# TBL Processing

## Question 1.

### a.

A screen shot of a graph

AI-generated content may be incorrect.

**Figure 1:** Plot of temporally averaged x-velocity, ubar

A red and yellow smoke

AI-generated content may be incorrect.  
**Figure 2:** Plot of instantaneous x-velocity

A blue and green line

AI-generated content may be incorrect.

**Figure 3:** Plot of instantaneous fluctuating x-velocity

### b.

The freestream velocity was calculated as the maximum of the temporally-averaged x-velocity as . The boundary layer thickness was calculated by iterating over all the x-indices, finding the minimum height where , and averaging those heights. The computed value is . The displacement and momentum thickness were calculated as, respectively:

Where both displacement and momentum thickness were calculated at each x-station and averaged to arrive at a final value.

### c.

A graph with a blue line

AI-generated content may be incorrect.

**Figure 4:** Wall-normal profile of mean velocity

A graph with a line

AI-generated content may be incorrect.

**Figure 5:** Wall-normal fluctuating velocity

The wall-normal average velocity profile starts near zero. It starts at a non-zero value since the first measurement is not conducted on the wall, but near the wall. The velocity is only zero at the wall. The gradient is initially very steep, up to y/delta~0.02. The gradient then levels out and recovers to near the freestream value as y/delta🡪1. With a turbulent boundary layer, we expect high levels of momentum mixing to lower the gradients near the boundary layer edge, and increase the gradients near the wall. This behavior is observed.

The wall-normal fluctuating velocity profile appears to start at a peak. At the wall, the fluctuating velocity is 0. Like the mean velocity profile, the first measurement is off the wall, so a zero fluctuating velocity is not observed. The initial values are very high, and the peak appears to happen right next to the wall. Plotting on a log-scaled horizontal axis might reveal different near-wall behavior. Fluctuations decrease farther from the wall, but they do not decrease to zero because of non-zero levels of freestream turbulence.

### d.

The equation can be rearranged for x to give:

This gives a downstream distance for x of .

### e.

By assuming that the downstream distance is at the initial x-station of the data, a theoretical ratio between the boundary-layer thickness at the start and end of the data can be calculated:

This indicates an approximate variation of ~10% in boundary layer thickness.

A graph of a line

AI-generated content may be incorrect.

**Figure 6:** Profiles of wall-normal mean velocity at the start and end of data

The plot of the first and last sections of streamwise-averaged mean velocity support the above assertion. In the range of velocities close to the freestream velocity (8-10m/s), the boundary layer segment near the end reaches the velocity at a higher y-value. This supports the conclusion of a thicker boundary layer. The near-wall gradients appear to be steeper, which might be due to higher turbulent intensity, or measurement artifacts.

## Question 2.

### a.

Under the condition , the friction velocity can be expressed as . This results in a value of and a corresponding value of . This relation is valid in the viscous sublayer, for . Therefore, this approximation is likely good. However, since the y-plus value was calculated using the friction velocity which assumed it was in the VSL, it might be inaccurate.

### b.

The corresponding Reynolds number is .

### c.

A graph of a graph

AI-generated content may be incorrect.

**Figure 7:** Fitted profile for u+

The above fit was computed using Matlab’s fminsearch subroutine. The resultant friction velocity and Reynolds number are and .

### d.

Combining equations 3 and 4 gives an expression for the friction velocity in terms of the downstream Reynolds number:

Using this relation, the friction velocity is calculated as and at the start and end of the domain. The start is assumed to be at the downstream distance calculated in Question 1. This matches the calculations from parts a and c very well, considering that it is based off an experimental correlation. The correlated friction velocity is within of the measured and fitted velocities.

Over the extent of the domain, the friction velocity is expected to decrease by , which is less than 1% of the total friction velocity. The x-dependence is very weak in the equation for the correlated friction velocity, which explains the insensitivity to downstream distance.

### e.

With a laminar boundary layer, the expected friction coefficient is , compared to the turbulent boundary layer where . The friction coefficient is 7.56 times higher for a turbulent boundary layer.

## Question 3.

### a.

The wake strength is found to be .

### b.

A graph of a function

AI-generated content may be incorrect.

**Figure 8:** Viscous sublayer, log-law, and combined wake/log-law fits on velocity profile

The approximations match the velocity profile well in the regions they are derived for. In the VSL, the first datapoint is at , and the fit nearly perfectly coincides with the first datapoint. In the log-law region, both the log-law fit and the combined log-law and wake fit capture the behavior of the velocity profile. Towards the end of the wake region, the wake fit does not level out with the experimental velocity profile, and so it overpredicts the velocity near the end of the wake.

### c.

A graph with a line

AI-generated content may be incorrect.

**Figure 9:** Difference between measured and fitted y plus values.

In the log-law region, the difference between fitted and measured y-plus is very small, on the order of . The agreement is good evidence in support of Prandtl’s hypothesis that in the log-law region, neither the inner nor outer variables have a large impact on the velocity profile.

## Code

if ~exist('U', 'var')

load('data\_full.mat');

end

close all;

u\_bar = mean(U,3);

u\_prime = U-u\_bar;

t\_0 = 150;

%% Calculate part b %%

% u\_inf = mean(u\_bar(225:end,:), 'all');

u\_inf = max(u\_bar, [], 'all');

first\_exceed\_indices = arrayfun(@(col) find(u\_bar(:, col) > 0.99 \* u\_inf, 1), 1:size(u\_bar, 2), 'UniformOutput', false);

y\_values = cell2mat(cellfun(@(idx) Y(idx), first\_exceed\_indices(~cellfun('isempty', first\_exceed\_indices)), 'UniformOutput', false));

delta\_99 = mean(y\_values);

delta\_displacement = 0;

count = 0;

for i = 1:size(X,2)

index\_max = first\_exceed\_indices{i};

if isempty(index\_max) % Skip if empty

continue;

end

sum = 0;

count = count+1;

for j = 2:index\_max

sum = sum + (Y(j) - Y(j-1)) \* (1 - u\_bar(j,i) / u\_inf);

end

delta\_displacement = delta\_displacement + sum;

end

delta\_displacement = delta\_displacement / count;

delta\_momentum = 0;

count = 0;

for i = 1:size(X,2)

index\_max = first\_exceed\_indices{i};

if isempty(index\_max) % Skip if empty

continue;

end

sum = 0;

count = count+1;

for j = 2:index\_max

sum = sum + (Y(j) - Y(j-1)) \* (1 - u\_bar(j,i) / u\_inf) \* (u\_bar(j,i)/u\_inf);

end

delta\_momentum = delta\_momentum + sum;

end

delta\_momentum = delta\_momentum / count;

turbulence\_intensity = 100 \* mean(abs(u\_prime), 'all')/u\_inf;

%% Calculate part c %%

wall\_norm\_mean = mean(u\_bar, 2);

wall\_norm\_fluctuating = mean(u\_prime.^2, [3,2]);

%% Calculate part d %%

x\_approx = (delta\_99/0.375)^1.25 \* (u\_inf/nu)^0.25;

%% Calculate part e

delta\_ratio\_start\_end = (x\_approx/(x\_approx+X(end)))^0.8;

%% Plot part a %%%%

figure

imagesc(X, Y, u\_bar)

set(gca, 'ydir', 'normal')

axis equal

xlabel('x (m)')

ylabel('y (m)')

colormap(jet)

cb = colorbar;

% clim([9 10]);

ylabel(cb, "U bar (m/s)")

figure

imagesc(X, Y, U(:,:,t\_0))

set(gca, 'ydir', 'normal')

axis equal

xlabel('x (m)')

ylabel('y (m)')

colormap(jet)

cb = colorbar;

ylabel(cb, "U (m/s)")

figure

imagesc(X, Y, u\_prime(:,:,t\_0))

set(gca, 'ydir', 'normal')

axis equal

xlabel('x (m)')

ylabel('y (m)')

colormap(jet)

cb = colorbar;

ylabel(cb, "U' (m/s)")

% figure

% imagesc(X, Y, mean(u\_prime.^2,3))

% set(gca, 'ydir', 'normal')

% axis equal

% xlabel('x (m)')

% ylabel('y (m)')

% colormap(jet)

% cb = colorbar;

% ylabel(cb, "U' averaged (m/s)")

%% Plot part C

figure

plot(Y/delta\_99, wall\_norm\_mean/u\_inf);

xlabel('$$\frac{y}{\delta\_{99}}$$', 'Interpreter', 'latex')

ylabel('$$\frac{\bar{u}}{U\_\infty}$$', 'Interpreter', 'latex')

figure

plot(Y/delta\_99, wall\_norm\_fluctuating/(u\_inf.^2));

xlabel('$$\frac{y}{\delta\_{99}}$$', 'Interpreter', 'latex')

ylabel('$$\frac{u^{\prime 2}}{U\_\infty^2}$$', 'Interpreter', 'latex')

%% Plot part e

figure

hold on

plot(Y, mean(u\_bar(:,1:100),2), 'DisplayName', 'Front (1:100)')

plot(Y, mean(u\_bar(:,end-100:end),2), 'DisplayName', 'End (last 100)')

hold off

xlabel('Y (m)')

ylabel('Mean Velocity \(\bar{u}\) (m/s)')

legend

close all;

%% Calculate Part A

u\_avg = mean(u\_bar, 2);

u\_tau\_a = sqrt(u\_avg(1)\*nu/Y(1));

y\_plus\_a = Y(1)\*u\_tau\_a/nu;

%% Calculate Part B

Re\_tau\_a = u\_tau\_a\*delta\_99/nu;

%% Calculate Part C

kappa = 0.41; A=5;

y\_plus\_initial = (Y \* u\_tau\_a) / nu;

start\_idx = find(y\_plus\_initial > 30, 1);

end\_idx = find(y\_plus\_initial > 0.15\*Re\_tau\_a, 1) - 1;

%Fitting code from chatgpt

model = @(utau) norm( ...

(u\_avg(start\_idx:end\_idx) / utau) ... % u+ formulation

- (1/kappa) \* log( (Y(start\_idx:end\_idx) \* utau) / nu ) - A ...

);

utau\_guess = 1.0;

u\_tau\_best = fminsearch(model, utau\_guess);

Re\_tau\_best = (u\_tau\_best \* delta\_99) / nu;

%% Calculate Part D

Re\_x\_start = x\_approx\*u\_inf/nu;

Re\_x\_end = (x\_approx+X(end))\*u\_inf/nu;

u\_tau\_start = u\_inf\*sqrt(0.01\*Re\_x\_start^-0.133);

u\_tau\_end = u\_inf\*sqrt(0.01\*Re\_x\_end^-0.133);

%% Calculate Part E

cf\_laminar = 0.664 \* (Re\_x\_start\*0.8+Re\_x\_end\*0.5)^-0.5;

cf\_turbulent = 2\*u\_tau\_best^2/u\_inf^2;

turb\_lam\_ratio = cf\_turbulent/cf\_laminar;

%% Plot Part C

y\_plus = (Y \* u\_tau\_best) / nu;

u\_plus = u\_avg / u\_tau\_best;

start\_idx = find(y\_plus > 30, 1);

end\_idx = find(y\_plus > 0.15\*Re\_tau\_best, 1) - 1;

figure;

plot(y\_plus, u\_plus, 'ko'); hold on;

y\_fit = linspace(y\_plus(start\_idx), y\_plus(end\_idx), 100);

u\_fit = (1/kappa) \* log(y\_fit) + A;

plot(y\_fit, u\_fit, 'r-', 'LineWidth', 2);

set(gca, 'XScale','log');

grid on;

xlabel('y^+');

ylabel('u^+');

legend('Data','Log-Law Fit','Location','best');

% clearvars -except u\_avg u\_tau\_best Re\_tau\_best X Y delta\_99 u\_inf y\_plus u\_plus kappa A;

close all;

%% Calculate Part A

start\_idx = find(y\_plus > 30, 1);

end\_idx = find(y\_plus > Re\_tau\_best, 1) - 1;

model = @(ws) norm( ...

u\_plus(start\_idx:end\_idx) ...

- (1/kappa) \* log(y\_plus(start\_idx:end\_idx)) - A ...

- (2\*ws/kappa) .\* (sin((pi/2) \* (Y(start\_idx:end\_idx) / delta\_99))).^2 ...

);

ws\_guess = 1.0;

ws = fminsearch(model, ws\_guess);

%% Plot Part B

figure;

plot(y\_plus, u\_plus, 'ko'); hold on;

plot(y\_fit, u\_fit, 'r-', 'LineWidth', 2);

y\_fit\_2 = linspace(y\_plus(start\_idx), y\_plus(end\_idx), 100);

u\_fit\_2 = (1/kappa) \* log(y\_fit\_2) + A + 2\*ws/kappa\*sin(pi/2 \* y\_fit\_2\*nu/u\_tau\_best/delta\_99).^2;

plot(y\_fit\_2, u\_fit\_2, 'g-', 'LineWidth', 2);

y\_fit\_3 = linspace(0,10,100);

u\_fit\_3 = y\_fit\_3;

plot(y\_fit\_3, u\_fit\_3, 'b-', 'LineWidth', 2);

% Determine the vertical boundaries for the regions

vsl\_end = 5;

buffer\_end = 30;

loglaw\_end = 0.15 \* Re\_tau\_best;

wake\_end = Re\_tau\_best;

% Get current y-axis limits for placing the lines and labels

yl = ylim;

% Draw vertical dashed lines at each boundary

line([vsl\_end vsl\_end], yl, 'Color', 'k', 'LineStyle', '--');

line([buffer\_end buffer\_end], yl, 'Color', 'k', 'LineStyle', '--');

line([loglaw\_end loglaw\_end], yl, 'Color', 'k', 'LineStyle', '--');

line([wake\_end wake\_end], yl, 'Color', 'k', 'LineStyle', '--');

% Place text labels at the center of each region (vertically centered)

mid\_y = mean(yl);

text( vsl\_end/2, mid\_y, 'VSL', 'HorizontalAlignment', 'center', 'FontSize', 10 );

text( (vsl\_end+buffer\_end)/2, mid\_y, 'Buffer', 'HorizontalAlignment', 'center', 'FontSize', 10 );

text( (buffer\_end+loglaw\_end)/2-50, mid\_y, 'Log-Law', 'HorizontalAlignment', 'center', 'FontSize', 10 );

text( (loglaw\_end+wake\_end)/2-300, mid\_y, 'Wake', 'HorizontalAlignment', 'center', 'FontSize', 10 );

text( (wake\_end+max(y\_plus))/2+2150, mid\_y, 'Freestream', 'HorizontalAlignment', 'center', 'FontSize', 10 );

% Set axis properties and labels

set(gca, 'XScale', 'log');

xlabel('y^+');

ylabel('u^+');

legend('Data','Log-Law Fit','Log-Law and Wake fit','y^+ = u^+','Location','best');

hold off;

%% Plot Part C

figure;

yl = ylim;

mid\_y = mean(yl);

plot(y\_plus, u\_plus - (1/kappa\*log(y\_plus)+A))

text( (buffer\_end+loglaw\_end)/2-50, mid\_y, 'Log-Law', 'HorizontalAlignment', 'center', 'FontSize', 10 );

line([buffer\_end buffer\_end], yl, 'Color', 'k', 'LineStyle', '--');

line([loglaw\_end loglaw\_end], yl, 'Color', 'k', 'LineStyle', '--');

set(gca, 'XScale', 'log');

xlabel('y^+');

ylabel('\Delta u^+');