GCH8108E
HW5 - Steady state Navier-Stokes problems
March 27, 2024
Pablo Elices Paz

This document is intended to answer the questions posed in the fifth homework of the GCH8108E course.

1 Finite element method for the incompressible Navier Stokes equations

The aim of this document is to solve the steady incompressible Navier Stoke equations:

$$\nabla \cdot \boldsymbol{u} = 0 \tag{1}$$
$$(\boldsymbol{u} \cdot \nabla)\boldsymbol{u} + \nabla p^* - \nu \nabla^2 \boldsymbol{u} = 0 \tag{2}$$

where p^* is the reduced pressure, $\nu = \mu/\rho$ is the kinematic viscosity and the gravity term has been neglected.

Contrary to the Stokes equations, which have been studied in detail in the lectures, these equations are a nonlinear system of equations due to the convective term $(\boldsymbol{u} \cdot \nabla)\boldsymbol{u}$. In this document, the equations are solved using the straightforward solution method.

1.1 Linearization of Navier-Stokes Equations

Here, we will follow the guidelines carried out in [1] and [2] to obtain the linearized system. Note that some parts are paraphrased from [2].

We define the residual function whose root is a solution of the Navier Stokes equations:

$$\boldsymbol{R}(\boldsymbol{u}, p^*) = \begin{pmatrix} \nabla \cdot \boldsymbol{u} \\ (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} + \nabla p^* - \nu \nabla^2 \boldsymbol{u} \end{pmatrix}. \tag{3}$$

Assuming the initial guess is good enough to guarantee the convergence of Newton's iteration and denoting $\boldsymbol{x} = (\boldsymbol{u}, p^*)$, Newton's iteration on a vector function can be defined as:

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - (\nabla \boldsymbol{R}(\boldsymbol{x}^k))^{-1} \boldsymbol{R}(\boldsymbol{x}^k), \tag{4}$$

where $\boldsymbol{x}^{k+1}$ is the approximate solution in step $k+1$, $\boldsymbol{x}^k$ represents the solution from the previous step, and $\boldsymbol{R}(\boldsymbol{x}^k)$ is the Jacobian matrix evaluated at $\boldsymbol{x}^k$.

The Newton iteration formula implies the new solution is obtained by adding an update term to the old solution. Instead of evaluating the Jacobian matrix and taking its inverse, we consider the update term as a whole, that is

$$\boldsymbol{\delta x}^k = -(\nabla \boldsymbol{R}(\boldsymbol{x}^k))^{-1} \boldsymbol{R}(\boldsymbol{x}^k), \tag{5}$$

where $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \boldsymbol{\delta x}^k$.

We can find the update term by solving the system

$$\nabla \boldsymbol{R}(\boldsymbol{x}^k)\boldsymbol{\delta x}^k = -\boldsymbol{R}(\boldsymbol{x}^k) \tag{6}$$

Here, the left of the previous equation represents the directional gradient of $\boldsymbol{R}(\boldsymbol{x}^k)$ along $\boldsymbol{\delta x}^k$ at $\boldsymbol{x}^k$. By definition, the directional gradient is given by

$$\nabla \boldsymbol{R}(\boldsymbol{x}^k)\boldsymbol{\delta x}^k = \lim_{\varepsilon \to 0} \frac{1}{\varepsilon} \left(\boldsymbol{R}(\boldsymbol{u}^k + \varepsilon \boldsymbol{\delta u}^k, p^{*k} + \delta p^{*k}) - \boldsymbol{R}(\boldsymbol{u}^k, p^{*k}) \right) \tag{7}$$

$$= \begin{pmatrix} \nabla(\boldsymbol{\delta u}^k) \\ (\boldsymbol{u}^k \cdot \nabla)\boldsymbol{\delta u}^k + (\boldsymbol{\delta u}^k \cdot \nabla)\boldsymbol{u}^k + \nabla \delta p^{*k} - \nu \nabla^2 \boldsymbol{\delta u}^k \end{pmatrix}. \tag{8}$$

A more detailed description can be found in Appendix A. Therefore, according to Eq. 6,

$$\begin{aligned} \nabla(\boldsymbol{\delta u}^k) &= -\nabla \cdot \boldsymbol{u}^k \\ (\boldsymbol{u}^k \cdot \nabla)\boldsymbol{\delta u}^k + (\boldsymbol{\delta u}^k \cdot \nabla)\boldsymbol{u}^k + \nabla \delta p^{*k} - \nu \nabla^2 \boldsymbol{\delta u}^k &= -(\boldsymbol{u}^k \cdot \nabla)\boldsymbol{u}^k - \nabla p^{*k} + \nu \nabla^2 \boldsymbol{u}^k \end{aligned} \tag{9}$$

where $\boldsymbol{u}^k$ and p^{*k} are the solutions from the previous iteration. Note that the only unknowns are the update terms $\boldsymbol{\delta u}^k$ and δp^{*k}.

1.2 Towards the weak form

We can now multiply our system of equations 9 by a scalar, q, and a vector, $\boldsymbol{q}$, test function and use Galerkin type finite element (the test functions are identical to the interpolation functions):

$$\boldsymbol{\delta u} = \sum_{j=0} \boldsymbol{\delta u_j} \boldsymbol{\phi_j}, \quad \boldsymbol{v} = \boldsymbol{\phi_i} \tag{10}$$

$$\delta p^* = \sum_{k=0} p_k^* \psi_k, \quad q = \psi_i \tag{11}$$

Integrating by parts and considering the boundary conditions in Γ equal to 0

$$\begin{aligned} \sum_{j=0} \boldsymbol{\delta u_j} \int_\Omega \psi_i (\nabla \cdot \boldsymbol{\phi_j}) \mathrm{d}\Omega &= \int_\Omega -\psi_i (\nabla \cdot \boldsymbol{u}) \mathrm{d}\Omega \\ \sum_{j=0} \boldsymbol{\delta u_j} \int_\Omega \left(\boldsymbol{\phi_i} (\boldsymbol{u} \cdot \nabla)\boldsymbol{\phi_j} + \boldsymbol{\phi_i} (\boldsymbol{\phi_j} \cdot \nabla)\boldsymbol{u} + \nu \nabla \boldsymbol{\phi_i}^T : \nabla \boldsymbol{\phi_j} \right) \mathrm{d}\Omega - \sum_{k=0} p_k^* \int_\Omega (\nabla \cdot \boldsymbol{\phi_i}) \psi_k, \mathrm{d}\Omega \\ = -\int_\Omega \boldsymbol{\phi_i} (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} \mathrm{d}\Omega + \int_\Omega (\nabla \cdot \boldsymbol{\phi_i}) p^* \mathrm{d}\Omega - \int_\Omega \nu \nabla \boldsymbol{\phi_i} : \nabla \boldsymbol{u} \mathrm{d}\Omega \end{aligned} \tag{12}$$

A more detailed explanation of how to arrive to these expressions can be found in Appendix B.

The system of equations 12 defines the weak form of the steady incompressible Navier Stoke equations.

1.3 Code implementation

Here, the building of the system matrix as well as the right-hand-term according to system 12 using `deal.ii` is explained. In addition, we will avoid commenting parts of the code already commented in other tasks, so we will only construct the definition of variables `cell_matrix` and `cell_rhs`.

Before starting, note that the system of equation 12 requires knowing, in addition to ψ and $\boldsymbol{\phi}$, the values of p^*, $\boldsymbol{u}$, $\nabla \cdot \boldsymbol{u}$ and $\nabla \boldsymbol{u}$:

```
fe_values[velocities].get_function_values(solution, previous_velocity);
```

```
fe_values[velocities].get_function_gradients(solution,
                                             previous_velocity_gradient);
```

```
for (unsigned int q = 0; q < n_q_points; q++)
{
  previous_velocity_divergence[q] = trace(previous_velocity_gradient[q]);
}
```

```
fe_values[pressure].get_function_values(solution, previous_pressure);
```

Once these variables are known, the test functions are explicitly defined for convenience inside the q and i loops:

```
for (unsigned int k = 0; k < dofs_per_cell; ++k)
  {
    div_phi_u[k]  = fe_values[velocities].divergence(k, q);
    grad_phi_u[k] = fe_values[velocities].gradient(k, q);
    phi_u[k]      = fe_values[velocities].value(k, q);
    phi_p[k]      = fe_values[pressure].value(k, q);
  }
```

In this way, the desired variables can be defined as follows:

- System matrix:

```
cell_matrix(i, j) +=
    (viscosity * scalar_product(grad_phi_u[i], grad_phi_u[j]) +
     phi_u[i] * (previous_velocity_gradient[q] * phi_u[j]) +
     phi_u[i] * (grad_phi_u[j] * previous_velocity[q])
     - div_phi_u[i] * phi_p[j]
     - phi_p[i] * div_phi_u[j]
     ) * fe_values.JxW(q);
```

- Right-hand term:

```
cell_rhs(i) +=
    (-viscosity * scalar_product(grad_phi_u[i],
      previous_velocity_gradient[q]) -
      phi_u[i]*(previous_velocity_gradient[q]*previous_velocity[q]) +
      div_phi_u[i] * previous_pressure[q] +
      phi_p[i] * previous_velocity_divergence[q]
      ) * fe_values.JxW(q);
```

1.4 Code considerations

Before going into the results obtained, a few considerations should be taken into account (see [3], [1]):

- The initial guess needs to be close enough to the solution for Newton's method to converge; hence, finding a good starting value is crucial to the nonlinear solver.

  In all cases studied, we have had no problems with our initial guess, however, this can occur. According with [2], when the viscosity ν is large, a good initial guess can be obtained by solving the Stokes equation with viscosity ν.

- Another problem related to the Newton's method is the step length α^k. In our solver, $\alpha^k = 1$, but according to [4], when the problem is strongly nonlinear, the solution may not converge if we always choose $\alpha^k = 1$. Typical values of the step length are $0 < \alpha^k < 1$. In [4], an iterative method is proposed, changing the step length until the difference between each i residual and the previous residual is reduced.

- The steady incompressible Navier Stoke equations are an example of saddle point problem, which means that the combination of some element type for pressure and some element type for velocity are not LBB (Ladyzhenskaya–Babuška–Brezzi) compatible. I.e, the elements used are Q_n for velocity and Q_{n-1} for pressure.

- As discussed in previous homework assignments, attention must be paid to the boundary conditions, since the unknowns of our system are $\boldsymbol{\delta u}$ and δp^*.

Having said that, the proposed problems can be studied, where the only thing that changes from the solver code are the boundary conditions.

2 Couette flow

Consider two-dimensional incompressible plane viscous flow between plates a distance $2h$ apart (see Fig. 1). The upper plate moves at velocity $\boldsymbol{u} = (1, 0)$ but there is no pressure gradient.

The present problem has an analytical solution (see, e.g. [5]):

$$u = y \quad 0 \leq y \leq 4 \tag{13}$$

$$\frac{\mathrm{d}p^*}{\mathrm{d}x} = 0 \tag{14}$$

Note that the analytical solution is independent of the viscosity of the fluid.

This is Couette flow due to a moving wall: a linear velocity profile with no slip at each wall, as sketched in Fig. 2.

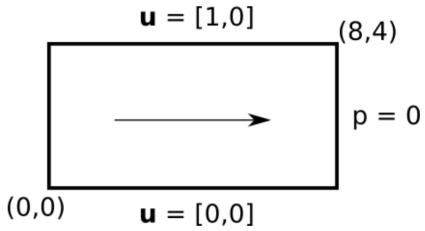


Figure 1: Schematic representing the Couette flow problem.

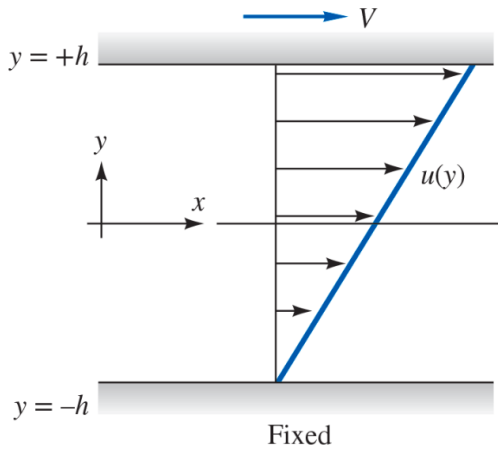


Figure 2: Incompressible viscous flow between parallel plates. No pressure gradient, upper plate moving. Image from [5].

The numerical simulation has been performed using a level of refinement equal to 4 (total number of cells equal to 256, see Fig. 3a). In addition, the tolerance for the Newton's method used is 1e-12. Two simulations have been conducted: the first for Re = 1, and the second for Re = 10, where $Re = 1/\nu$.

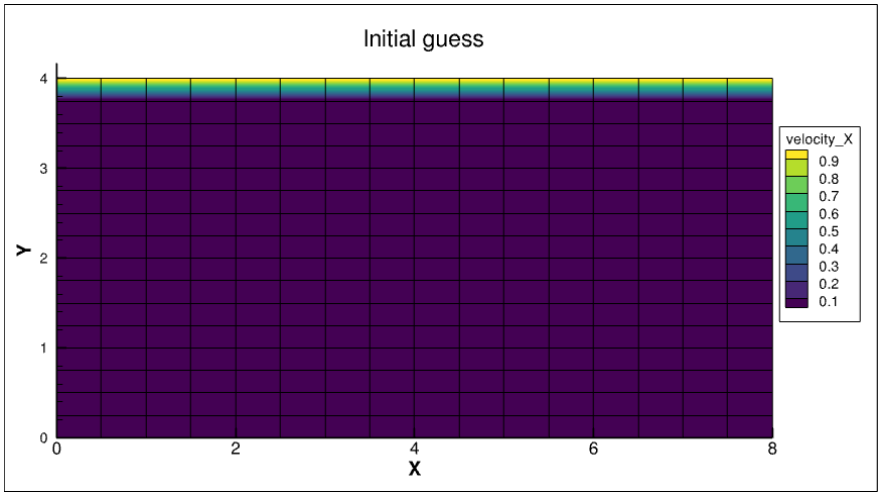
Both simulations converge in 2 iterations:

```
iter: 1, Linfty norm: 0.96875 L2 norm: 18.3214
iter: 2, Linfty norm: 1.24054e-14 L2 norm: 8.34039e-14
```

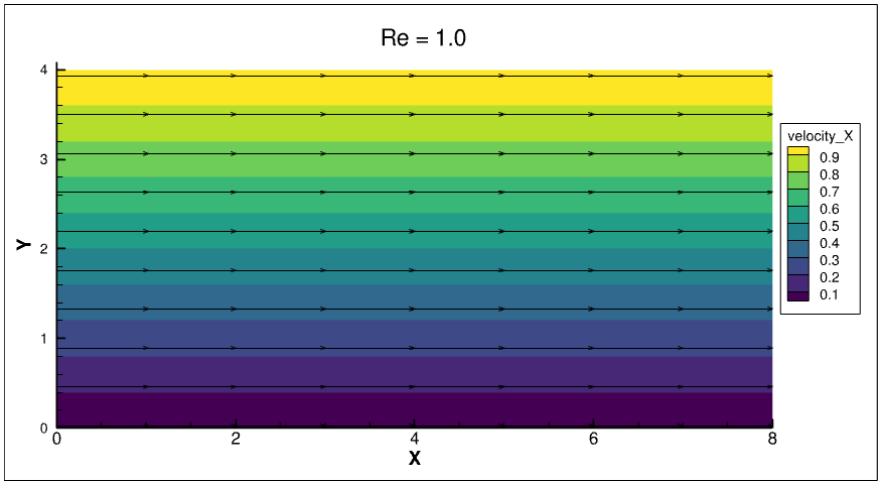
Listing 1: Output for the Couette flow simulation. Re=1

```
iter: 1, Linfty norm: 0.96875 L2 norm: 18.3214
iter: 2, Linfty norm: 5.88073e-15 L2 norm: 1.22168e-13
```

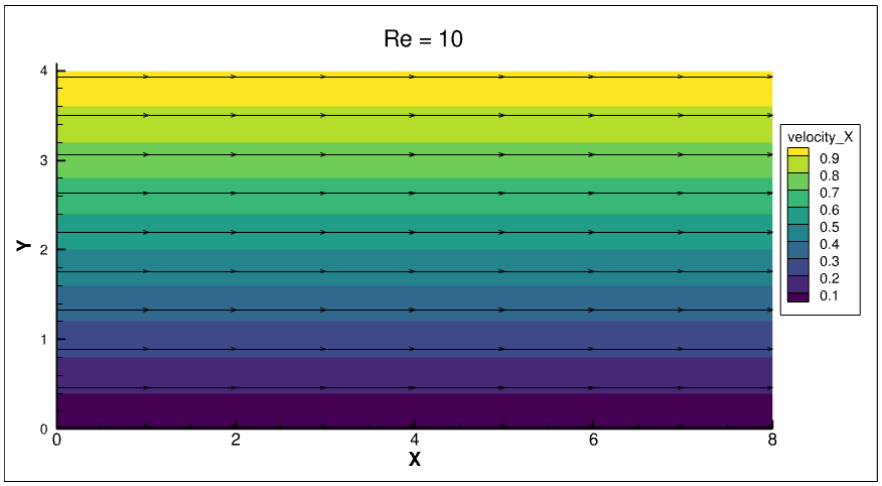
Listing 2: Output for the Couette flow simulation. Re=10



(a) Initial guess



(b) $Re = 1$



(c) $Re = 10$

Figure 3: Numerical solution for the Couette flow. Velocity field and streamlines.

Figure 3a shows the initial guess for both simulations. Figures 3b and 3c show the x-component velocity field for the two Reynolds numbers. The two simulations give as a result the analytical value of the velocity (machine accurate).

Note that Q_2 elements have been used for the velocity when the analytical solution is a linear distribution. This gives very accurate results (machine accurate). Figure 4 shows the velocity profile for Re=1.

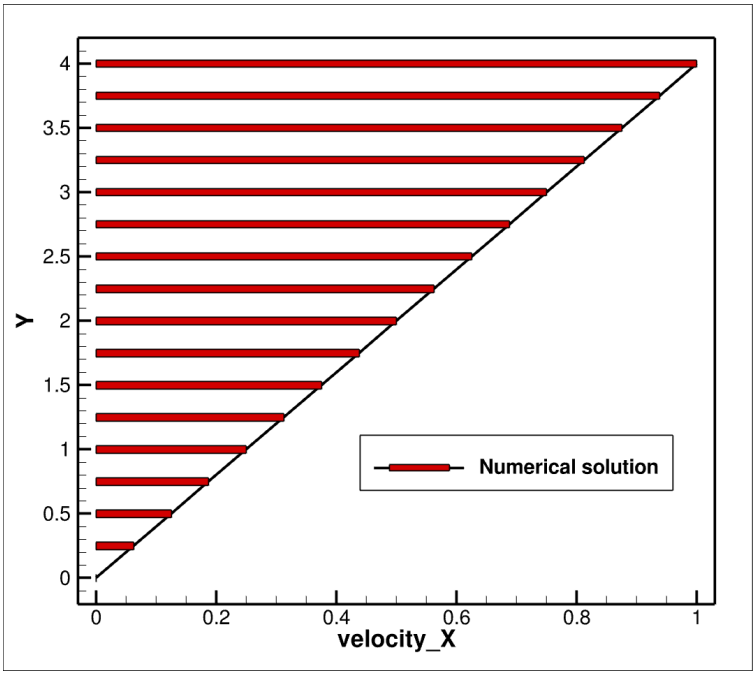
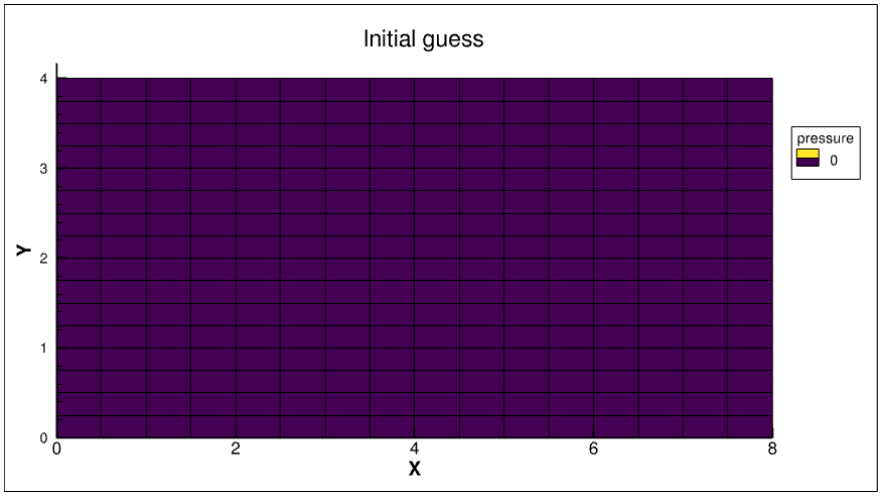
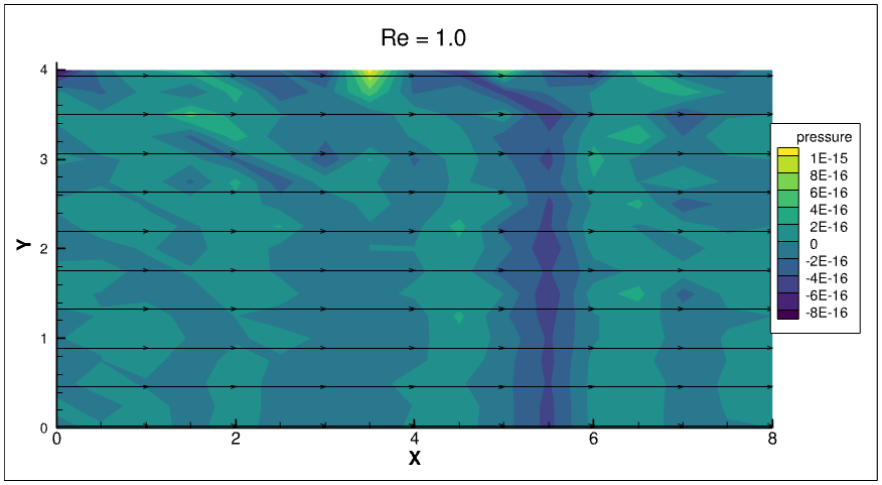


Figure 4: Numerical solution for the Couette flow. Velocity profile at $x = 4$. Re=1.

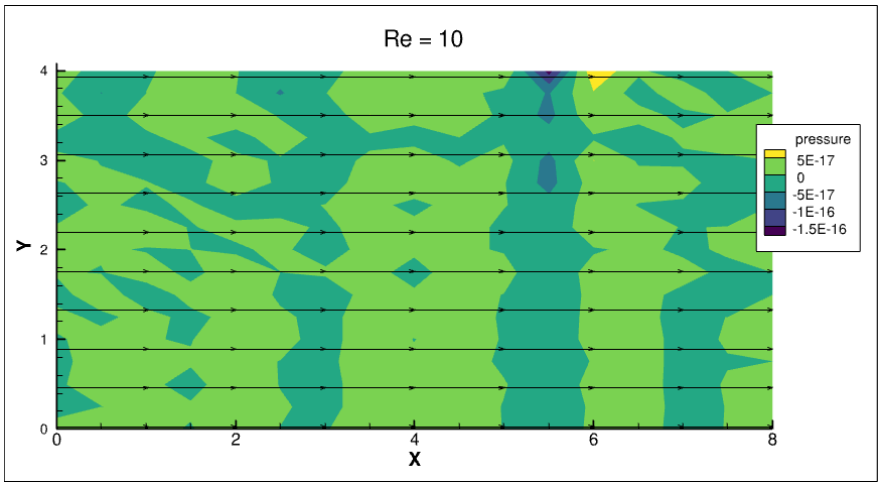
On the other hand, Figure 5a shows the initial guess for pressure field. As in the previous case, Figures 5b and 5c show the pressure field of the simulation. Again a machine-precision solution is obtained.



(a) Initial guess



(b) $Re = 1$



(c) $Re = 10$

Figure 5: Numerical solution for the Couette flow. Pressure field and streamlines.

3 Poiseuille flow

In this case, both plates are fixed ($\boldsymbol{u} = 0$), but the pressure varies in the x direction.

This problem has, as well, an analytical solution (see e.g. [5]) in the fully developed flow, i.e., far enough from inlet:

$$\nu \frac{\mathrm{d}^2 \boldsymbol{u}}{\mathrm{d}y^2} = \frac{\mathrm{d}p^*}{\mathrm{d}x} = \text{const} \tag{15}$$

$$\frac{\mathrm{d}p^*}{\mathrm{d}x} = -\frac{2\nu}{4^2}\frac{1}{u_{max}} = \frac{1}{8\mathrm{Re}}\frac{1}{u_{max}} \tag{16}$$

$$u = u_{max}\left(1 - \frac{y}{4}\right) y \tag{17}$$

Note that in this case, the pressure gradient depends on the Reynolds number.

Taking into account that the flow rate has to be the same at the inlet as in the full developed regime, the value of the pressure gradient as well as that of the velocity as a function of the inlet velocity, $\bar{V}$, can be obtained.

$$Q = \int_0^4 u dy = \frac{2}{3} 4 u_{max} = 4\bar{V} \tag{18}$$

$$u_{max} = \frac{3}{2}\bar{V} = 1.5 \tag{19}$$

$$\frac{\mathrm{d}p^*}{\mathrm{d}x} = \frac{1}{12\mathrm{Re}} \tag{20}$$

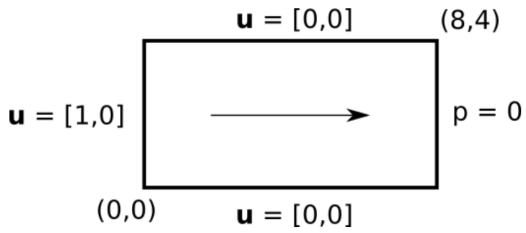


Figure 6: Schematic representing the Poiseuille flow problem.

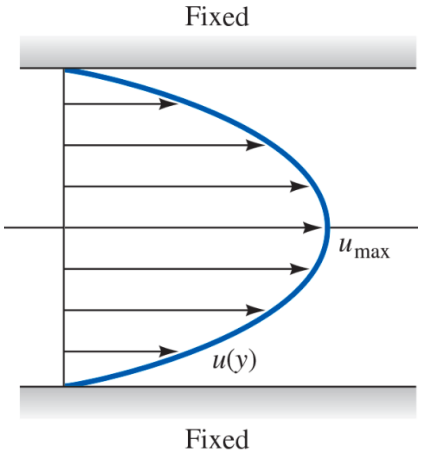


Figure 7: Incompressible viscous flow between parallel plates. Pressure gradient with both plates fixed. Image from [5].

The numerical simulation has been performed again using a level of refinement equal to 4 and with a tolerance for the Newton's method equal to 1e-12. As before, two simulations have been conducted: the first for $\mathrm{Re} = 1$, and the second for $\mathrm{Re} = 10$.

```
iter: 1, Linfty norm: 30.012 L2 norm: 82.7357
iter: 2, Linfty norm: 0.825836 L2 norm: 2.98089
iter: 3, Linfty norm: 0.000797141 L2 norm: 0.0031511
iter: 4, Linfty norm: 6.33917e-10 L2 norm: 1.98743e-09
iter: 5, Linfty norm: 7.80691e-15 L2 norm: 2.36312e-14
```

Listing 3: Output for the Poiseuille flow simulation. Re=1

```
iter: 1, Linfty norm: 2.80291 L2 norm: 35.501
iter: 2, Linfty norm: 0.760693 L2 norm: 4.53719
iter: 3, Linfty norm: 0.0309443 L2 norm: 0.223132
iter: 4, Linfty norm: 5.13018e-05 L2 norm: 0.000223379
iter: 5, Linfty norm: 1.1171e-10 L2 norm: 5.69925e-10
iter: 6, Linfty norm: 1.29817e-15 L2 norm: 5.59996e-15
```

Listing 4: Output for the Poiseuille flow simulation. Re=10

In this case, it requires more iterations to converge than the Couette flow. This is mainly because the initial guess is further away from the solution. In any case, convergence is obtained without problems.

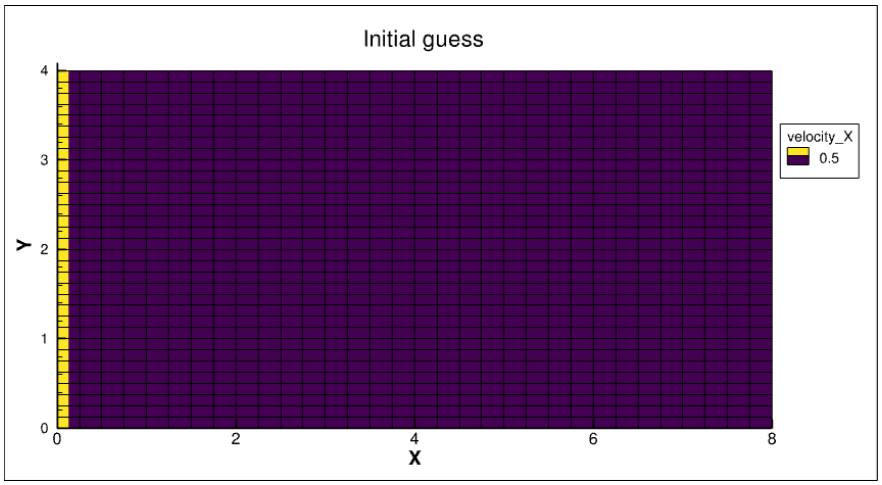
Figure 8 shows the velocity field and Figure 9 the pressure field (initial condition, $Re = 1$ and $Re = 10$). Note that for these figures, a level of refinement equal to 6 has been used. . In general, we can observe that the fully developed flow regime is reached earlier for a small Reynolds value.

Again, the solution is a quadratic function and a mesh convergence does not provide a better solution in the full developed zone. Figure 10 shows the velocity profile at $x = 7.5$ for different mesh sizes. In the figure it can be seen that the three solutions are overlapped. However, the level of refinement increases the maximum pressure value obtained (found in the lower and upper left corners) as shown in Figure 11.

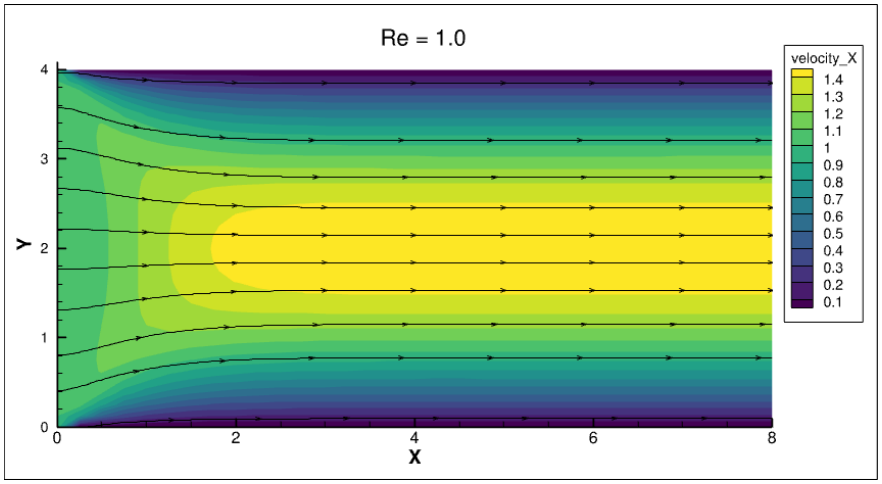
Reynolds	Re=1		Re=10	
Mesh level of refinement	4	6	4	6
Maximum pressure	29.2341	105.086	3.42929	11.0089

Table 1: Maximum pressure values. Poiseuille flow.

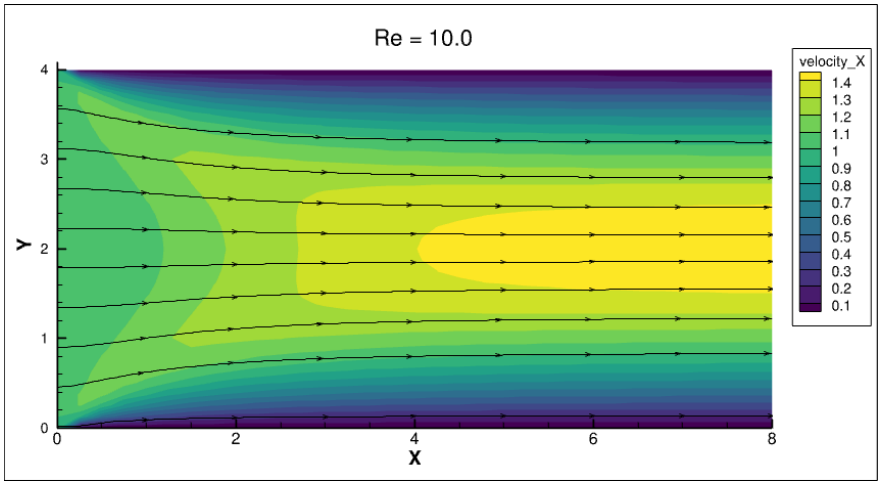
However, this is a punctual problem that, a priori, does not affect the rest of the solution.



(a) Initial guess

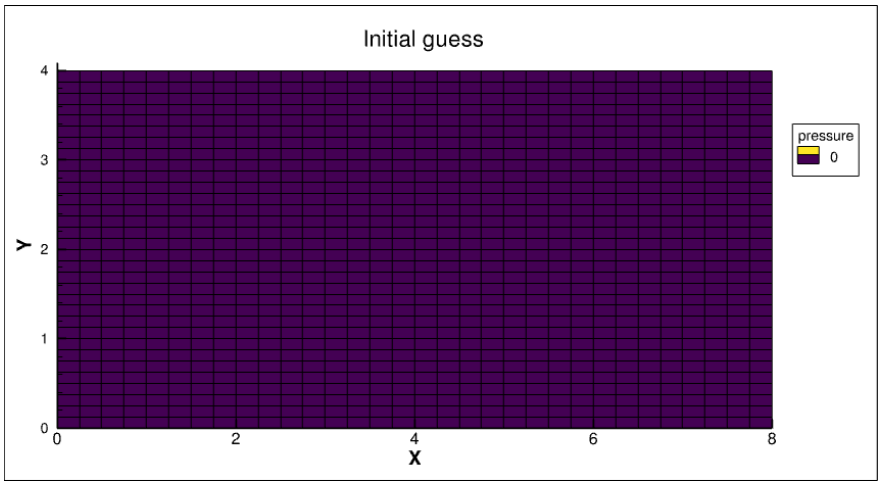


(b) $Re = 1$

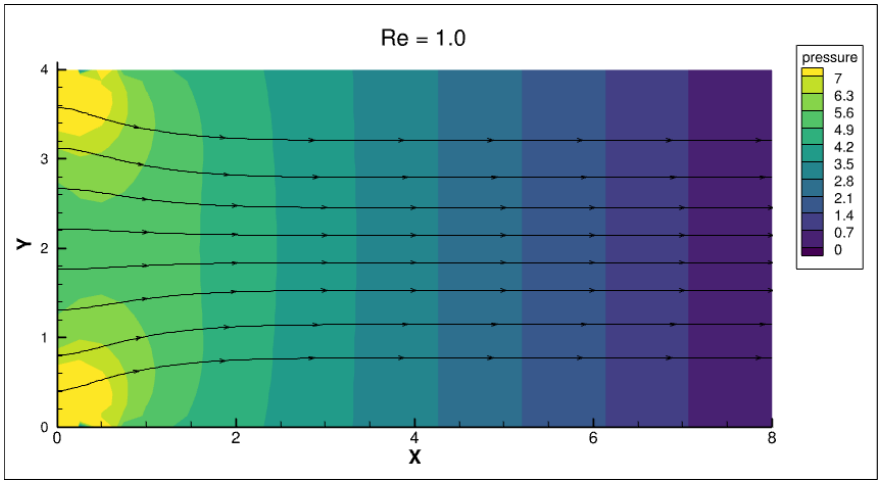


(c) $Re = 10$

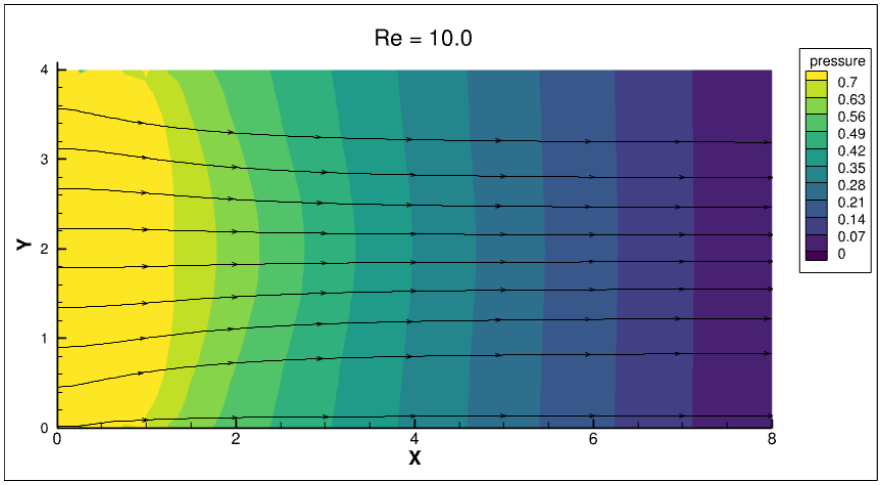
Figure 8: Numerical solution for the Poiseuille flow. Velocity field and streamlines.



(a) Initial guess



(b) $Re = 1$



(c) $Re = 10$

Figure 9: Numerical solution for the Poiseuille flow. Pressure field and streamlines.

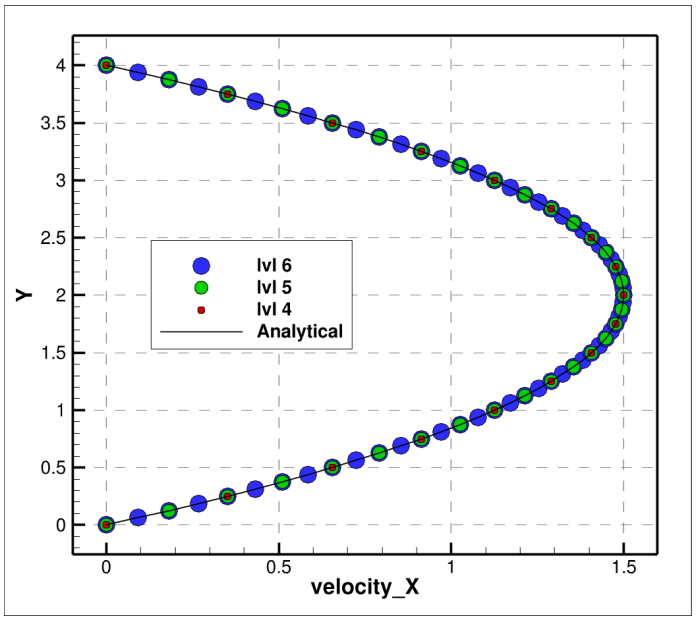
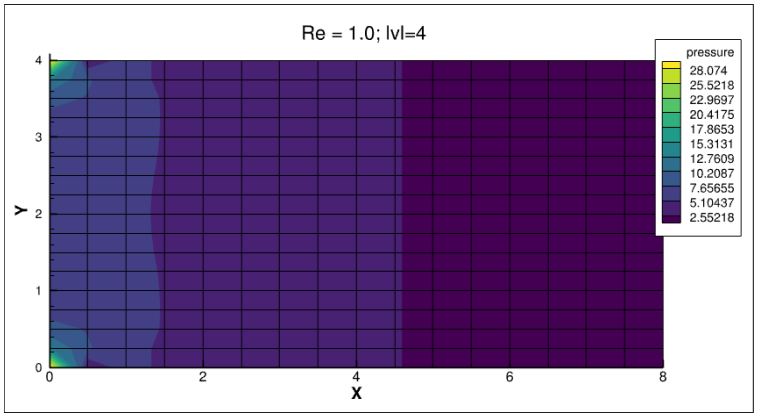
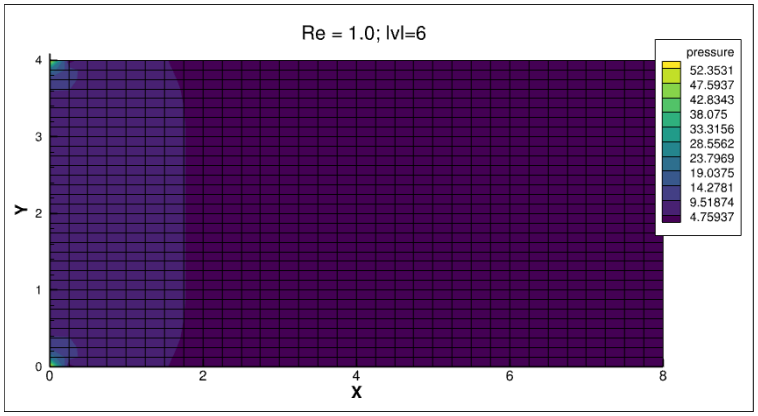


Figure 10: Numerical solution for the Poiseuille flow. Velocity profile at $x = 7.5$. Re=1.



(a) $lvl = 4$



(b) $lvl = 6$

Figure 11: Numerical solution for the Poiseuille flow for different mesh size. Pressure field.

4 Flow around the cylinder

This problem consists of a flow around a fixed cylinder with a constant upstream fluid velocity, see Figure 12. This problem no longer has an analytical solution.

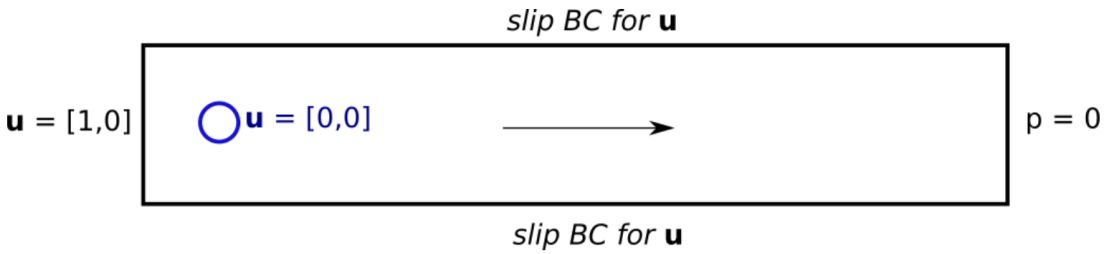


Figure 12: Schematic representing the flow around a cylinder problem.

Again, simulations have been carried out for different Reynolds numbers. In particular, three cases have been studied: Re $= 1, 10, 100$.

The creation of the mesh of the study domain has been carried out using the function of `deal.ii` (see [6]):

```
int number_of_initial_refinement = 4;
GridGenerator::channel_with_cylinder(triangulation, 0.03, 2, 2.0, true );
triangulation.refine_global(number_of_initial_refinement);
```

All the input values have been the default values, except for `colorize`, set as `true` in order to be able to set the boundary conditions easily. The mesh for a refinement level equal to 1 can be seen in Figure 13.

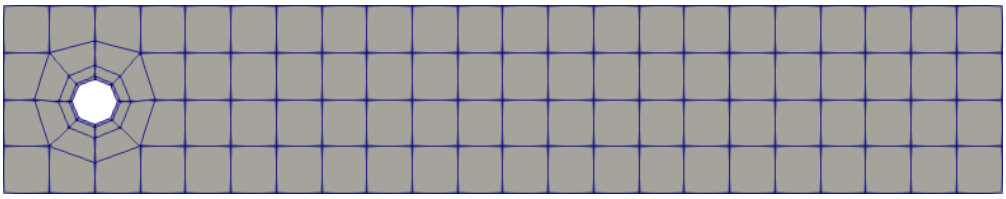


Figure 13: Mesh for the flow around cylinder, $lvl = 1$.

The numerical simulation has been performed again using a level of refinement equal to 4 and with a tolerance for the Newton's method equal to 1e-12. The output obtained for each simulation is shown bellow:

```
iter: 1, Linfty norm: 113.581 L2 norm: 5337.07
iter: 2, Linfty norm: 1.3182 L2 norm: 45.881
iter: 3, Linfty norm: 1.97009e-05 L2 norm: 0.000673898
iter: 4, Linfty norm: 2.98358e-13 L2 norm: 4.73766e-12
```

Listing 5: Output for the flow around cylinder simulation. Re=1

```
iter: 1, Linfty norm: 11.3581 L2 norm: 621.423
iter: 2, Linfty norm: 1.28846 L2 norm: 45.0883
iter: 3, Linfty norm: 0.00189957 L2 norm: 0.0702225
iter: 4, Linfty norm: 3.69878e-09 L2 norm: 8.17108e-08
iter: 5, Linfty norm: 4.78283e-14 L2 norm: 7.92767e-13
```

Listing 6: Output for the flow around cylinder simulation. Re=10

```
iter: 1, Linfty norm: 1.69114 L2 norm: 324.668
iter: 2, Linfty norm: 0.992065 L2 norm: 57.794
iter: 3, Linfty norm: 0.100798 L2 norm: 4.9711
iter: 4, Linfty norm: 0.000479428 L2 norm: 0.0166145
iter: 5, Linfty norm: 8.56957e-08 L2 norm: 5.69485e-07
iter: 6, Linfty norm: 1.27172e-12 L2 norm: 2.63916e-12
iter: 7, Linfty norm: 3.31282e-15 L2 norm: 6.71122e-14
```

Listing 7: Output for the flow around cylinder simulation. Re=100

The solution obtained for each simulation is shown below. In particular, the velocity, pressure and vorticity fields are shown. However, before showing the results it is worthwhile to do a literature review to see what the expected results are. In this paper we will focus on two references: [7] and [8].

In the limit of a non-viscous fluid (potential flow), the problem has an analytical solution and the streamlines are perfectly attached to the cylinder. The Cp distribution is therefore symmetrical if the mesh is sufficiently fine. Figure 14 shows a result of my personal research project for the Euler Flow case at M=0.1.

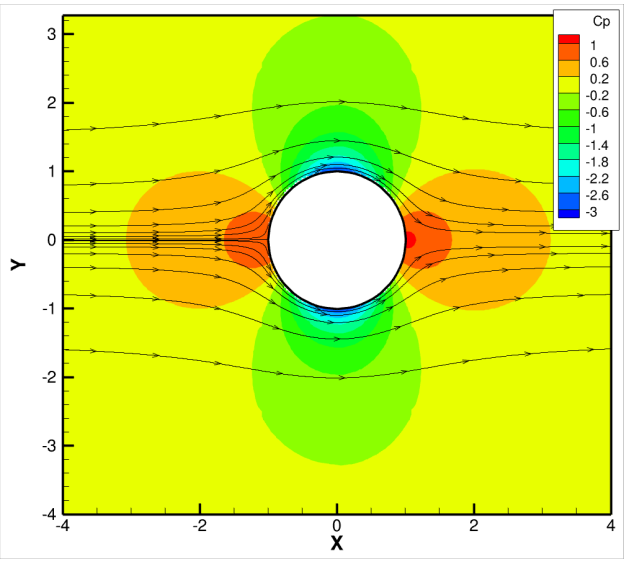


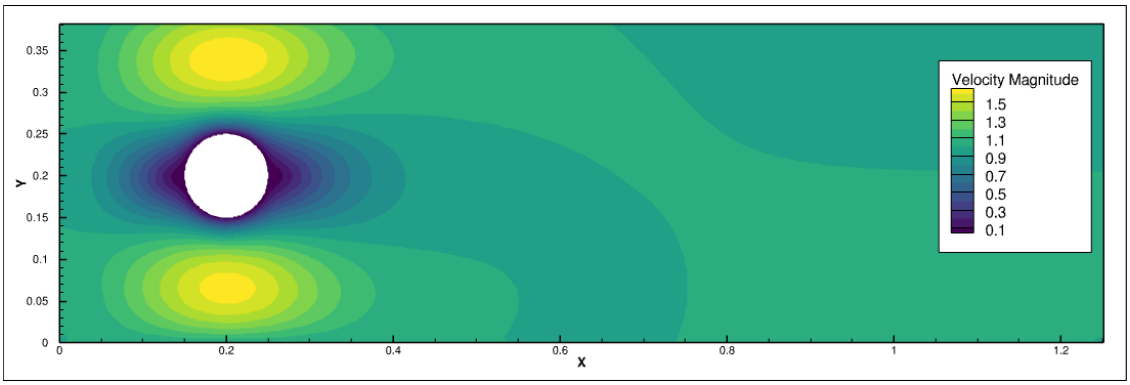
Figure 14: Euler Flow around cylinder, $M = 0.1$. Cp field.

For viscous flows, a recirculation zone is created at the back stagnation point. This zone grows as the Reynolds number grows (see, e.g., [7] or [8]).

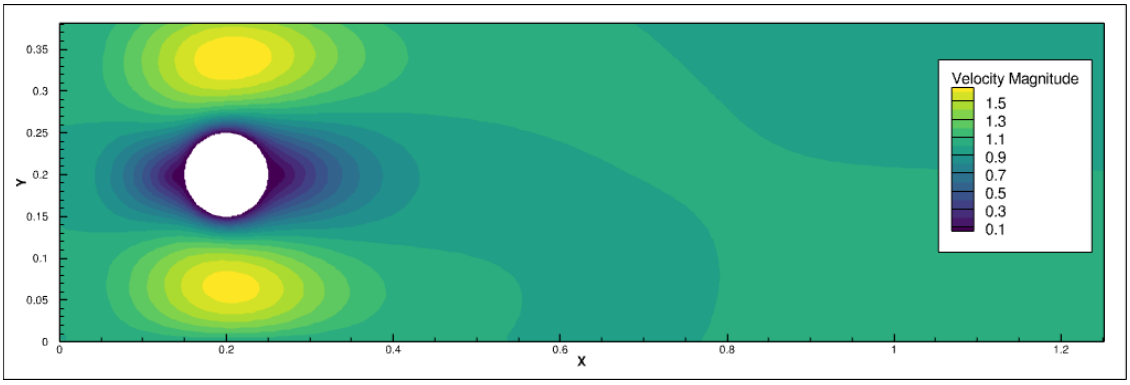
That said, the results can be analyzed.

4.1 Velocity

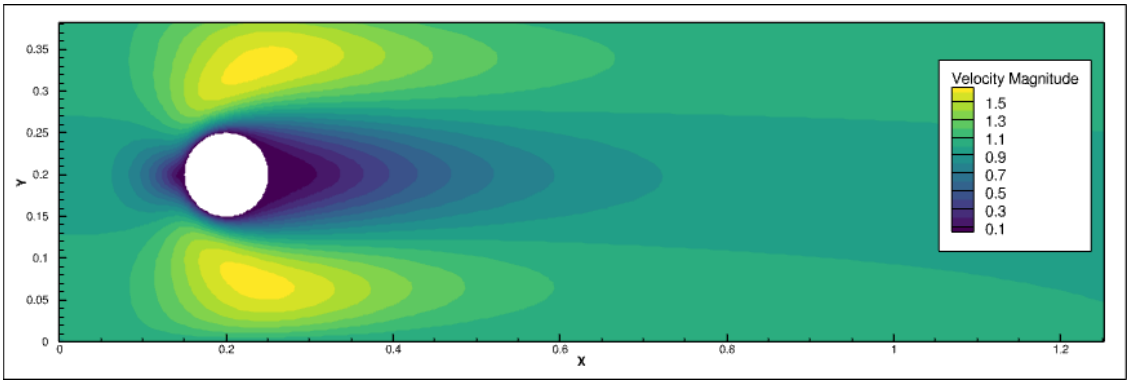
Figure 15 shows the velocity magnitude field of each simulation. the size of the recirculation zone between Figure 15a ($Re = 1$) and Figure 15c ($Re = 100$) is very different. As expected, as the Reynolds number increases this zone grows.



(a) $Re = 1$



(b) $Re = 10$

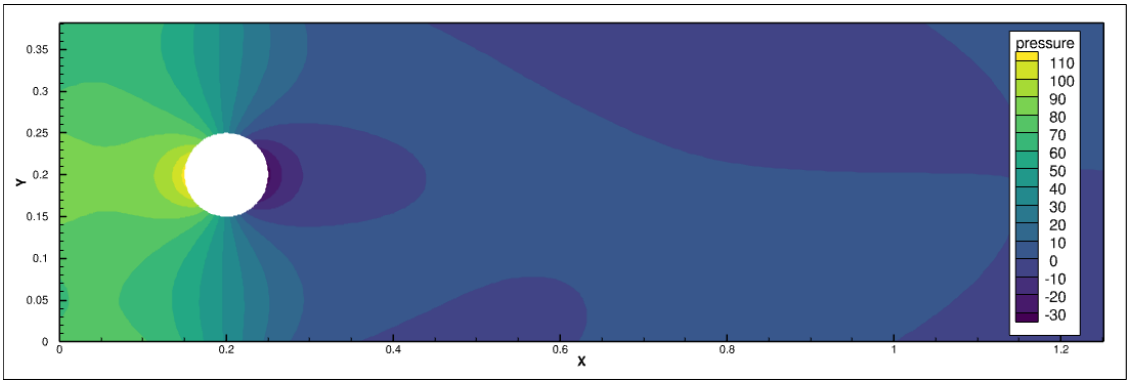


(c) $Re = 100$

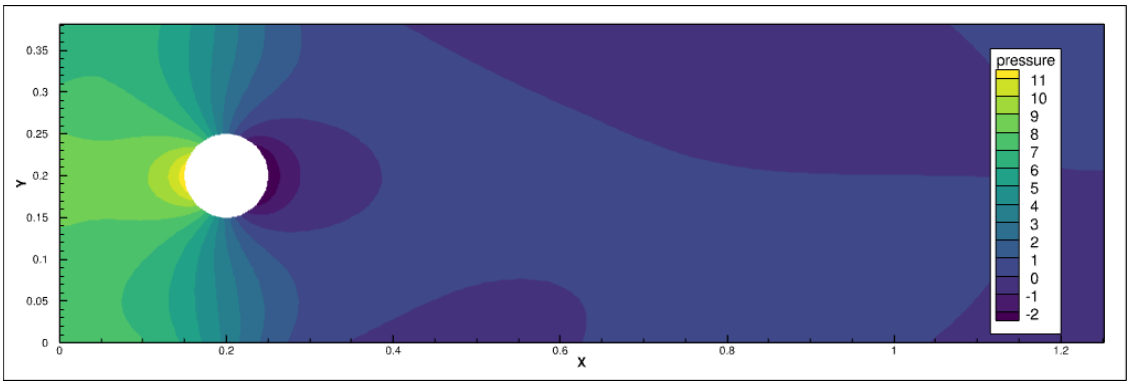
Figure 15: Numerical solution around the cylinder. Velocity field.

4.2 Pressure

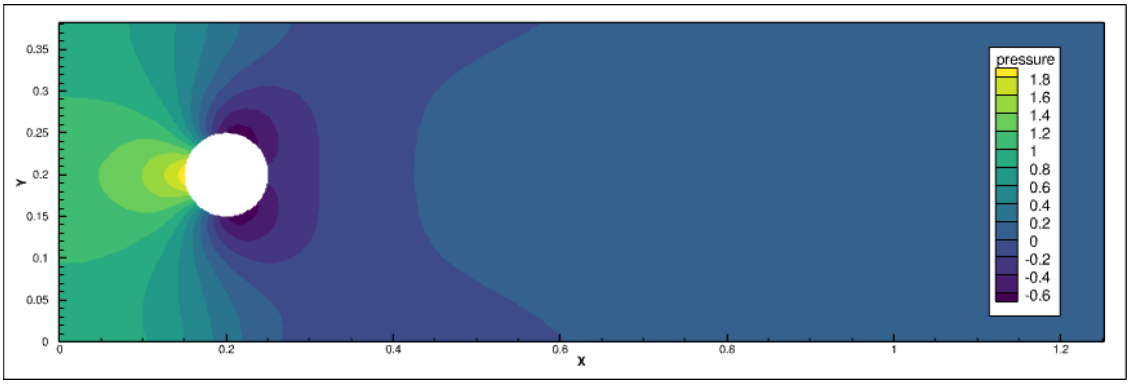
Figure 16 reveals that the highest pressure occurs at the front stagnation point, where the flow meets the cylinder wall head-on, while the lowest pressure is located at cylinder sides. Pressure experiences a partial recovery on the downstream side, but due to viscous losses causing flow separation and the formation of a recirculation zone, it remains significantly lower at the downstream stagnation point compared to the upstream counterpart.



(a) $Re = 1$



(b) $Re = 10$



(c) $Re = 100$

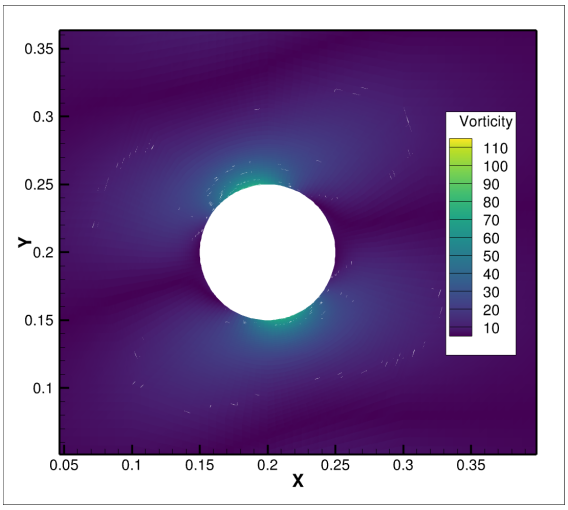
Figure 16: Numerical solution around the cylinder. Pressure field.

4.3 Vorticity

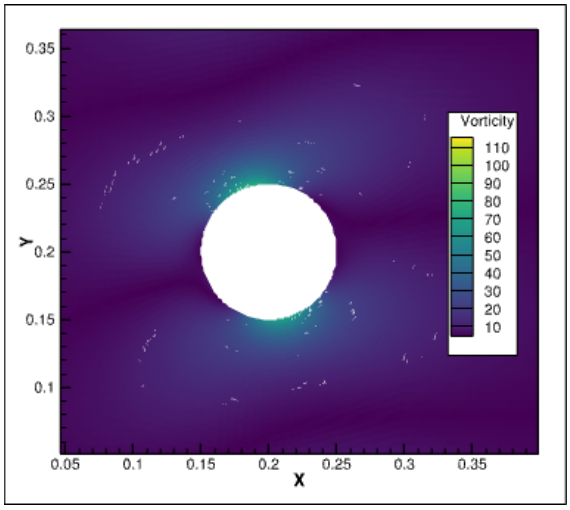
Finally, we have decided to show the vorticity around the cylinder. Vorticity is a vector field variable which is derived from the velocity vector. Mathematically, it is defined as the curl of the velocity vector (see e.g. [9]).

Since this is a bidemnsional case, the z component of the velocity, as well all derivatives with respect to z are 0. That is, vorticity has only one component on the z-axis. According with, [9] "the vorticity can be seen as a vector having magnitude equal to the maximum circulation at each point and to be oriented perpendicularly to this plane of circulation for each point."

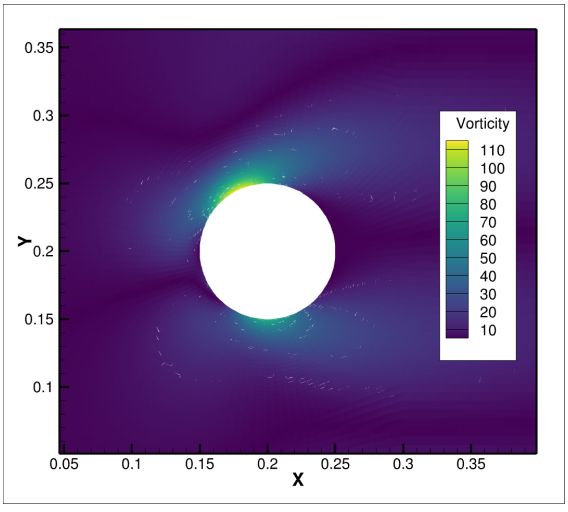
Figure 17 shows the results obtained by post-processing in Tecplot. However, the variable is not clearly observed, so a greater effort to plot this variable has to be undertaken.



(a) $Re = 1$



(b) $Re = 10$



(c) $Re = 100$

Figure 17: Numerical solution around the cylinder. Vorticity field.

References

[1] B. Blais, "The finite element method: Navier-Stokes equations." Polytechnique Montreal. Specialized Numerical Methods for Transport Phenomena. Lecture Notes. February 2024.

[2] deal.II, "The step-57 tutorial program." `https://www.dealii.org/current/doxygen/deal.II/step_57.html`. Accessed: Mars 2024.

[3] B. Blais, "The finite element method: Non-linear Problems." Polytechnique Montreal. Specialized Numerical Methods for Transport Phenomena. Lecture Notes. February 2024.

[4] deal.II, "The step-15 tutorial program." `https://www.dealii.org/current/doxygen/deal.II/step_15.html`. Accessed: Mars 2024.

[5] F. M. White and H. Xue, *Fluid mechanics*. New York, NY: McGraw-Hill Education, ninth edition ed., 2021.

[6] D. Arndt, W. Bangerth, M. Bergbauer, M. Feder, M. Fehling, J. Heinz, T. Heister, L. Heltai, M. Kronbichler, M. Maier, P. Munch, J.-P. Pelteret, B. Turcksin, D. Wells, and S. Zampini, "The `deal.II` library, version 9.5," *Journal of Numerical Mathematics*, vol. 31, no. 3, pp. 231–246, 2023.

[7] S. Sen, S. Mittal, and G. Biswas, "Numerical simulation of steady flow past a circular cylinder," 12 2010.

[8] B. Fornberg, "A numerical study of steady viscous flow past a circular cylinder," *Journal of Fluid Mechanics*, vol. 98, no. 4, p. 819–855, 1980.

[9] cfd-online, "Vorticity." `https://www.cfd-online.com/Wiki/Vorticity`. Accessed: Mars 2024.

Appendices

A Linearization of the incompressible Navier Stokes equations

- Continuity equation:

$$R(p^*) = \nabla \cdot \boldsymbol{u} \tag{21}$$
$$R(p^* + \varepsilon\delta p^*) = \nabla \cdot \boldsymbol{u} + \nabla \cdot (\varepsilon\boldsymbol{\delta u}) \tag{22}$$
$$\lim_{\varepsilon\to 0} \frac{R(p^* + \varepsilon\delta p^*) - R(p^*)}{\varepsilon} = \nabla(\boldsymbol{\delta u}) \tag{23}$$

- Momentum equation:

$$R(\boldsymbol{u}) = (\boldsymbol{u}\cdot\nabla)\boldsymbol{u} + \nabla p^* - \nu\nabla^2\boldsymbol{u} \tag{24}$$

$$R(\boldsymbol{u} + \varepsilon\boldsymbol{\delta u}) = ((\boldsymbol{u} + \varepsilon\boldsymbol{\delta u})\cdot\nabla)(\boldsymbol{u} + \varepsilon\boldsymbol{\delta u}) + \nabla(p^* + \varepsilon\delta p^*) - \nu\nabla^2(\boldsymbol{u} + \varepsilon\boldsymbol{\delta u}) \tag{25}$$
$$= (\boldsymbol{u}\cdot\nabla)\boldsymbol{u} + (\boldsymbol{u}\cdot\nabla)\varepsilon\boldsymbol{\delta u} + (\varepsilon\boldsymbol{\delta u}\nabla)\boldsymbol{u} + (\varepsilon\boldsymbol{\delta u}\nabla)\varepsilon\boldsymbol{\delta u} \tag{26}$$
$$+ \nabla p^* + \nabla\varepsilon\delta p^* - \nu\nabla^2\boldsymbol{u} - \nu\nabla^2(\varepsilon\boldsymbol{\delta u}) \tag{27}$$

$$\lim_{\varepsilon\to 0} \frac{R(\boldsymbol{u} + \varepsilon\boldsymbol{\delta u}) - R(\boldsymbol{u})}{\varepsilon} = (\boldsymbol{u}\cdot\nabla)\boldsymbol{\delta u} + (\boldsymbol{\delta u}\cdot\nabla)\boldsymbol{u} + \nabla\delta p^* - \nu\nabla^2\boldsymbol{\delta u} \tag{28}$$

B weak fornm of the steady incompressible Navier Stokes equations

- Continuity:

$$\int_\Omega q\nabla(\boldsymbol{\delta u})\mathrm{d}\Omega = \int_\Omega -q\nabla\cdot\boldsymbol{u}\mathrm{d}\Omega \tag{29}$$
$$\sum_{j=0}\boldsymbol{\delta u_j}\int_\Omega \psi_i(\nabla\boldsymbol{\phi_j})\mathrm{d}\Omega = \int_\Omega -\psi_i\nabla\cdot\boldsymbol{u}\mathrm{d}\Omega \tag{30}$$

- Momentum:
  - LHS:

$$\int_\Omega \boldsymbol{v}(\boldsymbol{u}\cdot\nabla)\boldsymbol{\delta u}\mathrm{d}\Omega + \int_\Omega \boldsymbol{v}(\boldsymbol{\delta u}\cdot\nabla)\boldsymbol{u}\mathrm{d}\Omega + \int_\Omega \boldsymbol{v}\nabla\delta p^*\mathrm{d}\Omega - \int_\Omega \boldsymbol{v}\nu\nabla^2\boldsymbol{\delta u}\mathrm{d}\Omega = \tag{31}$$

Doing the integral by parts:

$$\int_\Omega \boldsymbol{v}(\boldsymbol{u}\cdot\nabla)\boldsymbol{\delta u}\mathrm{d}\Omega + \int_\Omega \boldsymbol{v}(\boldsymbol{\delta u}\cdot\nabla)\boldsymbol{u}\mathrm{d}\Omega + \int_\Gamma \boldsymbol{v}(\delta p^*\boldsymbol{n})\boldsymbol{n}\mathrm{d}\Gamma - \int_\Omega (\nabla\boldsymbol{v})\delta p^*\mathrm{d}\Omega$$
$$- \int_\Gamma \nu\boldsymbol{v}(\nabla\boldsymbol{u}^T)\mathrm{d}\Gamma + \int_\Omega \nu(\nabla\boldsymbol{v})(\nabla\boldsymbol{u})\mathrm{d}\Omega \tag{32}$$

Considering $\int_\Gamma f \mathrm{d}\Gamma = 0$, Dirichlet boundary conditions:

$$\begin{aligned}\sum_{j=0} \boldsymbol{\delta u_j} \int_\Omega \boldsymbol{\phi_i}(\boldsymbol{u}\cdot\nabla)\boldsymbol{\phi_j}\mathrm{d}\Omega + \sum_{j=0} \boldsymbol{\delta u_j} \int_\Omega \boldsymbol{\phi_i}(\boldsymbol{\phi_j}\cdot\nabla)\boldsymbol{u}\mathrm{d}\Omega \\ - \sum_{k=0} p_k^* \int_\Omega (\nabla\cdot\boldsymbol{\phi_i})\psi_k, \mathrm{d}\Omega + \sum_{j=0} \boldsymbol{\delta u_j} \int_\Omega \nu\nabla\boldsymbol{\phi_i}^T : \nabla\boldsymbol{\phi_j}\mathrm{d}\Omega\end{aligned} \tag{33}$$

– RHS:

$$\int_\Omega -\boldsymbol{v}(\boldsymbol{u}\cdot\nabla)\boldsymbol{u}\mathrm{d}\Omega - \int_\Omega \boldsymbol{v}\nabla p^*\mathrm{d}\Omega + \int_\Omega \nu\boldsymbol{v}\nabla^2\boldsymbol{u}\mathrm{d}\Omega \tag{34}$$

Doing the integral by parts:

$$\begin{aligned}-\int_\Omega \boldsymbol{\phi_i}(\boldsymbol{u}\cdot\nabla)\boldsymbol{u}\mathrm{d}\Omega - \int_\Gamma \boldsymbol{\phi_i}(p^*\boldsymbol{n})\boldsymbol{n}\mathrm{d}\Gamma + \int_\Omega (\nabla\cdot\boldsymbol{\phi_i})p^*\mathrm{d}\Omega \\ + \int_\Gamma \nu\boldsymbol{\phi_i}(\nabla\boldsymbol{u}^T)\mathrm{d}\Gamma - \int_\Omega \nu(\nabla\boldsymbol{\phi_i})(\nabla\boldsymbol{u})\mathrm{d}\Omega\end{aligned} \tag{35}$$

Considering again Dirichlet boundary conditions:

$$-\int_\Omega \boldsymbol{\phi_i}(\boldsymbol{u}\cdot\nabla)\boldsymbol{u}\mathrm{d}\Omega + \int_\Omega (\nabla\cdot\boldsymbol{\phi_i})p^*\mathrm{d}\Omega - \int_\Omega \nu(\nabla\boldsymbol{\phi_i})(\nabla\boldsymbol{u})\mathrm{d}\Omega \tag{36}$$