Fluid Mechanics MTF053

Numerical Analysis of Fully Developed Channel Flow

Computer Assignment 1

Division of Fluid Dynamics
Department of Mechanics and Maritime Sciences
Chalmers University of Technology

Contents

1 Related course material

You can find information relevant for the assignment in the course book *F. M. White Fluid Mechanics 9^{th} ed.* (White, 2016) and in the following documents:

1. MTF053_C04.pdf (lecture notes chapter 4)
2. MTF053_C06.pdf (lecture notes chapter 6)
3. MTF053_Equation-for-Boundary-Layer-Flows.pdf
4. MTF053_Turbulence.pdf
5. MTF053_Formulas-Tables-and-Graphs.pdf

2 Assignment instructions

1. Read through this document
2. Download the following file: MTF053_CA1.ipynb (a `Jupyter NoteBook` that contains the `python` code needed to do the assignment)
3. Start `Jupyter Lab`: https://jupyter.org/try-jupyter/lab/ (you can also install `Jupyter Lab` on your own computer if you prefer that)
4. Upload the `Jupyter NoteBook` (MTF053_CA1.ipynb) in `Jupyter Lab`
5. Go through the assignment step by step in `Jupyter Lab`. Write down answers to questions in the `Jupyter NoteBook`. Derivations made using pen and paper or tablet can be included as photos/images in the `Jupyter NoteBook`.
6. When done with the assignment, download your modified `Jupyter NoteBook` from `Jupyter Lab` and submit it in Canvas along with any images/photos that you would like to add to the document. To download the `Jupyter NoteBook` right click on the notebook file in the file tree on the left side of the `Jupyter Lab` user interface and click on download in the drop down menu.

3 Introduction

In this exercise, you will study a fully developed channel flow (flow between two parallel plates) numerically. You will start with a laminar flow case as that problem can be solved analytically and thus it is possible to make a comparison and get a feeling for the accuracy of the numerical method. In the second part of the assignment, you will analyze a turbulent flow numerically and compare your results to provided measured data. The numerical part will be done using `Python` in `Jupyter Lab`.

3.1 Case definition

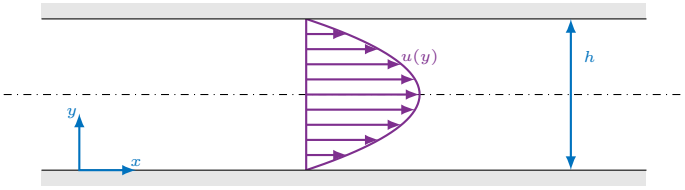

Figure 3.1: Channel flow definition

Fig. 3.1 above defines the geometry. A fluid flows through the channel from left to right and we want to find the velocity field. The flow is assumed to be steady-state, fully developed, and symmetric thus it is sufficient to simulate half the channel and one axial location, i.e. the flow is represented by a single velocity profile $u(y)$. The boundary conditions that should be applied are solid wall at $y = 0$ and centerline symmetry at $y = h/2$.

Geometry	
channel height	$h = 6.0\ cm$
Boundary Conditions	
solid wall (no slip)	$u = 0$ at $y = 0$
symmetry	$\frac{\partial u}{\partial y} = 0$ at $y = h/2$
Flow Properties	
fluid	air @ atmospheric pressure and $20°C$
Case I – laminar flow	
pressure gradient	$\partial p/\partial x = -0.01$ Pa/m, $(\partial p/\partial y = 0)$
Case II – turbulent flow	
pressure gradient	$\partial p/\partial x = -8.0$ Pa/m, $(\partial p/\partial y = 0)$

4 Case I – Laminar Flow

4.1 Analytical solution

The Navier-Stokes equations in two dimensions are given by the continuity equation and the x and y components of the momentum equation

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{4.1}$$

$$\rho\left(\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}\right) = -\frac{\partial p}{\partial x} + \rho g_x + \frac{\partial}{\partial x}\left(\mu\frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(\mu\frac{\partial u}{\partial y}\right) \tag{4.2}$$

$$\rho\left(\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y}\right) = -\frac{\partial p}{\partial y} + \rho g_y + \frac{\partial}{\partial x}\left(\mu\frac{\partial v}{\partial x}\right) + \frac{\partial}{\partial y}\left(\mu\frac{\partial v}{\partial y}\right) \tag{4.3}$$

Starting from the Navier-Stokes equations in two dimensions (Eqns. 4.1 - 4.3) assuming fully-developed steady-state laminar flow neglecting gravity effects, one can show that the differential equations describing our channel flow reduces to

$$\frac{\partial}{\partial y}\left(\mu\frac{\partial u}{\partial y}\right) = \frac{\partial p}{\partial x} \tag{4.4}$$

Task 1.1:

Show that the Navier-Stokes equations Eqns. 4.1 – 4.3 reduces to Eqn. 4.4 for fully-developed steady-state laminar channel flow in two dimensions neglecting gravity.

Examining Eqn. 4.4, it is obvious that for our specified application, it is possible to reduce the number of terms in Eqns. 4.1 – 4.3 significantly, which means that you will remove terms in the equations as part of the derivation. Please note, however, that all such modifications of the equations must be justified.

Hint: the lecture notes for chapter 4 (`MTF053_C04.pdf`) may be of use here.

Write down your answer in `Jupyter Lab`. Derivations made using pen and paper or tablet can be included as photos/images in the `Jupyter NoteBook`.

Task 1.2:

a) Solve Eqn. 4.4 analytically for the above-listed boundary conditions and flow properties (examples are given in the lecture notes for chapter 4 (`MTF053_C04.pdf`) and in the course text book)

b) Calculate the Reynolds number for the channel flow

$$Re = \frac{\rho D_h V_{av}}{\mu} = \frac{D_h V_{av}}{\nu}$$

Remember that we are investigating the flow in a two-dimensional channel, i.e., it is **not** a pipe flow. The correct hydraulic diameter D_h to be used in the Reynolds number for such a **non-circular pipe flow** can be found in `MTF053_Formulas-Tables-and-Graphs.pdf`. Also, the Reynolds number should be calculated using the average velocity, which can be obtained by integrating the velocity profile over the channel

$$V_{av} = \frac{1}{h}\int_0^h u(y)dy$$

Write down your answer in `Jupyter Lab`. Derivations made using pen and paper or tablet can be included as photos/images in the `Jupyter NoteBook`.

4.2 Numerical simulation

With the analytical solution in place, it's now time to attack the problem numerically. The numerical solution will then be compared to the analytical solution to validate the method. In the numerical approach, the flow velocity will be approximated in a discrete number of points. This is accomplished by dividing the domain (in our case a line from $y = 0$ to $y = 3.0$ cm) in a number, let's say N, cells. In each cell there is a node that may or may not be located in the cell center. In our case, we will divide the domain into N cells of equal size – a so-called equidistant mesh – and the nodes will be placed in the cell centers. The result looks like in Fig. 4.1 below. Note that two so-called ghost cells have been added outside of the computational domain (one on each side). The ghost cells will be used for the implementation of boundary conditions later.

The next step is to set up a system of algebraic equations with one equation per node, i.e., in total N equations. There are in principle three available methods to discretize an equation like Eqn. 4.4, namely the Finite-Element Method (FEM), Finite Difference, and the Finite Volume Method (FVM). All three methods would work in our case, but since FVM is the most common method for computational fluid dynamics (CFD) we will use the FVM approach here.

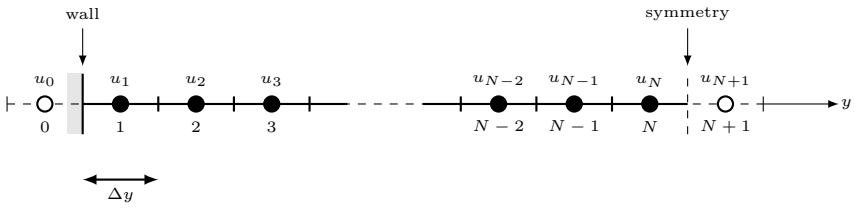

Figure 4.1: Computational domain

4.2.1 The Finite-Volume Method (FVM)

In the Finite-Volume Method (FVM), the governing equations are integrated over each of the cells in the computational domain, which leads to a set of equations that are solved numerically. Integrating Eqn. 4.4 over cell number i (control volume Ω_i), we get

$$\iiint_{\Omega_i} \frac{\partial}{\partial y}\left(\mu\frac{\partial u}{\partial y}\right) dV = \iiint_{\Omega_i} \frac{\partial p}{\partial x} dV$$

where dV i an infinitesimal volume within Ω_i. Using Gauss theorem, the left-hand side of the equation can be expressed as a surface integral

$$\iint_{\partial_i} \left(\mu\frac{\partial u}{\partial y}\right) \cdot \mathbf{n} dS = \iiint_{\Omega_i} \frac{\partial p}{\partial x} dV$$

where ∂_i is the surface of control volume Ω_i, $\mathbf{n}$ is a surface normal and dS is an infinitesimal surface element on ∂_i. Now, evaluating the integrals gives

$$b\Delta x\left[\left(\mu\frac{\partial u}{\partial y}\right)_{i+\frac{1}{2}} - \left(\mu\frac{\partial u}{\partial y}\right)_{i-\frac{1}{2}}\right] = b\Delta x\frac{\partial p}{\partial x}\Delta y_i \Rightarrow$$

$$\left(\mu\frac{\partial u}{\partial y}\right)_{i+\frac{1}{2}} - \left(\mu\frac{\partial u}{\partial y}\right)_{i-\frac{1}{2}} = \frac{\partial p}{\partial x}\Delta y_i \tag{4.5}$$

In Eqn. 4.5, fractional indices indicate cell faces (index $(i+1/2)$ represents the cell face between cell i and cell $(i+1)$ and index $(i-1/2)$ represents the cell face between cell i and cell $(i-1)$, see Fig. 4.1). Thus, in order to be able to close the system of equations and perform the calculations we need estimates of the velocity derivatives at the cell faces. An approximation of the derivative can be obtained using the velocities in the cells closest to the cell face as follows

$$\left(\mu\frac{\partial u}{\partial y}\right)_{i+\frac{1}{2}} = \mu\frac{u_{i+1}-u_i}{y_{i+1}-y_i}$$

$$\left(\mu\frac{\partial u}{\partial y}\right)_{i-\frac{1}{2}} = \mu\frac{u_i - u_{i-1}}{y_i - y_{i-1}}$$

Inserted in Eqn. 4.5 and assuming equidistant mesh ($y_{i+1} - y_i = y_i - y_{i-1} = \Delta y$), we get for cell i

$$\mu\frac{u_{i+1} - u_i}{\Delta y} - \mu\frac{u_i - u_{i-1}}{\Delta y} = \frac{\partial p}{\partial x}\Delta y$$

which can be rewritten as

$$u_{i+1} - 2u_i + u_{i-1} = \frac{\partial p}{\partial x}\frac{\Delta y^2}{\mu} \tag{4.6}$$

The expression given by Eqn. 4.6 is for a generic cell i and if applied to all cells we will get a system of N equations and N unknowns (the flow velocity in every cell). The remaining thing to do now in order to get a system of equations that we can solve is to implement the boundary conditions. If we evaluate Eqn. 4.6 for cell 1 (the cell closest to the wall in Fig. 4.1) we get

$$u_2 - 2u_1 + u_0 = \frac{\partial p}{\partial x}\frac{\Delta y^2}{\mu}$$

Where u_0 is the velocity in the ghost cell outside of the wall (the left-most cell in Fig. 4.1). We don't have a value for u_0, but we can calculate a value using the boundary condition. The wall boundary condition is a no-slip condition, i.e., the velocity should be zero at the cell face that coincides with the wall. Assuming that the fluid velocity at the wall is the average of the velocity in cell 0 and cell 1. We will assure that the wall velocity will be zero by setting the velocity u_0 equal to $-u_1$ (see equation below).

$$u(y = 0) = \frac{1}{2}(u_1 + u_0) = 0 \Rightarrow u_0 = -u_1$$

Now, let's have a look at the symmetry boundary. If we evaluate Eqn. 4.6 for cell N (the cell closest to the symmetry line in Fig. 4.1) we get

$$u_{N+1} - 2u_N + u_{N-1} = \frac{\partial p}{\partial x}\frac{\Delta y^2}{\mu}$$

As can be seen from the relation above, we need the velocity in cell $N + 1$, which again is outside of our computational domain (see Fig. 4.1). We will use the ghost cell on the right side

of the computational domain to implement the symmetry boundary condition. The appropriate boundary condition is that the velocity derivative should be zero at the cell face that coincides with the centerline of the channel. By setting u_{N+1} equal to u_N, we will fulfill the boundary condition.

$$\left.\frac{\partial u}{\partial y}\right|_{y=h/2} = \frac{u_{N+1} - u_N}{\Delta y} = 0 \Rightarrow u_{N+1} = u_N$$

Task 1.3:

Write the equation system that we get by evaluating Eqn. 4.6 in all of the N cells on matrix form, i.e., write Eqn. 4.6 as $A\mathbf{u} = \mathbf{b}$, where $\mathbf{u}$ is a vector with velocity values for all cells in the computational domain (N elements). What does the matrix A and the vector $\mathbf{b}$ look like?

Write down your answer in `Jupyter Lab`. Derivations made using pen and paper or tablet can be included as photos/images in the `Jupyter NoteBook`.

Now, to solve this problem, all we have to do is to invert the matrix A and then the velocity values are calculated as $\mathbf{u} = A^{-1}\mathbf{b}$

Task 1.4:

In the `Jupyter NoteBook` you'll find a `Python` code that solves the equation system derived in Task 1.3 using the `Python` functions `scipy.sparse.diags` and `scipy.sparse.linalg.spsolve`.

Before running the solver, you must make some minor changes in the code:

a) in the function called `laminar_channel_flow_solver`, specify the boundary entries in the coefficient matrix A

b) in the function called `get_analytic_laminar_velocity_profile`, implement calculation of analytical solution for laminar channel flow

c) in the main code for task 1.4, specify number of cells

With these changes made it is now time to run the code. To execute the code in a code cell in `Jupyter Lab`, click on the code cell to make it active and then click on the play button at the top of the notebook. Make sure to execute the code cells in sequence. The functions `laminar_channel_flow_solver` and `get_analytic_laminar_velocity_profile` must be executed before running the task 1.4 main code.

With both the numerical and analytical solution in place, we can estimate the discretization error. Let's define the error as the absolute value of the relative difference between the numerical velocity value and the analytical velocity value in the node closest to the centerline of the channel.

$$e = \left| \frac{u_{numeric} - u_{analytic}}{u_{analytic}} \right| \tag{4.7}$$

Task 1.5:

For this task, a `Python` code is provided in the `Jupyter NoteBook` that runs the laminar flow solver set up in the previous task and calculates the error (e) according to Eqn. 4.7 for 10 different values of N (number of cells) spanning from $N_{min} = 10$ to $N_{max} = 1000$. Note that the `Python` function `numpy.logspace` is used to distribute the cell counts. After running the laminar flow solver with 10 different cell counts and calculating the error for each case, a `loglog` plot showing the error as a function of cell count is generated.

a) How does the error decay with increased cell count?

b) When calculating the error, why is $u_{analytic}$ not calculated at the center of the duct?

Write down your answer in `Jupyter Lab`. Derivations made using pen and paper or tablet can be included as photos/images in the `Jupyter NoteBook`.

5 Case II – Turbulent Flow

Now we are ready for the real challenge – we will simulate the same flow but under turbulent conditions. Note that the only thing that has changed is that the pressure gradient is increased from 0.01 Pa/m to 8.0 Pa/m, i.e., the pressure gradient driving the flow is increased but geometry and fluid properties are the same as in the laminar case. In principle, we could use the same boundary conditions as well, but we will make some modifications for reasons that will be described in detail later. The increased pressure gradient will increase the average flow velocity and thus increase the Reynolds number such that the flow becomes turbulent. Since it will not be possible to obtain an analytic solution for the turbulent channel flow, we will use measured data obtained for a boundary layer over a flat plate for comparison.

5.1 The Reynolds-Averaged Navier-Stokes (RANS) Equations

Since the flow is now assumed to be turbulent, we will apply the Reynolds-averaging technique on the governing equations (Eqns. 4.1 - 4.3) to obtain the so-called Raynolds-Averaged Navier Stokes (RANS) equations. The first step of which is to express flow variables as the sum of an average and a fluctuating part

$$u = \overline{u} + u', \; v = \overline{v} + v', \; p = \overline{p} + p' \tag{5.1}$$

where the overlined quantities are the time averages and the primed quantities are unsteady fluctuations. Note, by definition, the averages of the fluctuating quantities are zero whereas the average of a product of fluctuating properties could be but is generally not zero.

$$\overline{u'} = \overline{v'} = \overline{w'} = 0, \; \overline{u'u'} \neq 0, \; \overline{u'v'} \neq 0, \; \overline{v'v'} \neq 0$$

Task 2.1:

Replace u, v, and p in the Navier-Stokes equations (Eqns. 4.1 – 4.3) with the expressions given in Eqn. 5.1 and time average the equations to get the Reynolds-Averaged Navier-Stokes (RANS) equations. Use the fact that the time average of a fluctuation is zero. Then, use the same arguments as in Task 1.1 to show that the equations can be simplified to the following relation for our specific problem

$$0 = -\frac{\partial \overline{p}}{\partial x} + \frac{\partial}{\partial y}\left(\mu \frac{\partial \overline{u}}{\partial y} - \rho \overline{u'v'}\right) \tag{5.2}$$

Hint: make use of the corresponding derivation available in the lecture notes for Chapter 6 (`MTF053_C06.pdf`).

Write down your answer in `Jupyter Lab`. Derivations made using pen and paper or tablet can be included as photos/images in the `Jupyter NoteBook`.

The terms $-\rho\overline{u'v'}$, $-\rho\overline{u'^2}$, and $-\rho\overline{v'^2}$ appearing in the RANS equations are called Reynolds stresses since they appear as stresses in the averaged equations. The Reynolds stresses are unknowns in the RANS equations. The introduction of new unknowns is a direct consequence of the averaging of the equations and is often referred to as the closure problem of turbulence – additional unknowns are introduced but the number of equations remains the same. Turbulence modeling as a means of closing the system of equations is a research field of its own and there are several approaches ranging from rather simple methods to very complicated ones. An example of a rather simple approach to model $-\rho\overline{u'v'}$ that works well for fully developed channel flow is Prandtl's mixing length model. A basic foundation for most turbulence models is the Boussinesq assumption, which states that the Reynolds stresses can be calculated from the strain rate tensor in analogy with how the viscous shear stresses in a Newtonian fluid are calculated. For our channel flow the Boussinesq assumption reduces to

$$-\rho\overline{u'v'} = \mu_{turb}\frac{\partial\overline{u}}{\partial y} \tag{5.3}$$

which can be compared with the viscous shear stress that, for a boundary layer flow, is obtained as

$$\tau = \mu\frac{\partial\overline{u}}{\partial y}$$

In Eqn. 5.3, μ_{turb} is the eddy viscosity that, in contrast to the fluid viscosity (μ), depends on the flow itself, i.e., it is not a fluid property, it is a property of the flow. According to Prandtl's mixing length model μ_{turb} can be obtained as

$$\mu_{turb} = \rho l_m^2\frac{\partial\overline{u}}{\partial y} \tag{5.4}$$

where l_m is the mixing length (the property that has given the model its name). In a boundary layer, the mixing length can be assumed to be a linear function of the distance from the wall and Eqn. 5.5 gives a well-established way to calculate the mixing length using the von Kármán constant ($\kappa = 0.41$).

$$l_m = \kappa y \tag{5.5}$$

Combined Eqns. 5.3 – 5.5 gives

$$-\rho\overline{u'v'} = \rho\kappa^2 y^2\frac{\partial\overline{u}}{\partial y} \tag{5.6}$$

and thus we have a way to calculate the Reynolds stress $-\rho\overline{u'v'}$ based on known flow properties. Implementing Prandtl's mixing length model into Eqn. 5.2, we get

$$\frac{\partial}{\partial y}\left(\left(\mu + \rho\kappa^2 y^2 \frac{\partial \overline{u}}{\partial y}\right)\frac{\partial \overline{u}}{\partial y}\right) = \frac{\partial \overline{p}}{\partial x} \tag{5.7}$$

For more details see `MTF053_Turbulence.pdf`.

So far so good. One problem remains to be solved though. As mentioned previously, the wall boundary condition needs to be modified. The reason is that the viscous sublayer, a region of the boundary layer very close to the wall (region I in Fig. 5.1), is extremely thin and resolving this layer using an equidistant mesh leads to a need for very small cells and thus the computational cost increases significantly. Moreover, the mixing length model doesn't work very well in the viscous sublayer. Instead of resolving the viscous sublayer, we will assume that the center of the cell closest to the wall is located within the log-layer (region III in Fig. 5.1), and thus the log law can be used. The ghost cell value to the left of the first cell (see Fig. 5.2) is thus calculated using the log law (see Eqn. 5.8).

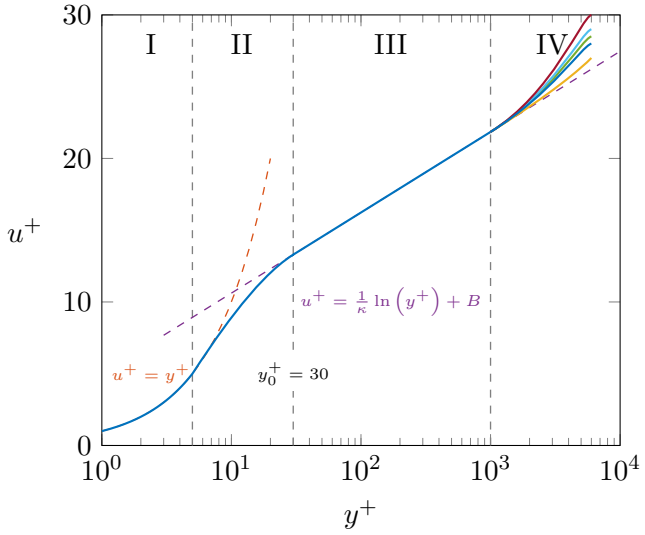

Figure 5.1: Boundary layer regions: viscous sublayer $y^+ < 5$ (I), buffer layer (II), log-law region $30 < y^+ < 1000$ (III), and the outer layer (IV).

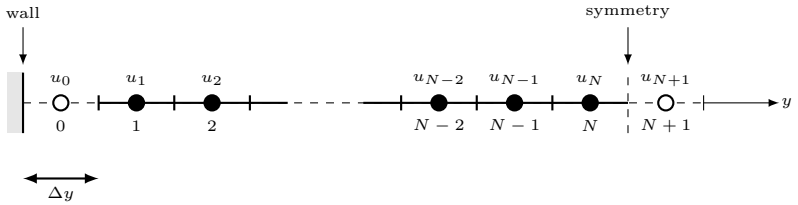

Figure 5.2: Computational domain for the turbulent boundary layer

$$\frac{\overline{u}_0}{u^*} = \frac{1}{\kappa}\ln\left(\frac{y_0 u^*}{\nu}\right) + B \Leftrightarrow u_0^+ = \frac{1}{\kappa}\ln\left(y_0^+\right) + B \tag{5.8}$$

where $B = 5.0$, u_0^+ is the non-dimensional velocity in cell 0, and y_0^+ is the non-dimensional wall-normal coordinate for cell 0. In general the non-dimensional parameters u^+ and y^+ are defined as

$$u^+ = \frac{\overline{u}}{u^*} \tag{5.9}$$

and

$$y^+ = \frac{y u^*}{\nu} \tag{5.10}$$

where $u*$ is the shear velocity, which, by definition, is related to the wall shear stress as

$$u^* \equiv \sqrt{\frac{\tau_w}{\rho}} \tag{5.11}$$

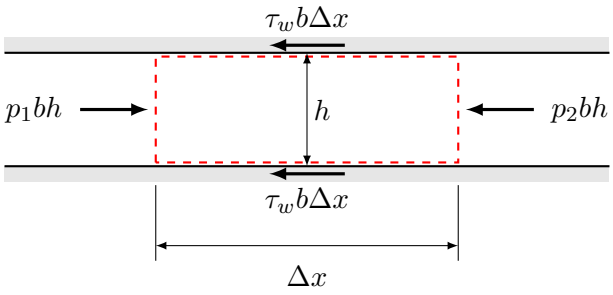

Figure 5.3: Forces on a control volume with length Δx (the channel height is h and the width into the paper is b).

The approach taken to define the boundary condition is that the cell center of the ghost cell (y_0) is selected such that $y_0^+ = 30$ and thereby we have ensured that the first cell is within region of the boundary layer referred to as the log region (see Fig. 5.1) and thus the log law given by Eqn. 5.8 is applicable. It should be noted, however, that we do not a have a value for the wall shear stress τ_w and thus u^* cannot be calculated. However, for the fully developed channel flow that we are analyzing it is possible to relate the wall shear stress to the pressure gradient. Setting up a control volume in the channel as shown in Fig. 5.3 and balancing the forces in the flow direction gives a relation between the pressure difference over the control volume and the wall shear stress as

$$(p_1 - p_2)bh - 2\tau_w b\Delta x = 0 \Rightarrow \tau_w = \frac{(p_1 - p_2)h}{2\Delta x} = \{p_1 - p_2 = -\Delta p\} = -\frac{\Delta p}{\Delta x}\frac{h}{2}$$

where τ_w can be replaced using the definition of u^* (Eqn. 5.11)

$$\rho u^{*2} = -\frac{\Delta p}{\Delta x}\frac{h}{2}$$

Now, let the length of the control volume (Δx) go to zero

$$u^* = \sqrt{\left(-\frac{\partial p}{\partial x}\right)\frac{h}{2\rho}} \tag{5.12}$$

which is the relation between u^* and the pressure gradient that we were looking for. Since we have specified that $y_0^+ = 30$, we can use the definition of y^+ (Eqn. 5.10) to calculate the corresponding y-coordinate (y_0) as a function of the pressure gradient.

$$y_0^+ = \frac{y_0 u^*}{\nu} = \frac{y_0}{\nu}\sqrt{\left(-\frac{\partial p}{\partial x}\right)\frac{h}{2\rho}} \Rightarrow y_0 = y_0^+\nu \Big/ \sqrt{\left(-\frac{\partial p}{\partial x}\right)\frac{h}{2\rho}} \tag{5.13}$$

Moreover, the log law (Eqn.5.8) gives us the velocity $\overline{u}_0$ as a function of the pressure gradient according to

$$\overline{u}_0 = \left(\frac{1}{\kappa}\ln\left(y_0^+\right) + B\right)u^* = \left(\frac{1}{\kappa}\ln\left(y_0^+\right) + B\right)\sqrt{\left(-\frac{\partial p}{\partial x}\right)\frac{h}{2\rho}} \tag{5.14}$$

The calculated coordinate y_0 will define the spatial discretization and $\overline{u}_0$ the ghost cell velocity defining the boundary condition.

5.2 Numerical simulation

As for the laminar flow case, we will use finite-volume discretization and the equations for each of the cells are obtained by integrating Eqn. 5.7 over the cells and inserting approximations of the velocity derivatives at the cell faces.

Integration of Eqn. 5.7 over cell i gives

$$\left(\left(\mu + \rho\kappa^2 y^2\frac{\partial\overline{u}}{\partial y}\right)\frac{\partial\overline{u}}{\partial y}\right)_{i+\frac{1}{2}} - \left(\left(\mu + \rho\kappa^2 y^2\frac{\partial\overline{u}}{\partial y}\right)\frac{\partial\overline{u}}{\partial y}\right)_{i-\frac{1}{2}} = \frac{\partial\overline{p}}{\partial x}\Delta y_i$$

Now, we need to insert cell face coordinates and approximations of the velocity gradients at the cell faces. In the following it is assumed that the mesh is equidistant and thus $\Delta y_i = \Delta y$

$$\left(\left(\mu + \rho\kappa^2 \left(\frac{y_{i+1} + y_i}{2}\right)^2 \frac{\overline{u}_{i+1} - \overline{u}_i}{\Delta y}\right) \frac{\overline{u}_{i+1} - \overline{u}_i}{\Delta y}\right) - \left(\left(\mu + \rho\kappa^2 \left(\frac{y_i + y_{i-1}}{2}\right)^2 \frac{\overline{u}_i - \overline{u}_{i-1}}{\Delta y}\right) \frac{\overline{u}_i - \overline{u}_{i-1}}{\Delta y}\right) = \frac{\partial \overline{p}}{\partial x} \Delta y$$

Collecting terms gives

$$\begin{aligned} &\left(\mu + \rho\kappa^2 \left(\frac{y_i + y_{i-1}}{2}\right)^2 \frac{\overline{u}_i - \overline{u}_{i-1}}{\Delta y}\right) \overline{u}_{i-1} - \\ &\left(2\mu + \rho\kappa^2 \left(\left(\frac{y_{i+1} + y_i}{2}\right)^2 \frac{\overline{u}_{i+1} - \overline{u}_i}{\Delta y} + \left(\frac{y_i + y_{i-1}}{2}\right)^2 \frac{\overline{u}_i - \overline{u}_{i-1}}{\Delta y}\right)\right) \overline{u}_i + \\ &\left(\mu + \rho\kappa^2 \left(\frac{y_{i+1} + y_i}{2}\right)^2 \frac{\overline{u}_{i+1} - \overline{u}_i}{\Delta y}\right) \overline{u}_{i+1} = \frac{\partial \overline{p}}{\partial x} \Delta y^2 \end{aligned} \tag{5.15}$$

In Eqn. 5.15 above, i takes all values from 1 to N. As in the laminar case, the system of equations obtained if applying Eqn. 5.15 to all cells in the computational domain can be expressed in matrix form as $A\mathbf{u} = \mathbf{b}$, but now matrix A will depend on $\mathbf{u}$ and $\mathbf{y}$, i.e. this problem cannot be solved directly so we will have to use an iterative method.

1. Initialize vector $\mathbf{u}$ with a velocity value for each cell (start guess)
2. Calculate the coefficients of A based on $\mathbf{u}$ and $\mathbf{y}$
3. Invert Matrix A
4. Calculate $\mathbf{u}$ as $\mathbf{u} = A^{-1}\mathbf{b}$

   Iterate 2-4 until converged

The provided `Python` function `turbulent_flow_solver` in the `Jupyter NoteBook` does just that.

Task 2.2:

In the main code for task 2.2 in the `Jupyter NoteBook`, set the number of cells to $N = 1000$. Run the turbulent flow solver for 50, 100, 200, and 300 iterations and monitor the development of the solution.

a) How does the number of iterations affect the velocity profile?

b) Suggest a measure of convergence for the numerical simulation and estimate the number of iterations needed to reach convergence.

c) When converged, the velocity profile has significantly lower velocities than the parabolic profile that was used as a starting guess. What is the reason for that? Suggest an alternative starting guess.

d) Calculate the Reynolds number for the turbulent flow.

Write down your answer in `Jupyter Lab`. Derivations made using pen and paper or tablet can be included as photos/images in the `Jupyter NoteBook`.

Task 2.3:

In the provided `Python` function `turbulent_flow_solve`, there is a variable called `relax` used as follows:

```
U = relax*spsolve(A,b)+(1.-relax)*Uold
```

where `U` is a vector containing the velocity values in all cells, `A` is a coefficient matrix, `b` is the right-hand side vector, `spsolve` is a `Python` function solving linear equation systems (in our case `AU=b`), and `Uold` is a vector containing the velocity values from the previous iteration. What does the `relax` variable do?

Write down your answer in `Jupyter Lab`. Derivations made using pen and paper or tablet can be included as photos/images in the `Jupyter NoteBook`.

Task 2.4:

As mentioned previously, u^+ is a non-dimensional velocity and y^+ is a non-dimensional wall-normal coordinate. The definitions of y^+, u^+, and the shear velocity u^* are repeated here for convenience.

$$y^+ = \frac{yu^*}{\nu}$$

$$u^+ = \frac{\overline{u}}{u^*}$$

$$u^* \equiv \sqrt{\frac{\tau_w}{\rho}}$$

a) Why is it common to present the boundary layer velocity profile in terms of y^+ and u^+ rather than y and $\overline{u}$?

b) What range of y^+ values corresponds to the viscous sublayer?

c) What range of y^+ values defines the log-region?

d) How does the turbulence viscosity μ_t compare to the fluid viscosity μ in the viscous sublayer and in the log-region, respectively?

e) How does the total shear stress vary with distance from the wall in the viscous sublayer and in the log-region, respectively?

f) How are the velocity profiles characterized mathematically in the viscous sublayer and in the log-region, respectively?

Write down your answer in `Jupyter Lab`. Derivations made using pen and paper or tablet can be included as photos/images in the `Jupyter NoteBook`.

Task 2.5:

In the `Jupyter NoteBook`, you have been provided with average velocity and corresponding wall-normal coordinates obtained from measurements of a turbulent boundary layer over a **flat plate** in a wind tunnel (also available in Appendix A in this document).

a) Obtain an estimate of the shear velocity (u^*) from the provided data. Follow the procedure described in section 5.3 below and make changes to the `Python` code provided in the `Jupyter NoteBook` where indicated.

   What u^* value do you get?

b) With the estimate of u^* from the experimental data, you can now convert the provided $\overline{u}$ and y values to u^+ and y^+ values. Compare these values with the corresponding data from your numerical simulation by plotting u^+ as a function of y^+ for the experiments and the numerical simulation in the same plot (use `semilogx` instead of `plot`).

   Compare the two profiles and comment on any differences and try to draw some conclusions about the accuracy of the numerical solution and/or the measurements based on your observations.

Write down your answer in `Jupyter Lab`. Derivations made using pen and paper or tablet can be included as photos/images in the `Jupyter NoteBook`.

5.3 Estimation of shear velocity from experimental data

In order to obtain an estimate of u^* from a velocity profile, you need to find a location within the log-layer. Assuming that the data is provided in the form of $\overline{u}$ and y vectors, go through the y-values starting with the one closest to the wall and follow the procedure suggested below.

1. make a first estimate of the wall shear stress τ_{w_0} using a rough estimate of the velocity derivative at the wall

$$\tau_{w_0} = \mu \left.\frac{\partial \overline{u}}{\partial y}\right|_{y=0} \approx \mu \frac{\overline{u}[j]}{y[j]}$$

   where j is the vector index (closest to the wall $j = 0$)

2. based on the estimate of the wall shear stress (τ_{w_0}), calculate the corresponding values of u^*, y^+, and u^+ (u_0^*, y_0^+, and u_0^+)

3. check if y_0^+ (the rough estimate of y^+) is within the log-region. If so, you can move to step 4 below and use τ_{w_0}, y_0^+, and u_0^+ as initial values for an iterative Newton-Raphson solver described in detail below. If y_0^+ is not within the log-layer, move to the next point in the boundary layer ($j = j + 1$) and repeat steps 1 and 2 above. Keep moving away from the wall until you find a location within the log-region.

4. The Newton-Raphson solver describe in section 5.3.1 will give a better estimate of τ_w. When converged, verify the the y^+ is still within the log-region. If that is not the case, move one step further out from the wall and repeat steps 1 to 4.

5.3.1 Newton-Raphson solver

The suggested outline gives a Newton-Raphson solver that finds a value for the wall shear stress (τ_w). When converged, the value of τ_w obtained from the solver can be used to get an estimate of u^*, which was what we were looking for.

To set up the Newton-Raphson solver, we will first rewrite the log-law

$$u^+ = \frac{1}{\kappa}\ln(y^+) + B$$

using the definitions of y^+, u^+, and u^* to get a function $f(\tau_w)$

$$f(\tau_w) = \frac{1}{\kappa}\ln y^+ - u^+ + B = \frac{1}{\kappa}\ln\left(\frac{y[j]\sqrt{\tau_w}}{\sqrt{\rho}\nu}\right) - \frac{\overline{u}[j]\sqrt{\rho}}{\sqrt{\tau_w}} + B$$

The derivative of the function f is obtained as

$$f'(\tau_w) = \frac{(1/\kappa)\sqrt{\tau_w} + \overline{u}[j]\sqrt{\rho}}{2\tau_w^{3/2}} = \frac{(1/\kappa) + u^+}{2\tau_w}$$

With the functions $f(\tau_w)$ and $f'(\tau_w)$ defined, we can set up an iterative Newton-Raphson solver to find τ_w using

$$\tau_{w_{n+1}} = \tau_{w_n} - \frac{f(\tau_{w_n})}{f'(\tau_{w_n})}$$

where $n+1$ and n are iteration numbers. Iterate until converged with the following convergence criterium:

$$\left|\frac{f(\tau_{w_n})}{f'(\tau_{w_n})}\right| \leq \tau_w \times 10^{-4}$$

References

F. M. White. *Fluid Mechanics*. McGraw-Hill, New York, 8 edition, 2016.

A Measured Turbulent Boundary Layer Data

y $[m]$	$\overline{u}$ $[m/s]$	u_{rms} $[m/s]$	y $[m]$	$\overline{u}$ $[m/s]$	u_{rms} $[m/s]$
0.00015000	3.05100000	0.77100000	0.00452500	6.82800000	0.65400000
0.00023100	3.53100000	0.82600000	0.00490000	6.90500000	0.64800000
0.00030000	3.91700000	0.85800000	0.00527500	6.98500000	0.63200000
0.00048100	4.57900000	0.87400000	0.00605000	7.14800000	0.62600000
0.00060000	4.87900000	0.85500000	0.00680000	7.26500000	0.61600000
0.00070600	5.08600000	0.83600000	0.00760000	7.41700000	0.59900000
0.00093100	5.38500000	0.82600000	0.00830000	7.52000000	0.58600000
0.00118100	5.63700000	0.77200000	0.00950000	7.69300000	0.56400000
0.00140600	5.80500000	0.75900000	0.01057000	7.82900000	0.54700000
0.00165600	5.92800000	0.73000000	0.01200000	7.98400000	0.52200000
0.00188100	6.05000000	0.71900000	0.01400000	8.23300000	0.48400000
0.00213100	6.13900000	0.70400000	0.01600000	8.44900000	0.44000000
0.00235600	6.22500000	0.69900000	0.01900000	8.68300000	0.37800000
0.00258100	6.32000000	0.69600000	0.02200000	8.89700000	0.30800000
0.00283100	6.42400000	0.68400000	0.02500000	9.03400000	0.22200000
0.00305600	6.46300000	0.67500000	0.03000000	9.12100000	0.11500000
0.00330600	6.53100000	0.67900000			
0.00377500	6.65700000	0.66300000			
0.00415000	6.75300000	0.66200000			