PhD course: Implementation of a finite volume solver for incompressible, two-dimensional laminar and turbulent flow

Lars Davidson
Division of Fluid Dynamics
Dept. of Mechanics and Maritime Sciences
Chalmers University of Technology
SE-412 96 Göteborg
http://www.tfd.chalmers.se/~lada, lada@chalmers.se

September 11, 2024

1 CFD course

1.1 Background

The traditional method for CFD in industry and universities is Reynolds-Averaged Navier-Stokes (RANS). It is a fast method and mostly rather accurate. In every industry which is working on fluid mechanics, CFD is used.

1.2 Aim

Unfortunately, most engineers and many researchers have limited knowledge of what a CFD CFD code is doing. The object of this course is to close that knowledge gap.

1.3 When?

This PhD course is given every second year (2015, 2017, 2019 ...) in Study period 3. If you want to take part, register on the course by sending me an Email. The course page can be found at PhD course in CFD

1.4 Lectures

There will be no lectures. There will be one meeting (1-2 hours) per week. In these meetings the students have the opportunity for questions and discussion. The first meeting will take place in week 1 (Study period 3).

2 Part I

Write a 2D Navier-Stokes solver for incompressible, laminar flow. Compute the flow in a square lid-driven cavity at $Re = U_{wall}L/\nu = 1000$; set $U_{wall} = L = 1$ and $\nu = 1/Re$. Compare with numerical simulations in [1]. A mesh with 12×12 nodes (constant size) should be used. Compare with data on the course www page. Write a small report (max 10 pages). This gives 4 ETC credits.

- Collocated grid arrangement, i.e non-staggered
- The convection-diffusion can be treated as in assignment K2 in the MTF072 CFD course

The convection scheme for first-order upwind is derived in the Appendix A

A flow chart could look like this. You should first look at the book [6] as well as the reports for CALC-BFC [3] and TEACH3D [4]. The reports can be downloaded here. You should also look at the `lecture notes` (Chapter 2-9).

1. Compute all geometrical quantities such as areas, volume, interpolation factors. I recommend that you put the boundary nodes *at* the boundaries, see Chapter 4, p. 10 in Lecture notes[1]
2. Set all variables (u, v, p) as well as the convections, $\dot{m}_e$ and $\dot{m}_n$, to zero.
3. Compute the coefficients, a_W, a_E, a_S and a_N. Use $\dot{m}_e$ and $\dot{m}_n$ in Eq. 2 (from the previous iteration) when computing the convective fluxes; the first iteration they are zero. The coefficients are the same for u and v. Initially you should use 1st-order upwind. Later on, you could try central differencing or second-order upwinding.
4. Compute the source terms of u and v, i.e. $-\partial p/\partial x$ and $-\partial p/\partial y$ and put them in S_U; for u is reads $S_U = -\partial p/\partial x \Delta V$. The boundary condition for the pressure is $\partial p/\partial n = 0$ at all boundaries.
5. Introduce implicit under-relaxation (see pp. 52-53, Chapter 6 in Lecture notes[1]), of say 0.5, and compute modified S_U^{mod} and a_P^{mod}.
6. Solve u and v (recall that S_U is different for u and v) . Use either TDMA or Gauss-Seidel (G-S) If you use TDMA (see Chapter 7), use it along both x (I lines) and y (J lines) directions. Note that you must probably make a couple of G-S sweeps for p' (10 or more) in order to make the equations converge, i.e.

```
do nn=1,nsweep(nphi)
   do i= 2,nim1
   do j= 2,njm1
```

[1] http://www.tfd.chalmers.se/~lada/comp_fluid_dynamics/lecture_notes.html

```
          rhs =ae(i,j)*phi(i+1,j,nphi)+aw(i,j)*phi(i-1,j,nphi)
     .        +an(i,j)*phi(i,j+1,nphi)+as(i,j)*phi(i,j-1,nphi)
     .        +su(i,j)
          phi(i,j,nphi)= rhs/ap(i,j)
         end do
         end do
     end do
```

where `nsweep(nphi) > 3` where nphi=u, v or p'.

7. Compute $\dot{m}_e^* = 0.5(u_E + u_P)A_e\rho$ and $\dot{m}_n^*$ with central differencing
8. Add Rhie-Chow as

$$\begin{aligned} d_p &= p_{EE} - 3p_E + 3p_P - p_W \\ d_v &= d_p A_e/4/a^u_{Pe} \\ \dot{m}_e &= \dot{m}_e^* + \rho A_e d_v \end{aligned} \tag{1}$$

see p. 55 in Chapter 6 (collocated grid) in Lecture notes[1]. a^u_{Pe} denotes a_P for the velocity interpolated to face e; A_e denotes the area of the east face.

9. Compute the coefficients for the p' equation. The boundary condition at all boundaries is $\partial p'/\partial n = 0$. This is achieved by setting the corresponding coefficient to zero (e.g. at the east boundary $a_E = 0$).
10. Compute the source term in the p' equation (i.e. net mass flux) as $S_U = -(\dot{m}_e - \dot{m}_w + \dot{m}_n - \dot{m}_s)$
11. The initial conditions of p' is zero, i.e. set $p' = 0$ in all points (at every iteration).
12. Solve p' using TDMA
13. Since we have Neumann boundary conditions on p' at all boundaries, the level of p' it not defined. Choose a cell (e.g. cell $(2, 2)$), and make $p' = 0$ (i.e. $p'_{new} = p' - p'_{I=J=2}$).
14. Correct $\dot{m}_e$ and $\dot{m}_n$. For $\dot{m}_e$ it reads

$$\dot{m}_e^{new} = \dot{m}_e + a_E(p'_P - p'_E) \tag{2}$$

where a_E is the coefficient in the p' equation which is computed as

$$a^u_{Pe} = 0.5(a^u_{P,I+1} + a^u_{P,I}), \quad a_E = \rho A_e^2/a^u_{Pe}$$

15. Correct the velocities, u and v. For u, it reads

$$u_P^{new} = u_P + \frac{p'_w - p'_e}{\Delta x a^u_P}\Delta V$$

16. Correct p as ($\alpha_{P'} \simeq 0.5$)

$$p_P^{new} = p_p + \alpha_{P'} p'_P$$

where α_P is the under-relaxation coefficient ($\alpha_P \simeq 0.5$).

17. Compute residuals for u and v as RHS minus LHS for each cell, i.e.

$$R = \sum_{I=2}^{I=NI-1} \sum_{J=2}^{J=NJ-1} |a_E u_E + a_W u_W + a_N u_N + a_S u_S + S_U - a_P u_P)|$$

and scale it with an appropriate value (see Chapter 4, p. 15 in Lecture notes[2])

The residual for the continuity equation is simply the magnitude of the net mass flux in each cell (which is the source term in the p' equation).

18. If the residuals are larger than, say, 10^{-3}, go to Item 3 and repeat (next iteration)

Compare your results with data on the course www page.

3 Part IIa: turbulence model

Here you should implement the two-equation $k-\omega$ model of Wilcox [7]. It reads

$$\begin{aligned}
\rho \frac{\partial \bar{v}_i k}{\partial x_i} &= P_k + \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] - \rho\beta^* k\omega \\
\rho \frac{\partial \bar{v}_i \omega}{\partial x_i} &= C_{\omega_1}\frac{\omega}{k}P_k + \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_t}{\sigma_\omega}\right)\frac{\partial \omega}{\partial x_j}\right] - C_{\omega 2}\rho\omega^2 \\
\mu_t &= \frac{\rho k}{\omega}, \quad P_k = \mu_t \frac{\partial \bar{v}_i}{\partial x_j}\left(\frac{\partial \bar{v}_i}{\partial x_j} + \frac{\partial \bar{v}_j}{\partial x_i}\right)
\end{aligned} \tag{3}$$

where $\beta^* = 0.09$, $c_{\omega 1} = 5/9$, $c_{\omega 2} = 3/40$ and $\sigma_k = \sigma_\omega = 2$. The momentum equation reads

$$\frac{\partial}{\partial x_j}\left(\bar{v}_i \bar{v}_j\right) = -\frac{1}{\rho}\frac{\partial \bar{p}_B}{\partial x_i} + \frac{\partial}{\partial x_j}\left[(\nu + \nu_t)\frac{\partial \bar{v}_i}{\partial x_j}\right] \tag{4}$$

where the cross-diffusion term has been neglected. The turbulent kinetic energy in the Boussinesq assumption is also neglected (or it can be assumed to be incorporated in the pressure, i.e. $\bar{p}_B = \bar{p} + 2k/3$, see [2]).

The boundary conditions for the turbulent quantities are $k = 0$ at the wall and $\omega_w = 6\nu/(c_{\omega 2} y^2)$ for the cells adjacent to the walls. The ω boundary condition is set by using S_P and S_U as $S_P = -1E10$, $S_U = 1E10 \cdot \omega_w$. You need to use some reasonable initial conditions on k and ω, otherwise the solution

[2] http://www.tfd.chalmers.se/~lada/comp_fluid_dynamics/lecture_notes.html

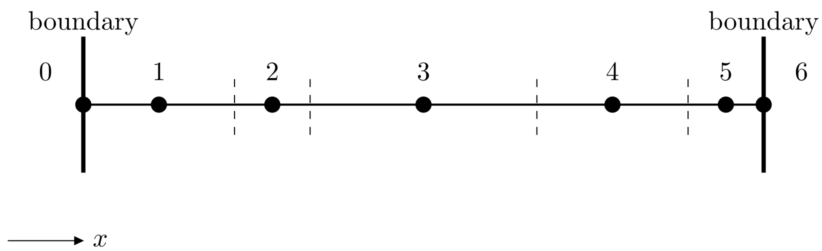


Figure 1: 1D grid with five cells (`ni=5`) using your own solver (e.g. Gauss-Seidel or TDMA). The bullets denote where the dependent variables (including the boundaries) are stored. Dashed lines denote control volume faces.

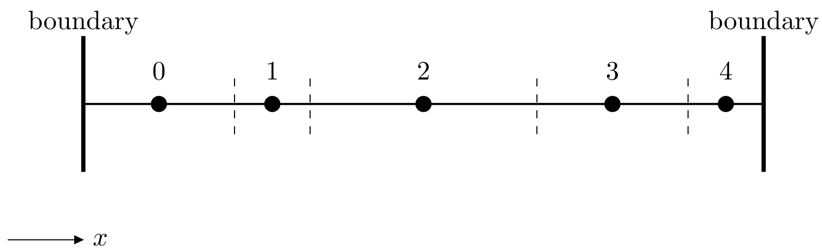


Figure 2: 1D grid with five cells (`ni=5`) using a sparse-matrix solved. The bullets denote where the dependent variables are stored which are labeled 0–4. Dashed lines denote control volume faces labeled 0–5.

may easily diverge. I recommend to use $k_{init} = 10^{-2}$ and $\omega_{init} = 100$. Under-relaxate turbulent viscosity as $\nu_t^{new} = \alpha\nu_t + (1-\alpha)\nu_t^{old}$. Note that you may need to use rather strong under-relaxation on all variables (even down to $\alpha = 0.1$).

The Reynolds number is 100 000. A mesh with 66×66 nodes are used with 15% stretching from the walls to the center.

Compare your results with data on the course www page.

Write a short report (max 10 pages). This part gives an additional 3.5 ETC credits.

4 Part IIb: Sparse-matrix Solver

In Part I, I assume that you use your own solver (Gauss-Seidel or TDMA). In this case, the grid looks like that in Fig. 1. Then the length of the solution array Φ is seven, since Φ is stored at the locations of the bullets including the boundaries. However, if you would set-up the matrix system, the boundaries must not be included because Φ is not solved there. An example of the coefficient matrix is given in Appendix B. In this case the grid is shown in Fig 2, i.e. Φ is not stored at the boundaries. Then the boundary conditions must be implemented via source terms as shown below.

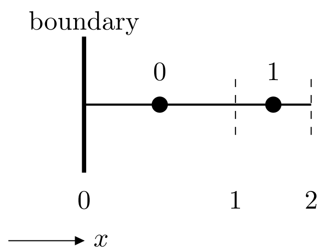


Figure 3: 1D grid. Boundary conditions at $x = 0$.

Suitable sparse-matrix solvers are found in Python, e.g. `linalg.lgmres`, `linalg.gmres` or the algebraic multigrid solver pyAMG [5].

4.1 Inlet boundary conditions using source term

Since we do not have any cells at the boundaries, the boundary conditions must be prescribed through source terms. By default, there is no flux through the boundaries and hence Neumann boundary conditions are set by default. Here, we describe how to set Dirichlet boundary conditions.

Consider discretization in a cell, P, adjacent to an inlet, see Fig. 3. Consider only convection. For the $\bar{v}$ equation at cell $i = 0$ we get

$$
\begin{aligned}
a_P \bar{v}_P &= a_W \bar{v}_W + a_E \bar{v}_E + S_U \\
a_P &= a_W + a_E - S_P, \quad a_W = C_w, \quad a_E = -0.5 C_e \\
C_w &= \bar{v}_W A_w \\
a_P &= C_w - 0.5 C_e
\end{aligned}
\tag{5}
$$

Note there's no 0.5 in front of C_w since the west node is located *at* the inlet. Since there is no cell west of $i = 0$, Eq. 5 has to be implemented with additional source terms

$$
\begin{aligned}
a_W &= 0 \\
S^u_{U,add} &= C_w \bar{v}_{in} \\
S^u_{P,add} &= -C_w
\end{aligned}
\tag{6}
$$

For $\bar{v}$ it reads

$$a_W = 0 \tag{7}$$

$$S^v_{U,add} = C_w \bar{v}_{in} \tag{8}$$

$$S^v_{P,add} = -C_w$$

$$\tag{9}$$

The additional term for the diffusion reads

$$\begin{aligned}
S^u_{U,add,diff} &= \frac{\nu_{tot}A_w}{\Delta x}\bar{v}_{in} \\
S^v_{U,add,diff} &= \frac{\nu_{tot}A_w}{\Delta x}\bar{v}_{in} \\
S_{P,add,diff} &= -\frac{\nu_{tot}A_w}{\Delta x}
\end{aligned} \tag{10}$$

where $S_{P,add,diff}$ is the same for $\bar{v}$ and $\bar{v}$.

4.2 Wall boundary conditions using source term

We use exactly the same procedure as in Section 4.1. At walls, there is no convection and the velocity is zero. Hence we simply use Eq. 10 with $\bar{v} = \bar{v} = 0$, i.e. (for west wall)

$$S_{P,add,diff} = -\frac{\nu A_w}{\Delta x}$$

Note that we use ν instead of ν_{tot} since the turbulent viscosity is zero at the wall.

A First-order upwind

The convective term reads

$$\frac{\partial v\Phi}{\partial y} = 0 \tag{11}$$

We integrate and get

$$Q_e - Q_w = 0 \tag{12}$$

where the fluxes are Q_n (north face) and Q_s (south face). First-order upwind gives

$$\begin{aligned}
Q_s &= \max(C_s, 0)\Phi_S - \max(-C_s, 0)\Phi_P \\
Q_n &= \max(C_n, 0)\Phi_P - \max(-C_n, 0)\Phi_N
\end{aligned} \tag{13}$$

where $C_s = (vA)_s$ and $Q_s = (vA\Phi)_s$. We get

$$\begin{aligned}
&Q_s - Q_n \\
&= \max(C_s, 0)\Phi_S - \max(-C_s, 0)\Phi_P - \max(C_n, 0)\Phi_P + \max(-C_n, 0)\Phi_N \\
&= \max(C_s, 0)\Phi_S + \max(-C_n, 0)\Phi_N - (\max(-C_s, 0) + \max(C_n, 0))\Phi_P
\end{aligned} \tag{14}$$

which can be written

$$\begin{aligned}
Q_s - Q_n &= C_s\Phi_S - C_n\Phi_P, \quad C_s, C_n > 0 \\
Q_s - Q_n &= C_n\Phi_N - C_s\Phi_P, \quad C_s, C_n < 0
\end{aligned} \tag{15}$$

(note that $-\max(-C_n, 0) = C_n$ if $C_n < 0$)

Add the continuity equation times Φ_P , i.e $(C_s - C_n)\Phi_P$, to the last line of Eq. 14 so that

$$
\begin{aligned}
&Q_s - Q_n \\
&= \max(C_s, 0)\Phi_S + \max(-C_n, 0)\Phi_N - (\max(C_s, 0) + \max(-C_n, 0))\Phi_P \\
&= \max(C_s, 0)(\Phi_S - \Phi_P) + \max(-C_n, 0)(\Phi_N - \Phi_P)
\end{aligned}
\tag{16}
$$

The first line is obtained because

$$\max(-x, 0) + x = \max(x), \quad \max(x, 0) - x = \max(-x) \tag{17}$$

Equation 16 (using Eq. 12) can now be written

$$Q_s - Q_n = a_S(\Phi_S - \Phi_P) + a_N(\Phi_N - \Phi_P) = 0 \tag{18}$$

where

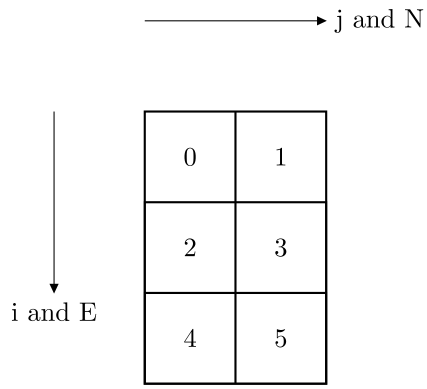
$$
\begin{aligned}
a_S &= \max(C_s, 0) \\
a_N &= \max(-C_n, 0)
\end{aligned}
\tag{19}
$$

Equation 18 can be written on usual SIMPLE finite volume form as

$$
\begin{aligned}
a_P\Phi_P &= a_N\Phi_N + a_S\Phi_S \\
a_P &= a_S + a_N
\end{aligned}
\tag{20}
$$

B Example of a coefficient matrix

B.1 2D grid, $ni \times nj = (3,2)$



$$
\begin{array}{cc}
 & \begin{array}{cccccc} C0 & C1 & C2 & C3 & C4 & C5 \end{array} \\
\begin{array}{c} L0: \\ L1: \\ L2: \\ L3: \\ L4: \\ L5: \end{array} &
\begin{bmatrix}
\color{red}{a_{P,0}} & -a_{N,0} & -a_{E,0} & 0 & \color{blue}{-a_{W,0}} & 0 \\
-a_{S,1} & \color{red}{a_{P,1}} & 0 & -a_{E,1} & 0 & \color{blue}{-a_{W,1}} \\
-a_{W,2} & -a_{S,2} & \color{red}{a_{P,2}} & -a_{N,2} & -a_{E,2} & 0 \\
0 & -a_{W,3} & -a_{S,3} & \color{red}{a_{P,3}} & 0 & a_{E,3} \\
\color{blue}{-a_{E,4}} & 0 & -a_{W,4} & 0 & \color{red}{a_{P,4}} & -a_{N,4} \\
0 & \color{blue}{-a_{E,5}} & 0 & -a_{W,5} & 0 & \color{red}{a_{P,5}}
\end{bmatrix}
\end{array}
\tag{21}
$$

Figure 4: Matrix, A, for 2D flow. $ni \times nj = (3,2)$. Cyclic in x. The coefficients due to cyclic boundary conditions are colored in blue.

References

[1] O. Botella and R. Peyret. Benchmark spectral results on the lid-driven cavity flow. *Computers & Fluids*, 27(4):421–433, 1998.

[2] L. Davidson. Fluid mechanics, turbulent flow and turbulence modeling. eBook, Division of Fluid Dynamics, Dept. of Mechanics and Maritime Sciences, Chalmers University of Technology, Gothenburg, 2021.

[3] L. Davidson and B. Farhanieh. CALC-BFC: A finite-volume code employing collocated variable arrangement and cartesian velocity components for computation of fluid flow and heat transfer in complex three-dimensional geometries. Rept. 95/11, Dept. of Thermo and Fluid Dynamics, Chalmers University of Technology, Gothenburg, 1995.

[4] L. Davidson and P. Hedberg. A general computer program for transient, three-dimensional, turbulent recirculating flow. Technical report, Dept. of Applied Thermo and Fluid Dynamics, Chalmers University of Technology, Gothenburg, 1986.

[5] L. N. Olson and J. B. Schroder. PyAMG: Algebraic multigrid solvers in Python v4.0, 2018. Release 4.0.

[6] H. K. Versteegh and W. Malalasekera. *An Introduction to Computational Fluid Dynamics - The Finite Volume Method.* Longman Scientific & Technical, Harlow, England, 1995.

[7] D. C. Wilcox. Reassessment of the scale-determining equation. *AIAA Journal*, 26(11):1299–1310, 1988.