

# Intro to Turbulent Flow HW1

Almog Dobrescu ID 214254252

May 26, 2025

## Contents

<b>1</b>	<b>Computing a Turbulent Flow</b>	<b>1</b>
1.1	Estimating boundary layer thickness $\delta$ . . . . .	1
1.2	Smallest Dynamically Important Scales . . . . .	1
1.3	Number of Grid Points . . . . .	1
1.4	Instantaneous Flow Over One Eddy . . . . .	2
<b>2</b>	<b>Heating a Room</b>	<b>3</b>
2.1	Characteristic Time Scale . . . . .	3
2.2	Induced Velocity Estimation . . . . .	3
2.3	Characteristic Time Scale For Room Heating . . . . .	4
<b>3</b>	<b>Help Karl Pearson</b>	<b>5</b>
3.1	Number of Realizations . . . . .	5
3.2	Average Displacement . . . . .	6
3.3	Autocovariance and Autocorrelation . . . . .	7
<b>A</b>	<b>Listing of The Computer Program</b>	<b>9</b>
A.1	Q1 . . . . .	9
A.2	Q2 . . . . .	9
A.3	Q3 . . . . .	9

## List of Figures

1	Calculating PDF . . . . .	5
2	PDF function as a function of realizations . . . . .	6
3	Average Displacement as a function of realizations . . . . .	6
4	Autocovariance and Autocorrelation . . . . .	7
5	Autocovariance and Autocorrelation ensemble . . . . .	7

## Listings

1	Code for Q1 . . . . .	9
2	Code for Q2 . . . . .	9
3	Code for Q3 . . . . .	9



# 1 Computing a Turbulent Flow

Considering a commercial airliner with a chord  $L = 5 \text{ [m]}$  cruising at  $U = 250 \left[ \frac{\text{m}}{\text{sec}} \right]$ .

## 1.1 Estimating boundary layer thickness $\delta$

According to Prandtl's one-seventh power law:

$$\frac{\delta}{x} \approx \frac{0.16}{Re_x^{\frac{1}{7}}} \quad (1)$$

At the trailing edge, the thickness of the boundary layer is:

$$\frac{\delta}{L} \approx \frac{0.16}{Re_L^{\frac{1}{7}}}, \quad Re_L = \frac{UL}{\nu} \quad (2)$$

To calculate the kinematic viscosity of air, let's assume the airliner is cruising at 30[kft]. At this height, the temperature is about  $-44.3^\circ[C]$  [1]. Hence, the kinematic viscosity is about  $1 \cdot 10^{-5}$  [2]

$$Re_L = \frac{250 \cdot 5}{1 \cdot 10^{-5}} = 1.25 \cdot 10^8 \quad (3)$$

$\Downarrow$

$$\delta \approx \frac{0.16 \cdot 5}{(1 \cdot 10^{-5})^{\frac{1}{7}}} = 0.0558[\text{m}] \quad (4)$$

## 1.2 Smallest Dynamically Important Scales

At the largest scales, the Reynolds number is very high, which indicates a turbulent regime. In a turbulent regime, there is no dissipation, which means:

$$Re_\ell = \frac{u\ell}{\nu} \gg 1 \quad (5)$$

However, we know that turbulent flows are dissipative, so there must be dissipation. For dissipation to accrue, the Reynolds number for the scales at which dissipation happens, the smallest scales, must be:

$$Re_\eta = \frac{v\eta}{\nu} \sim 1 \quad (6)$$

We can see that by fulfilling both demands for turbulent flow, there must be an energy transfer between scales, namely, an energy cascade.

## 1.3 Number of Grid Points

In order to accurately calculating the flow at the boundary layer, the size of one cell needs to be smaller than smallest eddy. From the energy cascade concept, we can conclude that the rate of change of the kinetic energy at the large scales:

$$\frac{du^2}{dt} \sim \frac{u^2}{\frac{\ell}{u}} \sim \varepsilon \quad (7)$$

is balanced by the energy dissipated at the small scales. By dimensional analysis, we find that:

$$\varepsilon \sim \nu \left( \frac{v}{\eta} \right)^2 \quad (8)$$



By combining Eq.7 and Eq.8 we get:

$$\frac{\ell}{\eta} \sim Re_\ell^{\frac{3}{4}} \quad \text{and} \quad \frac{v}{u} \sim Re_\ell^{-\frac{1}{4}} \quad \text{and} \quad \frac{\frac{\ell}{u}}{\frac{\eta}{v}} \sim Re_\ell^{\frac{1}{2}} \quad (9)$$

From Eq.9 we can calculate the size of one cell at the boundary layer:

$$\eta_x \sim \frac{L}{Re_L^{\frac{3}{4}}} = \frac{5}{(1.25 \cdot 10^8)^{\frac{3}{4}}} = 4.2295 \cdot 10^{-6} [\text{m}] \quad (10)$$

So, the number of grid point along the chord is:

$$N_x \sim \frac{L}{\eta} = \frac{5}{4.2295 \cdot 10^{-6}} = 1.1822 \cdot 10^6 \quad (11)$$

To calculate the number of grid point normal to the airfoil, we need to estimate the small scale of the boundary layer:

$$\eta_y \sim \frac{\delta}{Re_\delta^{\frac{3}{4}}} = \frac{0.0558}{\left(\frac{U\delta}{\nu}\right)^{\frac{3}{4}}} = 1.3745 \cdot 10^{-6} [\text{m}] \quad (12)$$

so the number of grid point normal to the airfoil:

$$N_y \sim \frac{\delta}{\eta} = \frac{0.0558}{4.2295 \cdot 10^{-6}} = 4.0574 \cdot 10^4 \quad (13)$$

The total number of grid point is therefore:

$$N = N_x N_y = \boxed{4.7965 \cdot 10^{10}} \quad (14)$$

## 1.4 Instantaneous Flow Over One Eddy

The turnover time of one eddy is defined as:

$$t_{large} = \frac{\delta}{U} \quad (15)$$

The time step needs to be smaller then the smallest time step. Therefore:

$$t_{small} = \frac{\eta_y}{v} \quad (16)$$

From Eq.9 we get:

$$\frac{t_{large}}{t_{small}} \sim Re_\delta^{\frac{1}{2}} \quad (17)$$

$\Downarrow$

$$N_t = \frac{t_{large}}{t_{small}} \sim Re_\delta^{\frac{1}{2}} \sim 1180 \text{ steps} \quad (18)$$

I finished this quistion in about 3 hours



## 2 Heating a Room

### 2.1 Characteristic Time Scale

The one dimensional heat equation for a non moving field is:

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x^2} \quad (19)$$

We will use dimensional analysis to determined the characteristic time scale for heating the room.

$$\begin{aligned} \frac{T_\infty}{t} &= \alpha \frac{T_\infty}{L^2} \\ t &= \frac{L^2}{\alpha} \end{aligned} \quad (20)$$

Assuming typical bedroom conditions [3]:

$$\begin{aligned} \alpha|_{T=25^\circ[C]} &= 22.39 \cdot 10^{-6} \left[ \frac{\text{m}^2}{\text{sec}} \right] , \quad L = 3 [\text{m}] \\ &\Downarrow \\ t &= 4.0197 \cdot 10^5 [\text{sec}] = 4.6524 [\text{day}] \end{aligned} \quad (21)$$

### 2.2 Induced Velocity Estimation

The momentum equation under Boussinesq approximation can be written as:

$$\frac{D}{Dt} \vec{u} = -\frac{1}{\rho} \nabla p + \nu \nabla^2 \vec{u} + g \frac{\Delta T}{T} \quad (22)$$

Where:

- $u$  is velocity induced by a space heater via buoyancy effects
- $T$  is the local temperature
- $\Delta T$  is the difference between  $T$  and the newly heated air

Assuming the buoyancy-driven flow is turbulent:

$$Re \gg 1$$

The momentum equation can be rewritten as:

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} = -\frac{1}{\rho} \nabla p + \nu \nabla^2 \vec{u} + g \frac{\Delta T}{T}$$

and by substituting the operators we get:

$$\frac{\partial u}{\partial t} + u \cdot \frac{\partial u}{\partial x} = -\frac{1}{\rho} \frac{\partial p}{\partial x} + \nu \frac{\partial^2 u}{\partial x^2} + g \frac{T - T_h}{T} \quad (23)$$

In order to estimate the magnitude of the induced velocity we will use dimensional analysis:

$$u = u_h \tilde{u} \mid t = T \tilde{t} \mid x = h \tilde{x} \mid p = \Lambda \tilde{p} \mid T = T_\infty \tilde{T}$$



$$\frac{u_h}{T} \frac{\partial \tilde{u}}{\partial \tilde{t}} + \frac{u_h^2}{h} \tilde{u} \cdot \frac{\partial \tilde{u}}{\partial \tilde{x}} = -\frac{\Lambda}{\rho h} \frac{\partial \tilde{p}}{\partial \tilde{x}} + \frac{\nu u_h}{h^2} \frac{\partial^2 \tilde{u}}{\partial^2 \tilde{x}} + g \frac{\tilde{T} - \frac{T_h}{T_\infty}}{\tilde{T}} \quad (24)$$

Multiplying by  $\frac{h^2}{\nu u_h}$ :

$$\frac{h^2}{\nu u_h} \frac{u_h}{T} \frac{\partial \tilde{u}}{\partial \tilde{t}} + \frac{h^2}{\nu u_h} \frac{u_h^2}{h} \tilde{u} \cdot \frac{\partial \tilde{u}}{\partial \tilde{x}} = -\frac{h^2}{\nu u_h} \frac{\Lambda}{\rho h} \frac{\partial \tilde{p}}{\partial \tilde{x}} + \frac{h^2}{\nu u_h} \frac{\nu u_h}{h^2} \frac{\partial^2 \tilde{u}}{\partial^2 \tilde{x}} + \frac{h^2}{\nu u_h} g \frac{\tilde{T} - \frac{T_h}{T_\infty}}{\tilde{T}} \quad (25)$$

$$\underbrace{\frac{u_h h}{\nu}}_{Re_h} \underbrace{\frac{h}{u_h T}}_{St_h} \frac{\partial \tilde{u}}{\partial \tilde{t}} + \underbrace{\frac{h u_h}{\nu}}_{Re_h} \tilde{u} \cdot \frac{\partial \tilde{u}}{\partial \tilde{x}} = -\underbrace{\frac{u_h h}{\nu}}_{Re_h} \frac{\Lambda}{\rho u_h^2} \frac{\partial \tilde{p}}{\partial \tilde{x}} + \frac{\partial^2 \tilde{u}}{\partial^2 \tilde{x}} + \underbrace{\frac{u_h h}{\nu}}_{Re_h} \frac{h}{u_h^2} g \frac{\tilde{T} - \frac{T_h}{T_\infty}}{\tilde{T}} \quad (26)$$

For a buoyancy-driven turbulent flow, the dominant terms are the advection and buoyancy terms. Hence, we can estimate the following relations:

$$\begin{aligned} \frac{Re_h \Lambda}{\rho u_h^2} &\sim 1, & \frac{Re_h h g}{u_h^2} &\sim Re_h \\ &\Downarrow & & \\ \Lambda &\sim \frac{\rho u_h^2}{Re_h}, & u_h &\sim \sqrt{h g} \end{aligned} \quad (27)$$

Therefore, the magnitude of the induced velocity, for  $h = 0.3[m]$  is:

$$u_h \sim \sqrt{0.3 \cdot 9.81} = 1.7155 \left[ \frac{m}{sec} \right] \quad (28)$$

### 2.3 Characteristic Time Scale For Room Heating

Let's consider a cloud of particles in turbulent flow. For two particles in that cloud, if the two particles spread by Fickian diffusion, then their spread can be described by the diffusivity constant  $D$ :

$$r^2 \sim Dt \quad (29)$$

From Richardson's 4/3's law, we now:

$$D_{turb} \sim \varepsilon^{\frac{1}{3}} r^{\frac{4}{3}} \quad (30)$$

Where  $\varepsilon = \frac{u^3}{L}$  from the lecture.

Therefore, the characteristic time scale for heating the room is:

$$t \sim \frac{L^2}{\varepsilon^{\frac{1}{3}} L^{\frac{4}{3}}} = \frac{L^2}{\frac{u_h}{L^{\frac{1}{3}}} L^{\frac{4}{3}}} = \frac{L}{u_h} = 1.7487[sec] \quad (31)$$

I finished this quistion in one day



### 3 Help Karl Pearson

#### 3.1 Number of Realizations

A Drunk walker takes  $n$  random steps. In our case,  $n = 100$ . In order to determine how many walks a walker need to take, in order for the PDF to converge, we will run tests for a range of realizations.

The PDF is calculated by counting all realizations inside a circle with radius  $r$  and a bigger circle with radius  $r + dr$ . In our case  $dr = 0.5$ :

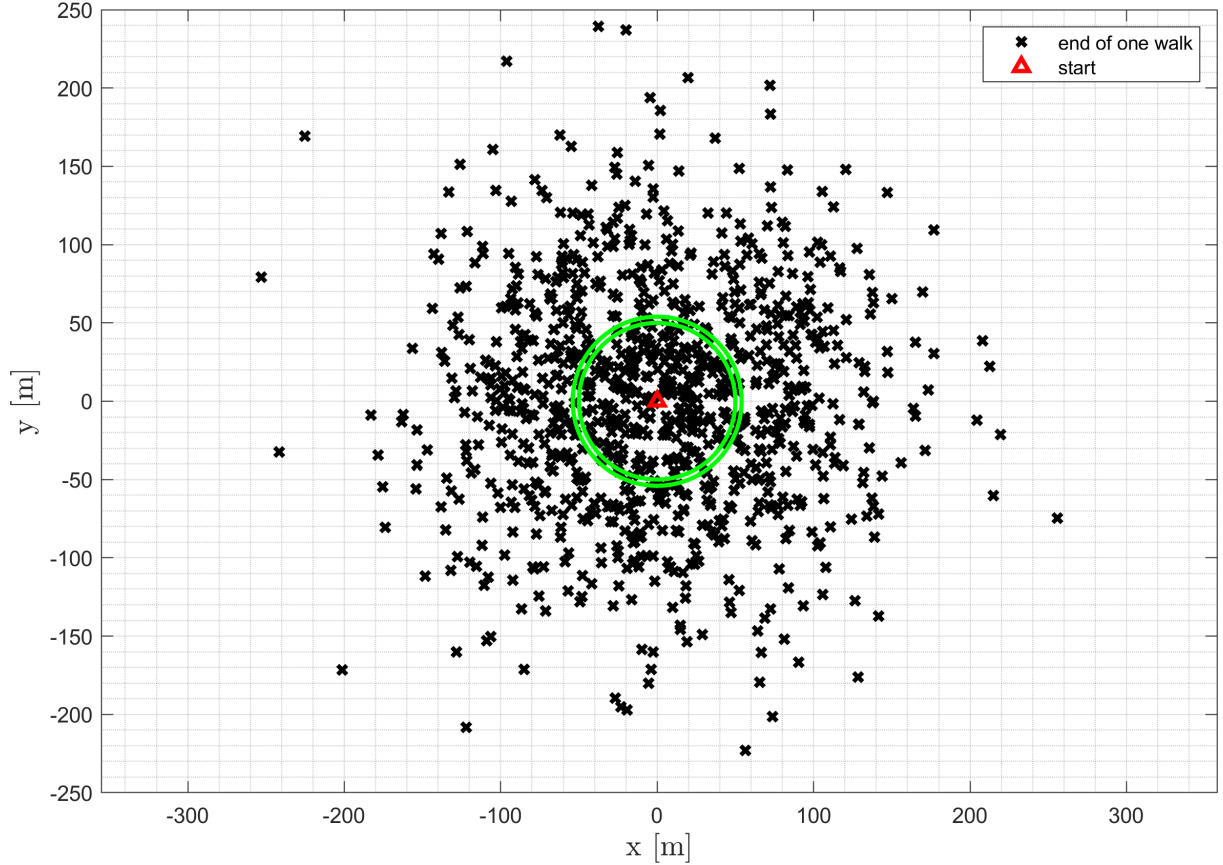


Figure 1: Calculating PDF

Lord Rayleigh's result for the ensemble PDF is:

$$\frac{2}{n} r e^{-\frac{r^2}{n}} \quad (32)$$

- for a step of  $1[m]$

We calculated the PDF function as a function of  $r$  for different numbers of realizations and compared the difference between the calculated PDF and Rayleigh's equation. The RMS is calculated by:

$$RMS = \sqrt{\sum_{r=0}^{r_{max}} [PDF_{analytical}(r) - PDF_{numerical}(r)]^2} \quad (33)$$

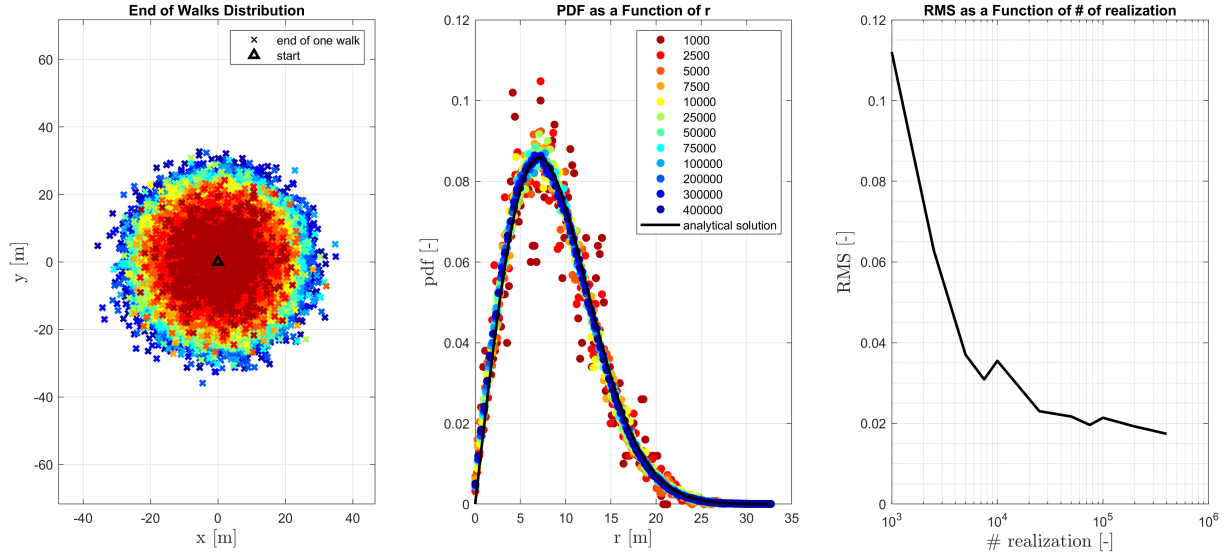


Figure 2: PDF function as a function of realizations

We can see that for number of realizations bigger than  $10^5$ , we can see that the RMS between the calculated PDF and Rayleigh's equation is small enough to consider converged.

### 3.2 Average Displacement

From the lectures, the average displacement  $\langle r \rangle$ :

$$\langle r \rangle = \mathbb{E}(r) = \int_{-\infty}^{\infty} r \cdot PDF(r) dr \quad (34)$$

Hence, the average displacement as a function of number of realizations is:

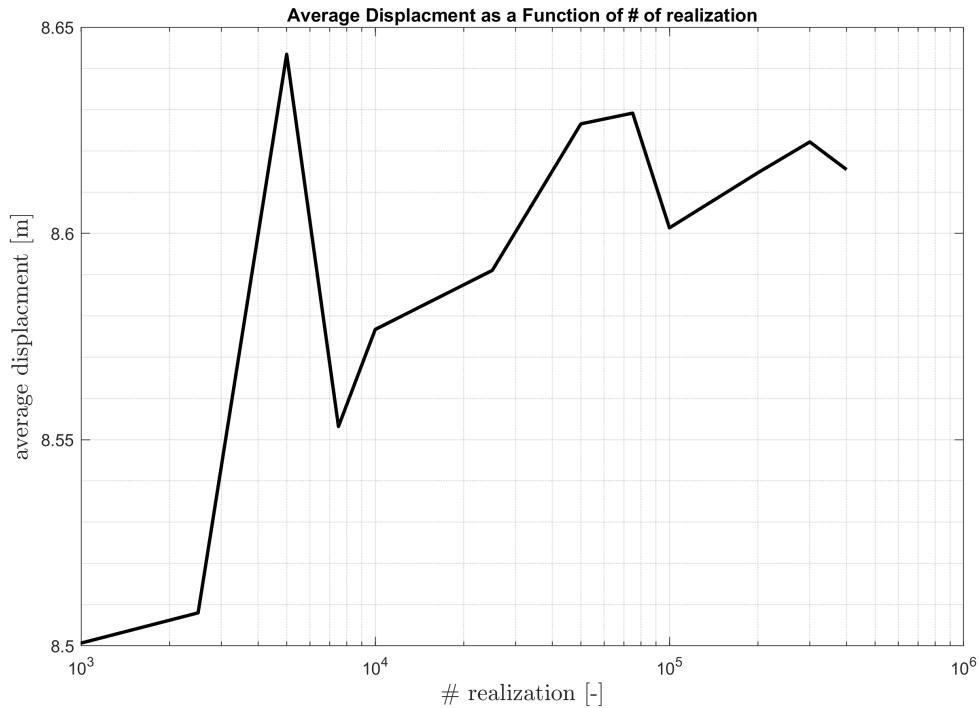


Figure 3: Average Displacement as a function of realizations

We can see that the average displacement converge to a value between 8.6 and 8.65.





### 3.3 Autocovariance and Autocorrelation

The covariance is calculated using the *cov* function in *MatLab*. The autocorrelation is the normalized autocovariance. The definition of the covariance is:

$$Q_{uv} = \mathbb{E}[(u - \mu_u)(v - \mu_v)] \quad (35)$$

The autocovariance is therefore:

$$Q(\Delta\ell) = \mathbb{E}[(r(\ell) - \mu)(r(\ell + \Delta\ell) - \mu)] \quad (36)$$

and the autocorrelation is:

$$R(\Delta\ell) = \frac{Q(\Delta\ell)}{\max\{Q(\Delta\ell)\}} \quad (37)$$

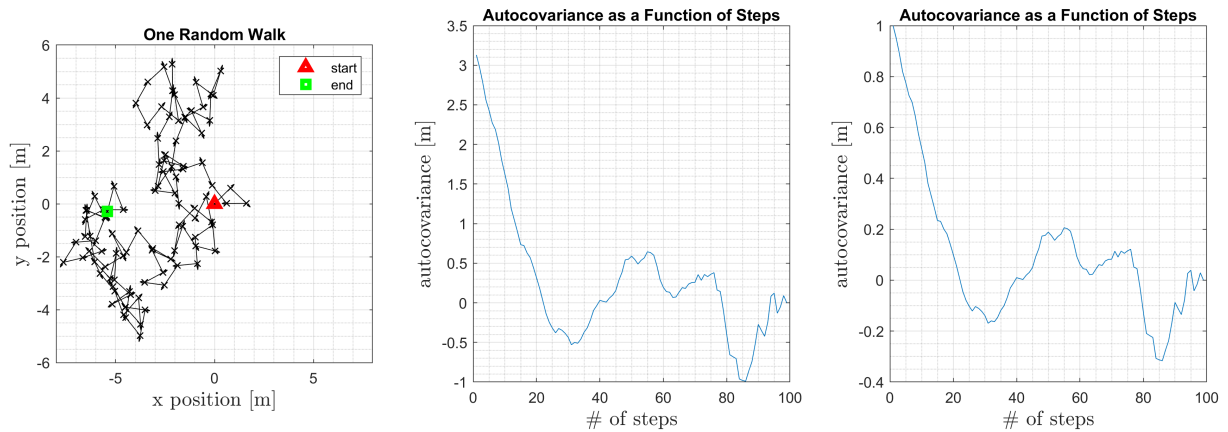


Figure 4: Autocovariance and Autocorrelation

In order to determine how many walks are needed in the ensemble for the autocorrelation to converge we will calculate the autocorrelation over a range of realizations:

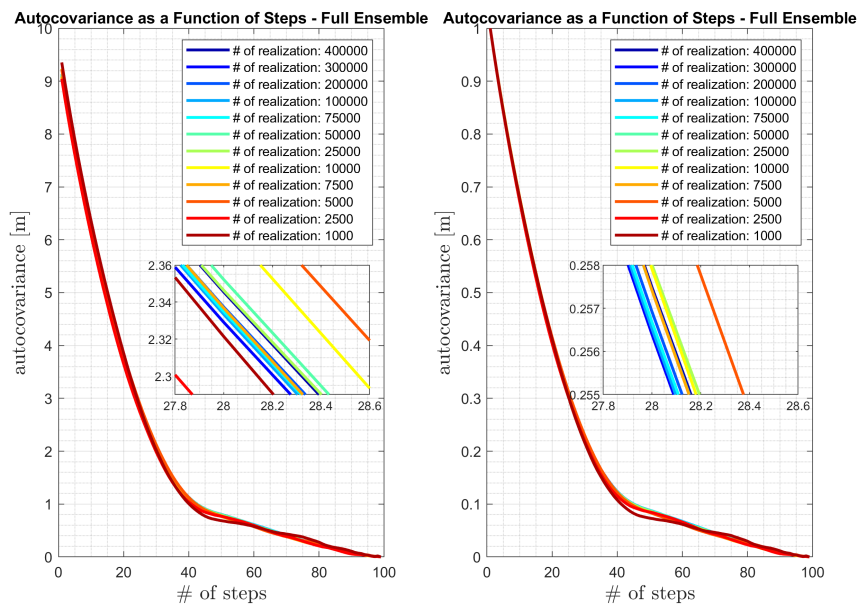


Figure 5: Autocovariance and Autocorrelation ensemble

We can see that for more than  $10^4$  realizations, the autocovariance does not have any significant changes.

I finished this question in 3 day



## References

- [1] T. E. ToolBox, “U.s. standard atmosphere: Temperature, pressure, and air properties vs. altitude.” [https://www.engineeringtoolbox.com/standard-atmosphere-d\\_604.html](https://www.engineeringtoolbox.com/standard-atmosphere-d_604.html), 2003.
- [2] E. Edge, “Viscosity of air, dynamic and kinematic.” [https://www.engineersedge.com/physics/viscosity\\_of\\_air\\_dynamic\\_and\\_kinematic\\_14483.htm](https://www.engineersedge.com/physics/viscosity_of_air_dynamic_and_kinematic_14483.htm), 2025.
- [3] T. E. ToolBox, “Air - thermal diffusivity vs. temperature and pressure.” [https://www.engineeringtoolbox.com/air-thermal-diffusivity-d\\_2011.html](https://www.engineeringtoolbox.com/air-thermal-diffusivity-d_2011.html), 2018.



## A Listing of The Computer Program

### A.1 Q1

```

1 clc; clear; close all
2
3 U      = 250;      % [m/sec]
4 L      = 5;        % [m]
5 nu_30k = 1e-5;     % [m^2/sec]
6
7 Re_L    = U * L / nu_30k
8 delta   = 0.16 * 5 / Re_L^(1/7)
9 Re_delta = U * delta / nu_30k
10 eta_x   = 5 / Re_L^(3/4)
11 eta_y   = delta / Re_delta^(3/4)
12 N_x     = L / eta_x
13 N_y     = delta / eta_y
14 N       = N_x * N_y
15 N_t     = Re_delta^0.5

```

Listing 1: Code for Q1

### A.2 Q2

```

1 clc; clear; close all
2
3 L      = 3;          % [m]
4 h      = 0.3;        % [m]
5 g      = 9.81;       % [m/sec^2]
6 alpha_25C = 22.39e-6; % [m^2/sec]
7 t_sec   = L^2/alpha_25C % [sec]
8 t_day   = t_sec/60/60/24 % [day]
9
10 u_h     = sqrt(h*g)   % [m/sec]
11 t_heating = L/u_h     % [sec]

```

Listing 2: Code for Q2

### A.3 Q3

```

1 clc; clear; close all;
2
3 %% Q.3.a PREFORM AND PRINT A RUN #####
4 step_size = 1; % [m]
5 num_of_steps = 1e4;
6 num_of_walks = 1e3;
7 r = 50;
8 dr = 4;
9
10 fig1 = figure('Name', '1', 'Position', [100, 250, 900, 600]);
11 hold all
12
13 steps = [];
14 [end_x_vec, end_y_vec, ~] = one_run(0, 0, step_size, num_of_steps, num_of_walks);
15
16 [count_of_band, count_of_outer_circ] = num_points_in_band_and_outer_circ(r, dr, end_x_vec, end_y_vec);

```



```

17
18 plot(end_x_vec, end_y_vec, 'x', 'LineWidth', 2, 'Color', 'k')
19 plot(0,0, '^', 'LineWidth', 2, 'Color', 'r')
20 draw_circ(0, 0, r, 'g', 2)
21 draw_circ(0, 0, r+dr, 'g', 2)
22
23 ax = gca;
24 MaxX = max(abs(ax.XLim));
25 MaxY = max(abs(ax.YLim));
26 axis([-MaxX MaxX -MaxY MaxY]);
27
28 xlabel('x [m]', 'FontSize', 14, 'Interpreter', 'latex')
29 ylabel('y [m]', 'FontSize', 14, 'Interpreter', 'latex')
30 legend({'end of one walk', 'start'})
31 axis equal
32 box on
33 grid on
34 grid minor
35 % exportgraphics(fig1, 'images/graph1.png', 'Resolution', 300);
36
37 %% Q.3.a PREFORM AND PRINT A RUN AND PDF #####
38 step_size = 1; % [m]
39 num_of_steps = 1e2;
40 dr = 0.5;
41 num_of_walks_vec = [4e5, 3e5, 2e5, 1e5, 7.5e4, 5e4, 2.5e4, 1e4, 7.5e3, 5e3, 2.5e3, 1e3];
42 % num_of_walks_vec = [7.5e4, 5e4, 2.5e4, 1e4, 5e3, 1e3];
43 num_of_radiuses = 1e2;
44
45 end_x_vec_vec = {};
46 end_y_vec_vec = {};
47 distances_vec_of_vec = {};
48 rs_vec = {};
49 pdf_vec_vec = {};
50 for index = 1:length(num_of_walks_vec)
51     num_of_walks = num_of_walks_vec(index);
52     steps = [];
53     [current_end_x_vec, current_end_y_vec, current_distances_vec] = one_run(0, 0,
54         step_size, num_of_steps, num_of_walks);
55     end_x_vec_vec{index,1} = current_end_x_vec;
56     end_y_vec_vec{index,1} = current_end_y_vec;
57     distances_vec_of_vec{index,1} = current_distances_vec;
58     rs_vec{index,1} = linspace(0, min(max(abs(current_end_x_vec)), max(abs(
59         current_end_y_vec))), num_of_radiuses);
60     pdf_vec_vec{index,1} = calc_pdf(rs_vec{index,1}, dr, current_end_x_vec,
61         current_end_y_vec);
62 end
63
64 fig2 = figure('Name', '2', 'Position', [150, 250, 1500, 600]);
65 hold all
66 colors = jet(length(num_of_walks_vec));
67 % colors = cool(length(num_of_walks_vec))*0.8;
68 lg = {};
69 rms = [];
70 for index = 1:length(num_of_walks_vec)
71     num_of_walks = num_of_walks_vec(index);
72     lg{end+1} = sprintf('%g', double(num_of_walks_vec(end-index+1)));
73     end_x_vec = end_x_vec_vec{index,1};
74     end_y_vec = end_y_vec_vec{index,1};

```



```

72
73 subplot(1,3,1) % #####
74 hold all
75 p = plot(0,0, 'x', 'LineWidth', 1, 'Color', 'k', 'HandleVisibility', 'callback');
76 plot(end_x_vec, end_y_vec, 'x', 'LineWidth', 2, 'Color', colors(index,:))
77 s = plot(0,0, '^', 'LineWidth', 2, 'Color', 'k');
78
79 ax = gca;
80 MaxX = max(abs(ax.XLim));
81 MaxY = max(abs(ax.YLim));
82 axis([-MaxX MaxX -MaxY MaxY]);
83 xlabel('x [m]', 'FontSize', 14, 'Interpreter', 'latex')
84 ylabel('y [m]', 'FontSize', 14, 'Interpreter', 'latex')
85 title('End of Walks Distribution')
86 legend([p, s], {'end of one walk', 'start'})
87 axis equal
88 % axis square
89 box on
90 grid on
91 grid minor
92
93 subplot(1,3,2) % #####
94 hold all
95
96 rs = rs_vec{length(num_of_walks_vec)-index+1, 1};
97 analytical_solution = 2/num_of_steps.*rs.*exp(-rs.^2/num_of_steps);
98 pdf_vec = pdf_vec_vec{length(num_of_walks_vec)-index+1, 1};
99
100 plot(rs, pdf_vec, '*', 'LineWidth', 2, 'Color', colors(length(num_of_walks_vec)-index
    +1,:))
101 if index == length(num_of_walks_vec)
102     plot(rs, analytical_solution, '-', 'LineWidth', 2, 'Color', 'k')
103     lg{end+1} = 'analytical solution';
104 end
105
106 xlabel('r [m]', 'FontSize', 14, 'Interpreter', 'latex')
107 ylabel('pdf [—]', 'FontSize', 14, 'Interpreter', 'latex')
108 title('PDF as a Function of r')
109 legend(lg)
110 box on
111 grid on
112 grid minor
113
114 subplot(1,3,3) % #####
115
116 rms(end+1) = sqrt(sum((analytical_solution-pdf_vec).^2));
117
118 semilogx(flip(num_of_walks_vec(end-length(rms)+1:end)), rms, '-', 'LineWidth', 2, '
    Color', 'k')
119
120 xlabel('\# realization [—]', 'FontSize', 14, 'Interpreter', 'latex')
121 ylabel('RMS [—]', 'FontSize', 14, 'Interpreter', 'latex')
122 title('RMS as a Function of # of realization')
123 box on
124 grid on
125 grid minor
126 end
127 % exportgraphics(fig2, 'images/graph2.png', 'Resolution', 300);

```



```

128 % exportgraphics(fig1, 'images/graph1.png','Resolution',300); exportgraphics(fig2, '
    images/graph2.png','Resolution',300);
129
130 %% Q.3.b Average Displacement #####
131 % step_size = 1; % [m]
132 % num_of_steps      = 1e2;
133 % dr                 = 0.5;
134 % % num_of_walks_vec = [4e5, 3e5, 2e5, 1e5, 7.5e4, 5e4, 2.5e4, 1e4, 7.5e3, 5e3, 2.5e3, 1
    e3];
135 % num_of_walks_vec = [7.55e4, 5e4, 2.5e4, 1e4, 5e3, 1e3];
136 % num_of_radiuses  = 1e2;
137 %
138 % end_x_vec_vec     = {};
139 % end_y_vec_vec     = {};
140 % distances_vec_of_vec = {};
141 % rs_vec            = {};
142 % pdf_vec_vec       = {};
143 % for index = 1:length(num_of_walks_vec)
144 %     num_of_walks = num_of_walks_vec(index);
145 %     steps = [];
146 %     [current_end_x_vec, current_end_y_vec, current_distances_vec] = one_run(0, 0,
    step_size, num_of_steps, num_of_walks);
147 %     end_x_vec_vec{index,1} = current_end_x_vec;
148 %     end_y_vec_vec{index,1} = current_end_y_vec;
149 %     distances_vec_of_vec{index,1} = current_distances_vec;
150 %     rs_vec{index,1} = linspace(0, min(max(abs(current_end_x_vec)),max(abs(
    current_end_y_vec))), num_of_radiuses);
151 %     pdf_vec_vec{index,1} = calc_pdf(rs_vec{index,1}, dr, current_end_x_vec,
    current_end_y_vec);
152 % end
153
154 ave_disp = [];
155 for index = 1:length(num_of_walks_vec)
156     ave_disp(index) = trapz(rs_vec{index,:}, rs_vec{index,:}.*pdf_vec_vec{index,:});
157 end
158
159 fig3 = figure('Name', '3', 'Position', [300, 250, 900, 600]);
160
161 semilogx(num_of_walks_vec, ave_disp, '-', 'LineWidth', 2, 'Color', 'k')
162
163 xlabel('\# realization [—]', 'FontSize', 14, 'Interpreter', 'latex')
164 ylabel('average displacment [m]', 'FontSize', 14, 'Interpreter', 'latex')
165 title('Average Displacment as a Function of # of realization')
166 box on
167 grid on
168 grid minor
169 % exportgraphics(fig3, 'images/graph3.png','Resolution',300);
170 % exportgraphics(fig3, 'images/graph3.png','Resolution',300); exportgraphics(fig1, '
    images/graph1.png','Resolution',300); exportgraphics(fig2, 'images/graph2.png','
    Resolution',300);
171
172 %% Q.3.c autocovariance and autocorrelation #####
173 step_size = 1; % [m]
174 number_of_steps = 1e2;
175
176 fig4 = figure('Name', '4', 'Position', [450, 250, 1350, 400]);
177
178 subplot(1,3,1) % #####

```



```

179
180 hold all
181 steps = [];
182 distances = [];
183 steps(1,:) = [0,0];
184 for i=1:number_of_steps
185     [x,y] = one_step(steps(i, 1), steps(i, 2), step_size);
186     steps(end+1,:) = [x,y];
187     distances(end+1) = sqrt(x^2+y^2);
188 end
189
190 plot(steps(:,1), steps(:,2), 'x', 'LineWidth', 0.75, 'Color', 'k', 'HandleVisibility','
    off')
191 quiver(steps(1:end-1,1), steps(1:end-1,2), diff(steps(:,1)), diff(steps(:,2)), '
    MarkerSize', 0.5, 'Color', 'k', 'MaxHeadSize', 0.05, 'HandleVisibility','off')
192 plot(steps(1,1),steps(1,2), '^', 'LineWidth', 3, 'Color', 'r')
193 plot(steps(end,1),steps(end,2), 's', 'LineWidth', 3, 'Color', 'g')
194
195 ax = gca;
196 MaxX = max(abs(ax.XLim));
197 MaxY = max(abs(ax.YLim));
198 axis([-MaxX MaxX -MaxY MaxY]);
199
200 xlabel('x position [m]', 'FontSize', 14, 'Interpreter', 'latex')
201 ylabel('y position [m]', 'FontSize', 14, 'Interpreter', 'latex')
202 title('One Random Walk')
203 legend({'start', 'end'})
204 axis square
205 box on
206 grid on
207 grid minor
208
209 Q = [];
210 for i = 1:length(distances)
211     A = distances(1:length(distances)-i);
212     B = distances(1+i:length(distances));
213     C = cov(A, B);
214     Q(i) = C(1,2);
215 end
216
217 subplot(1,3,2) % #####
218 plot(1:length(distances), Q)
219
220 xlabel('\# of steps', 'FontSize', 14, 'Interpreter', 'latex')
221 ylabel('autocovariance [m]', 'FontSize', 14, 'Interpreter', 'latex')
222 title('Autocovariance as a Function of Steps')
223 box on
224 grid on
225 grid minor
226
227 subplot(1,3,3) % #####
228 plot(1:length(distances), Q/Q(1))
229
230 xlabel('\# of steps', 'FontSize', 14, 'Interpreter', 'latex')
231 ylabel('autocovariance [m]', 'FontSize', 14, 'Interpreter', 'latex')
232 title('Autocovariance as a Function of Steps')
233 box on
234 grid on

```



```

235 grid minor
236 % exportgraphics(fig4, 'images/graph4.png','Resolution',300);
237 % exportgraphics(fig4, 'images/graph4.png','Resolution',300); exportgraphics(fig3, '
    images/graph3.png','Resolution',300); exportgraphics(fig1, 'images/graph1.png','
    Resolution',300); exportgraphics(fig2, 'images/graph2.png','Resolution',300);
238
239 %% Q.3.c ensemble #####
240 % step_size = 1; % [m]
241 % num_of_steps      = 1e2;
242 % dr                = 0.5;
243 % % num_of_walks_vec = [4e5, 3e5, 2e5, 1e5, 7.5e4, 5e4, 2.5e4, 1e4, 7.5e3, 5e3, 2.5e3, 1
    e3];
244 % num_of_walks_vec = [7.55e4, 5e4, 2.5e4, 1e4, 5e3, 1e3];
245 % num_of_radiuses  = 1e2;
246 %
247 % end_x_vec_vec      = {};
248 % end_y_vec_vec      = {};
249 % distances_vec_of_vec = {};
250 % rs_vec             = {};
251 % pdf_vec_vec        = {};
252 % for index = 1:length(num_of_walks_vec)
253 %     num_of_walks = num_of_walks_vec(index);
254 %     steps = [];
255 %     [current_end_x_vec, current_end_y_vec, current_distances_vec] = one_run(0, 0,
    step_size, num_of_steps, num_of_walks);
256 %     end_x_vec_vec{index,1} = current_end_x_vec;
257 %     end_y_vec_vec{index,1} = current_end_y_vec;
258 %     distances_vec_of_vec{index,1} = current_distances_vec;
259 %     rs_vec{index,1} = linspace(0, min(max(abs(current_end_x_vec)),max(abs(
    current_end_y_vec))), num_of_radiuses);
260 %     pdf_vec_vec{index,1} = calc_pdf(rs_vec{index,1}, dr, current_end_x_vec,
    current_end_y_vec);
261 % end
262
263 Q_vec = {};
264 Q_ensemble = {};
265 for index = 1:length(num_of_walks_vec)
266     fprintf('Walk number %d\n', index);
267     distances_vec = distances_vec_of_vec{index,1};
268     Q_ensemble{index,1} = zeros(1,length(distances));
269     for j = 1:length(distances_vec)
270         j
271         distances = distances_vec{j,1};
272         Q = [];
273         for i = 1:length(distances)
274             A = distances(1:length(distances)-i);
275             B = distances(1+i:length(distances));
276             C = cov(A, B);
277             Q(i) = C(1,2);
278         end
279         Q_ensemble{index,1} = Q_ensemble{index,1} + Q;
280     end
281     Q_ensemble{index,1} = Q_ensemble{index,1} / num_of_walks_vec(index);
282 end
283 %%
284 fig5 = figure('Name', '5', 'Position', [600, 250, 900, 600]);
285
286 colors = jet(length(num_of_walks_vec));

```





```

287 % colors = cool(length(num_of_walks_vec))*0.8;
288 subplot(1,2,1) % #####
289 hold all
290 lg = {};
291 for index = 1:length(num_of_walks_vec)
292     Q = Q_ensemble{index,1};
293     plot(1:length(distances), Q, '-', 'LineWidth', 2, 'Color', colors(index,:))
294     lg{end+1} = sprintf('# of realization: %d', num_of_walks_vec(index));
295 end
296
297 xlabel('\# of steps','FontSize',14,'Interpreter','latex')
298 ylabel('autocovariance [m]','FontSize',14,'Interpreter','latex')
299 title('Autocovariance as a Function of Steps – Full Ensemble')
300 legend(lg)
301 box on
302 grid on
303 grid minor
304
305 zoom = axes('position',[0.25 0.36 0.2 0.2]);
306 box on % put box around new pair of axes
307 hold all
308 for index = 1:length(num_of_walks_vec)
309     Q = Q_ensemble{index,1};
310     plot(1:length(distances), Q, '-', 'LineWidth', 2, 'Color', colors(index,:))
311 end
312 zoom.XLim = [27.8, 28.6];
313 zoom.YLim = [2.29, 2.36];
314 grid on
315 grid minor
316
317 subplot(1,2,2) % #####
318 hold all
319 lg = {};
320 for index = 1:length(num_of_walks_vec)
321     Q = Q_ensemble{index,1};
322     plot(1:length(distances), Q/Q(1), '-', 'LineWidth', 2, 'Color', colors(index,:))
323     hold on
324     lg{end+1} = sprintf('# of realization: %d', num_of_walks_vec(index));
325 end
326
327 xlabel('\# of steps','FontSize',14,'Interpreter','latex')
328 ylabel('autocovariance [m]','FontSize',14,'Interpreter','latex')
329 title('Autocovariance as a Function of Steps – Full Ensemble')
330 legend(lg)
331 box on
332 grid on
333 grid minor
334
335 zoom = axes('position',[0.69 0.36 0.2 0.2]);
336 box on % put box around new pair of axes
337 hold all
338 for index = 1:length(num_of_walks_vec)
339     Q = Q_ensemble{index,1};
340     plot(1:length(distances), Q/Q(1), '-', 'LineWidth', 2, 'Color', colors(index,:))
341 end
342 zoom.XLim = [27.8, 28.6];
343 zoom.YLim = [0.255, 0.258];
344 grid on

```



```

345 grid minor
346
347 % exportgraphics(fig5, 'images/graph5.png','Resolution',300);
348 % exportgraphics(fig5, 'images/graph5.png','Resolution',300); exportgraphics(fig4, '
    images/graph4.png','Resolution',300); exportgraphics(fig3, 'images/graph3.png','
    Resolution',300); exportgraphics(fig1, 'images/graph1.png','Resolution',300);
    exportgraphics(fig2, 'images/graph2.png','Resolution',300);
349
350
351
352
353
354
355
356 % FUNCTIONS #####
357 function [des_x, des_y] = one_step(src_x, src_y, step_size)
358     theta = rand()*2*pi;
359     des_x = src_x + step_size * cos(theta);
360     des_y = src_y + step_size * sin(theta);
361 end
362
363 function [end_x, end_y, distances] = one_walk(start_x, start_y, step_size, num_of_steps)
364     x = start_x;
365     y = start_y;
366     distances = [];
367     for i=1:num_of_steps
368         [x,y] = one_step(x, y, step_size);
369         distances(i) = sqrt(x^2+y^2);
370     end
371     end_x = x;
372     end_y = y;
373 end
374
375 function [end_x_vec, end_y_vec, distances_vec] = one_run(start_x, start_y, step_size,
    num_of_steps, num_of_walks)
376     fprintf('performing a run ...\n');
377     steps = [];
378     distances_vec = {};
379     for i=1:num_of_walks
380         if ~mod(i, num_of_walks/10)
381             fprintf('    complited: %2.0f%%\n', i/num_of_walks*100);
382         end
383         [x,y,distances] = one_walk(start_x, start_y, step_size, num_of_steps);
384         steps(end+1,:) = [x,y];
385         distances_vec{i, 1} = distances;
386     end
387     end_x_vec = steps(:,1);
388     end_y_vec = steps(:,2);
389 end
390
391
392 function [count_of_band, count_of_outer_circ] = num_points_in_band_and_outer_circ(r, dr,
    x_vec, y_vec)
393     if length(x_vec) ~= length(y_vec)
394         fprintf('x and y are not the same length\n');
395         return
396     end
397

```



```

398     count_of_inner_circ = 0;
399     count_of_outer_circ = 0;
400     for i = 1:length(x_vec)
401         dist_squre = x_vec(i)^2+y_vec(i)^2;
402         if dist_squre <= r^2
403             count_of_inner_circ = count_of_inner_circ+1;
404         end
405         if dist_squre <= (r+dr)^2
406             count_of_outer_circ = count_of_outer_circ+1;
407         end
408     end
409
410     count_of_band = count_of_outer_circ - count_of_inner_circ;
411 end
412
413 function draw_circ(center_x, center_y, r, color, line_width)
414     pos = [[center_x, center_y]-r, 2*r, 2*r];
415     rectangle('Position',pos,'Curvature',[1 1], 'EdgeColor', color, 'LineWidth',
416               line_width)
417 end
418
419 function pdf_vec = calc_pdf(rs, dr, x_vec, y_vec)
420     if length(x_vec) ~= length(y_vec)
421         fprintf('x and y are not the same length\n');
422         return
423     end
424     fprintf('calc pdf, dr = %4f\n', dr);
425
426     pdf_vec = [];
427     for i=1:length(rs)
428         r = rs(i);
429         [count_of_band, count_of_outer_circ] = num_points_in_band_and_outer_circ(r, dr,
430                                           x_vec, y_vec);
431         cdf_inner = (count_of_outer_circ-count_of_band) / length(x_vec); % points inside
432                               circ
433         cdf_outer = (count_of_outer_circ) / length(x_vec); % points outside circ
434         pdf_vec(i) = (cdf_outer - cdf_inner) / dr;
435     end
436 end

```

Listing 3: Code for Q3