TBL Processing

Question 4.

a.

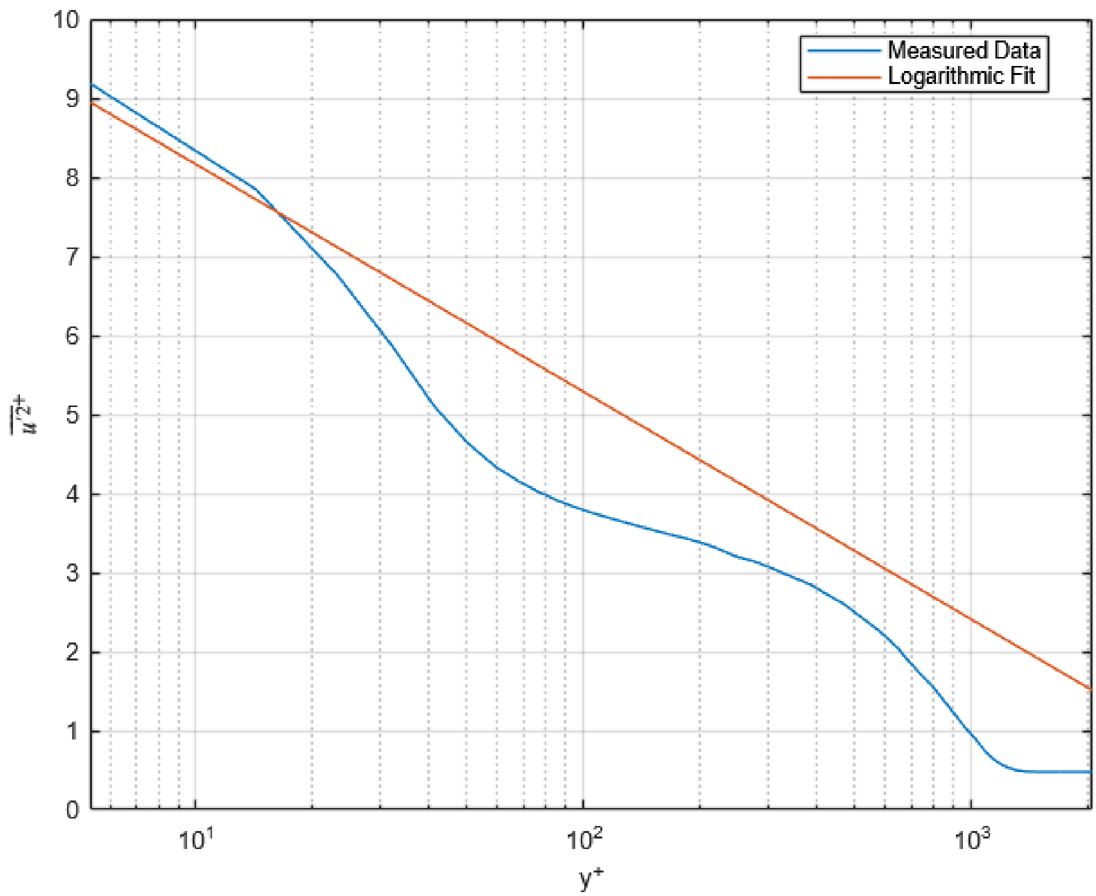

Figure 1: Streamwise fluctuations vs y-plus.

In comparison with Fig. 8.11 in the book, some notable features are missing. In the near-wall region, no peak is observed. In the book, peaks appear around a y-plus of 20. There are fewer data points in this data set collected around the low y-plus values, which might account for the missing peak.

In addition, there is no outer peak observed. The quasi-linear region appears to occur from y+ of 70-300. This region is below the near-wall fluctuation levels, which lines up with the low-Re curve in Fig. 8.11a, where no second peak is observed. The second peak is not distinctly observed until Re > 10000.

The logarithmic fit does not accurately capture the trends. My best guess is that the friction velocity estimate is not perfect, which would change Re_tau and therefore move the log-fit line, as well as change the scaling of the data, since they are normalized by the friction velocity.

b.

The mixing-length model can be applied in the flow, and the mixing length can be estimated from the experimental data as:

$$-\overline{u'v'} = l_m^2 \left(\frac{d\bar{u}}{dy}\right)^2$$

$$l_m = \frac{1}{\left|\frac{d\bar{u}}{dy}\right|}\sqrt{\left|-\overline{u'v'}\right|}$$

In addition, the von-Karmon mixing length model can also be applied.

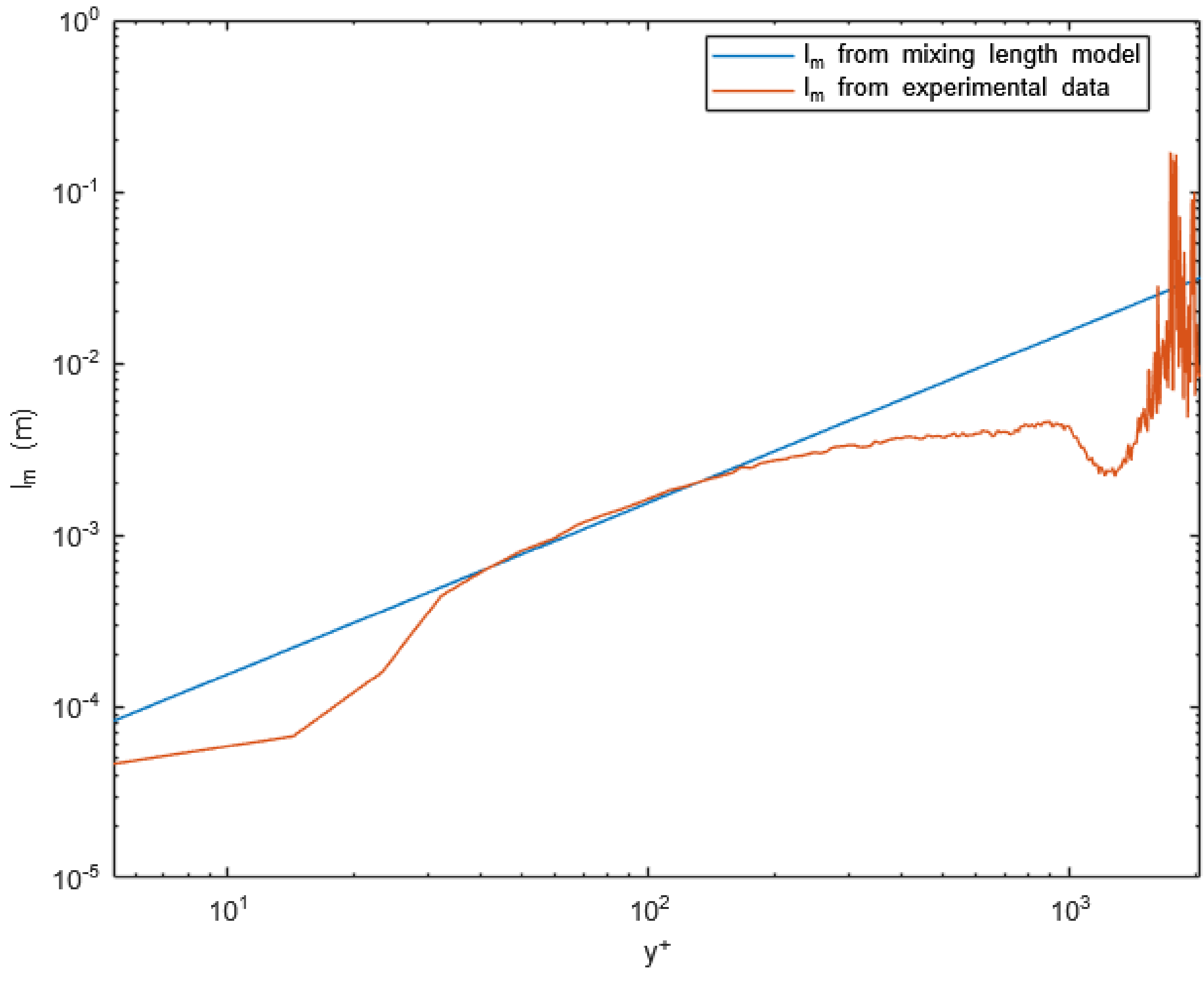

Figure 2: Mixing length from experimental data and VK model.

The graph shows excellent agreement in the log-layer region. From y+ of ~30-200, the mixing length is nearly identical when estimated with the experimental data and VK model.

In the viscous layer and buffer layer, the VK model overpredicts the experimental mixing length, although the sparsity of experimental data in this region could contribute to the lack of agreement. In the wake and freestream, the agreement is poor. In a TBL with a laminar freestream, the mixing length would decrease moving upwards into the freestream, but the VK model predicts monotonic growth. Overall the VK model works best in the log-layer region.

Question 5.

a.

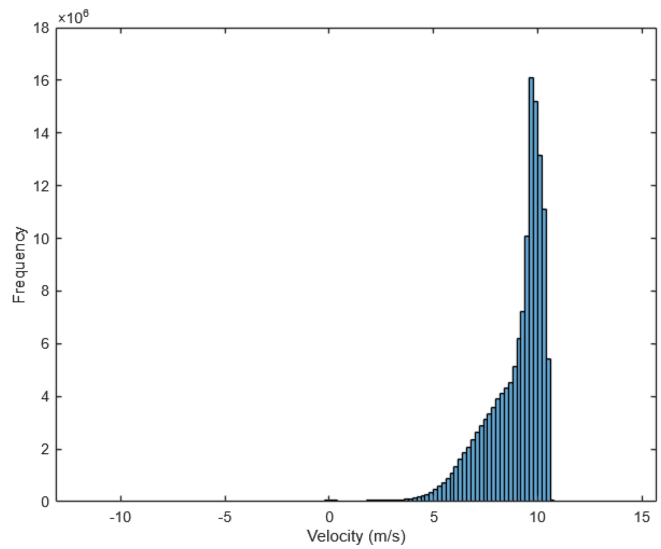

Figure 3: Histogram of all data

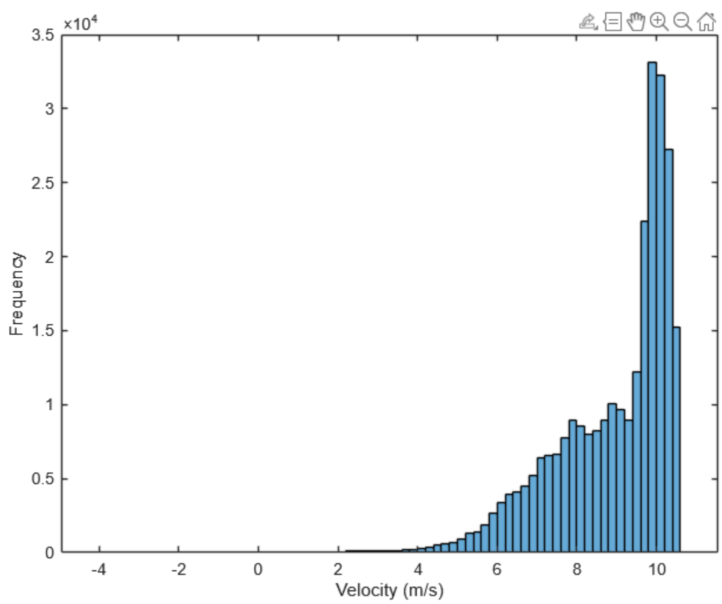

Figure 4: Histogram of data in frame 102

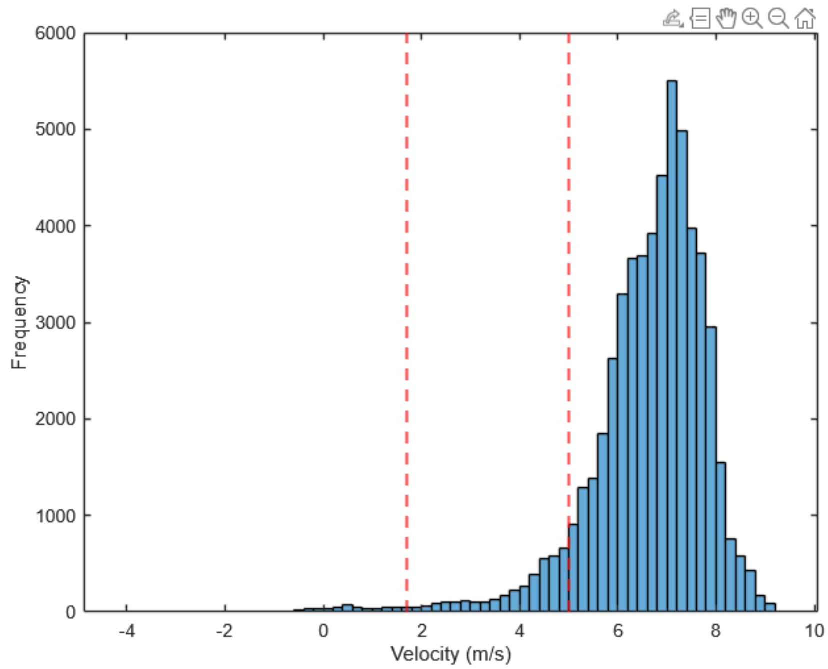

Figure 5: Histogram of data in frame 102 for x/d<0.25

In the three histograms, the peaks become less prominent and noisier the less data is included. If all the temporal and spatial data were included, there would be only one peak, around the mean velocity. For a smaller range of data, correlated fluctuations that are part of a coherent structure are more visible. Therefore, more peaks are visible in the third histogram.

In the third histogram, three UMZs are indicated.

b.

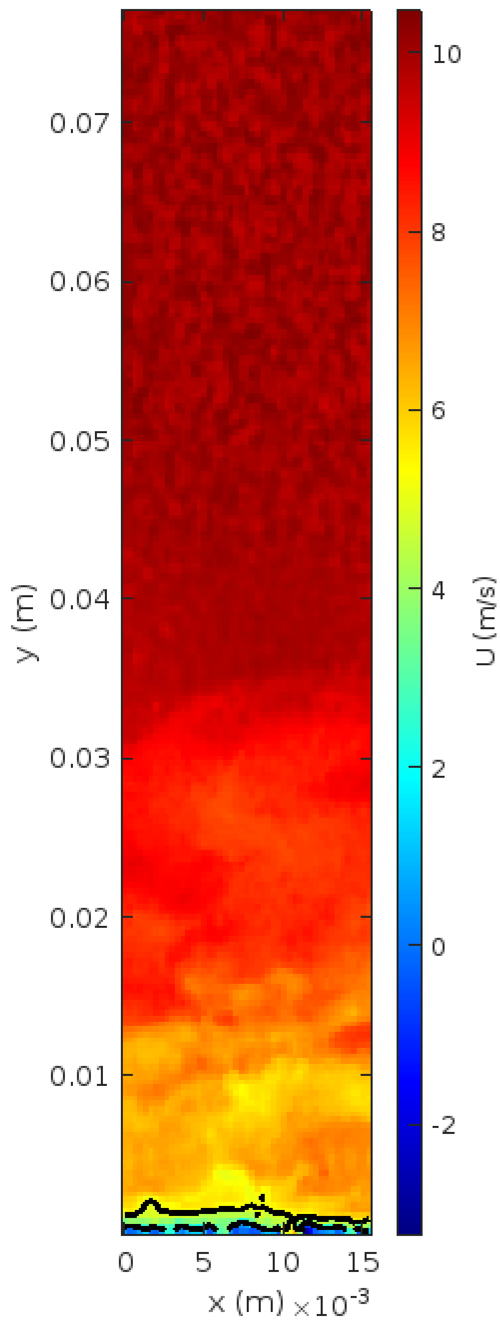

Figure 6: Contour plot of UMZs in flow

In this plot which outlines the UMZs, the coherent structures are visible. The first is in the wall-attached small eddies. The second is in a relatively near-wall region of approximately 4 m/s, and the third is in the wake region. Two of the three UMZs are concentrated near the wall, which might indicate that the coherent structures are more prevalent near a wall.

c.

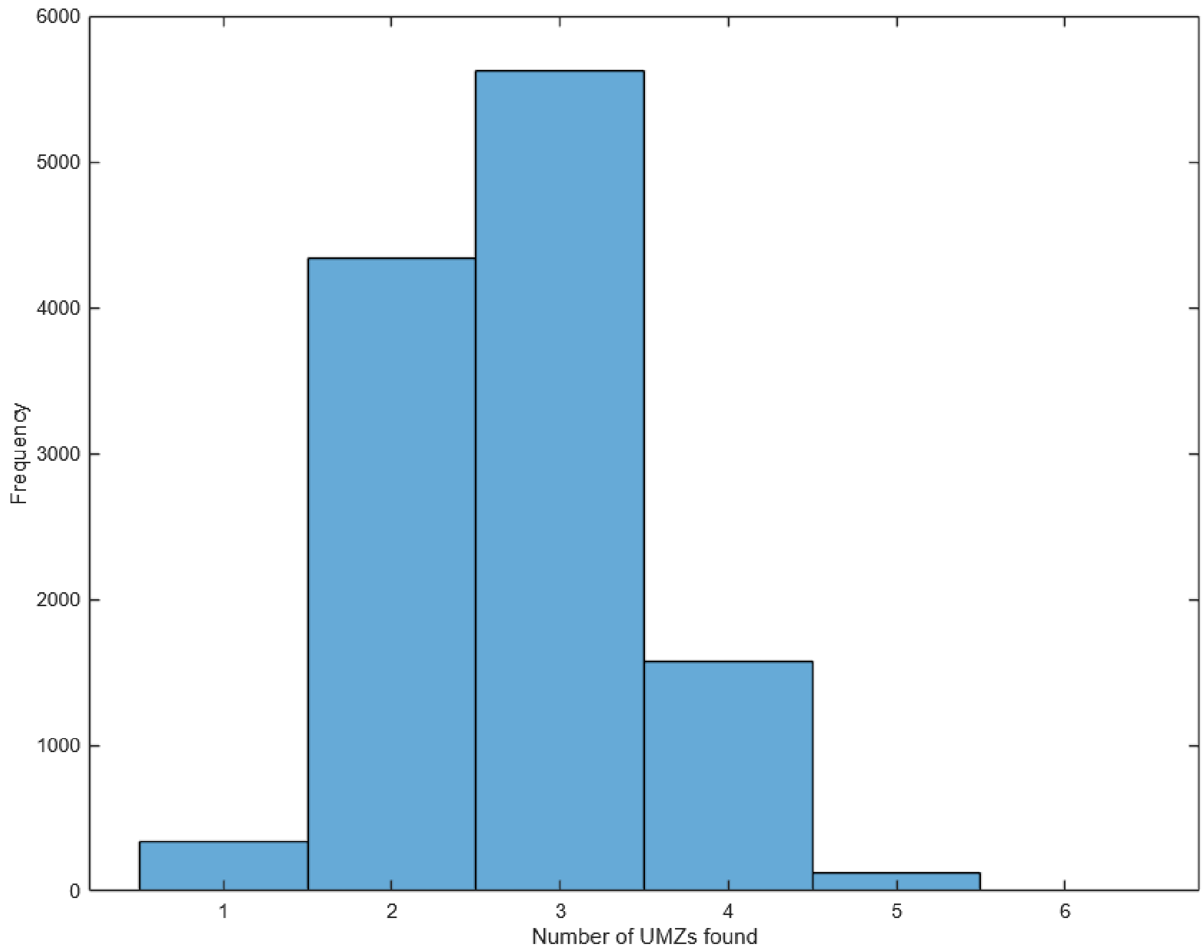

Figure 7: Histogram of UMZs

The mean number of UMZs is 2.74 with a bin size of 0.2m/s and a minimum peak prominence of 100.

The data from Silva et al. show a peak in detected UMZs at 4 for a similar Re_tau flow (1200). The mean number of UMZs is 3.2. The difference is most likely from differences in detection methods, for example using a different minimum peak prominence (which has strong effects on the detected UMZs) or a different bin size.

Overall, the trend is the same, with a peak at a low number of UMZs and then a tail as the number of UMZs increases.

d.

e.

Question 6.

a.

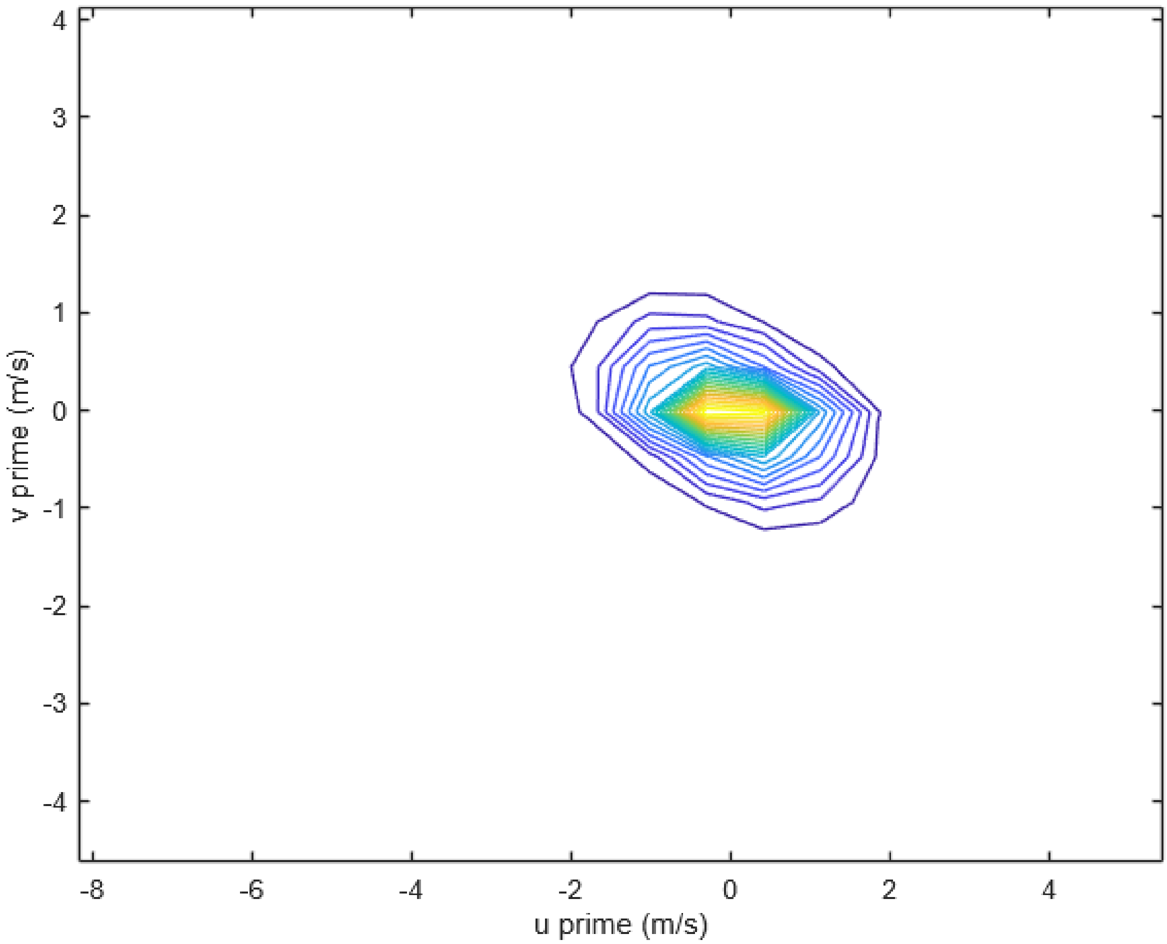

Figure 8: Contour plot of histogram of u’v’ data

b.

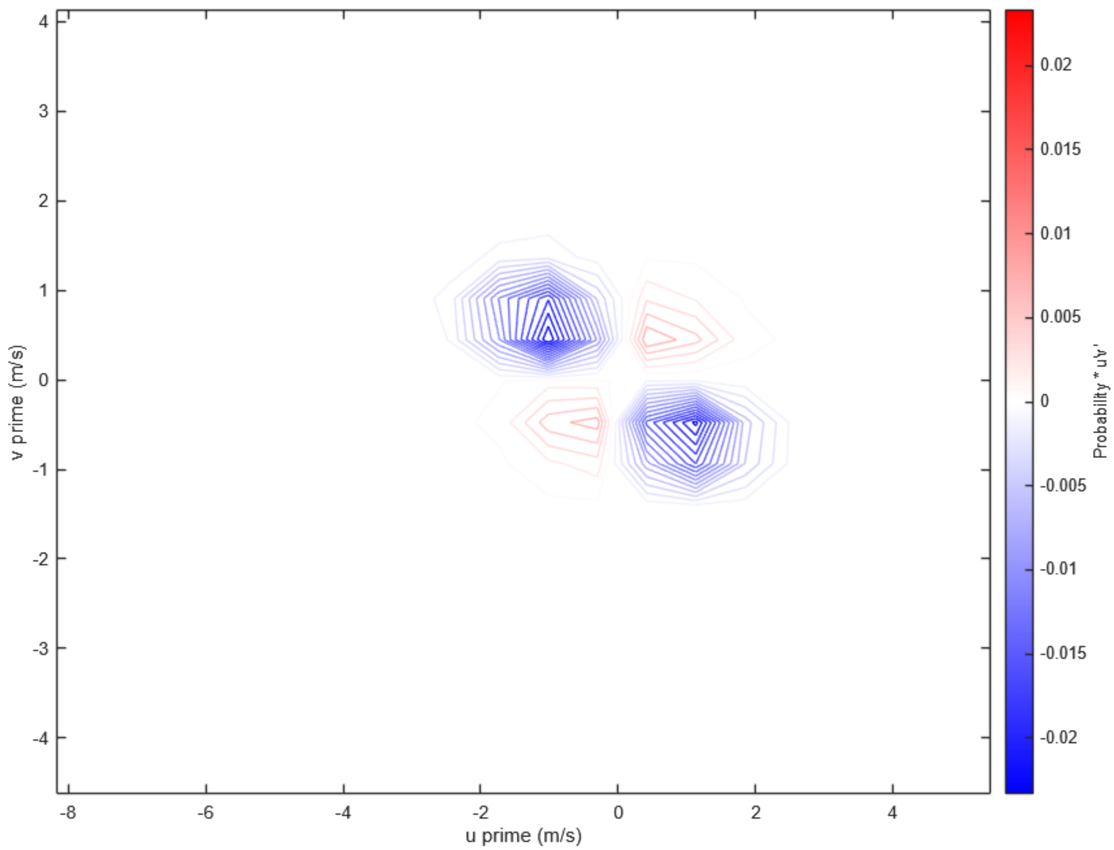

Figure 8: Premultiplied contour plot of histogram of u'v' data

c.

Data from the logarithmic region were plotted in the above contour histogram and premultiplied histogram. The first plot reveals that most fluctuations are concentrated near the 0-0 level, or that most fluctuations are small. This makes intuitive sense, since fluctuations tend to be small, so the product of fluctuations is rarely large.

The more interesting plot is the second one, which is the probability multiplied by the magnitude of the product of fluctuations. The strongest areas are negative (in quadrant 2 and 4), and the peaks in these areas are not located at 0,0. This indicates that the biggest contribution to the total Reynolds stresses comes from somewhat large u'v' values. The faintness of the positive contours in quadrants 1 and 3 indicates an expected result, that positive fluctuations are rare. The negative contours indicate where turbulent kinetic energy is produced, which is in quadrant 2 and 4.

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^+');
```