Chapter 10

Hands-on exercises

10.1 Introduction

This final chapter contains four hands-on exercises. The exercises are designed to cover a wide variety of turbulent flows (in most cases using actual real data sets!), analysis methods, as well as modelling techniques, as covered in the previous chapters. All exercises are designed to be solved in Matlab, and have been developed over many years as report assignments within the Turbulence Theory/Turbulent Flows course taught at the Technical University of Denmark (DTU).

Exercise 1 considers analysis of the flow and turbulence properties stemming from measurements made in an **open channel flow**, and is hence of direct relevance to material covered in Chapters 2 and 3. Exercise 2 considers the development and application of a simple model of **turbulent dispersion**, with obvious relevance to Chapter 8. Exercise 3 considers the statistical, correlation, and spectral analysis of measured velocities in a **turbulent air jet**. It is therefore especially relevant to the concepts introduced in Chapter 4. Finally, Exercise 4 focuses directly on **turbulence modelling** of unsteady (oscillatory) **wave boundary layer** flows, including comparison with data measured in an oscillating tunnel flow on both **smooth** and **rough beds**. It is hence especially relevant to material from Chapters 5, 6 and 9.

The authors sincerely hope that these exercises give the interested reader a true “hands-on” experience in the analysis and modelling of turbulent flows. In working through the exercises, it is likewise intended that students experience in the joys (and sometimes challenges) of working with and interpreting actual data measured in laboratory settings.

10.2 Data sets

As discussed above, three of the four exercises involve analysis of and/or comparison against actual real life data sets, which have been measured in laboratories at the Technical University of Denmark (DTU). Details on the contents of each data set will be described directly in the relevant exercise statements that follow; see the forthcoming Tables 10.2, 10.3 and 10.5. The individual data sets correspond to:

- Exercise 1: Steady turbulent open channel flow (filename: `Exercise1.mat`);
- Exercise 3: Turbulent air jet flow (filename: `Exercise3.mat`);
- Exercise 4: Oscillatory wave boundary layer flow (Jensen et al., 1989, filename: `Exercise4.mat`).

All of the digital material intended to accompany the present book (the three data sets above, the MatRANS model described in the next section, as well as the Matlab examples from Chapter 4) can be found in the data collection available online at:

`https://doi.org/10.11583/DTU.c.4508648`

10.3 MatRANS k-ω turbulence model in Matlab

Several of the exercises also involve applications of an actual turbulence closure model, and for this purpose the authors provide the MatRANS model (a simple one-dimensional vertical, 1DV, model solving Reynolds-averaged Navier-Stokes equations, coupled with a k-ω model in Matlab). This model is as originally described by Fuhrman et al. (2013), with subsequent developments to the turbulence modelling capabilities made by Williams and Fuhrman (2016) and Kirca et al. (2016), who extended the model to include the "low Reynolds number" version.

This model has been described in detail in the present book in Chapter 5, Section 5.12, with the final 1DV formulation corresponding to the framed equations in that section (primarily Eqs. 5.147, 5.162 and 5.163, combined

Table 10.1 Description of contents from MatRANS output file specified in the `OutFileName` variable. When loaded this will contain the single structure array `MatRANS` described below. Note that the period T and free stream velocity magnitude U_{0m} are only relevant for wave boundary layer simulations.

Structure array	Field	Book notation	Size	Description
`MatRANS`			1×1	Selected output from MatRANS simulation
	`n_t`	n_t	1×1	Number of saved time levels (s)
	`n_y`	n_y	1×1	Number of y grid points (s)
	`t`	t	$n_t \times 1$	Saved time levels (s)
	`y`	y	$1 \times n_y$	Model y positions (m)
	`u`	$\overline{u}(t, y)$	$n_t \times n_y$	Horizontal velocities (m/s)
	`k`	$k(t, y)$	$n_t \times n_y$	Turbulent kinetic energy density (m^2/s^2)
	`omega`	$\omega(t, y)$	$n_t \times n_y$	Specific dissipation rate (1/s)
	`nu_t`	$\nu_t(t, y)$	$n_t \times n_y$	Kinematic eddy viscosity (m^2/s)
	`tau0`	$\overline{\tau}_0(t)$	$n_t \times 1$	Bed shear stress (Pa)
	`nu`	ν	1×1	Fluid kinematic viscosity (m^2/s)
	`rho`	ρ	1×1	Fluid density (kg/m^3)
	`k_s`	k_s	1×1	Bottom roughness (m)
	`h_m`	h_m	1×1	Model domain height (m)
	`T`	T	1×1	Wave period (s)
	`U0m`	U_{0m}	1×1	Free stream velocity magnitude (m)

with the eddy viscosity defined in Eq. 5.164). The model can be downloaded (filename: `MatRANS.zip`) from the data collection link provided at the end of Section 10.2.

Examples to be used in the following exercises are provided already set up, and can be found in the `MatRANS/TurbulenceBook/` folder, which is then sub-divided into `Exercise1` and `Exercise4` folders. Running the model is straightforward, accomplished by simply executing the `runMatRANS.m` script! Any alterations required to the input will be explained directly in the relevant exercise statements (namely, Exercise 4).

In all cases, upon completion selected results from the provided book examples will be exported to a single output file, with name corresponding to the `OutFileName` input string variable (e.g. `out_MatRANS.mat`). This output file consists of a single Matlab data structure array called `MatRANS`, with fields described in detail in Table 10.1. All field names have been chosen to closely coincide with the book notation. With this, it should be straightforward for users familiar with Matlab to make all desired animations or plots requested in the following exercises. Code to make some example plots of the flow/turbulence profiles are provided at the end of the `runMatRANS.m` input execution scripts to help facilitate this.

10.4 Exercise 1: Analysis of a turbulent boundary layer in an open channel

Experiments

For this exercise you will consider a series of velocity measurements made in a steady and (streamwise) uniform boundary layer flow in the tilting flume at the Hydraulic Engineering Laboratory at the Technical University of Denmark (DTU). A two-component Laser Doppler Velocimeter (LDV), also known as Laser Doppler Anemometer (LDA), was used to measure the u and v components of the water velocity, where x is in the streamwise direction and y in the vertical direction. The flow depth in the experiments was maintained at $h = 7$ cm, and the flow velocity at the water surface at $U_0 \approx 30$ cm/s. The cross section of the flume is rectangular, with width $b = 30$ cm. The u and v velocities were measured at the following 23 y (cm) locations: 0.02, 0.03, 0.05, 0.07, 0.1, 0.14, 0.18, 0.2, 0.25, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 1.0, 1.25, 1.5, 2.0, 2.5, 3.0, 4.0, 5.0. The sampling length was approximately 180 s for each point, and the sampling interval was set automatically by the LDA equipment (but will be in all circumstances smaller than O(20 ms)). For the purposes of this exercise, you may assume that the kinematic water viscosity is $\nu = 10^{-6}$ m^2/s.

Data input, processing and analysis

All data processing is intended to be done in Matlab. The data set to be analyzed is provided in the workspace file `Exercise1.mat` from the steady turbulent open channel flow data set. This can be obtained as presented in Section 10.2. This file can be easily imported into Matlab using the `load` command. Once loaded, the data is contained entirely in the single Matlab data structure variable `Channel`, the contents of which are described in full detail in Table 10.2.

The naming of the various fields comprising the `Channel` variable is seen to closely coincide with the book notation; Data can hence hopefully be easily and intuitively accessed. For example: `Channel(5).u` will return the raw u values from the fifth measurement point, etc. Other data can be similarly accessed by replacing the field `u` with any of those indicated in Table 10.2. Similarly, data from other measurement positions can be accessed simply by replacing the `5` with any other integer ranging from `1` to `23`.

Table 10.2 Description of contents for Matlab data set contained in `Exercise1.mat`. The number of measurements (sample length) N for each position varies.

Structure array	Field	Book notation	Size	Description
`Channel`			1×23	Data set for 23 y measurement positions
	`y`	y	1×1	Measurement position (m)
	`u`	u	N	Streamwise velocities (m/s)
	`v`	v	N	Vertical velocities (m/s)
	`t`	t	N	Measurement times (s)
	`tt`	t_t	N	Travel times (s)
	`h`	h	1×1	Channel depth (m)
	`nu`	ν	1×1	Kinematic water viscosity (m^2/s)

Since the data set includes irregularly sampled data, some form of weighting technique must be used to properly estimate statistical quantities (mean, variance, etc.). For example, weighting based simply on the the transit time t_t (corresponding to the time taken for a measured flow particle to pass through the measurement volume) will correspond (e.g. for the streamwise component) to using for the mean:

$$\overline{u} = \frac{\sum_{i=1}^{N} u_i t_{t,i}}{\sum_{i=1}^{N} t_{t,i}} \tag{10.1}$$

and for the r.m.s. of the turbulent fluctuations:

$$\sqrt{\overline{u'^2}} = \left(\frac{\sum_{i=1}^{N} u'^2 t_{t,i}}{\sum_{i=1}^{N} t_{t,i}} \right)^{1/2} \tag{10.2}$$

where N is the sample length. Analogous expressions may be used for other quantities (e.g. $\sqrt{\overline{v'^2}}$ and $\sqrt{-\overline{u'v'}}$; For the latter case e.g. u'^2 should be replaced with $-u'v'$ within the numerator of Eq. 10.2).

Action items

(1) Plot the mean velocity $\overline{u}$ as function of y using linearly scaled axes. In addition to the raw data, you may include the points $u(y = 0) = 0$ (no slip at the bed) and $u(y = h) = U_0 = 0.30$ m/s for completeness.

(2) From the velocity profile $\overline{u}(y)$ in item 1 calculate the depth-averaged velocity V via

$$V = \frac{1}{h} \int_0^h \overline{u} dy \tag{10.3}$$

(Hint: the Matlab function `trapz` may be used to compute the integral.)

(3) Estimate the friction velocity U_f from the Darcy-Weisbach equation

$$U_f = \sqrt{\frac{f}{2}}V \tag{10.4}$$

where f is the friction coefficient, calculated from the Blausius formula

$$f = \frac{0.0557}{Re^{0.25}} \tag{10.5}$$

where $Re = r_h V/\nu$ is the Reynolds number, with $r_h = A/P$ the hydraulic radius (where $A = hb$ is the cross-sectional flow area, and $P = 2h + b$ is the wetted perimeter). Eq. 10.5 is a good approximation to the more general flow-resistance equation, Eq. 3.130, for $Re < 0.25 \times 10^5$ (Schlichting, 1979, p. 597 and Fig. 20.1).

(4) Plot $\overline{u}$ versus y on a semi-log scale (i.e. logarithmic scale for y and linear scale for $\overline{u}$). Identify the approximate lower and upper bounds of the logarithmic region. The upper bound should be taken as the top of the constant stress region ($y \leq 0.1h$ or equivalently $y^+ \leq 0.1Re_\tau$, where $y^+ = yU_f/\nu$ and $Re_\tau = hU_f/\nu$) whereas the lower bound should be taken as where $y^+ = 30$, where U_f is estimated from item 3 above (this bound should be re-checked afterwards!). By fitting a straight line to the data in the logarithmic-layer portion of the velocity profile (i.e. of the form $\overline{u} = m\log(y) + b$), and comparing the result with Eq. 3.42, calculate U_f directly from the data (Hint: use Matlab's `polyfit` function to determine the coefficients m and b above). Compare this U_f with the estimation made in item 3.

(5) Using the U_f from the log-fitting exercise (item 4) above re-plot the velocity profile in a dimensionless fashion i.e. as $\overline{u}/U_f$ vs. y^+ as well as vs. y/h. Identify various regions of the flow e.g. the viscous sublayer, the buffer layer, the logarithmic layer, and the outer region.

(6) Plot the van Driest velocity distribution from Eq. 3.108 on the same diagram and compare the experimentally determined velocity profile with the van Driest profile. (Hint: Calculate the required cumulative integral numerically in Matlab using `cumtrapz`.)

(7) Plot the turbulence data in wall units, namely $\sqrt{\overline{u'^2}}/U_f$, $\sqrt{\overline{v'^2}}/U_f$ and $\sqrt{-\overline{u'v'}}/U_f$ as a function of y^+ over the range $0 \leq y^+ \leq 100$. Can you see that these turbulence quantities become approximately constant once the logarithmic region is reached?

(8) Plot the turbulence data in terms of outer-flow parameters, namely $\sqrt{\overline{u'^2}}/U_f$, $\sqrt{\overline{v'^2}}/U_f$ and $\sqrt{-\overline{u'v'}}/U_f$ as a function of y/h, now spanning the entire flow depth i.e. $0 \leq y/h \leq 1$.

(9) Also, do the same for k/U_f^2, where $k = 1/2\left(\overline{u'^2} + \overline{v'^2} + \overline{w'^2}\right)$ is the turbulent kinetic energy (per unit mass), as a function of y/h. For this purpose, you may use the approximation

$$\overline{w'^2} = 1.8\overline{v'^2} \tag{10.6}$$

for the missing transverse component of the fluctuating velocity.

(10) Calculate the Reynolds stress through

$$-\rho\overline{u'v'} = \overline{\tau} - \mu\frac{\partial\overline{u}}{\partial y} \tag{10.7}$$

where $\overline{\tau}$ is given by

$$\overline{\tau} = \rho U_f^2\left(1 - \frac{y}{h}\right) \tag{10.8}$$

(Hint: For calculating $\partial\overline{u}/\partial y$ use Matlab's `gradient` function.) Plot the non-dimensional Reynolds stress $-\rho\overline{u'v'}/U_f^2$ as a function of y/h. Compare the latter with that obtained directly from the measurements in item 7.

(11) For the steady and streamwise uniform boundary layer under consideration, write out the simplified form of the energy equation for turbulence, Eq. 2.31. This should lead to (among other things) a turbulence energy production term corresponding to $-\rho\overline{u'v'}\partial\overline{u}/\partial y$. Plot this quantity as a function of y/h.

(12) Run the MatRANS k-ω model for this case i.e. simply execute the `MatRANS/TurbulenceBook/Exercise1/runMatRANS.m` script in Matlab, which is already appropriately set up for this case. See again Table 10.1 for a complete description of the model output. Compare the experimental data from above with the turbulence model results for $\overline{u}/U_f$ and k/U_f^2 vs. y/h.

10.5 Exercise 2: A simple numerical model of dispersion in a turbulent boundary layer flow

For this exercise consider the following: A certain amount of dispersant is dumped into a river or channel. The time when the dumping takes place is $t = 0$. It is assumed that the dispersant is distributed uniformly over the cross-section at $x = 0$ at the instant of dumping. The dispersant will be convected in the (streamwise) x-direction. At the same time, it will be subject to dispersion in the x-direction due to the combined effect of turbulent diffusion and the velocity gradient in the y-direction. As a result, the cloud will become more and more dispersed as it is convected downstream.

Flow model

In this exercise, you will study this problem by a numerical simulation of the dispersion process in a channel flow on the computer. For this purpose the same channel flow as in Exercise 1 will be considered, with flow velocity and turbulence properties taken from the end state of the MatRANS turbulence model results used previously (Exercise 1, Action item 12; Note that this simulation is set up such that the flow depth is $h = 0.07$ m and friction velocity is $U_f = 0.0134$ m/s once the steady state is reached). Hence, it is assumed that the `runMatRANS.m` script mentioned there has already been executed, and that the resulting workspace file containing the `MatRANS` structure array (with contents again described in full detail in Table 10.1) has been created. This structure array thus contains the computed horizontal mean flow velocity $\overline{u}$, the turbulent kinetic energy density k, and the specific dissipation rate ω at discrete y positions.

From the basic flow model results the variance of the vertical turbulent fluctuations can be assumed to follow

$$\overline{v'^2} = \sigma_v^2 = \frac{1}{3}k \tag{10.9}$$

Additionally, the mixing length ℓ_m can be taken from Eqs. 9.56 and 9.33 to be

$$\ell_m = \frac{1}{\beta^{*1/4}} \frac{\sqrt{k}}{\omega} \tag{10.10}$$

where the fully turbulent value $\beta^* = 0.09$ (Eq. 9.39) for the closure coefficient can be utilized. This should be valid in the logarithmic region and above.

From the direct model output above physical quantities can be obtained directly at the discrete y positions contained in `MatRANS.y`. Relevant quantities at other y positions can then easily be interpolated as needed (Hint: Matlab's `interp1` function can be utilized, but use of `griddedInterpolant` is much faster and recommended). As it is only the end (steady state) flow conditions that are to be used, it may be convenient to assign the end state flow results as simple vector variables e.g. `u=MatRANS.u(end,:)`, and similarly for `k` and `omega`, though this is not strictly necessary.

Random walk model

With the basic flow properties essentially given, the primary aim of the present exercise is to develop and apply a so-called random walk model for simulating the longitudinal dispersion process in a channel flow. For this purpose consider the motion of a single particle having position $(x_p^{(n)}, y_p^{(n)})$ at time t, where the superscript (n) refers to an integer time index i.e. not raising to a power. The evolution of this particle to a new position $(x_p^{(n+1)}, y_p^{(n+1)})$ at $t + \Delta t$ can be obtained from the following algorithm, which is reasonably similar to that used in Kirca et al. (2016):

- The small time increment Δt can be locally calculated from the mixing length ℓ_m and the standard deviation of the vertical turbulent fluctuations σ_v (both based on position $y = y_p^{(n)}$) according to

$$\Delta t = \frac{\ell_m}{\sigma_v} \tag{10.11}$$

- The instantaneous vertical fluctuation can be obtained by

$$v_p = \alpha_r \sigma_v - w_s \tag{10.12}$$

  where α_r is a normally distributed random number having unit standard deviation (Hint: Use Matlab's `randn` function; You may wish to set a seed at the beginning using Matlab's `rng` function, such that your results are repeatable). In the above w_s is the settling velocity of the particles (hence the special case where $w_s = 0$ corresponds to neutrally buoyant particles.)
- The new vertical particle position at time $t+\Delta t$ can then be computed via

$$y_p^{(n+1)} = y_p^{(n)} + v_p \cdot \Delta t \tag{10.13}$$

  If the obtained $y_p^{(n+1)} > h$ or $y_p^{(n+1)} < y_{bot}$ then the particle should be simply mirrored back into the domain via reflection about the exceeded upper or lower boundary, such that $y_{bot} \leq y_p^{(n+1)} \leq h$. Here y_{bot} defines the lower y limit for the dispersion model domain. For the present purposes this can be taken as where $y_{bot}^+ = y_{bot} U_f/\nu = 70$ i.e. such that y_{bot} defines the beginning of the logarithmic layer.
- The average horizontal (translation) velocity of the particle u_p over the time increment Δt can then be estimated as

$$u_p = \frac{\overline{u}\left(y = y_p^{(n)}\right) + \overline{u}\left(y = y_p^{(n+1)}\right)}{2} \tag{10.14}$$

- Finally, the new horizontal position of the particle can be determined from

$$x_p^{(n+1)} = x_p^{(n)} + u_p \cdot \Delta t \tag{10.15}$$

after which the time level index can be updated from (n) to $(n+1)$, and the process described above repeated.

The algorithm above describes a model for a single particle. However, as many particles as desired can be considered in succession, and the results then combined to demonstrate the dispersion process of a cloud. For the purposes of the present exercise, you should construct your model such that the dispersant will be modelled with N particles released one-by-one, with initial positions distributed uniformly over the modelled flow depth $y_{bot} < y < h$ at $x = 0$ at time $t = 0$.

In the presentation of all results, please use the following non-dimensional quantities:

$$Y = \frac{y}{h}, \qquad X = \frac{x}{h}, \qquad T = \frac{tU_f}{h} \tag{10.16}$$

Released particles should be modelled until the dimensionless time $T_{end} = 25$ is reached. It is noted that the time step Δt (hence ΔT) will be irregularly spaced and vary for each particle modelled. Therefore, once the end time is reached for a given particle, it is suggested to re-sample the particle X and Y positions at regular time intervals, such that the particle positions representing the dispersant cloud can be re-assembled at common times. (Hint: This can again be easily achieved using either the `interp1` or `griddedInterpolant` function.)

Action items

(1) Load the workspace from the MatRANS simulation from Exercise 1. Plot the resulting velocity profile ($\overline{u}/U_f$ vs. y/h), the turbulent kinetic energy density profile (k/U_f^2 vs. y/h), and the resulting mixing length profile (ℓ_m/h vs. y/h) at the end state. As a check, compare the results for the first two of these plots with those presented in Fig. 9.4.

(2) From the velocity profile compute the dimensionless depth-averaged velocity V/U_f, where

$$V = \frac{1}{h}\int_0^h \overline{u} dy \tag{10.17}$$

(Hint: Use `trapz` to compute the integral.)

(3) Consider first cases involving neutrally buoyant particles, i.e. with settling velocity $w_s = 0$. Run the model for some small values of N and show the resulting particle paths of a few particles.

(4) Perform simulations where $N = 100$, 200, 500, 1000, 2000, 5000 (or more!) uniformly distributed particles are released. Plot the mean particle position $\overline{X}$ and the variance $\overline{(X - \overline{X})^2}$ at time $T = 25$ as a function of the number of particles N, and see how many particles are needed for $\overline{X}$ and $\overline{(X - \overline{X})^2}$ to tend to practically constant (converged) values with increasing N.

(5) For the largest N achieved above (in reasonable computational time), calculate $\overline{X}$, $\overline{Y}$ and $\overline{(X - \overline{X})^2}$ at times $T = 5$, 10, 15, 20, and 25. Plot these quantities as a function of time T.

(6) Calculate the (dimensionless) ultimate mean velocity of the cloud from

$$U_m^* = \frac{U_m}{U_f} = \frac{d\overline{X}}{dT} \tag{10.18}$$

How does U_m^* compare with V/U_f from item 2?

(7) Make a probability density function for the end particle positions X and Y (Hint: Consider for simplicity using Matlab's `histogram` function). Compare the probability density functions for X and Y, respectively, with a Gaussian (normal) and uniform distribution.

(8) From the plot of $\overline{(X - \overline{X})^2}$ vs. T above, calculate the non-dimensional longitudinal dispersion coefficient from

$$D_1^* = \frac{D_1}{hU_f} = \frac{1}{2}\frac{d}{dT}\overline{(X - \overline{X})^2} \tag{10.19}$$

(Hint: consider using `polyfit`.) How does D_1^* compare with expectations based on Eq. 8.120. (Hint: If the model is implemented correctly, they should be rather close!)

(9) Now consider a released dispersant that is not neutrally buoyant, but rather one composed of slightly heavy particles having settling velocity $w_s = 0.15U_f$. Repeat item 7 above and make probability density functions for X and Y at the end state for the heavy particles. Compare these with those from the neutrally buoyant particles, and explain the differences.

(10) Finally, calculate the dimensionless ultimate mean velocity (U_m^*) and the longitudinal dispersion coefficient (D_1^*) of the heavy dispersant. Compare these with those of the neutrally-buoyant dispersant and physically explain the reasons for the differences observed.

Table 10.3 Description of contents in Matlab data contained in `Exercise3.mat`.

Structure array	Field	Book notation	Size	Description
`Jet`			1×12	Data set for 12 exit velocities
	`u`	u	500000×1	Flow velocities (m/s)
	`t`	t	500000×1	Measurement times (s)
	`V`	V	1×1	Exit velocity (m/s)
	`D`	D	1×1	Exit diameter (m)
	`nu`	ν	1×1	Kinematic air viscosity (m^2/s)
	`L`	L	1×1	Measurement position (m)

10.6 Exercise 3: Statistical, correlation, and spectral analysis of turbulent air jet flow

Introduction

For this exercise, you will consider a turbulent air velocity signal measured in a free round jet flow in the DTU Mechanical Engineering Fluid Mechanics Laboratory. The experiment was conducted using a jet with exit diameter $D = 0.03$ m with exit velocity V. Velocities were measured utilizing a single hot wire (Constant Temperature Anenometry, CTA, Dantec Dynamics probe type 55P11) sensor, at a fixed distance of $L = 0.6$ m ($20D$) away from the exit, with a sampling frequency of 50,000 Hz. For the purposes of this exercise, you may assume a kinematic fluid (air) viscosity of $\nu = 1.51 \times 10^{-5}$ m^2/s.

The full data set is comprised of 12 tests with various fixed values for V (varied over the range 16.0 m/s $\leq V \leq$ 38.4 m/s). It is provided in the `Exercise3.mat` Matlab workspace file, which can be obtained as presented in Section 10.2. The data is contained in a Matlab structure array variable called `Jet`, with fields fully detailed in Table 10.3. It can be accessed e.g. for Test 12 as: `Jet(12).u` (for the velocities), and similarly for the other fields. All data are in SI units.

For the purposes of this exercise you are to consider primarily Test 12, which had an exit velocity $V = 38.4$ m/s. (Hint: For help in performing many of the action items below, it may be useful to consult the various Matlab examples provided in Chapter 4.)

Action items

(1) Load the data and subtract the mean value from the raw velocity signal for Test 12. Plot the resultant turbulent fluctuating velocity

$u' = u(t) - \overline{u}$ time series. Plot both the full signal, as well as zoomed-in segments, such that you can clearly identify the passing of some individual eddies or coherent structures in the flow.

(2) Report basic statistics for the signal i.e. the mean (of the raw signal) $\overline{u}$, variance σ_u^2, skewness S_u, and kurtosis F_u in a table (see Section 4.1). Also report the turbulence intensity $\sigma_u/\overline{u}$, see e.g. Eq. 4.53.

(3) Plot the probability density function (p.d.f.) $p(u')$ for the turbulent signal, and compare this with the normal distribution from Eq. 4.3.

(4) Plot the time correlation function $R_E(\tau)$ (Eq. 4.38) for the measured turbulent signal. (For the purposes of this exercise it is suggested to compute $R_E(\tau)$ until the first zero crossing, see Program 3 in Chapter 4.) From this function determine the Eulerian macro time scale T_E and the Eulerian micro time scale τ_E. Can you identify regions of passing coherent structures corresponding approximately to these time scales in the zoomed-in time series plotted in item 1 above?

(5) Making use of Taylor's frozen turbulence approximation, e.g. Eq. 4.55, work out approximations for the corresponding macro (Λ_f) and micro (λ_f) length scales of turbulence. Discuss the applicability of this approximation, based on the turbulence intensity $\sigma_u/\overline{u}$ computed in item 2 above.

(6) Assuming isotropic turbulence as an approximation, can you also estimate the Kolmogorov time scale τ_K, the dissipation rate (per unit mass) ε, and the Kolmogorov length scale η_K? (Hint: See Section 4.3.3.)

(7) Make a log-log plot of the raw (using fast Fourier transform FFT) energy density spectra $S(f)$ in the frequency domain. Be sure to check that the integral of the spectrum yields the variance! (Hint: Please make sure you are analyzing u' and not u!) Also add a smoothed out version of this spectrum (e.g. using Matlab's `pwelch` function).

(8) Assuming Taylor's frozen turbulence approximation applies such that wave numbers can be determined by $k = 2\pi f/\overline{u}$, convert the frequency domain spectrum $S(f)$ above to a one-dimensional wave number spectrum $F(k)$, according to Eq. 4.125. Be sure to check that the integral of this spectrum now yields half the variance, in accordance with the convention from Eq. 4.124. Compare the resulting wave number spectrum with the von Karman model spectrum given in Eq. 4.131.

(9) On the wave number spectrum, identify the energy containing range, the universal equilibrium range, the inertial subrange, and the dissipation subrange.

(10) Convert the characteristic length scales found above to characteristic wave numbers by taking their inverse i.e. using $k_{macro} \simeq \Lambda_f^{-1}$, $k_{micro} \simeq \lambda_f^{-1}$ and $k_{Kolmogorov} \simeq \eta_K^{-1}$. Identify where these are located on your wave number spectrum $F(k)$. Do these fall into the expected ranges identified above?

(11) Repeat the analysis above considering all of the other measurement signals having other values for the exit velocity V (Test 1–12). For each test record the three characteristic length scales Λ_f, λ_f and η_K as well as the Reynolds number $Re = VD/\nu$. Make scatter plots (log-log scaled) of the resulting length scale ratios Λ_f/η_K and Λ_f/λ_f versus Re. Also add trend lines such that $\Lambda_f/\eta_K \sim Re^{3/4}$ and $\Lambda_f/\lambda_f \sim Re^{1/2}$, as suggested by Eqs. 4.92 and 4.93. Do the measured scale ratios follow the expected Reynolds number dependence?

10.7 Exercise 4: Turbulence modelling of oscillatory wave boundary layer flows

Introduction

In this exercise you will perform a numerical modelling study of oscillatory wave boundary layers for various Reynolds numbers, as well as on both smooth and rough beds. In doing so, you will obtain first-hand experience in turbulence modelling, by properly setting up and running the provided MatRANS k-ω model in Matlab (Fuhrman et al., 2013). You will likewise compare your model results with either theory or laboratory measurements made in DTU's oscillating tunnel environment (Jensen et al., 1989).

Model description

The model to be used in this assignment is the so-called MatRANS model, originally developed by Fuhrman et al. (2013). The model solves one-dimensional vertical (1DV) Reynolds-averaged Navier-Stokes (RANS) equations, coupled with two-equation (Wilcox, 2006) k-ω turbulence closure ($k = 1/2\overline{u_i'u_i'}$ being the turbulent kinetic energy density, ω being the specific dissipation rate), in the Matlab environment. The model is as originally described in Fuhrman et al. (2013), with extension to include the "low Reynolds number" version, as described by Williams and Fuhrman (2016) and Kirca et al. (2016). The resulting model equations are described in full detail in Section 5.12, where the final governing 1DV equations correspond to the framed (boxed) equations in that section (primarily Eqs. 5.147, 5.162 and 5.163, combined with the eddy viscosity defined in Eq. 5.164). The model is again available for download at the link provided at the end of Section 10.2.

Note that while the full model includes a method for approximating convective terms, as described in Section 5.12 (underlined terms), these are to be neglected in the present exercise (achieved by keeping `streaming=0` in the model input). This is appropriate for situations where the flow is statistically uniform in the streamwise direction, as in a tunnel environment.

The driving pressure gradient is as described in Eqs. 5.159 and 5.161, again with all underlined terms neglected and with $P_x = 0$, such that a free stream velocity of the form

$$U_0 = U_{0m} \sin\left(\frac{2\pi}{T} t\right) \tag{10.20}$$

will be driven, where T is the wave period. As boundary conditions, a frictionless rigid lid is imposed at the top boundary, whereby vertical derivatives of $\overline{u}$, k, and ω are set to zero. Alternatively, the bottom boundary is considered a friction wall, and a no-slip boundary condition is imposed, i.e. velocity variables are set to $\overline{u} = 0$, as well as $k = 0$. The bottom boundary condition for the specific dissipation rate ω is given according to Eq. 9.47.

For the purposes of this exercise you are to utilize the transitional (low Reynolds number) version of the model (achieved by keeping `turb=2` in the model input, as provided), wherein selected closure coefficients are flow dependent as specified in Eq. 9.43–9.46, and all others are specified according to Eqs. 9.39–9.41 (with $\beta = \beta_0$ as for two-dimensional flows). This single model should be capable of simulating both laminar and turbulent regimes, including transition from one to another, without modification.

Model input and output

All model input variables are set in the `runMatRANS.m` file, a working example of which is provided in the:

`MatRANS/TurbulenceBook/Exercise4/Case1`

directory, which is hopefully self-explanatory. The only input variables that should require changing to complete this exercise are: `U_1m` (corresponding to the free stream velocity magnitude U_{0m}), `dy0` (corresponding to the near wall grid size Δy), and in a single case `k_s` (corresponding to Nikuradse's

Table 10.4 Summary of wave boundary layer conditions to be considered. The corresponding test from the experimental work of Jensen et al. (1989) is also indicated. All calculations below use the reported kinematic viscosity $\nu = 1.14 \times 10^{-6}$ m^2/s.

Case	Test no. from Jensen et al. (1989)	T (s)	U_{0m} (m/s)	k_s (m)	Re	a/k_s
1	3	9.72	0.23	smooth	7.2×10^4	–
2	7	9.72	0.68	smooth	6.3×10^5	–
3	10	9.72	2.0	smooth	5.4×10^6	–
4	13	9.72	2.0	0.00084	5.4×10^6	3700

equivalent sand grain roughness k_s). All physical variables in the model have SI units.

Following simulation of each case, a single output file will be created as explained in full detail in Section 10.3, and organized as indicated in Table 10.1. Following each simulation, it is strongly encouraged that you view an animation of the results e.g. by executing the provided `animateMatRANS.m` script, which also serves as an example on how to access and plot the model results.

Turbulence modelling cases

For the present exercise you will consider a series of four cases summarized in Table 10.4, where the Reynolds number Re is defined according to Eq. 5.1. The model is to be set up to mimic wave boundary layers in DTU's oscillating water tunnel facility, see Fig. 5.21. For the purposes of this exercise, you are to use a model height corresponding to half the physical tunnel height (0.29 m) i.e. $h_m = 0.145$ m, hence it is only the bottom boundary layer that will be simulated (in reality, wave boundary layers will develop at both the top and bottom boundaries in a tunnel environment!). For each case, you are to simulate five wave periods, utilizing only the results from the last period for your analysis. This duration is sufficient to effectively achieve cyclic behavior in the model domain. (Please note that it is implied that the phase starts over at $\omega t = 0°$ at the beginning of each period.)

Data set

Note that each of the selected cases corresponds to experimental conditions considered e.g. in Jensen et al. (1989), hence selected (processed) data from these experiments will be utilized for comparison. The data set is provided in the Matlab workspace file `Exercise4.mat`. This can be obtained as pre-

Table 10.5 Description of contents in Matlab data contained in `Exercise4.mat`. The number of measurement positions for each case n_y varies for each variable and from case to case.

Structure array	Field	Book notation	Size	Description
`WBL`			1×4	Selected data for 4 experiments
	`test`	–	1×1	Test no. from Jensen et al. (1989)
	`T`	T	1×1	Period (s)
	`U0m`	U_{0m}	1×1	Free stream velocity magnitude (m/s)
	`k_s`	k_s	1×1	Bottom roughness (m) (or `smooth`)
	`omegat`	ωt	1×13	Phase angles (°) for velocities
	`y_u`	y	$n_y \times 13$	y positions (m) for $\overline{u}$
	`u`	$\overline{u}(y, \omega t)$	$n_y \times 13$	Ensemble averaged velocities (m/s)
	`y_uuvv`	y	$n_y \times 13$	y positions (m) for $\overline{u'u'}$, $\overline{v'v'}$
	`uu`	$\overline{u'u'}(y, \omega t)$	$n_y \times 13$	Reynolds normal stresses (m^2/s^2)
	`vv`	$\overline{v'v'}(y, \omega t)$	$n_y \times 13$	Reynolds normal stresses (m^2/s^2)
	`y_uv`	y	$n_y \times 13$	y positions (m) for $\overline{u'v'}$
	`uv`	$\overline{u'v'}(y, \omega t)$	$n_y \times 13$	Reynolds shear stresses (m^2/s^2)
	`omegat_tau0`	ωt	varies	Phase angles (°) for τ_0
	`tau0`	τ_0	varies	Bed shear stress (Pa)
	`nu`	ν	1×1	Kinematic water viscosity (m^2/s)
	`rho`	ρ	1×1	Water density (kg/m^3)

sented in Section 10.2. Once loaded, the data is contained in the single Matlab structure array `WBL`, with fields as indicated in Table 10.5, allowing (hopefully) easy and intuitive access for analyzing and plotting. For example, the following Matlab commands:

```
c=3; n=7; WBL(c).omegat(n),
plot(WBL(c).u(:,n),WBL(c).y_u,'o')
```

will: (1) output the phase being considered to the screen (in this example $\omega t = 90°$, since `n=7` and the velocity data is provided at 15° increments starting at 0°) for Case 3 (since `c=3`) and (2) plot the ensemble averaged velocity profile $\overline{u}(y)$ for this case at this phase. Additionally, the time series of the bed shear stress $\overline{\tau_0}(\omega t)$ can be obtained by plotting the field `tau0` versus `omegat_tau0`. All fields included in the `WBL` structure array are intended to be self explanatory and are consistent with the book notation.

Action items

(1) Based on your knowledge of wave boundary layers from Chapter 5, for each of the four cases make an *a priori* estimate of the wave friction factor f_w and the expected maximum friction velocity U_{fm}. For this

purpose it is imperative that you utilize an expression for f_w that is appropriate for the expected flow regime (i.e. laminar, Eq. 5.59; smooth-turbulent, Eq. 5.60; or rough-turbulent, Eq. 5.69). [Note that if a case falls into a transitional regime, e.g. Case 2, then a conservative choice of f_w should be used i.e. one resulting in the largest U_{fm}.]

(2) Use the estimated U_{fm} values to determine the required near-wall grid spacing Δy for each case, such that Eq. 9.49 is satisfied. For Cases 1–3 (smooth beds) also determine an appropriate roughness k_s (model variable `k_s`) such that $k_s^+ = k_s U_{fm}/\nu = 0.1$ is satisfied. Based on your *a priori* calculations, what is the expected maximum roughness Reynolds number k_s^+ for Case 4? Is this case in the hydraulically rough regime? Please list all of the values found for items 1 and 2 in a table.

(3) **Case 1: Laminar wave boundary layer,** $Re = 7.2 \times 10^4$.
In the provided `Case1` directory verify that the model input for Case 1 in the `runMatRANS.m` script is correct and in accordance with your calculations above. Run the simulation simply by executing the `runMatRANS.m` script. Simulation results will be saved into the output workspace file `out_MatRANS.mat`, as designated by the `OutFileName` model input variable. When the simulation is complete be sure to animate the results to make sure everything looks reasonable.

Make plots comparing the computed and theoretical laminar velocity profiles (u/U_{0m} vs. y/a) at four phases: $\omega t = 0°$, $45°$, $90°$, and $135°$, where now $\omega = 2\pi/T$ is the wave angular frequency. Additionally, make a plot comparing the computed bed shear stress time series ($\tau_0/(\rho U_{0m}^2)$ versus ωt) with the laminar theory.

(4) **Case 2: Transitional wave boundary layer,** $Re = 6.3 \times 10^5$.
Now copy your `Case1` directory to a new directory called `Case2`, adjust the necessary model input, and run the simulation for Case 2. After animating the results, make a plot again comparing the computed bed shear stress time series over the final period with the laminar theory, as well as with the provided data of Jensen et al. (1989) (their test 7). Similarly, based on the modeled/measured bed shear stresses, calculate the time evolution of f_w^* defined in Eq. 5.22 for $0° \le \omega t \le 130°$. Compare the model results with the constant f_w from laminar theory (Eq. 5.59) as well as against f_w^* based on the measurement data. From these plots, at which phase ωt do you see the laminar-to-turbulent transition begin? How does this compare with expectations based on e.g. Fig. 5.17.

(5) **Case 3: Smooth-turbulent wave boundary layer,** $Re = 5.4 \times 10^6$.
Similarly, create a new `Case3` directory, adjust the necessary model input, and run/animate the simulation for Case 3. Compare the computed velocity ($\overline{u}/U_{0m}$ vs. y/a) and turbulent kinetic energy (k/U_{0m}^2 vs. y/a) profiles at the four phases: $\omega t = 0°$, $45°$, $90°$, and $135°$ with the provided oscillatory tunnel measurements from Jensen et al. (1989) (their test 10). Note that as only two components of velocity were measured, you may use the approximation (Justesen, 1991)

$$k = 0.65\left(\overline{u'^2} + \overline{v'^2}\right) \tag{10.21}$$

to obtain the turbulent kinetic energy (per unit mass) k. Similarly, compare the modelled/measured profiles of the dimensionless Reynolds stress $-\overline{u'v'}/U_{0m}^2$ at the same four phases as above. (Note that this can be computed from the model results as: $-\overline{u'v'} = \nu_t \partial\overline{u}/\partial y$; Hint: use Matlab's `gradient` function to compute the velocity gradient). Finally, make a plot comparing the computed and measured bed shear stress time series ($\tau_0/(\rho U_{0m}^2)$ versus ωt). At approximately what phase ωt do you see laminar-to-turbulent transition occur now? (Hint: It should occur almost immediately!)

(6) **Case 4: Rough-turbulent wave boundary layer,** $Re = 5.4 \times 10^6$.
Now copy your `Case3` directory to a new `Case4` directory, adjust the near-wall grid size and wall roughness k_s, and run/animate the simulation for Case 4. Compare the computed velocity, turbulent kinetic energy, and Reynolds stress profiles at the same four phases as before with the provided oscillatory tunnel measurements from Jensen et al. (1989) (now their test 13). Similarly, compare the computed and measured bed shear stress time series, as before.

(7) Finally, for the specific phase $\omega t = 90°$ create new plots directly comparing the computed velocity and turbulent kinetic energy profiles from Cases 3 and 4. Can you observe/explain the physical effects of the bottom roughness on the flow and turbulence?

As indicated above, all of the preceding simulations are intended to be made utilizing the implemented "low Reynolds number" (transitional) version of the k-ω model (controlled by setting the input variable `turb=2`). Readers interested in additionally comparing against the "standard" (high Reynolds number version i.e. with constant closure coefficients in accordance with Eq. 9.39–9.41) are welcome/encouraged to repeat the simulations of e.g. Cases 3 and 4 with the input simply changed to utilize `turb=1`.

Similarly, the turbulence model can be turned off entirely by setting `turb=0`, which should thus yield very nearly the same results as above for Case 1. (If the models are otherwise properly set up, this should be the only change required!)

10.8 References

(1) Fuhrman, D.R., Schløer, S. and Sterner, J. (2013): RANS-based simulation of turbulent wave boundary layer and sheet-flow sediment transport processes. Coast. Eng., vol. 73, 151–166.
(2) Jensen, B.L., Sumer, B.M. and Fredsøe, J. (1989): Turbulent oscillatory boundary layers at high Reynolds numbers. J. Fluid Mech., vol. 206, 265–297.
(3) Justesen, P. (1991): A note on turbulence calculations in the wave boundary layer. J. Hydraul. Res., vol. 29, 699–711.
(4) Kirca, V.S.O., Sumer, B.M., Steffensen, M., Jensen, K.L. and Fuhrman, D.R. (2016): Longitudinal dispersion of heavy particles in an oscillating tunnel and application to wave boundary layers. J. Ocean Eng. Mar. Energy., vol. 2, 59–83.
(5) Schlichting, H. (1979): Boundary Layer Theory. McGraw-Hill.
(6) Wilcox, D.C. (2006): Turbulence Modeling for CFD, 3rd edn. DCW Industries.
(7) Williams, I.A. and Fuhrman, D.R. (2016): Numerical simulation of tsunami-scale wave boundary layers. Coast. Eng., vol. 110, 17–31.