Project 1 - MEK4470

Shako Farhad

March 24, 2017

Excercise 1

In incompressible flow the density and pressure is decoupled. The energy and the rest of the system is similarly decoupled. The incompressible continuity and momentum equations are given by as follows ([1]).

$$\frac{\partial \bar{u}}{\partial t} + \nabla \cdot \bar{u}^2 - \nabla \cdot (\nu \nabla \bar{u}) = -\nabla p \tag{1}$$

$$\nabla \cdot \bar{u} = 0 \tag{2}$$

The non-linearity in the convection term, $\nabla \cdot (\bar{u}^2)$, is linearized by using an iterative technique, where we set

$$\nabla \cdot (\bar{u}^2) \approx \nabla \cdot (\bar{u}^* \bar{u}^k)$$

Here $\bar{u}^*$ is the known solution and $\bar{u}^k$ is the unknown solution. The algorithm runs until we have $\bar{u}^* = \bar{u}^k$.

There is no pressure equation, but equation 2 imposes a scalar constraint on equation 1. Since there is no pressure equation, we use equation 2 and 1 to derive a pressure equation. We start by discretizing equation 1 and keeping the pressure gradient in its original form.

$$a_p^{\bar{u}} \bar{u}_p + \sum_N a_N^{\bar{u}} \bar{u}_N = r - \nabla p$$

We set $H(\bar{u}) = r - \sum_N a_N^{\bar{u}} \bar{u}_N$ and get

$$a_p^{\bar{u}} \bar{u}_p = H(\bar{u}) \nabla p$$

$$\bar{u}_p = (a_p^{\bar{u}})^{-1} (H(u) - \nabla p)$$

Now using equation 2, we get a pressure equation for incompressible flow.

$$\nabla \cdot ((a_p^{\bar{u}})^{-1} \nabla p = \nabla \cdot ((a_p^{\bar{u}})^{-1} H(\bar{u})) \tag{3}$$

In the loop

At each timestep we solve the momentum equations using the pressure from the previous time step. Below we have set up the momentum equation. Here 'phi' is the same as $\bar{u}^*$ in the momentum equation.

```
fvVectorMatrix UEqn
(
      fvm::ddt(U)
    + fvm::div(phi, U)
    - fvm::laplacian(nu, U)
);

solve(UEqn == -fvc::grad(p));
```

We first loop the pressure corretor step 'nCorr' times. Second we loop the non-orthogonal corrector step 'nNonOrthCorr' times, where we calculate the new pressure and then correct 'phi' for the next pressure corrector step. The continuity error is calculated and written out, and the approximate velocity field using the corrected pressure gradient is corrected.

Turbulence model

Classes for non-newtonian turbulent flow is included with,

```
#include "singlePhaseTransportModel.H"
#include "turbulenceModel.H"
```

A 'turbulence' object is constructed from the class 'autoPtr<incompressible::turbulenceModel>', and this reads the 'constant/turbulenceProperties' and 'RASProperties' or 'LESProperties' dictionaries in order to determine which turbulence model to incorporate.

Excercise 2

The equations used are as follows:

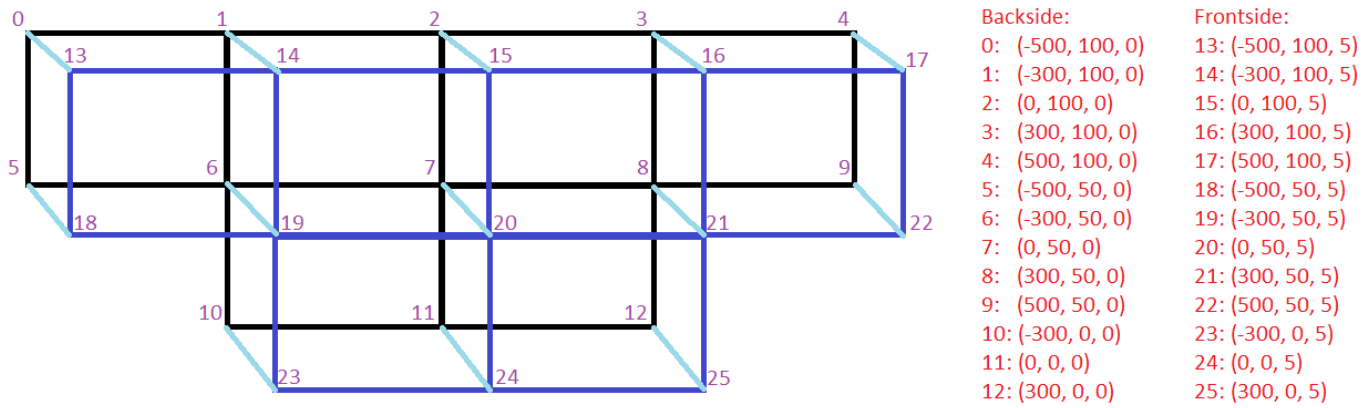
$$\nabla \cdot U^2 - \nabla \cdot R = -\nabla p \tag{4}$$

$$\nabla \cdot U = 0 \tag{5}$$

where p is the pressure and R is the viscous stress.

The initial conditions are $U = 0$ m/s and $p = 0$. The boundary conditions are $U = 10$ m/s at inlet, $p = 0$ at outlet, and no slip on the walls. Since we are simulating air, we have the kinematic of viscosity given as $\nu = 14$ m/s^2.

In figure 1 we can see how the block mesh was setup, and there were two runs of the code, one with mesh size set to 10 in x and y direction, and the second one with mesh size set to 100. So the mesh is increased by a 100 times. While we could see the general flow of air, a lot of detailed vorticity was completely non existent in the 10x10 mesh. The mesh size does/does not have an impact on the solution.



Backside:	Frontside:
0: (-500, 100, 0)	13: (-500, 100, 5)
1: (-300, 100, 0)	14: (-300, 100, 5)
2: (0, 100, 0)	15: (0, 100, 5)
3: (300, 100, 0)	16: (300, 100, 5)
4: (500, 100, 0)	17: (500, 100, 5)
5: (-500, 50, 0)	18: (-500, 50, 5)
6: (-300, 50, 0)	19: (-300, 50, 5)
7: (0, 50, 0)	20: (0, 50, 5)
8: (300, 50, 0)	21: (300, 50, 5)
9: (500, 50, 0)	22: (500, 50, 5)
10: (-300, 0, 0)	23: (-300, 0, 5)
11: (0, 0, 0)	24: (0, 0, 5)
12: (300, 0, 0)	25: (300, 0, 5)

Figure 1: This conceptualized figure is the basis for the blockMeshDict file in the openFoam implementation of the problem.

From figure 2 we can see that the pressure is very high everywhere except for at the outlet, where we set it to 0. In figure 3 we can see that at the end of the simulation the pressure is more evenly distributed. The high pressure is at the inlet and the low pressure is at the outlet. This is simply because the pressure is relieved at the outlet and pressure is created by the air that is pouring in. What is interesting is that we can see that the vortices's create local reductions and increases in pressure. Especially noticeable is the left corner, where there is a somewhat pressure drop. Right next to that blue pressure drop, there is another vortex with a slight pressure increase. These pressure differences certainly do explain why the air is moving the way it is. The air is being sucked in to the low pressure areas and repelled from the higher pressure areas.

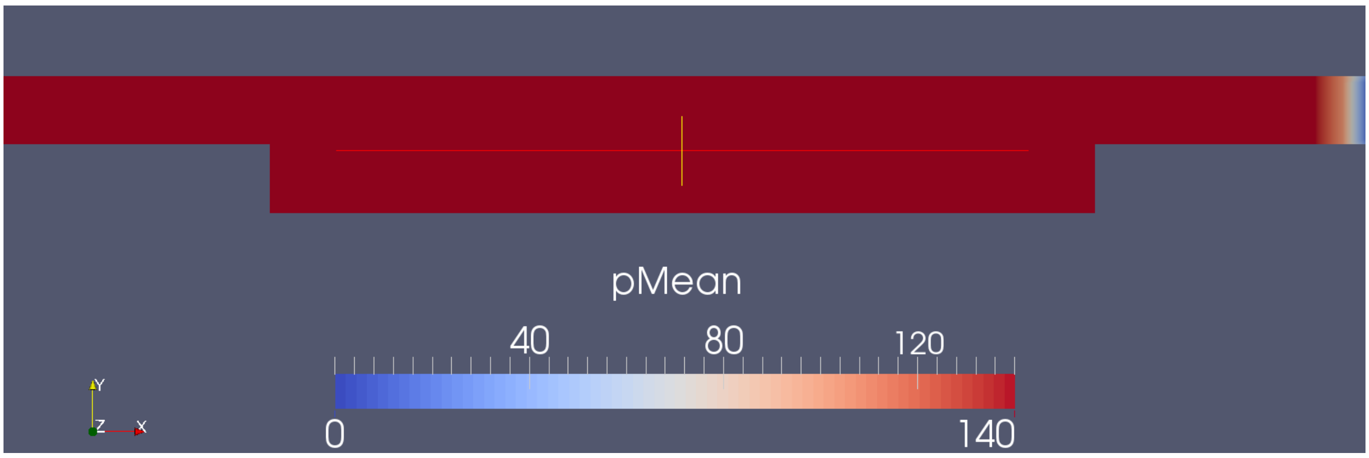


Figure 2: The average pressure profile at the beginning of the simulation with pisoFoam and LES turbulence model. Mesh size is $10x10$.

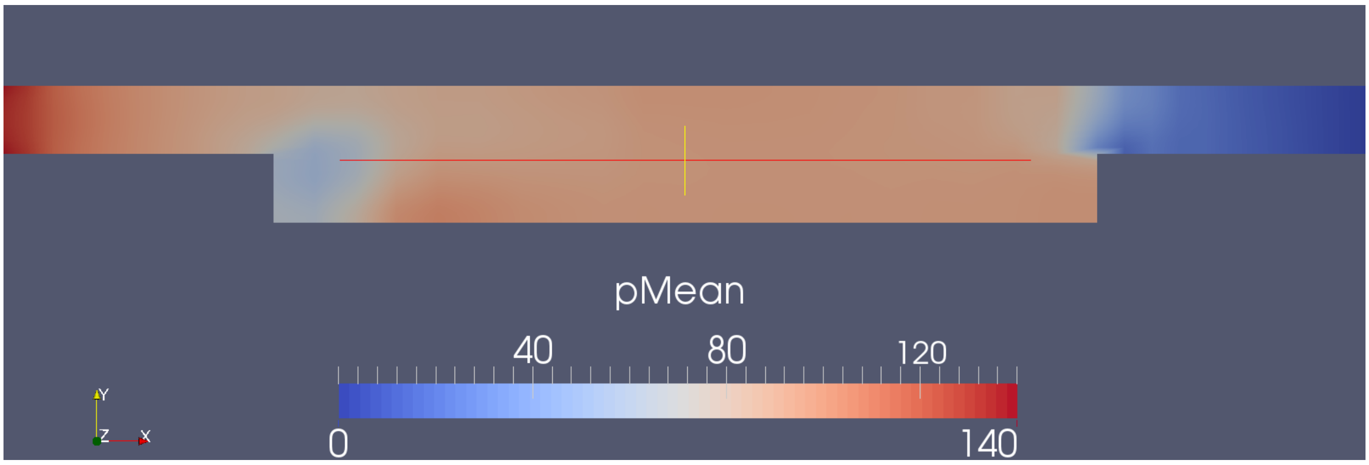


Figure 3: The average pressure profile at the end of the simulation with pisoFoam and LES turbulence model. Mesh size is $10x10$.

In figure 4 the mesh size is $10x10$ and in figure 7 the mesh shize is $100x100$. We can clearly see that a lot of the vortices's are gone and have been smoothed out. This indicates that the LES turbulence modeling is very much dependant on mesh size. If we want to see more details in the vortices's we have to increase the mesh size to an adequate level

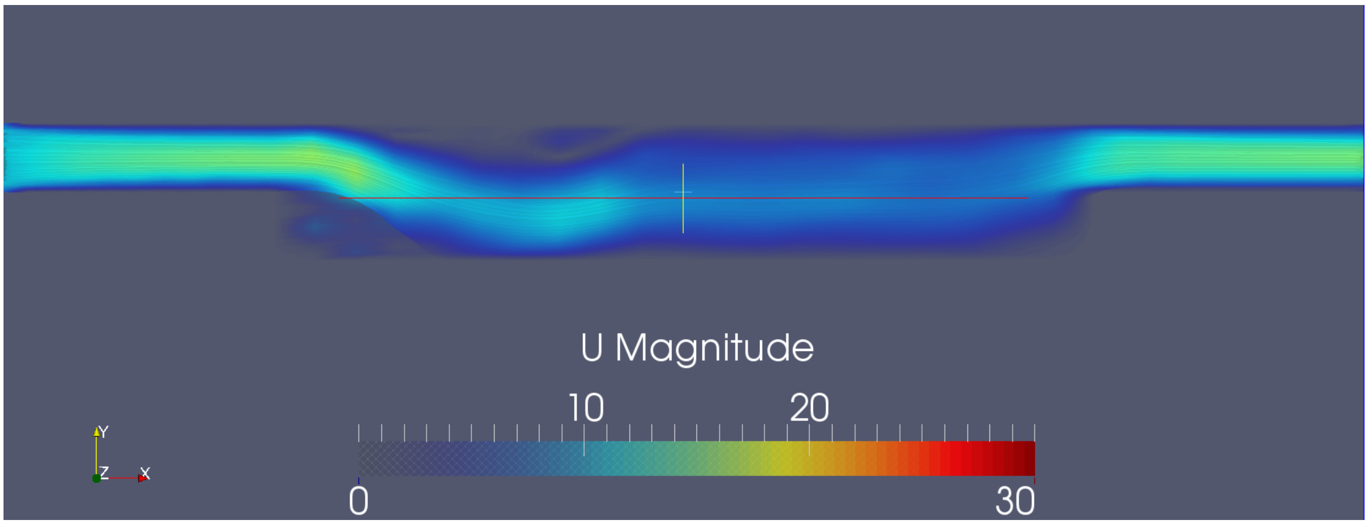


Figure 4: A velocity profile snapshot from a simulation with pisoFoam and LES turbulence model. Mesh size is $10x10$.

Excercise 3

Unfortunately the kOmega would not function, and therefore only one RANS model has been investigated which is the kEpsilon model. All of the boundary conditions and initial conditions are the same as in Excercise 2. The main difference is that we start off with $k = 0$ and $\varepsilon = 0$. The simpleFoam solver automatically updates these to correct values. Since RANS does a time average of the velocity profile, the mesh size does not matter in particular. In figure 6 we have used a $100x100$ mesh size which is most likely overkill.

Excercise 4

RANS stands for Reynolds-averaged Navier–Stokes and what it basically does is to take a time-average of the turbulence, rather than to look at the instantenous turbulence of the flow. LES stands for Large eddy simulation and it does the same thing as RANS but it takes a spatial average of the turbulence. This way both RANS and LES save computational power by ignoring the small scales and looking at the bigger picture.

While all the algorithms (piso, simple, pimple) solve the same governing equations, the algorithms principally differ in how they loop over the equations. The pimple solver makes it possible to have a higher Courant number. The simple solver requires the Courant number to be less than 1, but this is not the case with the pimple solver. So when it is needed to look at very long time lengths, as is needed fro RANS (which gives a time average), we use the pimple solver. With the simple solver we can not just increase the time step because then we face instability. With the pimple solver we can just increase the 'nOuterCorrectors' number so that enough corrections are done on the pressure within one time step, before going to the next one. This ensures that we can have high time steps, but at the same time, not be unstable.

Excercise 5

From figure 6 and 7 we can see that the vortexes are completely eliminated. The RANS turbulence model completely smooths out all of these details. The average velocity profile given in figure 5 gives us something similar to the RANS turbulence model, but it is not as extreme. The important takeaway is that the RANS turbulence model is extremely fast compared to the LES turbulence model, and it gives us insight into the overall flow direction and distribution. This is still valuable even though a lot of detail is lost.

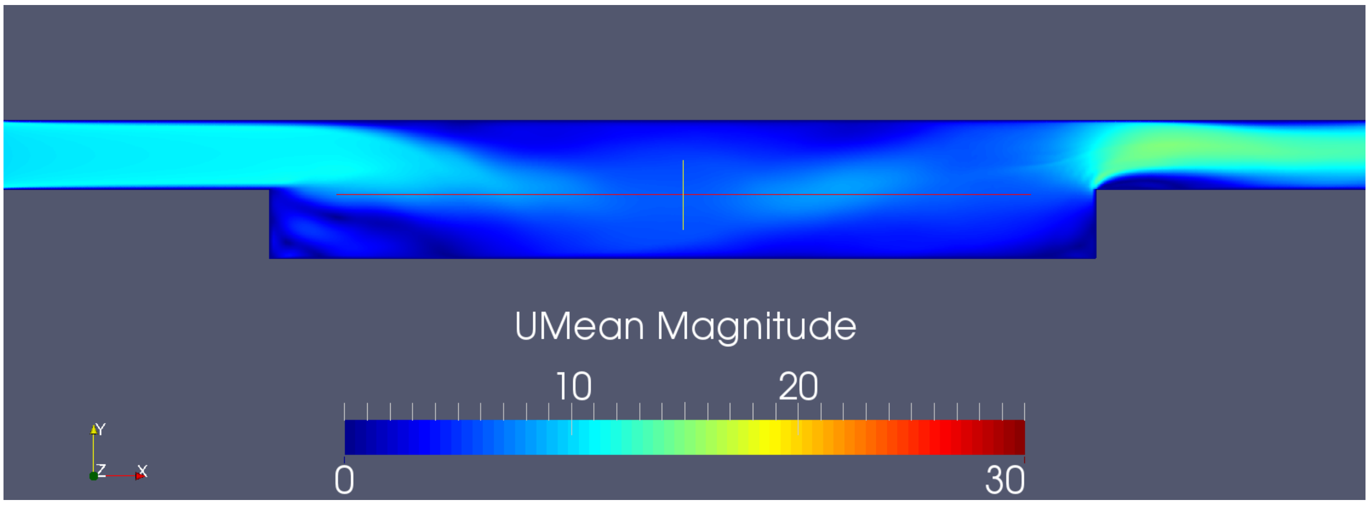


Figure 5: The average velocity over the field at the last frame of the simulation done with pisoFoam solver and LES turbulence model. Mesh size is $100x100$.

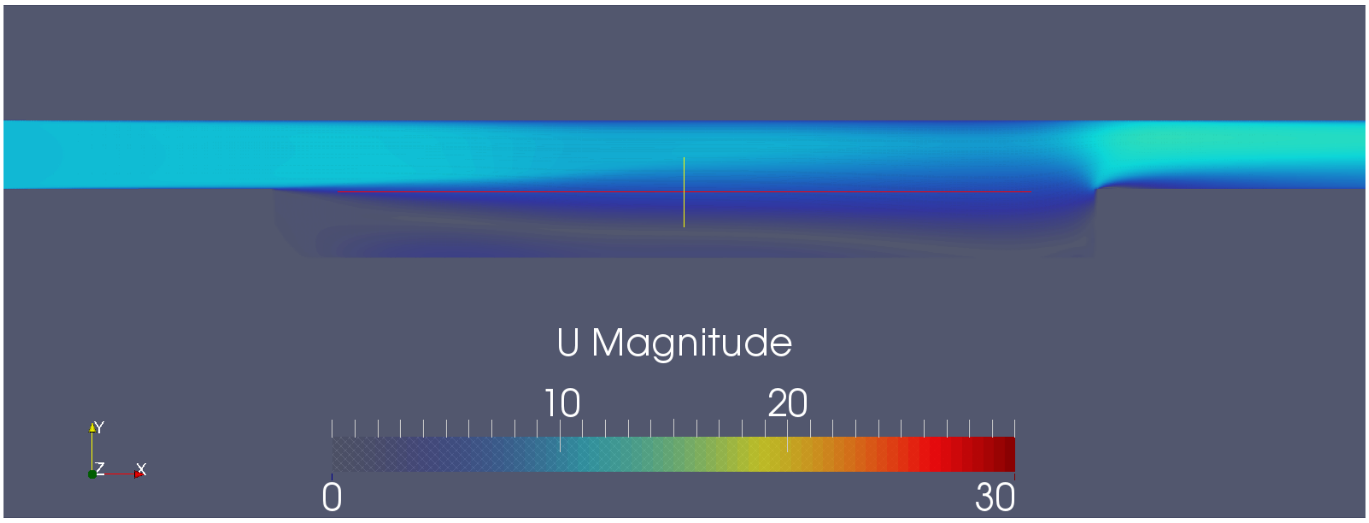


Figure 6: Velocity profile with simpleFOAM and RANS kEpsilon turbulence model. This is the last frame, after convergence has been reached. Mesh size is 100x100.

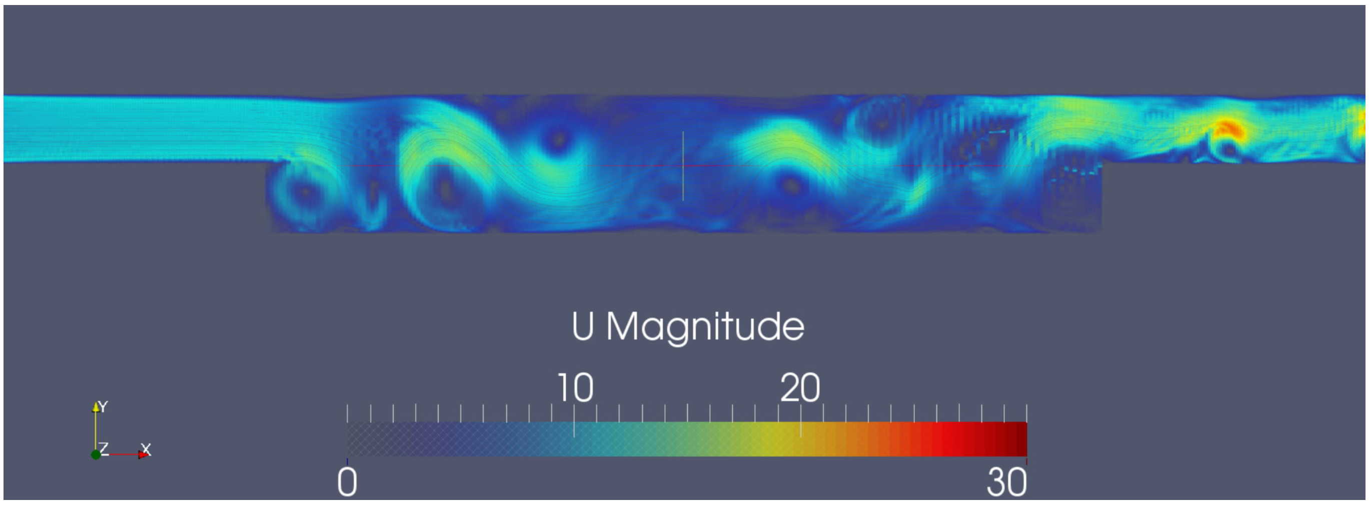


Figure 7: The velocity profile given by pisoFoam with LES turbulence model. Mesh size is 100x100.

Excercise 6

LES takes a spatial average of the turbulence and ignores the small length scale variations. The most simplified form of a problem is usually in 2D, and ignoring the small length scale variations would simplify the problem even more. This could lead to a lot of inaccuracy when comparing the results to the real world. Therefore LES is not usually used in 2D.

Appendix

In the below link you can find the animations, the plots and code.
`https://github.com/ShakoFarhad/Project-1-MEK4470`

References

[1] Chalmers Tekniska Högskola Håkan Nilsson. A look inside icoFoam (and pisoFoam). http://www.tfd.chalmers.se/~hani/kurser/OS_CFD_2013/aLookInsideIcoFoam.pdf.