bmatrix} \begin{bmatrix} v_n + c & 0 & 0 \\ 0 & v_n - c & 0 \\ 0 & 0 & v_n \end{bmatrix} \begin{bmatrix} \frac{c - v_n}{2c} & \frac{n_x}{2c} & \frac{n_y}{2c} \\ \frac{c + v_n}{2c} & -\frac{n_x}{2c} & -\frac{n_y}{2c} \\ \frac{n_y u - n_x v}{c} & -\frac{n_y}{c} & \frac{n_x}{c} \end{bmatrix} \end{aligned} \tag{4}$$

The stabilization term in Roe flux is

$$\begin{aligned} -\frac{1}{2}\left|\frac{\partial F}{\partial \mathbf{u}}\right|_{\text{Roe}}(\mathbf{u}_R - \mathbf{u}_L) &= -\frac{1}{2} \begin{bmatrix} 1 & 1 & 0 \\ u + n_x c & u - n_x c & -n_y c \\ v + n_y c & v - n_y c & n_x c \end{bmatrix} \begin{bmatrix} |v_n + c| & 0 & 0 \\ 0 & |v_n - c| & 0 \\ 0 & 0 & |v_n| \end{bmatrix} \begin{bmatrix} \frac{c - v_n}{2c} & \frac{n_x}{2c} & \frac{n_y}{2c} \\ \frac{c + v_n}{2c} & -\frac{n_x}{2c} & -\frac{n_y}{2c} \\ \frac{n_y u - n_x v}{c} & -\frac{n_y}{c} & \frac{n_x}{c} \end{bmatrix} \\ &\quad \times \begin{bmatrix} h_R - h_L \\ h_R u_R - h_L u_L \\ h_R v_R - h_L v_L \end{bmatrix} \\ &= -\frac{1}{2}\Big(C_1|\lambda_1|\vec{r}_1 + C_2|\lambda_2|\vec{r}_2 + C_3|\lambda_3|\vec{r}_3\Big) \end{aligned} \tag{5}$$

where

$$\begin{cases} C_1 = \frac{c - v_n}{2c}\Delta(h) + \frac{n_x}{2c}\Delta(hu) + \frac{n_y}{2c}\Delta(hv) \\ C_2 = \frac{c + v_n}{2c}\Delta(h) - \frac{n_x}{2c}\Delta(hu) - \frac{n_y}{2c}\Delta(hv) \\ C_3 = \frac{n_y u - n_x v}{c}\Delta(h) - \frac{n_y}{c}\Delta(hu) + \frac{n_x}{c}\Delta(hv) \end{cases}$$

Roe flux is then given by

$$\widehat{F}_{\text{Roe}} = \frac{1}{2}(F_L + F_R) - \frac{1}{2}\Big(C_1|\lambda_1|\vec{r}_1 + C_2|\lambda_2|\vec{r}_2 + C_3|\lambda_3|\vec{r}_3\Big)$$

in which $C_i, \lambda_i, \vec{r}_i (i = 1,2,3)$ are determined by Roe state:

$$h = \frac{h_R + h_L}{2}, \quad u = \frac{\sqrt{h_R}u_R + \sqrt{h_L}u_L}{\sqrt{h_R} + \sqrt{h_L}}, \quad v = \frac{\sqrt{h_R}v_R + \sqrt{h_L}v_L}{\sqrt{h_R} + \sqrt{h_L}}$$

2.1.3 Implementation and Test

Suppose that

$$\mathbf{u}_L = \begin{bmatrix} 1/2g \\ 0 \\ 0 \end{bmatrix} \quad , \quad \mathbf{u}_R = \begin{bmatrix} 3/2g \\ 0 \\ 0 \end{bmatrix} \quad , \quad \vec{n} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Roe flux calculated analytically:

$$\mathbf{u} = \begin{bmatrix} 1/g \\ 0 \\ 0 \end{bmatrix}$$

$$\frac{\partial \mathbf{F}}{\partial \mathbf{u}} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \Longrightarrow \Big|\frac{\partial \mathbf{F}}{\partial \mathbf{u}}\Big| = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\widehat{F}_{\text{Roe}} = \frac{1}{2}\Big(\begin{bmatrix} 0 \\ \frac{g}{2}(\frac{1}{2g})^2 + \frac{g}{2}(\frac{3}{2g})^2 \\ 0 \end{bmatrix} - \frac{1}{2}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} 1/g \\ 0 \\ 0 \end{bmatrix}\Big) = \frac{1}{g}\begin{bmatrix} -\frac{1}{2} \\ \frac{5}{8}^2 \\ 0 \end{bmatrix}$$

Similarily, if $\vec{n} = [0,1]^T$, we will have

$$\widehat{F}_{\text{Roe}} = \frac{1}{g}\begin{bmatrix} -\frac{1}{2} \\ 0 \\ \frac{5}{8} \end{bmatrix}$$

The output of my Roe flux test function is listed below:

```
Normal vector pointing towards +x direction:
Roe flux multiplied by g is
 -0.5
0.625
    0
Normal vector pointing towards +y direction:
Roe flux multiplied by g is
 -0.5
    0
0.625
```

2.2 Finite Volume solver and Free-stream test/preservation test

2.2.1 Utility

(1) **read_gri.cpp**: read a '.gri' file, returns a struct **mesh** which contains location of nodes, E2N matrix and boundary information (boundary type, boundary nodes number, etc).

(2) **edgehash.cpp**: identifies interior and boundary edges, and their connectivities, in a triangular mesh given an element-to-node array.

(3) **process_gri.cpp**: read in a '.gri' file, outputs

- I2E: a mapping from interior faces to elements. Each row contains four integers, elemL, faceL, elemR, faceR.
- B2E: a mapping from boundary faces to elements and boundary groups. Each row contains three integers, elem, face, bgroup.
- In: normal vectors for interior faces. Normal direction is from the L to the R element.
- Bn: normal vectors for boundary faces. Normal points outward from the adjacent element.
- Area: the area of each element.

(4) **roe.cpp**: given states in the left and right element, determine the numerical flux from left cell to right cell.

(5) **read_param.cpp**: read inputs from file, those inputs set up (1) CFL, (2) boundary type, (3) how does simulation outputs and (4) simulation time.

2.2.2 Solver

FVsolver.cpp: pseudo code is listed below:

```
1. read and process '.gri' file to get "mesh, I2E, B2E, In, Bn, Area"
2. read in input parameters from input file
3. precompute length of every face (interior & boundary)
4. for (int n=0; n<Nt; n++){

        R = 0; s = 0; Force = 0;
        // loop over interior faces
        for(...){
                F = roe(uL, uR, nx, ny, &smax);
                R_left += F*dl;     R_right -= F*dl;
                s_left += smax*dl; s_right += smax*dl;
        }
```

```
        // loop over boundary faces
        for(...){
                F = Fb; smax = sqrt(g*h);
                R_left += F*dl; s_left += smax*dl;

                // calculate force, i is determined by bgroup number
                Force[i] += F*rho*dl;
        }

        // determine time step (dt) and number of iteration (Nt)
        if (n==0) {calculate dt, Nt}

        // loop over elements, forward Euler
        for(element i){
                Ui -= dt*Ri/Areai;
        }
}
```

2.2.3 Freestream test/ preservation test

I have set up two types of tests

- Full state boudnary: $h = 1, (u, v) =$ some nonzero constants. $dt = 0.00114192\ s$
- Wall boundary: $h = 1, (u, v) = 0,$ $dt = 0.00132701\ s$

The results are shown in Figure [1] - [6]. We can see that residual and forces on all buildings are near machine-precision, that means our solver has successfully passed free-stream test/ preservation test.

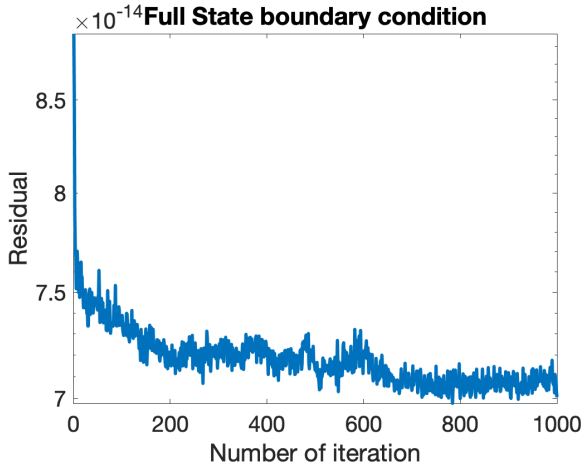

Figure 1:

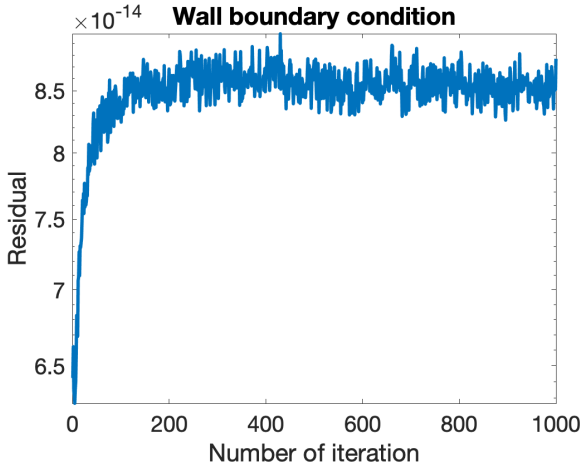

Figure 2:

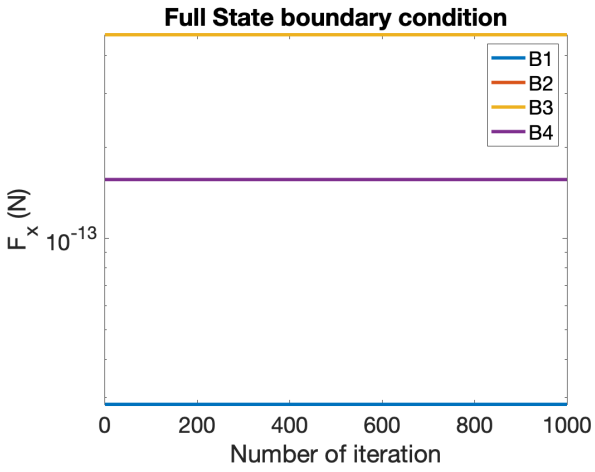

Figure 3:

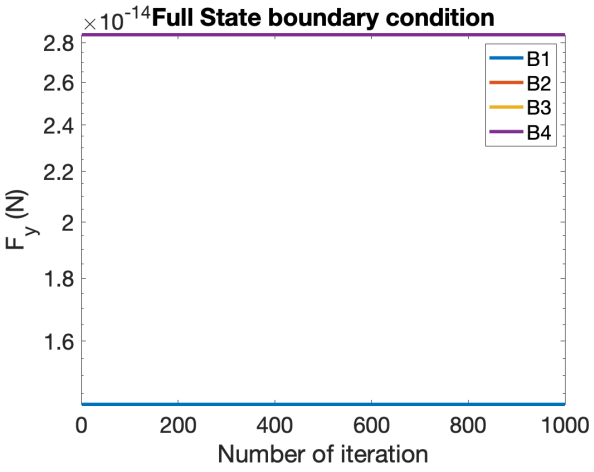

Figure 4:

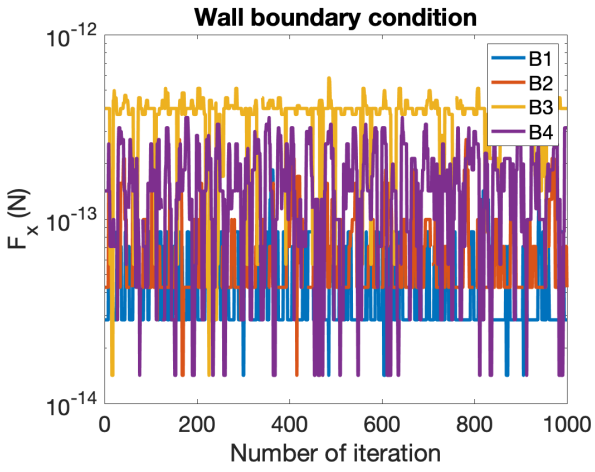

Figure 5:

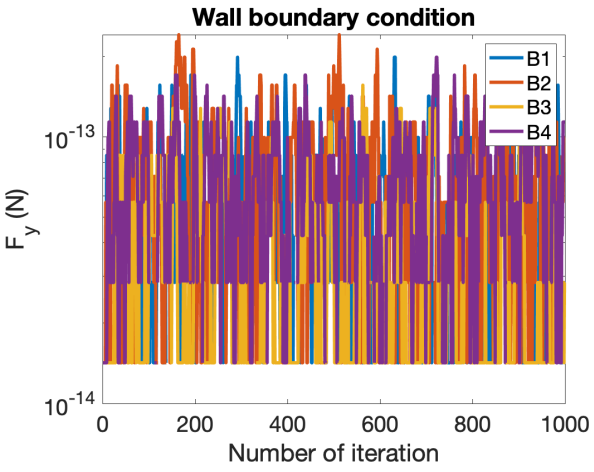

Figure 6:

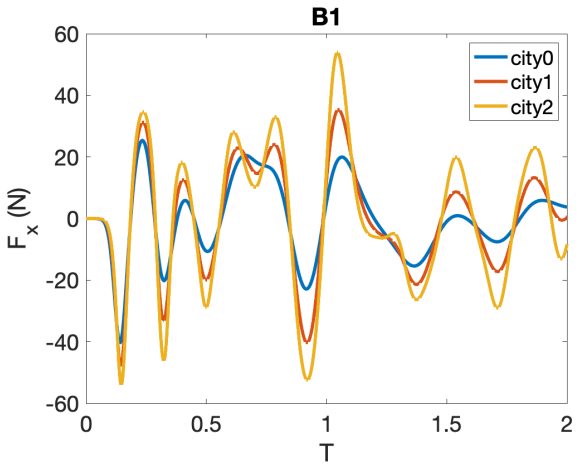

Figure 7:

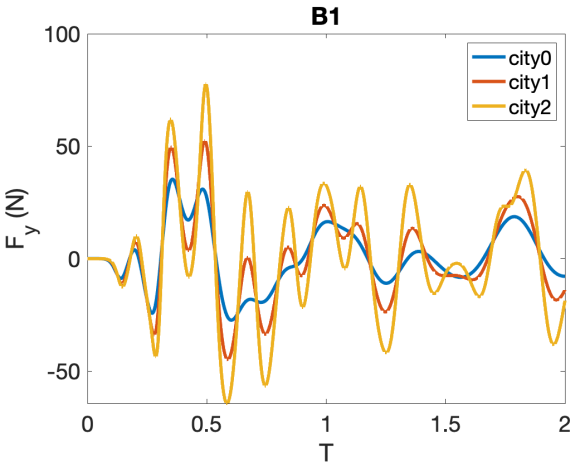

Figure 8:

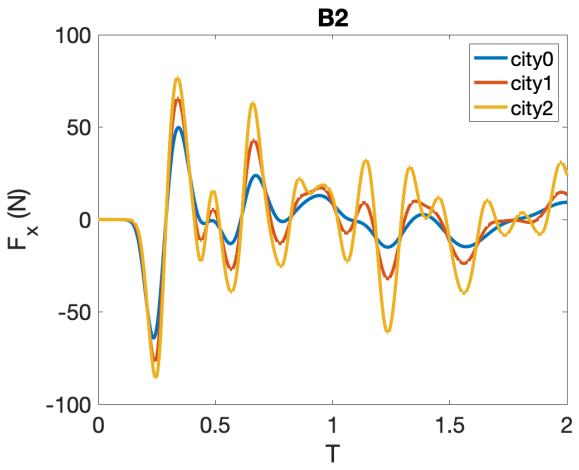

Figure 9:

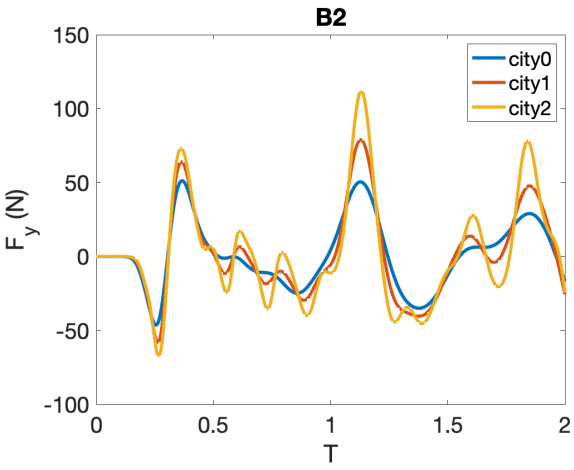

Figure 10:

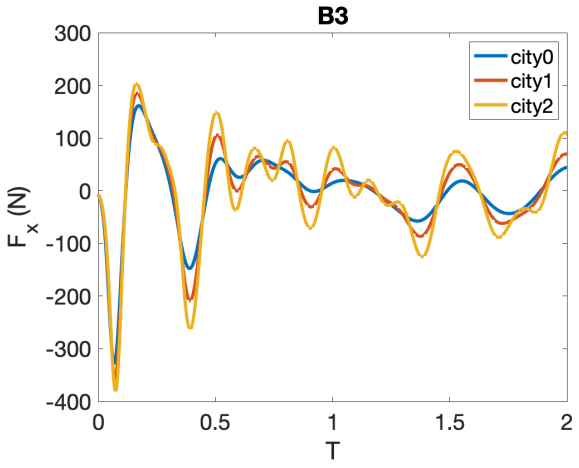

Figure 11:

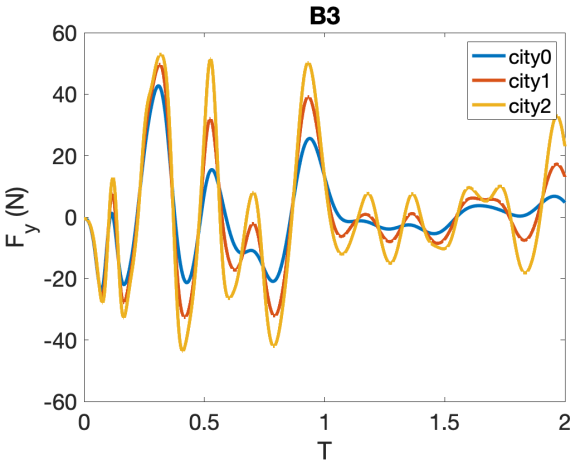

Figure 12:

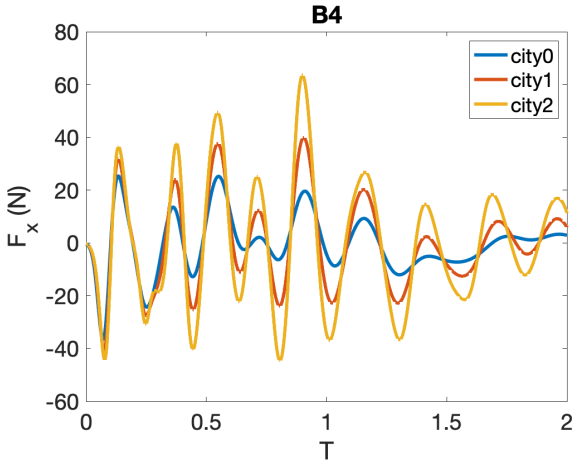

Figure 13:

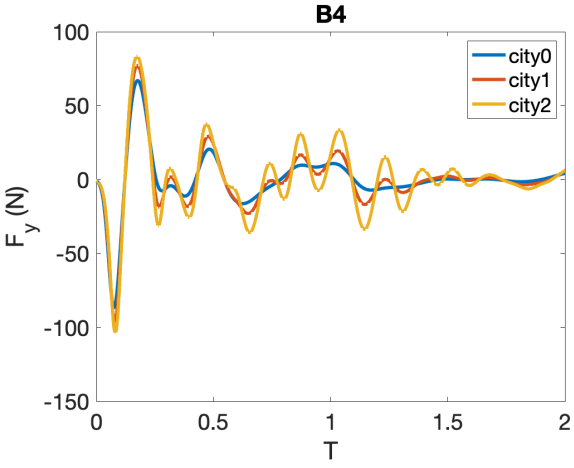

Figure 14:

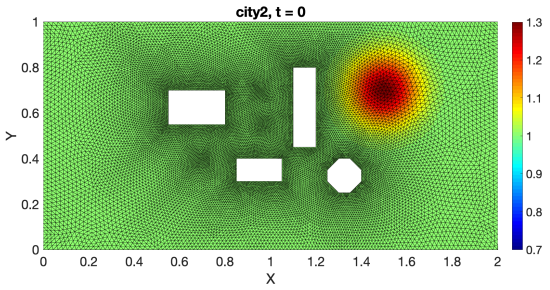

Figure 15:

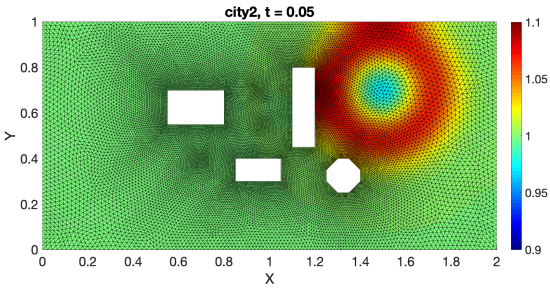

Figure 16:

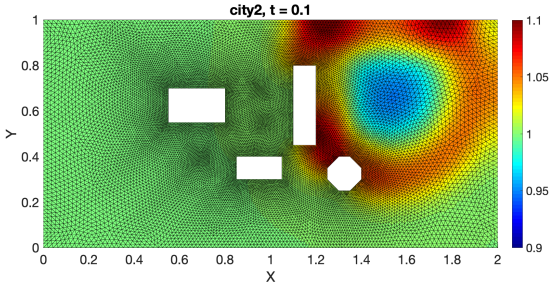

Figure 17:

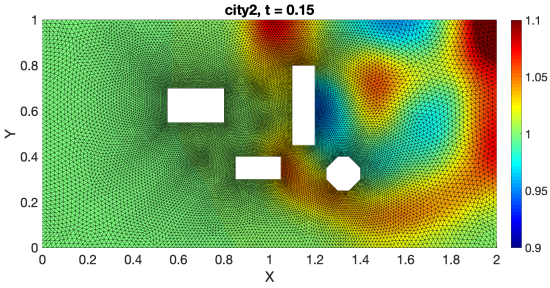

Figure 18:

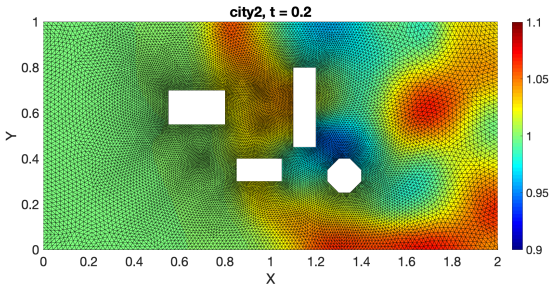

Figure 19:

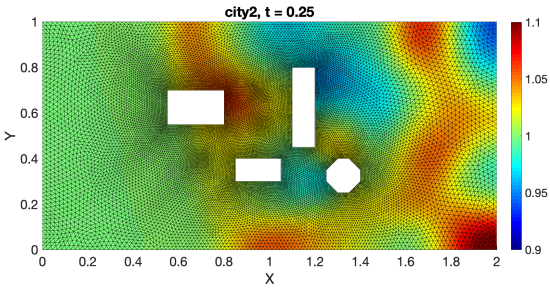

Figure 20:

2.3 Forces on buildings

The results are shown in Figure [7] - [14].

Initial trend of F_x, F_y: because at $t = 0$, the bump is at the upper right corner, so both F_x, F_y should be negative, we notice that in our plots, F_x, F_y on four buildings start from 0, then decrease to some negative values, this is conform to our expectation.

Difference between meshes: we notice that for the first 0.1 or 0.2 second, all three meshes look similar. However, in later times, higher resolution mesh tends to have peaks of larger amplitude, that's what we expect because finer mesh has less numerical diffusion such that larger gradient in simulation can be preserved, and larger gradient of water height leads to stronger forces on buildings.

2.4 Water height contour

The results are shown in Figure [15] - [20].

I think the results are consistent with my expectation

- t = 0.00 $\sim$ 0.10 s, we can see the expansion of initial high water level (peak) and a pit/hole left behind the wavefront ('crater' in the center).
- t = 0.10 $\sim$ 0.15 s, we observe the wave bouncing back off the surfaces of B3, B4 and outer boundary, and the returning waves meet at somewhere close to (1.5, 0.7) to form another peak.
- t = 0.15 $\sim$ 0.25 s, we can still see the propogation and reflection of water waves.