Project SA1 - Aircraft Wing Analysis First Report

Schoen Cao (zc282), Kai Tan (ykt25)

May 22, 2019

1 Exercise 1

Listing 1: ueintbit.m

```
function f = ueintbit(xa,ua,xb,ub)

    %Calculate for variables
    ubar = (ua+ub)/2;
    du = ub-ua;
    dx = xb-xa;

    %Solve for integral
    f = (ubar^5 + 5/6*ubar^3*(du)^2 + 1/16*ubar*(du)^4)*dx;
end
```

Listing 2: Script for Exercise 1

```
clear
close all

%Input variables
ReL = 2500;
x = linspace(0,1,101);
ue = 1;

%Iterate for theta/L
for i=1:length(x);
    theta(i) = sqrt(.45/ReL*(ue)^-6*ueintbit(0,ue,x(i),ue));
end

%Plot of Analytical Solution
hold on
plot(x,theta);
xlabel('Non-dimensional position, x/L');
ylabel('Non-dimensional momentum thickness, theta/L');
title('Non-dimensional Momentum Thickness against Non-dimensional Position')

%Calculate Blasius Solution
thetab = 0.664/ReL^.5 .* (x).^.5;

%Plot Blasius Solution
plot(x,thetab);
legend('Analytical','Blasius','location','Southeast');
hold off
```

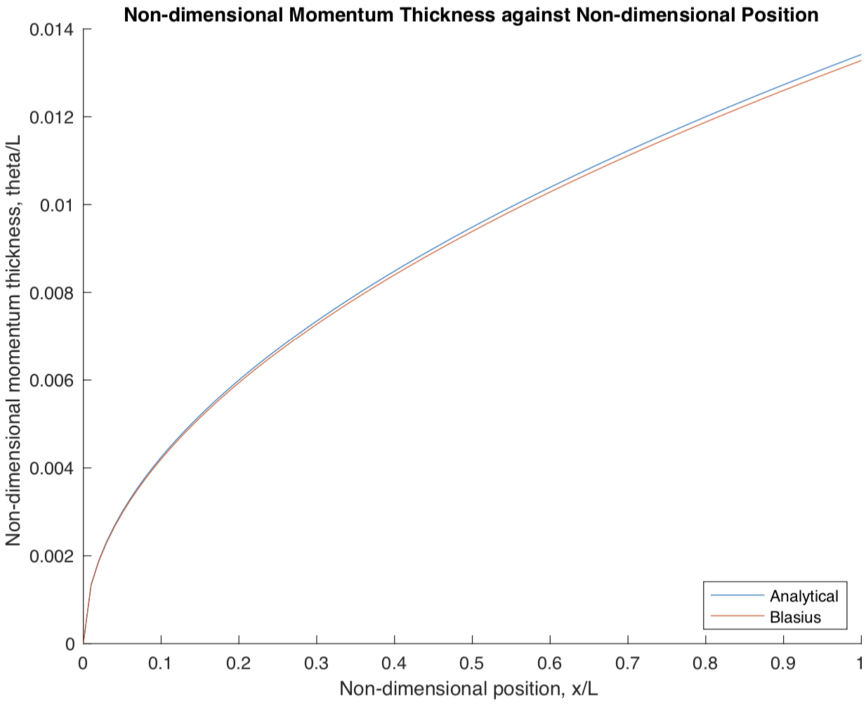

Figure 1: Plot of Non-Dimensional Momentum Thickness Against Non-Dimensional Position

2 Exercise 2

Listing 3: Script for Exercise 2

```
clear
close all

%Input Conditions, change according to required by Exercise
Re = 1e6;
duedx = -.2;

%Initial Conditions and Discretisation steps
n = 101;
laminar = true;
x = linspace(0,1,n);
ue0 = 1;

%Initialise transition and separation indicators
int = 0;
ils = 0;

%Generate a matrix of ue values
for i = 1:n;
    ue(i) = duedx*x(i)+ue0;
end

%To initialise loop
i = 1;
while laminar && i < n; %laminar loop
    i = i+1;    %Increase interation counter

    %Solve for theta, Rethet, m, H, He
    theta(i) = sqrt(.45/Re*(ue(i))^-6*ueintbit(0,ue(1),x(i),ue(i)));
    Rethet = theta(i)*Re*(ue(i));
   m = -Re*(theta(i))^2*duedx;
   H = thwaites_lookup(m);
   He(i) = laminar_He(H);

   %Check for transition
    if log(Rethet) >= 18.4*He(i) - 21.74;
        laminar = false;    %Flow no longer laminar
        int = i;    %Save iteration when transition occurs
        display([x(i), Rethet/1000]);
    end
end
```

$d(u_e/U)/d(x/L)$	Re_L (10^6)	x/L	Re_θ (10^3)
-0.2	1	0.58	0.5666
-0.2	10	0.24	1.0802
-0.2	100	0.04	1.3498
0	10	0.37	1.2903
0	100	0.04	1.3416
0.2	100	0.04	1.3337

Table 1: Transition Locations and Corresponding Re_θ for Various Conditions

3 Exercise 3

Listing 4: Script for Exercise 3

```
clear
close all

%Input Conditions, change according to required by Exercise
Re = 1e6;
duedx = -.5;

%Initial Conditions and Discretisation steps
n = 101;
laminar = true;
x = linspace(0,1,n);
ue0 = 1;

%Initialise transition and separation indicators
int = 0;
ils = 0;

%Generate a matrix of ue values
for i = 1:n;
    ue(i) = duedx*x(i)+ue0;
end

%To initialise loop
i=1;
while laminar && i < n; %laminar loop
    i = i+1;    %Increase interation counter

    %Solve for theta, Rethet, m, H, He
    theta(i) = sqrt(.45/Re*(ue(i))^-6*ueintbit(0,ue(1),x(i),ue(i)));
    Rethet = theta(i)*Re*(ue(i));
   m = -Re*(theta(i))^2*duedx;
   H = thwaites_lookup(m);
   He(i) = laminar_He(H);

    %Check for transition
    if log(Rethet) >= 18.4*He(i) - 21.74;   %Transition condition
        laminar = false;    %Flow no longer laminar
        int = i;    %Save iteration where transition occurs

    %Check for separation
    elseif m >= 0.09;   %Separation Condition
        laminar = false;    %Flow no longer laminar
        ils = i;    %Save iteration where transition occurs
    end
end

if int ~= 0;
    disp(['Natural transition at ' num2str(x(int)) ' with Rethet ' num2str(Rethet)]);
elseif ils ~= 0;
    disp(['Laminar seperation at ' num2str(x(ils)) ' with Rethet ' num2str(Rethet)]);
end
```

Re_L	Separation Location (x/L)
10^4	0.25
10^5	0.25
10^6	0.25

Table 2: Separation Location for Various Flow Conditions

Listing 5: Script for Exercise 3 to Find Re_L at which Transition Supplants Laminar Separation

```
clear
close all

%Input Conditions, change according to required by Exercise
Re = linspace(1e6,1e7,9001);
duedx = -.5;

%Initial Conditions and Discretisation steps
n = 101;
x = linspace(0,1,n);
ue0 = 1;

%Generate a matrix of ue values
for i = 1:n;
    ue(i) = duedx*x(i)+ue0;
end

%To initialise loop
for k = 1:length(Re);
    i=1;

    %Initialise indicators at the start of every loop
    int = 0;
    ils = 0;

    %Reset laminar flag at the start of every loop
    laminar = true;

    %Laminar loop
    while laminar && i < n;
        i = i+1;    %Increase interation counter

        %Solve for theta, Rethet, m, H, He
        theta(i) = sqrt(.45/Re(k)*(ue(i))^-6*ueintbit(0,ue(1),x(i),ue(i)));
        Rethet = theta(i)*Re(k)*(ue(i));
        m = -Re(k)*(theta(i))^2*duedx;
        H = thwaites_lookup(m);
        He(i) = laminar_He(H);

            %Check for transition
            if log(Rethet) >= 18.4*He(i) - 21.74;    %Transition condition
                laminar = false;    %Flow no longer laminar
                int = i;    %Save iteration where transition occurs

            %Check for separation
            elseif m >= 0.09;    %Separation Condition
                laminar = false;    %Flow no longer laminar
                ils = i;    %Save iteration where transition occurs
            end
    end

    %Display Re at which transition will supplant laminar separation
    if int ~= 0; %If transition occurs before separation, ils = 0
        disp(['Re_L at which transition supplants laminar transion is ' ...
        ,num2str(Re(k))])
        break %break loop once required Re is found
    end
end
```

From this code, it was found that the minimum Re_L for transition to supplant laminar separation is 1.792×10^6

4 Exercise 4

Listing 6: thickdash.m

```
function dthickdx = thickdash(xmx0,thick);
    %import global variables
    global Re ue0 duedx

    %find theta and deltaE as defined
    theta = thick(1);
    deltaE = thick(2);

    %calculate He
    He = deltaE/theta;

    %calculate H
    if He >= 1.46;
        H = (11*He+15) / (48*He-59);
    else
        H = 2.803;
    end

    %calculate ue
    ue = ue0 + duedx * xmx0;

    %calculate Re_theta
    Rethet = Re * ue * theta;

    %calculate cf
    cf = 0.091448*((H-1)*Rethet)^(-.232)*exp(-1.260*H);

    %calculate cdiss
    cdiss = 0.010019*((H-1)*Rethet)^(-1/6);

    %calculate f
    dthickdx = zeros(2,1);
    dthickdx(1) = cf/2 - (H+2) / ue * duedx * theta;
    dthickdx(2) = cdiss - 3/ue * duedx * deltaE;
end
```

Listing 7: Script for Exercise 4

```
clear
close all

%Defining global variables value
global Re ue0 duedx

Re = 1e7;
ue0 = 1;
duedx = 0;

%Initial values of theta and delta_E
x0 = 0.01;
thick0(1) = 0.037*x0*(Re*x0)^(-1/5);
thick0(2) = 1.80 * thick0(1);

[delx thickhist] = ode45(@thickdash,[0 0.99],thick0);

for i = 1:length(delx);

    theta_7(i) = 0.037 * (x0+delx(i)) * (Re * (x0+delx(i)))^(-1/5);
    theta_9(i) = 0.023 * (x0+delx(i)) * (Re * (x0+delx(i)))^(-1/6);
end
```

```
hold on
plot(delx+x0,thickhist(:,1));
plot(delx+x0,theta_7);
plot(delx+x0,theta_9);
xlabel('non dimensional position x/L');
ylabel('non dimensional momentum thickness \theta/L');
legend('Differential equation','1/7^{th} Power Law Estimate','1/9^{th} Power Law ...
Estimate','location','Southeast');
hold off
```

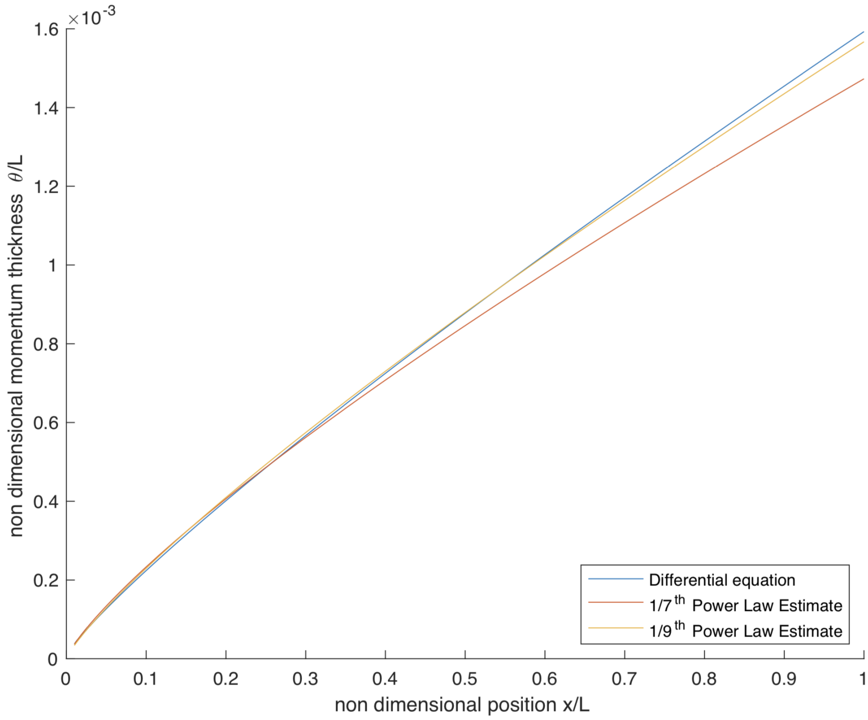

Figure 2: Non-Dimensional Momentum Thickness against Non-Dimensional Position for Various Solutions

5 Exercise 5

Listing 8: Script for Exercise 5b

```
clear
close all

%Defining global variables value
global Re ue0 duedx

Re = 1e7;
ue0 = 1;
duedx = -.3;

%Boundary values of theta and delta_E
x0 = 0.01;
thick0(1) = 0.037*x0*(Re*x0)^(-1/5);
thick0(2) = 1.80 * thick0(1);

[delx thickhist] = ode45(@thickdash,[0 0.99],thick0);

%Calculate He
for i = 1:length(delx);
    He(i) = thickhist(i,2)/thickhist(i,1);
end

%Plotting duedx = -.6
clear He
duedx = -.6;
[delx thickhist] = ode45(@thickdash,[0 0.99],thick0);
for i = 1:length(delx);
    He(i) = thickhist(i,2)/thickhist(i,1);
end

%Plot theta and deltaE
figure(1);
hold on
plot(delx+x0,thickhist(:,1));
plot(delx+x0,thickhist(:,2));
xlabel('non dimensional position x/L');
ylabel('non dimensional thickness');
legend('\theta','\delta_E','location','Northwest');
title(['Re_L=',num2str(Re),' d(u_e/U)/d(x/L)=',num2str(duedx)]);
hold off

%He plot to be done command window and by changing the input variables
```

By plotting the graphs for H_E again x/L using the command window, the turbulent separation for the various flow conditions can be found. The results are tabulated in Table 3 and Table 4.

$d(u_e/U)/d(x/L)$	Turbulent Separation Location x/L
-0.3	NA
-0.6	0.814
-0.9	0.533

Table 3: Turbulent Separation Location for Various Velocity Gradient at $Re_L = 10^7$

Re_L	Turbulent Separation Location x/L
10^6	0.735
10^7	0.810
10^8	0.903

Table 4: Turbulent Separation Location for Various Re_L at $d(u_e/U)/d(x/L) = -0.6$

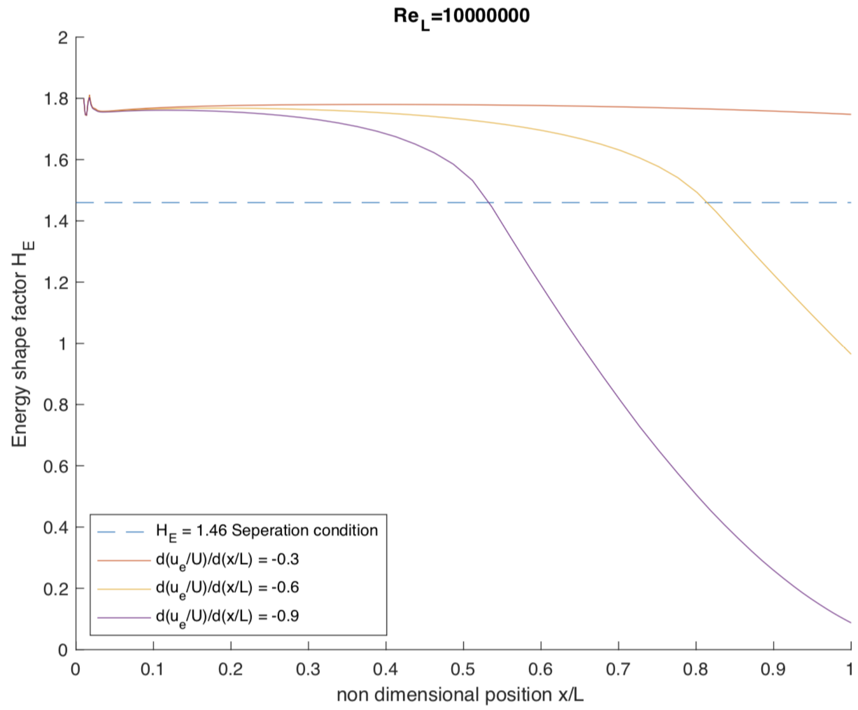

Figure 3: Plot of Turbulent Separation for Various Velocity Gradient at $Re_L = 10^7$

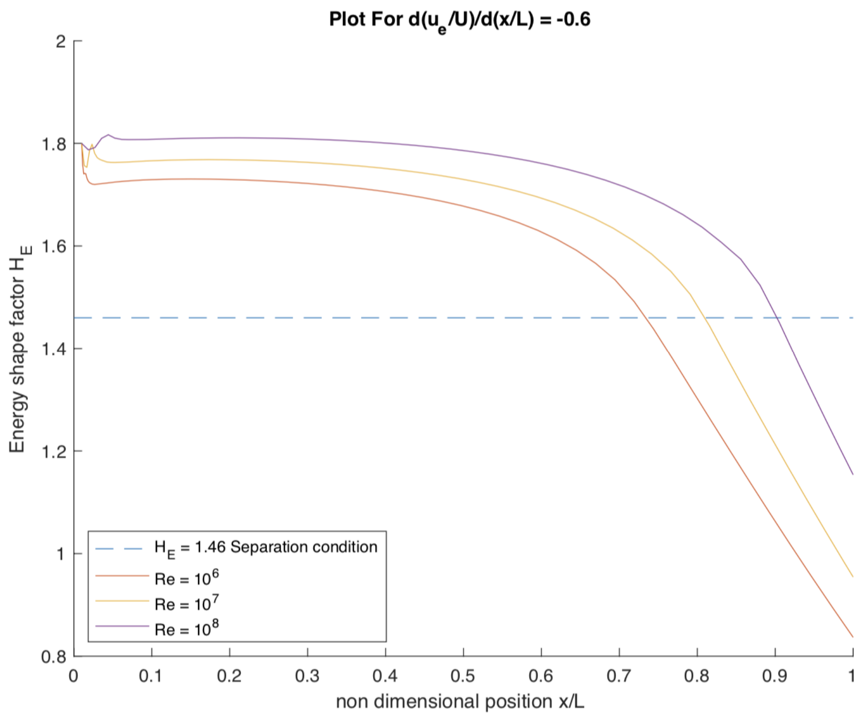

Figure 4: Plot of Turbulent Separation Location for Various Re_L at $d(u_e/U)/d(x/L) = -0.6$

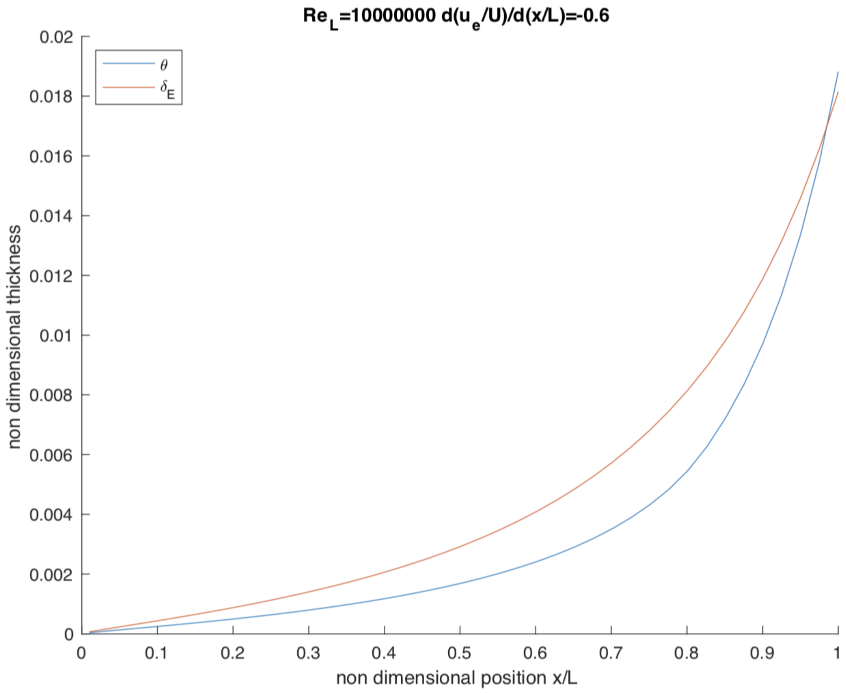

Figure 5: Plot of $\frac{\delta_E}{L}$ and $\frac{\theta}{L}$ Against Non-Dimensional Position

6 Exercise 6

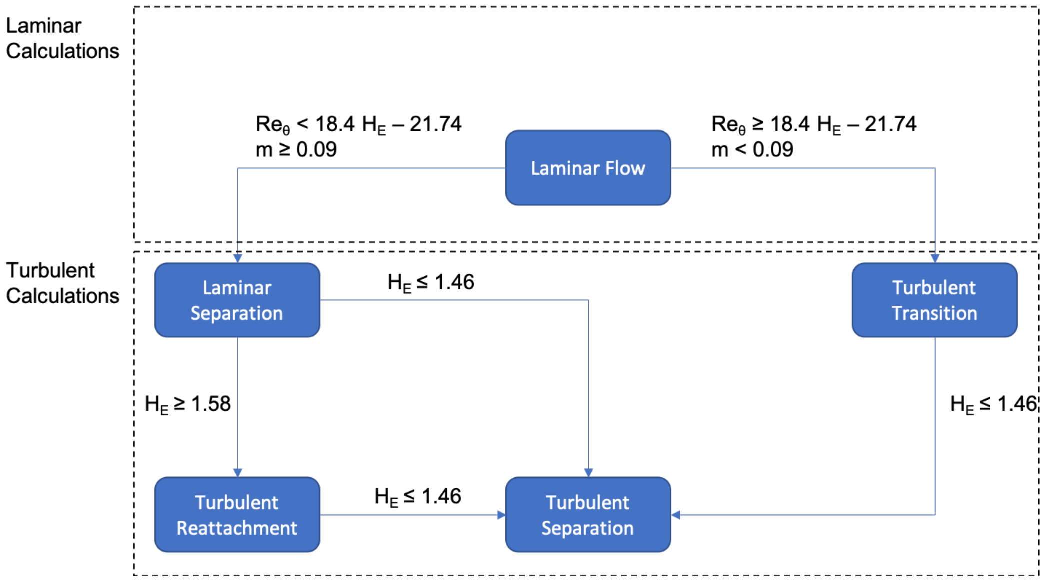

Figure 6: Flow Chart of Development of Flow

Listing 9: Script for Exercise 6

```
clear
close all

%Defining global variables value
global Re ue0 duedx

%Define simulation conditions
Re = 1e5;
duedx = -0.25;
ue0 = 1;

%Iteration setting & initial conditions
n = 101;
laminar = true;
x = linspace(0,1,n);

%%initialsing indicators
int = 0;    %natural transition
ils = 0;    %laminar seperation
itr = 0;    %turbulent reattachment
its = 0;    %turbulent seperation

%generating ue matrix
for i = 1:n;
    ue(i) = duedx*x(i)+ue0;
end

%initialising i
i = 1;
while laminar && i < n; %laminar loop
    i = i+1;    %interpretation
    theta(i) = sqrt(.45/Re*(ue(i))^-6*ueintbit(0,ue(1),x(i),ue(i)));
    Rethet = theta(i)*Re*(ue(i));
   m = -Re*(theta(i))^2*duedx;
   H = thwaites_lookup(m);
   He(i) = laminar_He(H);
    if log(Rethet) >= 18.4*He(i) - 21.74;    %laminar check
        laminar = false;    %laminar flag & end loop
```

```
        int = i;    %set pointer
        disp(['Turbulent Transition at x/L = ' num2str(x(int)) ' at Re_L '...
        num2str(Re)]);

    elseif m >= 0.09;   %seperation check
        laminar = false;    %also end laminar loop & turbulent formula
        ils = i;
        He(i) = 1.51509;    %set He to seperated value
        disp(['Laminar Separation at x/L = ' num2str(x(ils)) ' at Re_L '...
        num2str(Re)])
    end
end

%Value for He for Laminar Flow
He(1) = 1.57258;

%Calculate deltaE matrix
deltaE = He.*theta;

%Turbulent Loop
while its == 0 && i < n;
    thick0(1) = theta(i);   %y matrix, value at elemental plate's start
    thick0(2) = deltaE(i);
    i = i+1;
    [delx thickhist] = ode45(@thickdash,[0,x(i)-x(i-1)],thick0);
    theta(i) = thickhist(length(delx),1);   %assign value at elemental plate's end
    deltaE(i) = thickhist(length(delx),2);
    He(i) = deltaE(i)/theta(i);

    %Check for turbulent reattachment
    if ils > 0 && He(i) >= 1.58 && itr == 0;
        itr = i;
        disp(['Turbulent Reattachmemt at x/L = ' num2str(x(itr)) ...
        ' at Re_L ' num2str(Re)]);
    end

    %Check for turbulent separation
    if He(i) <= 1.46;   %turbulent seperation check
        its = i;
        H=2.803;    %H at seperation
        disp(['Turbulent Separation at x/L= ' num2str(x(its)) ...
        ' at Re_L ' num2str(Re)]);
    end
end

while i < n;    %final loop
    theta(i+1) = theta(i)*(ue(i)/ue(i+1))^(H+2);    %theta for cf=0
    i = i+1;
    He(i) = He (its);   %H assumed to remain constant since He is constant
end

%Plot graph using command window
```

Listing 10: Script for Exercise 6 to Find Critical Velocity Gradient

```
clear
close all

%Defining global variables value
global Re ue0 duedx

%Define simulation conditions
Re = 1e5;
```

```
duedxtest = linspace(-0.55,-0.25,31); %Create an array for test gradient
ue0 = 1;

%Iteration setting & initial conditions
n = 101;
laminar = true;
x = linspace(0,1,n);

for k = 1:length(duedxtest); %loop for various velocity gradient conditions
    duedx = duedxtest(k);

    %generating ue matrix
    for i = 1:n;
        ue(i) = duedx*x(i)+ue0;
    end

    %initialising i
    i = 1;
    laminar = true;
    int = 0;    %natural transition
    ils = 0;    %laminar seperation
    itr = 0;    %turbulent reattachment
    its = 0;    %turbulent seperation
    while laminar && i < n; %laminar loop
        i = i+1;    %interpretation
        theta(i) = sqrt(.45/Re*(ue(i))^-6*ueintbit(0,ue(1),x(i),ue(i)));
        Rethet = theta(i)*Re*(ue(i));
        m = -Re*(theta(i))^2*duedx;
        H = thwaites_lookup(m);
        He(i) = laminar_He(H);
        if log(Rethet) >= 18.4*He(i) - 21.74;   %laminar check
            laminar = false;    %laminar flag & end loop
            int = i;    %set pointer
            %disp(['Turbulent Transition at x/L = ' num2str(x(int)) ' at Re_L ' num2str(Re)]

        elseif m >= 0.09;   %seperation check
            laminar = false;    %also end laminar loop & turbulent formula
            ils = i;
            He(i) = 1.51509;    %set He to seperated value
            %disp(['Laminar Separation at x/L = ' num2str(x(ils)) ' at Re_L ' num2str(Re)])
        end
    end

    %Value for He for Laminar Flow
    He(1) = 1.57258;

    %Calculate deltaE matrix
    deltaE = He.*theta;

    %Turbulent Loop
    while its == 0 && i < n;
        thick0(1) = theta(i);   %y matrix, value at elemental plate's start
        thick0(2) = deltaE(i);
        i = i+1;
        [delx thickhist] = ode45(@thickdash,[0,x(i)-x(i-1)],thick0);
        theta(i) = thickhist(length(delx),1);
        deltaE(i) = thickhist(length(delx),2);
        He(i) = deltaE(i)/theta(i);

        %Check for turbulent reattachment
        if ils > 0 && He(i) >= 1.58 && itr == 0;
            itr = i;
            %disp(['Turbulent Reattachmemt at x/L = ' num2str(x(itr)) ...
```

```
                '␣at␣Re_L␣' num2str(Re)]);
        end

        %Check for turbulent separation
        if He(i) <= 1.46;   %turbulent seperation check
                its = i;
                H=2.803;    %H at seperation
                %disp(['Turbulent Separation at x/L= ' num2str(x(its)) ...
                '␣at␣Re_L␣' num2str(Re)]);
        end
        if i==101 && its ~=0
                disp(['Critical␣Velocity␣Gradient␣is␣' num2str(duedx)]);
                break
        end

    end
end
```

Critical Velocity Gradient is found to be -0.51 (2 significant figures)

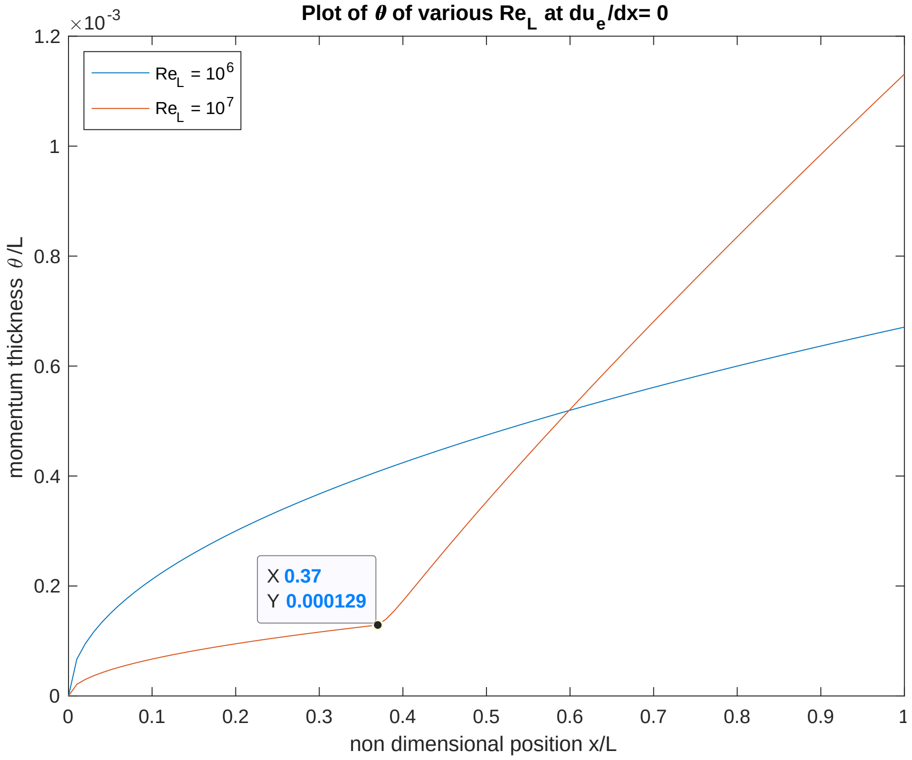

Figure 7: Plot of $\frac{\theta}{L}$ Against $\frac{x}{L}$ for Various Re and d(ue/U)/d(x/L) = 0

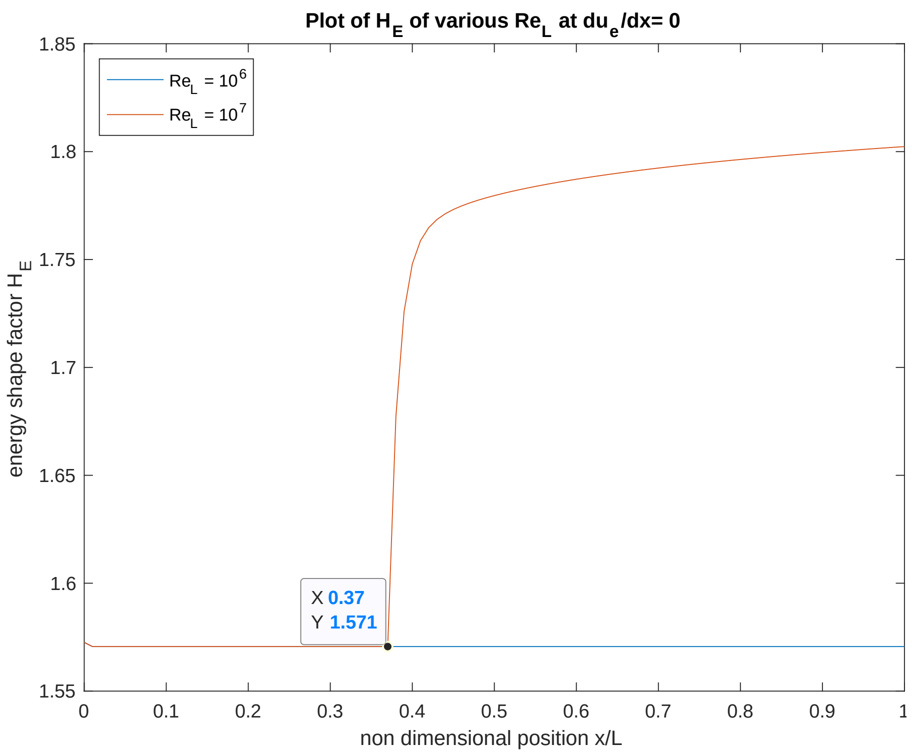

Figure 8: Plot of H_E Against $\frac{x}{L}$ for Various Re and d(ue/U)/d(x/L) = 0

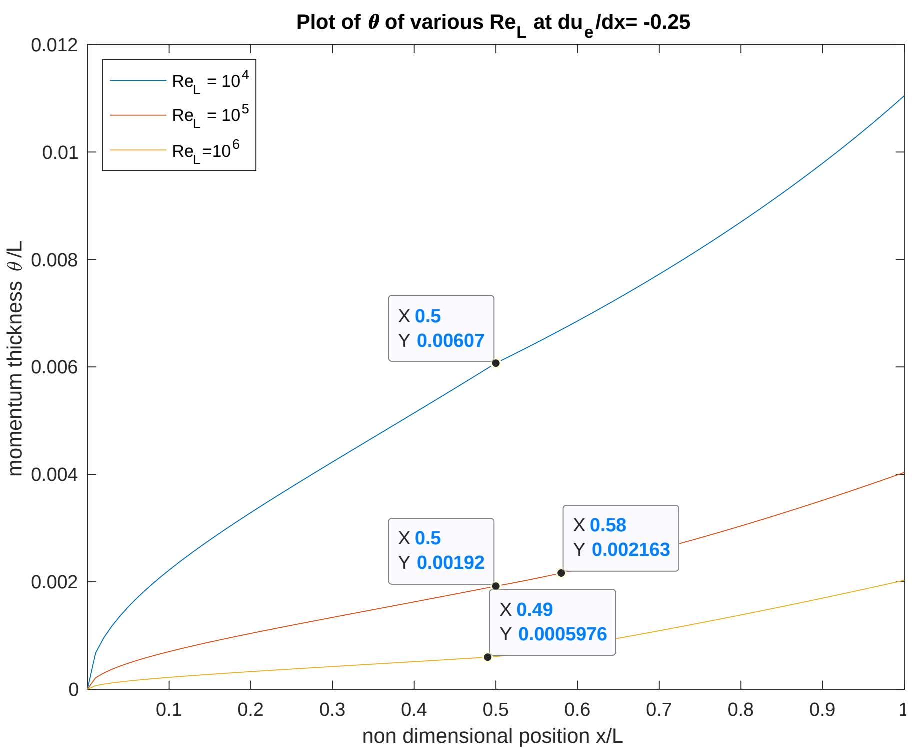

Figure 9: PPlot of $\frac{\theta}{L}$ Against $\frac{x}{L}$ for Various Re and d(ue/U)/d(x/L) = -0.25

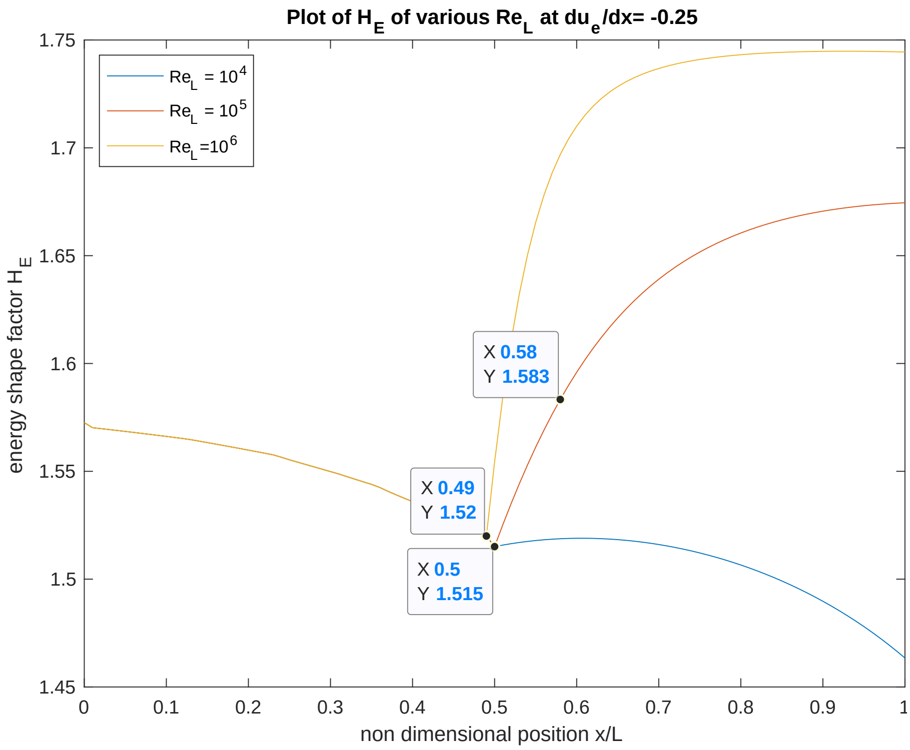

Figure 10: Plot of H_E Against $\frac{x}{L}$ for Various Re and d(ue/U)/d(x/L) = 0