Department of Aeronautics, Imperial College London

Computing assignment : AERO70008 Computational Fluid Dynamics

Direct Numerical Simulation of a 2D heat exchanger based on circular cylinders

FIGURE 1 – *Heat exchanger with a circular cylinder arrangement. (copyright ©ONERA. All rights reserved.)*

Objective

The objective of this course work assignment is to use and to modify a finite-difference code written in modern Fortran for the solution of the 2D compressible Navier-Stokes equations. The flow configuration to be studied is a proxy of a heat-exchanger based on cylindrical cylinders.

1 The compressible Navier-Stokes equations

For a perfect fluid of constant dynamic viscosity μ and of constant thermal conductivity λ, the compressible Navier-Stokes equations can be expressed as the following

$$\begin{aligned}
\frac{\partial \rho}{\partial t} &= -\frac{\partial(\rho u)}{\partial x} - \frac{\partial(\rho v)}{\partial y} \\
\frac{\partial(\rho u)}{\partial t} &= -\frac{\partial p}{\partial x} - \frac{\partial\left(\rho u^2\right)}{\partial x} - \frac{\partial(\rho u v)}{\partial y} + \mu\left(\frac{4}{3}\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{1}{3}\frac{\partial^2 v}{\partial x \partial y}\right) \\
\frac{\partial(\rho v)}{\partial t} &= -\frac{\partial p}{\partial y} - \frac{\partial(\rho u v)}{\partial x} - \frac{\partial\left(\rho v^2\right)}{\partial y} + \mu\left(\frac{\partial^2 v}{\partial x^2} + \frac{4}{3}\frac{\partial^2 v}{\partial y^2} + \frac{1}{3}\frac{\partial^2 u}{\partial x \partial y}\right) \\
\frac{\partial(\rho e)}{\partial t} &= -\frac{\partial(\rho e u)}{\partial x} - \frac{\partial(p u)}{\partial x} - \frac{\partial(\rho e v)}{\partial y} - \frac{\partial(p v)}{\partial y} + \lambda\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right)
\end{aligned}$$

$$
\begin{aligned}
& +\mu\, u \left(\frac{4}{3}\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{1}{3}\frac{\partial^2 v}{\partial x \partial y}\right) + \mu\, v \left(\frac{\partial^2 v}{\partial x^2} + \frac{4}{3}\frac{\partial^2 v}{\partial y^2} + \frac{1}{3}\frac{\partial^2 u}{\partial x \partial y}\right) \\
& +2\,\mu \left[\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2\right] - \frac{2}{3}\,\mu \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)^2 + \mu \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)^2 \\
p \quad = \quad & r\rho T \qquad (1)
\end{aligned}
$$

Where e is the total energy with

$$e = C_v T + \frac{1}{2}\left(u^2 + v^2\right) \qquad (2)$$

Introducing $\gamma = C_p/C_v$ and by using $r = C_p - C_v$, we have

$$e = \frac{1}{\gamma - 1}\frac{p}{\rho} + \frac{1}{2}\left(u^2 + v^2\right) \qquad (3)$$

whereas the ideal gas law can be written as

$$p = \rho\ \frac{\gamma - 1}{\gamma}\ C_p T \qquad (4)$$

1.1 Conservative formulation and primary variables

We are dealing here with the conservative form of the compressible Navier-Stokes equations, i.e. we are dealing with ρ, ρu, ρv et ρe. After each time step, it is necessary to update the primary variables u, v, p and T. This is done in the subroutine `state` with

— $u \longleftarrow \dfrac{\rho u}{\rho}$

— $v \longleftarrow \dfrac{\rho v}{\rho}$

— $p \longleftarrow (\gamma - 1)\ \left(\rho e - \dfrac{1}{2}(\rho u\ u + \rho v\ v)\right)$

— $T \longleftarrow \dfrac{\gamma}{\gamma - 1}\ \dfrac{p}{\rho C_p}$

2 Flow configuration

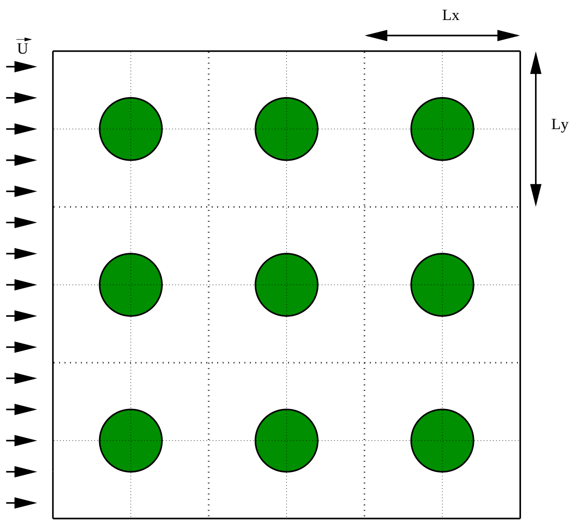

FIGURE 2 – *Schematic view of the flow configuration.*

In this numerical work, we follow the temporal evolution of a uniform flow $\vec{U}$ inside a heat exchanger based on cylindrical cylinders. We consider a spatial domain of size $3L_x \times 3L_y$. The velocity field is $\vec{u}(x,y,t)$ which components are (u,v) in a Cartesian reference frame $R(O,\vec{e}_x,\vec{e}_y)$ of coordinates (x,y), see figure 2.

In order to simplify the problem, we assume that the heat exchanger can be modelled as an infinite array of cylinders. Therefore only the flow around a single cylinder will be studied with the use of periodic boundary conditions in the two spatial directions

$$\vec{u}(x,y,t) = \vec{u}(x+L_x,y,t) \quad , \quad \vec{u}(x,y,t) = \vec{u}(x,y+L_y,t) \quad \forall t > 0 \tag{5}$$

The computation domain where the compressible Navier-Stokes are solved is therefore limited to $(L_x \times L_y)$. This configuration allows a correct reproduction of the dynamics of the wake behind a cylinder of diameter d. Using periodic boundary conditions, it is possible to replicate three times the simulations in the two spatial directions to simulate a heat exchanger with nine cylinders.

3 Parameters of the simulation

3.1 Dimensionless quantities

Concerning the normalisation of the governing equations, we pick by convention ρ_∞, c_∞, d and C_p as primary variables. Basically, when programming the initial condition, each primary variable is equal to 1. Then the numerical values of the other variables can be deduced from the primary variables. The characteristic of the flow could be determined from three dimensionless numbers :

— The Reynolds number $Re = \dfrac{\rho_\infty\ U_0\ d}{\mu}$;

— The Mach number $M = \dfrac{U_0}{c_\infty}$;

— The Prandtl number $Pr = \dfrac{\mu\ C_p}{\lambda}$.

3.2 Initial condition

The initial condition for the velocity field is a uniform flow field

$$\vec{u}(x,y,0) = U_0 \quad \vec{e}_x. \tag{6}$$

For the density and the temperature, we will simply take $\rho(x,y,0) = \rho_\infty$ et $T(x,y,0) = T_\infty$. Note that in order to trigger instabilities a small random noise is added to the velocity field in the initial condition (lines 585-587 of the `2D_compressible.f90` file).

3.3 Size of the domain and physical parameters

We will consider a square computational domain with $L_x \times L_y = 4d \times 4d$, where d is the diameter of the cylinder. The Reynolds number $Re = U_0 d/\nu$ will be set to 200, the Prandtl number $Pr = \mu C_p/\lambda$ to 0.7, the Mach number $M = U_0/c_\infty$ to 0.2 ($U_0 = 0.2c_\infty$)) and the parameter γ will be set to 1.4.

4 Numerical methods

4.1 Mesh

We consider a regular Cartesian mesh (x_i, y_i) to represent the computational domain

$$\begin{array}{rcl} x_i & = & (i-1)\ \Delta x \quad , \quad i = 1,..,n_x \\ y_j & = & (j-1)\ \Delta y \quad , \quad j = 1,..,n_y \end{array} \tag{7}$$

and a succession of consecutive time step t_n

$$t_n \quad = \quad (n-1)\ \Delta t \quad , \quad n = 1,..,n_t \tag{8}$$

Each variable of the flow $\phi(x,y,t)$ will be determined by its values (x_i, y_j) at the time t_n, with

$$\phi^n_{i,j} = \phi(x_i, y_j, t_n) \tag{9}$$

4.2 Second order finite-difference schemes

To solve the equations, we use centred second-order finite-difference schemes. For the first derivative, the schemes are

$$\left.\frac{\partial\phi}{\partial x}\right|_{i,j} = \frac{\phi_{i+1,j}-\phi_{i-1,j}}{2\Delta x} \quad , \quad \left.\frac{\partial\phi}{\partial y}\right|_{i,j} = \frac{\phi_{i,j+1}-\phi_{i,j-1}}{2\Delta y} \tag{10}$$

and for the second derivative

$$\left.\frac{\partial^2\phi}{\partial x^2}\right|_{i,j} = \frac{\phi_{i+1,j}-2\phi_{i,j}+\phi_{i-1,j}}{\Delta x^2} \quad , \quad \left.\frac{\partial^2\phi}{\partial y^2}\right|_{i,j} = \frac{\phi_{i,j+1}-2\phi_{i,j}+\phi_{i,j-1}}{\Delta y^2} \tag{11}$$

4.3 Temporal integration

The time integration can be based on a second-order Adams-Bashforth scheme

$$\frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} \phi\ dt = \frac{3}{2}\ \phi^n - \frac{1}{2}\ \phi^{n-1}, \tag{12}$$

or a third-order Runge-kutta scheme

$$\frac{1}{\Delta t}\int_{t_k}^{t_{k+1}} \phi\ dt = a_k\ \phi^k + b_k\ \phi^{k-1} \tag{13}$$

with three sub-time steps $k = 1,2,3$ with $t_1 = t_n$ et $t_4 = t_{n+1}$ The coefficients a_k et b_k are defined in order to have a third-order scheme and are equal to

$$a_1 = \frac{8}{15} \qquad b_1 = 0 \tag{14}$$

$$a_2 = \frac{5}{12} \qquad b_2 = -\frac{17}{60} \tag{15}$$

$$a_3 = \frac{3}{4} \qquad b_3 = -\frac{5}{12} \tag{16}$$

5 General tips

5.1 Skeleton of the code

The programming language is `Fortran`. It is strongly recommended to develop in the code as clearly as possible with some descriptive comments if possible. The number of mesh points in the x direction is `nx` and `ny` in the y direction. The values ϕ_{ij} from a variable ϕ are represented in a 2D array `phi(nx,ny)`. For example the main 2D arrays are ρ, ρu, ρv, ρe, u, v, p et T.

— Subroutine `param` : Initialisation of the parameters of the simulation such as the Reynolds number, the size of the domain or the primary variables.

— Subroutine `initl` : Initialisation of the velocity, pressure, temperature at $t = 0$ as seen in section (3.2). Note that to take into account the cylinder inside the computational domain, a direct forcing approach is used where the velocity field is frozen to zero. The idea is to impose a zero velocity field inside the cylinder and at the wall of the cylinder. This is done via an extra array `eps` which is equal to 1 in the immersed solid and to 0 in the fluid. Finally, a very small perturbation is imposed on the velocity field v in order to trigger instabilities.

— Subroutine `fluxx` : Dedicated to the computation of the right hand side of equations (1).

— Subroutine `adams` : Dedicated to the time advancement using a second-order Adams-Bashforth scheme.

— Subroutine `rkutta` : Dedicated to the time advancement using a third-order Runge-kutta scheme.

— Subroutine `state` : This subroutine is updating at each time step the velocity, pressure and temperature.

— Subroutines `derx`, `derxx`, `dery` and `deryy` : Dedicated to the computation of the first and second derivatives in the two spatial directions using centred second-order schemes.

5.2 Visualisation of the data

The visualisation of the data is an important part of the project. You are free to use the visualisation software of your choice. However, it is recommended to use `gnuplot` or `python` which are free to use. The `plot_gnu` file will help you to generated `png` figures with `gnuplot` whereas the `plot_py` file will help you to generated `png` figures with `python` to be included in your report. Feel free to modify these files if you want to change the `png` figures. The code is generating vorticity snapshots every `imodulo=2500` time steps. The files generated are called `vort0000`, `vort0001`, `vort0002`, etc. The value of `imodulo` can be changed (see line 34 of the `2D_compressible.f90` file).

5.3 How to compile the code and generate the figures

First do not forget to make a backup copy of the code, just in case.

— To compile the code : `gfortran -O3 2D_compressible.f90` (replace `gfortran` with your Fortran compiler).
— It will generate an executable file called `a.out`.
— If you are using `gnuplot` to generate the pgn files for the figures : `gnuplot plot_gnu`.
— If you are using `python` to generate the pgn files for the figures : `python3 plot_py`.

6 Tasks

1. **Run a first simulation** with $Re = 200$ (line 614 of the `2D_compressible.f90` file) and $n_x \times n_y = 129 \times 129$ for $nt = 10000$ time steps with a second order Adams-Bashforth scheme (`itemp=1`) and a CFL equal to 0.25 (line 47 of the `2D_compressible.f90` file). **Generate 4 visualisations of the vorticity field for** $nt = 2500, 5000, 7500, 10000$ **and put them in your report**. **Briefly comment on the results in less than 100 words**. (Note : no changes in the code are needed for this question). [5%]
2. We want to increase the CFL number in order to reduce the cost of the simulation. Using the same second order Adams-Bashforth scheme (`itemp=1`), is it possible to run a simulation with a CFL of 0.75 ? If yes, **generate 4 visualisations of the vorticity field for** $nt = 2500, 5000, 7500, 10000$**, put them in your report** and **briefly comment on the results in less than 100 words**. If not, **try to explain why in less than 100 words**. [10%]
3. **Complete the subroutine `rkutta`** in which a third-order Runge-Kutta scheme will be used for the time advancement (use the skeleton provided in the code). The scheme is described in section 4.3. **Run a simulation** with the same parameters as in the first simulation but with a CFL=0.75 and the newly implemented third-order Runge-Kutta scheme (`itemp=2`, line 38 of the `2D_compressible.f90` file)). **Generate 4 visualisations of the vorticity field for** $nt = 2500, 5000, 7500, 10000$ **and put them in your report. Briefly comment on the results in less than 100 words and copy-paste the subroutine `rkutta` in your report**. [15%]
4. We want to use centered fourth-order schemes for the first and second derivatives in the two spatial directions. **Write four new subroutines**, using the skeletons provided in the code : `derix4`, `deriy4`, `derxx4` and `deryy4`. In the subroutine `fluxx`, replace the second-order first and second derivatives with the new fourth-order ones. Then run a simulation with same parameters as the first simulation (question 1) but with the new the fourth-order schemes. **Generate 4 visualisations of the vorticity field for** $nt = 2500, 5000, 7500, 10000$ **and put them in your report. Briefly comment on the results and discuss the differences (if any) with the first simulation (question 1) in less than 100 words**. **Copy-paste the four new subroutines in your report**. For this question, make sure to follow standard modern Fortran implementation. [15%]
5. What happens to the flow if you run the simulation with the setup in question 1 for a very long time ? **Justify your answer in less than 100 words**. [10%]
6. By making as less changes as possible in the code, run a simulation in the set up of question 1 but with the flow going from right to left instead of going from left to right (basically with the flow going in the opposite direction). **Generate 4 visualisations of the vorticity field for** $nt = 2500, 5000, 7500, 10000$ **and put them in the report. Copy-paste the modified code in your report**. For this question, make sure to follow standard modern Fortran implementation. To get full marks, you need to make as less changes as possible. [20%]
7. Modify the code in order to simulate a single cylinder on a domain $L_x \times L_y = 20d \times 12d$, with $n_x \times n_y = 513 \times 257$ mesh nodes and with inflow/outflow boundary conditions in the streamwise direction. You will have to modify the derivative subroutines `derix` and `derxx` to account for the new inlet/outlet boundary

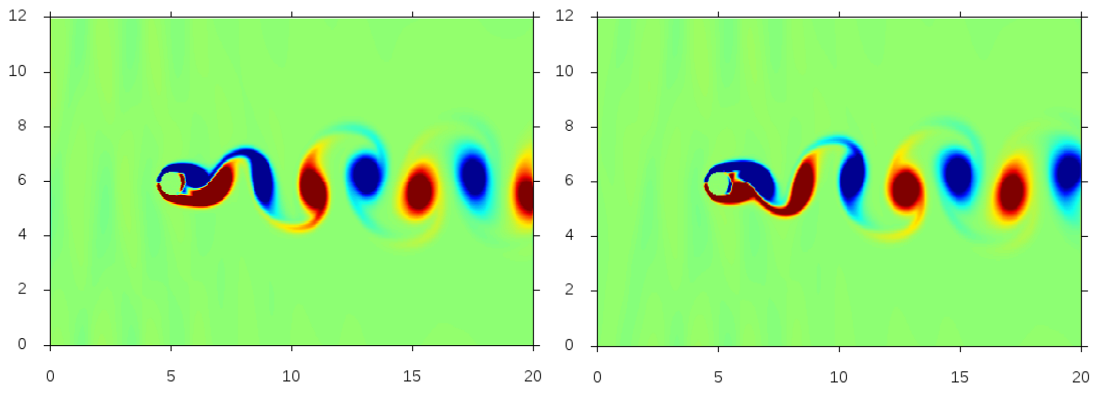

FIGURE 3 – *Instantaneous visualisations of the flow around a single cylinder with inflow/outflow boundary conditions in the streamwise direction.*

conditions (periodic boundary conditions are not relevant anymore in the streamwise direction, the use of non-centred schemes for the first and last points can be an option). You may want to create a new subroutine `boundary` to impose the inlet and outlet boundary conditions in the streamwise direction. Inlet and outlet boundary conditions need to be imposed for *rho*, *rou*, *rov*, *roe*. For the inlet boundary condition, you can look at the subroutine `initl` and use the same profiles for $i = 1$. For the outlet boundary condition, you can use a simple 1D convection equation

$$\frac{\partial \phi}{\partial t} + U_c \frac{\partial \phi}{\partial x} = 0$$

where ϕ is the quantity to be convected at a speed $U_c = U_0$ outside of the computational domain. To solve this equation, you can use an Euler scheme for the time derivative and a first order scheme for the spatial derivative. For this final simulation, you can use the same parameters of the first question : $Re = 200$, CFL=0.25, second order Adams-Bashforth scheme for time advancement, same initial condition. **Use a computational domain $(L_x \times L_y) = (20d \times 12d)$ with the centre of the circular cylinder located at $(5d, 6d)$. The post-processing tools to generate the figures will have to be changed in order to generate two visualisations of the vorticity field once it has reached the end of the computational domain in the streamwise direction (see two examples in figure 3). Make sure that the aspect ratio of your visualisations is correct. Along with the visualisations which needs to be included in the report, copy-paste the new subroutines `derix` and `derxx` as well as the new subroutine `boundary` (or any parts of the code that you have changed for this question).** Screenshots of your code are acceptable for this question. [25%]

7 Comments

— For your report, **please just answer the questions within the word limit** and **do not forget to copy-paste your code developments**.

— You can use up to one figure per question, on top of the flow visualisations, but this is not a requirement (full marks can be awarded without figures but not without the flow visualisations).

— For this project, it is recommended to use `gfortran` which is freely available if you are using a Mac or Linux. Linux is installed on several computers in the Department. If you want to use your laptop with Windows, you might want to use Microsoft Visual Studio and Intel oneAPI Base toolkit, which are integrated development environments from Microsoft and Intel (they will required few Gb of storage). Another option for Windows laptops is to install a virtual machine with Ubuntu (open source operating system on Linux) or to have Ubuntu installed on a USB stick.