AAE 334 HW6: 3D XFLR5 Analysis

Dr. Blaisdell

Friday March 6, 2020

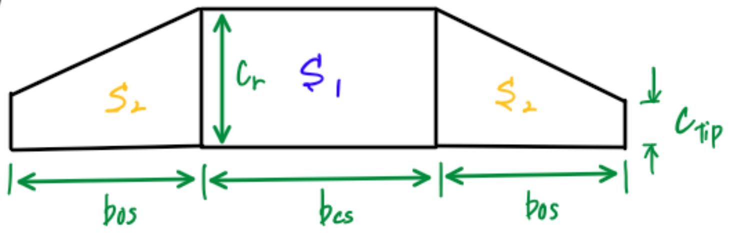
Tomoki Koike

*All Calculations will be handled using MATLAB

1. You will run XFLR5 to perform lifting line analysis on the wing from the general aviation aircraft shown on the attached sheet. The instructions for performing a wing analysis are given in the instructions file in the folder Homework/HW6 on Blackboard. The airplane has a wingspan of 36 ft. 1 in. The root chord length is cr = 5 ft. 6 in. The wing has a constant chord length over the center section, which is 16 ft. 9 in. long. Then the wing is tapered to the wing tip, where the chord length is 4 ft. 0 in. We will assume the leading edge of the entire wing is straight (no sweep), except as noted below. The airfoil section is a NACA 2412 for the entire wing. We will consider the design cruise conditions as follows:
   - Steady level flight at an altitude of 8,500 ft. (assume a standard atmosphere and use Sutherland's law to determine the viscosity of air at the relevant temperature)
   - Air speed of 124 kts.
   - Weight of 2550 lbs
     a. Determine the wing area (in ft2 and m2), wing aspect ratio, freestream Mach number, and wing lift coefficient needed to support the weight of the plane (using the wing planform area as the reference area and ignoring the tail). Is the freestream Mach number below 0.3? (It should be. Therefore, we will analyze the flow as incompressible, setting the Mach number to 0.) Determine the chord length Reynolds number of the wing root airfoil section and the chord length Reynolds number of the wing tip airfoil section

b = 36 ft 1 inch = 433 inch = 10.998 m
C_r = 5 ft 6 inch = 66 inch = 1.6764 m
C_{tip} = 4 ft 0 inch = 48 inch = 1.2192 m
b_{cs} := length of center section = 16 ft 9 inch = 5.1054 m
b_{os} := length of outer section = $b - b_{cs}$ = 5.8928 m

wing area S



$$S_1 = C_r \cdot b_{cs}$$
$$S_2 = (C_r + C_{tip}) \cdot b_{os}$$
$$\boxed{S = S_1 + S_2 = 25.6220 \ m^2}$$

wing aspect ratio AR

$$\boxed{AR = \frac{b^2}{S} = 4.7210}$$

freestream Mach Number, M_a

computing the standard conditions @ given altitude we know that

$$\gamma = 1.4$$
$$R = 287.05 \ J/kgK$$
$$T_a \ (@ \ 8500 ft) = 271.3098 \ K$$

and given that

$$U_\infty = 124 \ kts = 63.7911 \ m/s$$

$$\boxed{M_\infty = \frac{U_a}{\sqrt{\gamma R T_a}} = 0.1929}$$

lift coefficient C_L

$$C_L = \frac{W}{\frac{1}{2}\rho_\infty U_\infty^2 S}$$

where $W = 1.1343 \times 10^4 \ N$, $\rho_\infty = 0.9480$

$$\boxed{C_L = 0.2295}$$

Discussion:

- The calculated Mach number is lower and approximately the half of 0.3.

Reynolds number

defining $T_{ref} = 15°C = 288.15K$

$\mu_{ref} = 1.789 \times 10^{-5}$ Pa-s @ sea level

using Sutherland's formula

$$\mu = \mu_{ref} \left(\frac{T_a}{T_{ref}}\right)^{\frac{3}{2}} \frac{T_{ref} + S}{T_a + S}$$

where S = Sutherland's constant for air
$= 120$

thus, we can compute

$$\mu = 1.7048 \times 10^{-5} \text{ Pa-s}$$

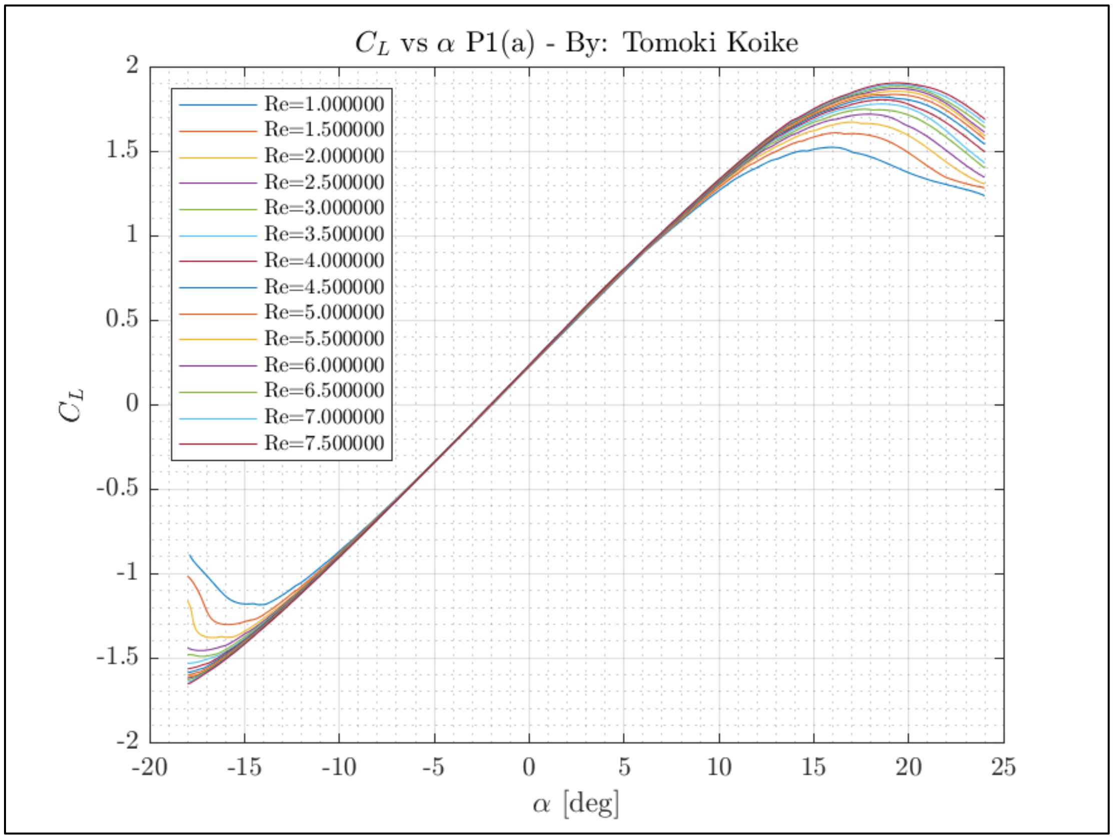
<i> wing root airfoil section

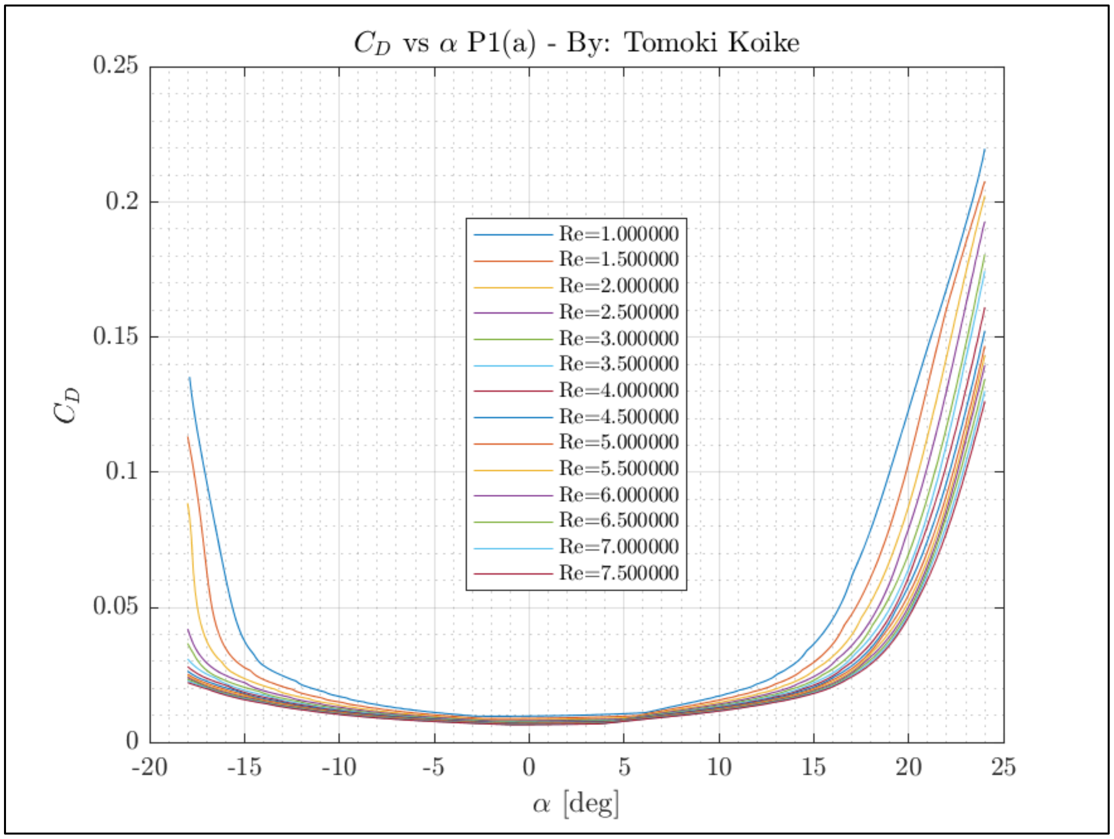
$$\boxed{Re_{,root} = \frac{\rho_\infty U_\infty C_r}{\mu} = 5.9469 \times 10^6}$$

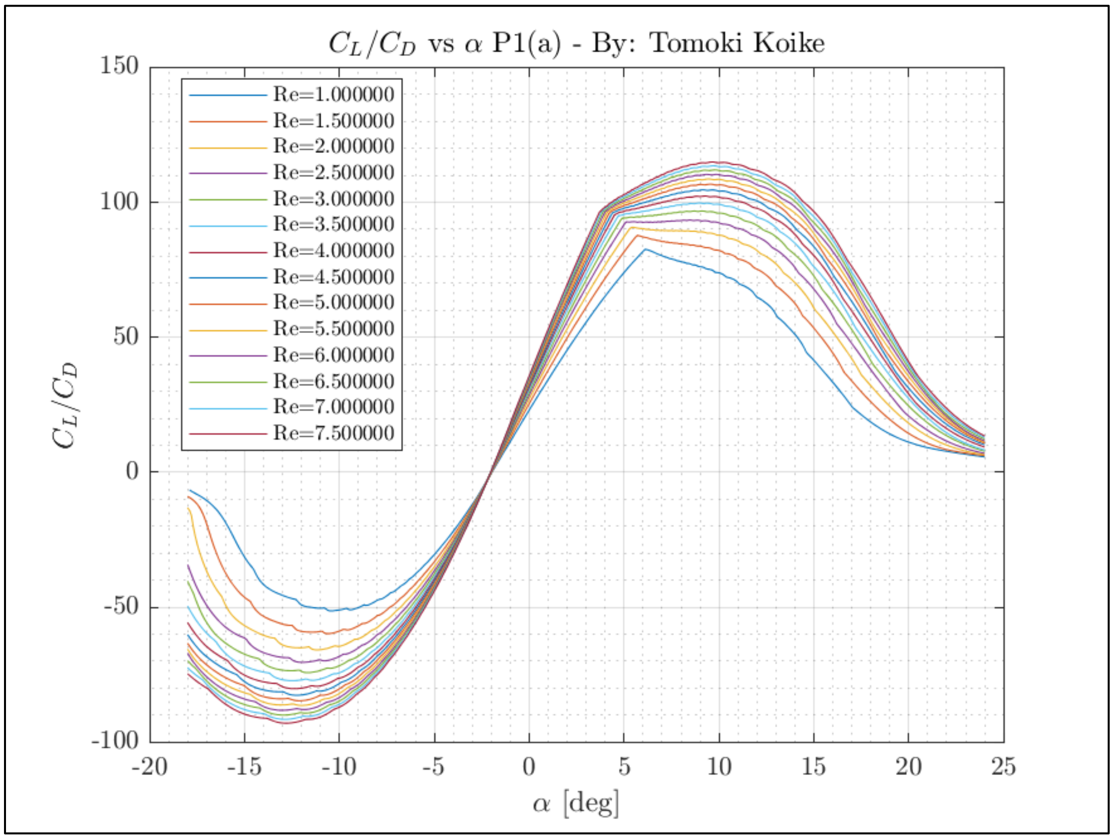
<ii> wing tip airfoil section

$$\boxed{Re_{,tip} = \frac{\rho_\infty U_\infty C_{tip}}{\mu} = 4.3250 \times 10^6}$$

b. We first need to run a batch airfoil analysis, as outlined in the instructions. (Note: use a refined geometry with 201 panels.) Since we will want to run the wing analysis at other flight conditions, we need a wider range of conditions than just the two Reynolds number found in part (a). Run a batch analysis for Reynolds numbers ranging from 1, 000, 000 to 7, 500, 000 with an increment of 500, 000. Set the boundary layer trip locations on the upper and lower surface to xtr/c = 0.20. This will provide nearly fully turbulent boundary layers at all conditions. (While this increases viscous drag, it helps avoid or reduce boundary layer separation.) Run a sequence of angles of attack from α = −18 ∘ to 24 ∘ with an increment of Δα = 0.1 ∘. Save the airfoil polars as described in the instructions. Plot the lift curve, drag polar, and lift to drag ratio from the batch analysis. How do the maximum lift coefficient, minimum drag coefficient and maximum lift to drag ratio change with Reynolds number?



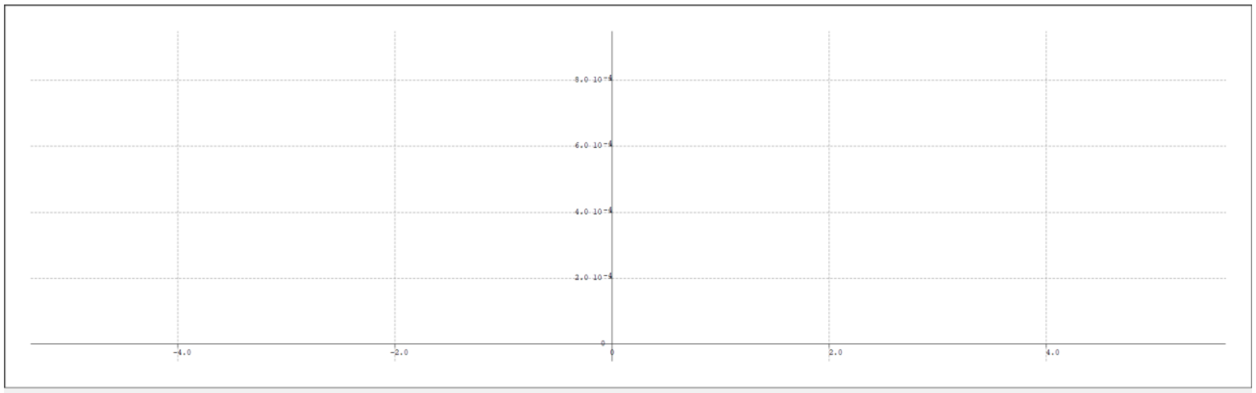

C_D vs α P1(a) - By: Tomoki Koike
C_D
α [deg]
0.25
0.2
0.15
0.1
0.05
0
-20
-15
-10
-5
0
5
10
15
20
25
Re=1.000000
Re=1.500000
Re=2.000000
Re=2.500000
Re=3.000000
Re=3.500000
Re=4.000000
Re=4.500000
Re=5.000000
Re=5.500000
Re=6.000000
Re=6.500000
Re=7.000000
Re=7.500000

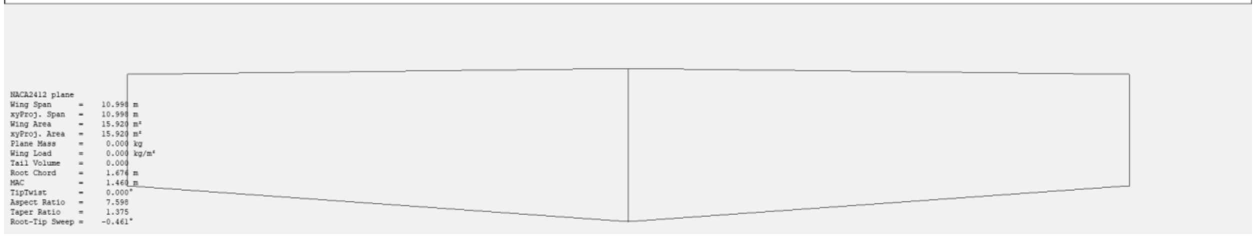

C_L/C_D vs α P1(a) - By: Tomoki Koike
C_L/C_D
α [deg]
150
100
50
0
-50
-100
-20
-15
-10
-5
0
5
10
15
20
25
Re=1.000000
Re=1.500000
Re=2.000000
Re=2.500000
Re=3.000000
Re=3.500000
Re=4.000000
Re=4.500000
Re=5.000000
Re=5.500000
Re=6.000000
Re=6.500000
Re=7.000000
Re=7.500000

Discussion:

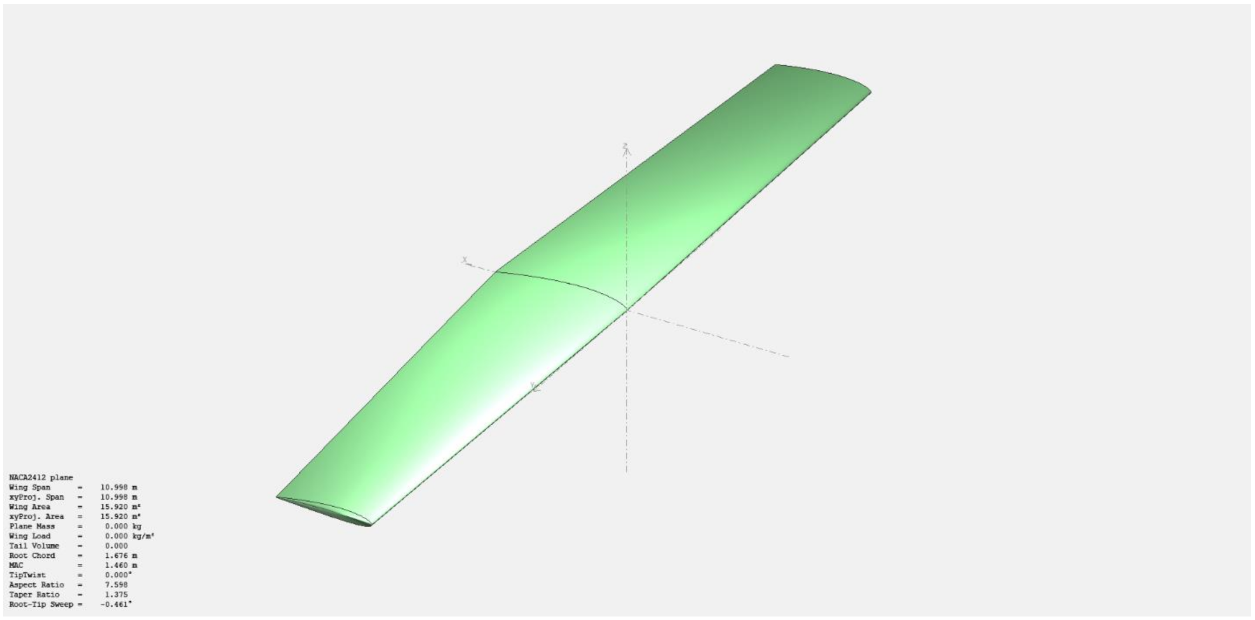
- The maximum lift coefficient increases as the Reynolds number increases
- The minimum drag coefficient become smaller as the Reynolds number increases
- The maximum lift-to-drag ratio increases as the Reynolds number increases

c. Follow the instructions and create the wing geometry using the NACA 2412 airfoil. Print a planform view of your wing and a perspective view.





```
NACA2412 plane
Wing Span      =    10.998 m
xyProj. Span   =    10.998 m
Wing Area      =    15.920 m²
xyProj. Area   =    15.920 m²
Plane Mass     =     0.000 kg
Wing Load      =     0.000 kg/m²
Tail Volume    =     0.000
Root Chord     =     1.676 m
MAC            =     1.460 m
TipTwist       =     0.000°
Aspect Ratio   =     7.598
Taper Ratio    =     1.375
Root-Tip Sweep =    -0.461°
```

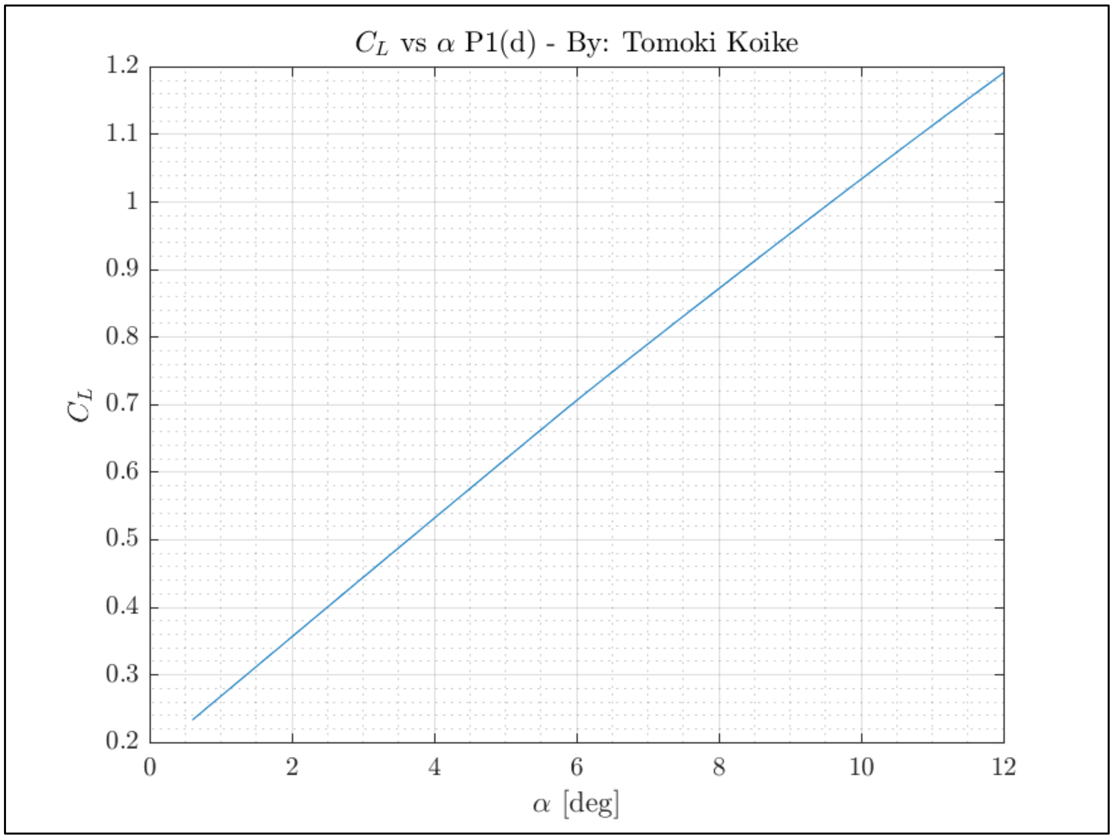


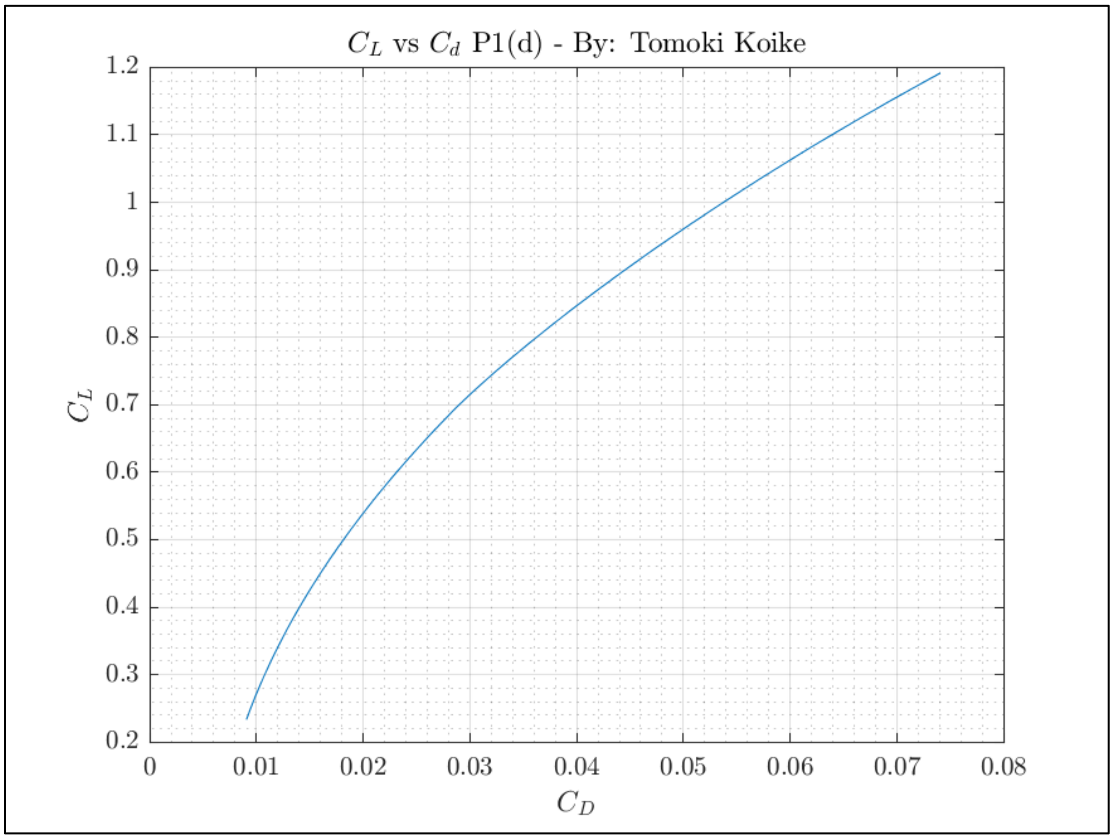
```
NACA2412 plane
Wing Span      =    10.998 m
xyProj. Span   =    10.998 m
Wing Area      =    15.920 m²
xyProj. Area   =    15.920 m²
Plane Mass     =     0.000 kg
Wing Load      =     0.000 kg/m²
Tail Volume    =     0.000
Root Chord     =     1.676 m
MAC            =     1.460 m
TipTwist       =     0.000°
Aspect Ratio   =     7.598
Taper Ratio    =     1.375
Root-Tip Sweep =    -0.461°
```

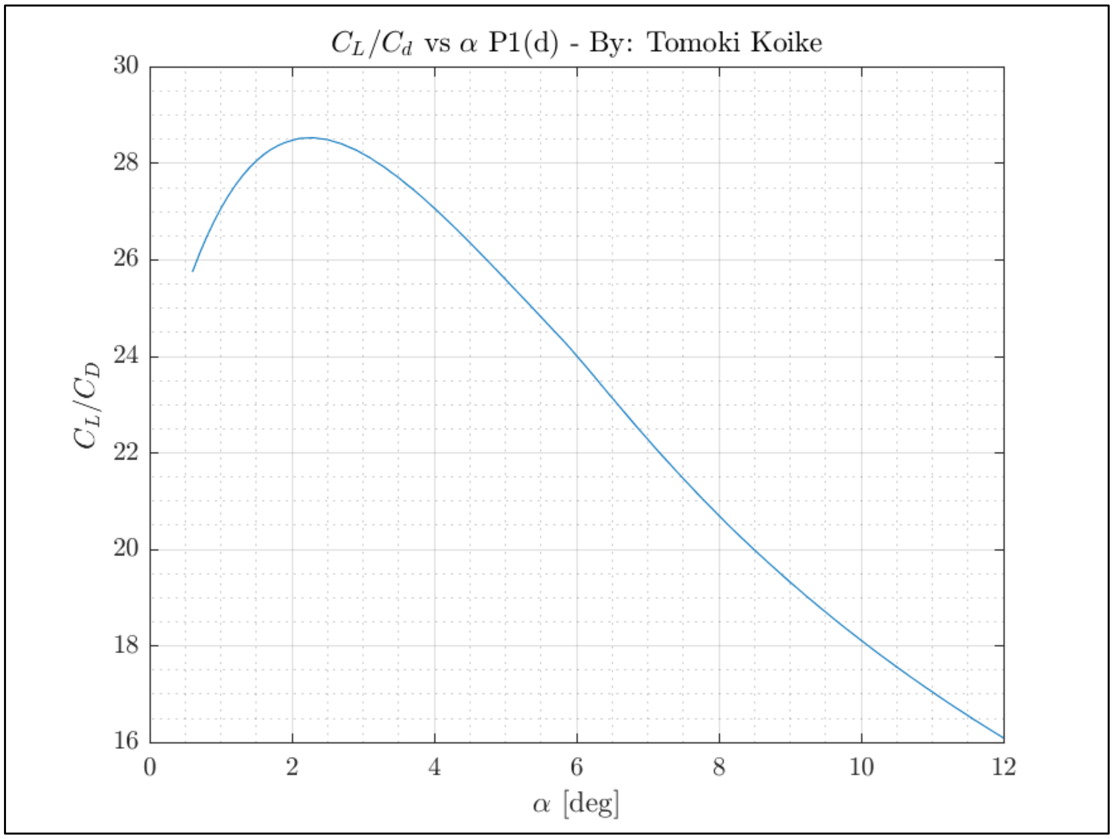
d. Run a Type 2 wing analysis, entering the mass of the airplane and the other needed flight parameters. (Use the atmospheric properties you determine from the Standard Atmosphere Tables in Anderson and Sutherland's law for viscosity (available on-line) rather than the conditions available from XFLR5.) When you enter the mass, set the center of gravity to x = 18.25 in = 0.46355 . downstream of the wing root leading edge. Use a range of angles of attack from $\alpha = 0^\circ$ to 12° using an increment of $\Delta\alpha = 0.1^\circ$. (There may be angles of attack for which the airfoil data is not available and the solution will not converge, but we will have the range of angles of attack we need.)
Plot the wing lift curve, drag polar and lift to drag ratio. Determine the maximum lift to drag ratio of the wing. How does this value compare to that of the airfoil? If one accounts for the drag of the fuselage, horizontal and vertical tails, and other appendages, such as antennae, the drag coefficient would more than double. Assuming the drag of the entire airplane is twice that of the wing, determine the maximum lift to drag ratio of the airplane? (This is a more realistic value for the lift to drag ratio of an airplane.) At what angle of attack does the maximum lift to drag ratio of the wing occur? What is the wing lift coefficient at this angle of attack? How does this lift coefficient compare to the one computed in part (a)? Use a finite difference approximation to compute the lift curve slope near the angle of attack for maximum lift to drag ratio. (Do this by computing $dCL/d\alpha \approx \Delta CL/\Delta\alpha$ using values of α on either side of the desired angle of attack.) How does the value of the lift curve slope compare to the theoretical lift curve slope for an elliptically loaded wing with NACA 2412 airfoil sections and the aspect ratio found in part (a)? (The theoretical lift curve slope of an elliptically loaded wing is given by

$$\frac{dC_L}{d\alpha} = \frac{a_0}{1 + \frac{a_0}{\pi AR}}$$

where a0 is the airfoil lift curve slope, which we will take to be a0 = 6.588 (per radian).)

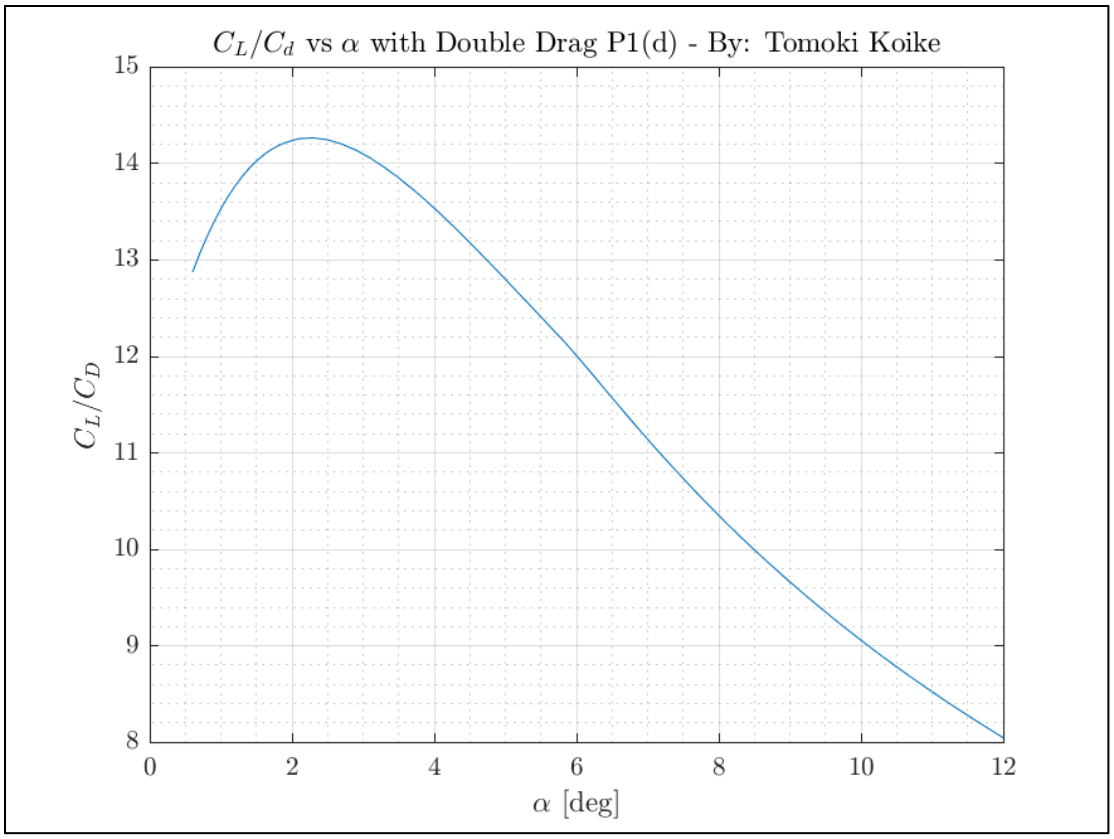

C_L vs α P1(d) - By: Tomoki Koike
C_L
α [deg]
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
1.1
1.2
0
2
4
6
8
10
12


C_L vs C_d P1(d) - By: Tomoki Koike
C_L
C_D
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
1.1
1.2
0
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08



Discussion:

- Using MATLAB, we found the maximum lift-to-drag ratio to be: 28.5341
- The maximum lift-to-drag ratio for the highest Reynolds number is 114.9686. From this we can compute the difference to be 86.4345. This is a substantial deviation.



Discussion:

- When the drag coefficient is doubled the maximum lift-to-drag ratio is going to become half 14.2671 @ the an AoA of 2.300
- At the AoA computed above the lift coefficient is 0.3830
- This is about 1.5 times higher than the lift coefficient computed in 1(a). This might be due to the crudeness of the analytical solution in 1(a)
- Using the finite difference approximation we can compute the lift curve slope to be 5.0376; whereas the theoretical value calculated from the elliptical loading is 4.5617 which has a 10.4325% difference

e. On the right side of the toolbar (near the top of the window) is a dropdown list of the angles of attack for which results are available. Select the angle of attack for the maximum lift to drag ratio of the wing. Then click on . A window will pop up with tabulated data. The variables of interest to us are the following: CD, the drag coefficient VCD, the viscous drag coefficient ICD, the induced drag coefficient Cm, the moment coefficient about the moment reference center (the c.g.) Tabulate the values of these quantities at this angle of attack and at α = 10 ◦ . Compute the ratio of the induced drag coefficient to the total drag coefficient for the two angles of attack. Compute the span efficiency factor e from CDi = C 2 L /(πeAR). Discuss how angle of attack affects the ratio of induced drag to total drag and the value of the span efficiency factor. Compute and tabulate the total dimensional drag for these two angles of attack. How do they compare? What are the corresponding flight speeds for these two angles of attack? How do they compare? The Breguet range equation relates the range of an aircraft to its velocity, lift to drag ratio, specific impulse of the propulsion system, and the initial and final weights of the airplane (due to the consumption of fuel). (See http://web.mit.edu/16.unified/www/FALL/thermodynamics/notes/node98.ht ml) From this equation velocity times lift to drag ratio is important. What is the ratio of the product of velocity times lift to drag ratio for the two angles of attack (with the product for α = 10 ◦ divided by the product for the angle of attack corresponding to the maximum lift to drag ratio)?

data_T = 2×6 table

	Alpha	CD	VCD	ICD	Cm	CL
1	2.3000	0.0134	0.0071	0.0063	-0.0350	0.3830
2	10.0000	0.0571	0.0108	0.0463	-0.0039	1.0342

Now using the function

$$C_{Di} = \frac{C_L^2}{\pi e AR}$$

we get the following relation

$$e = \frac{C_L^2}{\pi AR C_{Di}}$$

Using MATLAB we obtain e(@2.3deg) = 0.7368, e(@10deg) = 1.2630 data_T = 2×7 table

	Alpha	CD	VCD	ICD	Cm	CL	e
1	2.3000	0.0134	0.0071	0.0063	-0.0350	0.3830	0.7368
2	10.0000	0.0571	0.0108	0.0463	-0.0039	1.0342	1.2630

Discussion:

- The ICD and CD ratio for each angle of attacks were 0.4786 and 0.8103 respectively. From this we can tell that as the angle of attack increases prior to the stall angle the ratio increases as well.
- The span efficiency factor also increases with the angle of attack but not as much as the ICD/CD ratio

Now we can calculate the drags using the properties we have computed before and using CD above and the velocity to maintain steady level flight. The tabulated result is below (*D is in units of N). Also, the corresponding flight speeds should be the Vinf values from XFLR5. Thus, the result is the following.

data_T = 2×9 table

	Alpha	CD	VCD	ICD	Cm	CL	e	Vinf	D
1	2.3000	0.0134	0.0071	0.0063	-0.0350	0.3830	0.7368	62.6520	663.2636
2	10.0000	0.0571	0.0108	0.0463	-0.0039	1.0342	1.2630	38.1250	2.8221e+03

Discussion:

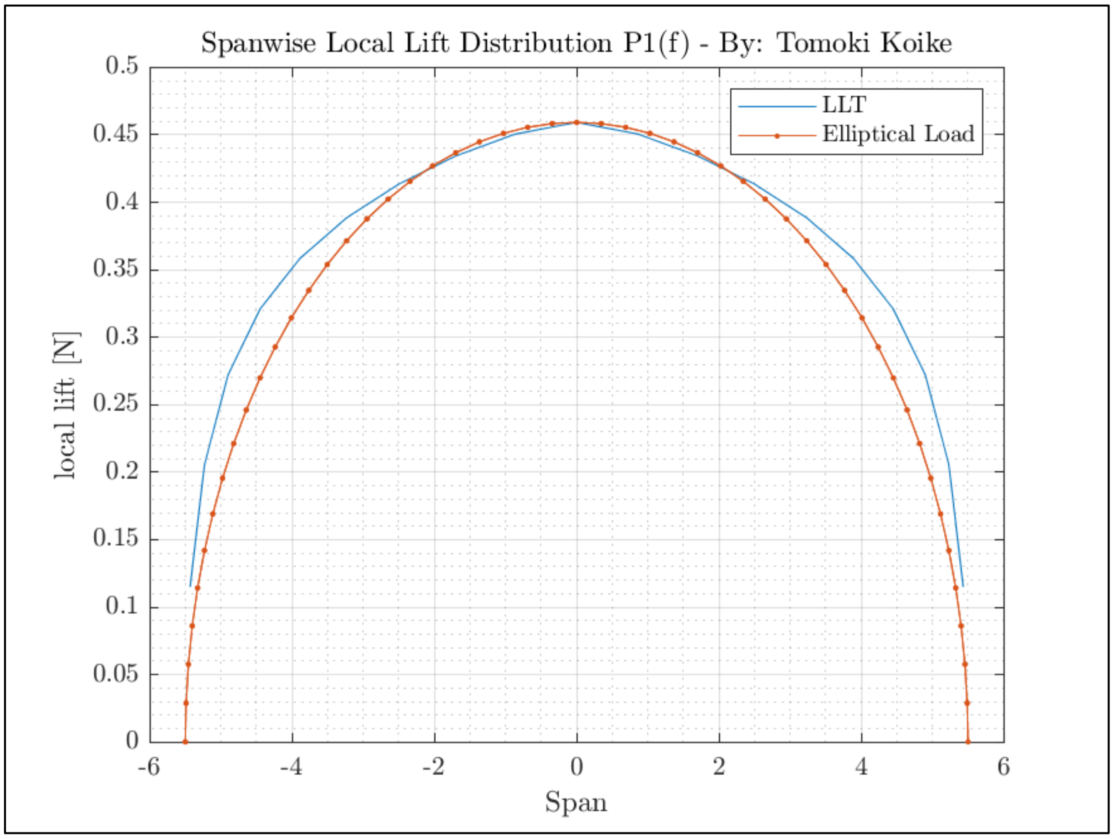
- The total dimensional drag is larger for when the angle of attack is larger. This means that the angle of attack of 2.3 degrees is favorable then 10 degrees to reduce the amount of drag acting on the aircraft.
- The flight speed is faster when the angle of attack is smaller.

The product of velocity times L/D are

@ alpha = 2.3 => 1787.9 m/s

@ alpha = 10 => 690.5 m/s

f. Select the OpPoint view from the toolbar. Choose for the lift distribution ("Local Lift"), L ′ (y). Choose the angle of attack that corresponds to maximum lift to drag ratio of the wing. Add the elliptic load distribution curve following the instructions. Plot the comparison and discuss similarities and differences between the load distribution for this wing and an elliptic load distribution. Look at the spanwise load distribution for α = 10 ∘ . Is there much of a difference due to the change in angle of attack?



Discussion:

- For the locations where it is close the root of the wing, the LLT and elliptical load distribution overlap with each other and show same values; however, the farther it is from the root the two load distributions deviate. For this deviation the LLT values are larger than the elliptical load distribution values.
- By observing the case for when the angle of attack is 10.0 degrees the values overall become larger; however, the similarities and the differences of the LLT and elliptical load distribution are identical.

g. Assume the horizontal tail is made of thin symmetric airfoil sections. Then the center of pressure is the same as the aerodynamic center, which is at the tail quarter chord location. Assume the quarter chord locations of the tail airfoils are lined up at the same axial position, although that is not exactly the case from the drawing. The tail aerodynamic center is located 15 ft. 6 in. behind the wing leading edge. What dimensional value of lift is needed on the tail to balance the moment about the c.g. at the angle of attack that corresponds to the maximum lift to drag ratio? What value is needed at α = 10∘ ?

For the moments about the cg for this aircraft for the 2 lifts generated by the wing and tail to balance out we calculate the moments produced by each lift. To calculate the lift on the wing we must integrate the lift we found in the previous problem 1(f). Using MATLAB we integrate the total local lift to be

Local_lift_tot = 205220 N

from XFLR5 we know the center of gravity is X_CG = 0.464 m from backend of wing and center of pressure is

X_CP = 0.593 m

$$M_{LE} = X\_CP \int_{-b/2}^{b/2} L'(y)\,dy$$

divide this by wingspan

@ alpha = 2.3 degrees >> M_LE = 10165 N-m

This is going to equal the moment of tail about the center of pressure X_CP, computing this with MATLAB we get

d = 15ft 6 inch – X_CP

d = 4.1314

thus,

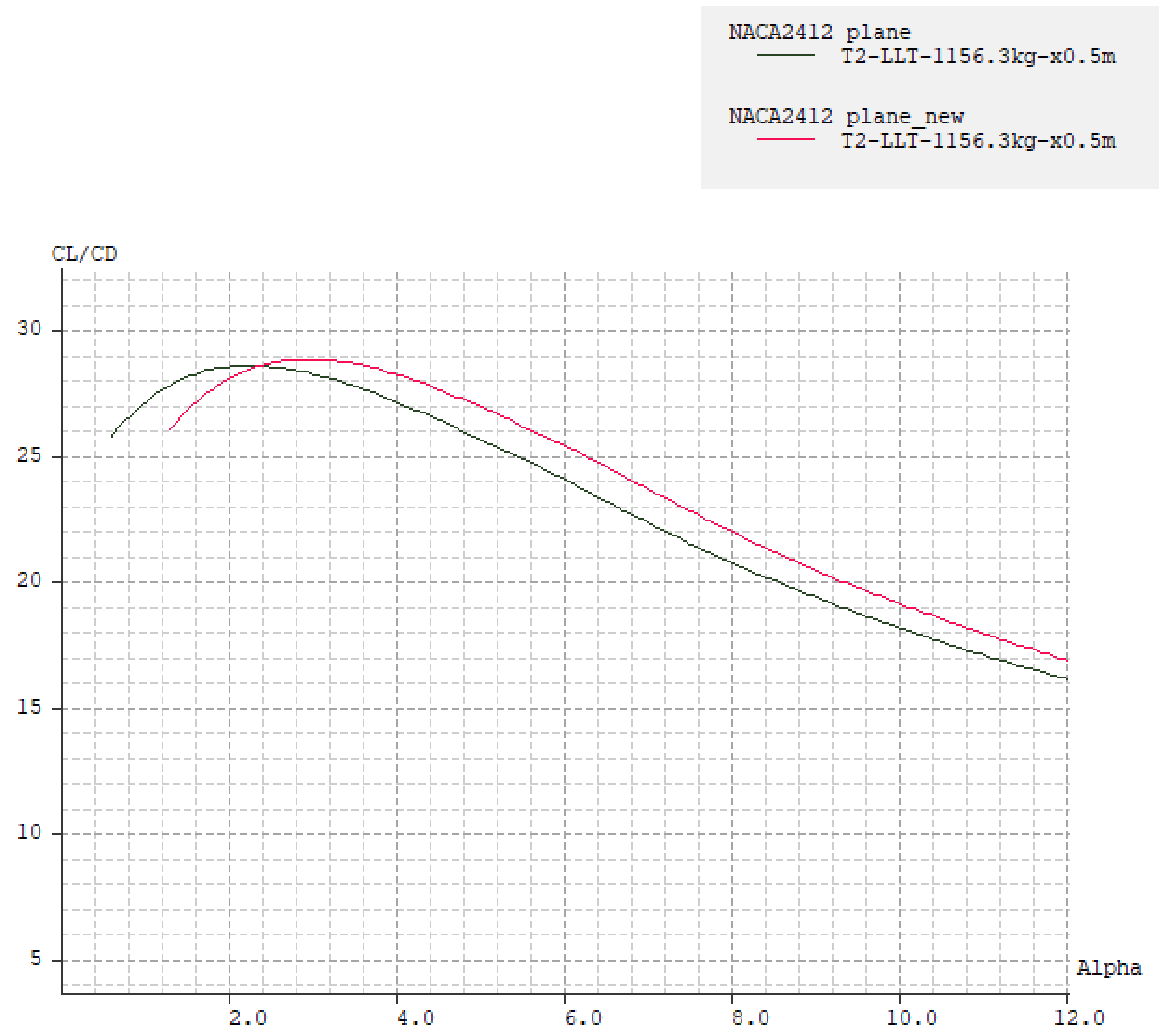
L_tail(@2.3) = M_LE / d = 2678.2 N

With the method we can find the lift @ alpha = 10 degrees this time X_CP = 0.456 m

L_tail = 5383.0 N

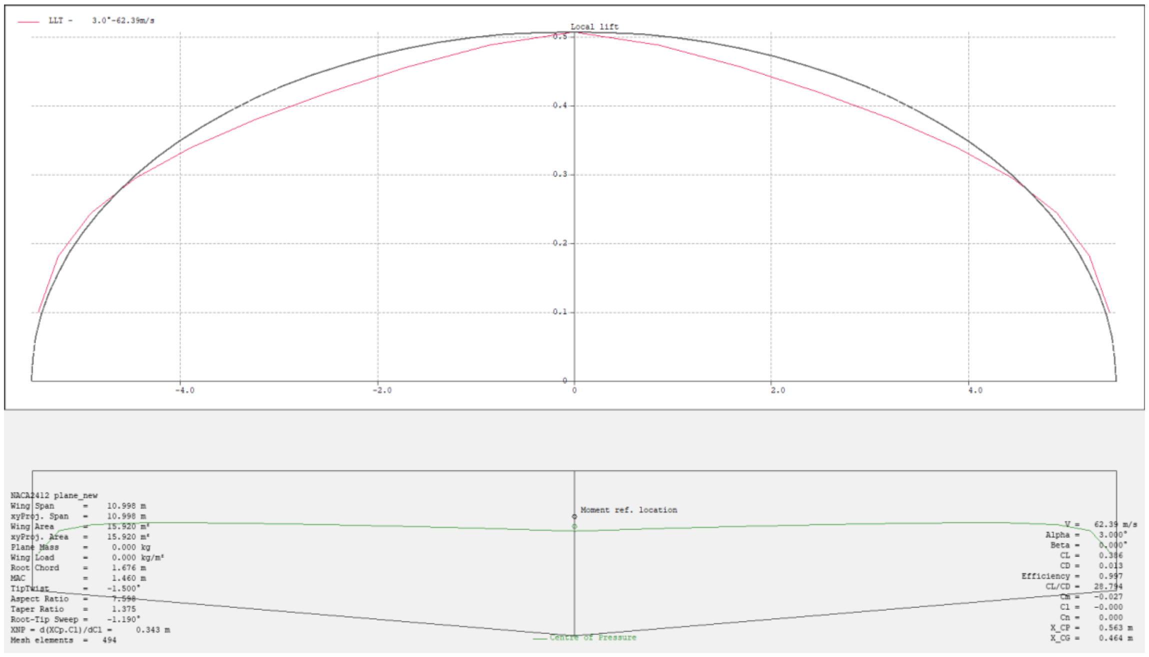
2. (Make sure your project from problem 1 is saved.) Edit the wing and save it with a different name. To do this click on In this problem you will include a small amount of twist (with negative values corresponding to washout of the wingtips). Entering a value of twist introduces a linear distribution of wing twist from one end of a wing section to the next. You should keep the wing root at zero twist angle. Set the wing tip to have -1.5° of twist. (Set the twist at the junctions of the middle and outer wing sections so that the twist is proportional to the spanwise location, i.e., so that the twist varies linearly and smoothly from the wing root to the wing tip.) Once you have created the new wing, run the analysis. How does the maximum lift to drag ratio compare to the wing from problem 1? At what angle of attack does this occur? Does this make sense, given that the wing is twisted? Plot the spanwise load distribution at this angle of attack, with the elliptic curve. Is the load distribution closer or further away from the elliptic distribution? Compute the span efficiency factor e from $C_{Di} = C_L^2/(\pi e AR)$ at this angle of attack and compare it to the value found in problem 1 at the angle of maximum lift to drag ratio. What is the ratio of the induced drag to the total drag at this angle of attack and how does that compare to what was found in problem 1? Because the induced drag coefficient varies with lift coefficient, it is better to compare the wing performance at the same lift coefficient. Use linear interpolation of the data that is available for the twisted wing to determine the angle of attack where the lift coefficient equals the value from the table in problem 1 for the non-twisted wing at its maximum lift to drag ratio. Compute the ratio of the induced drag to the total drag using the interpolated data for the twisted wing at this angle of attack where the CL values match and describe how this value of the ratio compares to the value found in problem 1?

We get the following plots for the lift-to-drag ratio



Discussion:

- The maximum lift-to-drag ratio increases to 28.7943 from 28.534.
- This is a slight increase
- This occurs at the angle of 3 degrees
- This makes sense because the practical means to have a twist is to add absolute angle of attack to the wing which improves the stall angle for the overall wing



```
NACA2412 plane_new
Wing Span        =    10.998 m
xyProj. Span     =    10.998 m
Wing Area        =    15.920 m²
xyProj. Area     =    15.920 m²
Plane Mass       =     0.000 kg
Wing Load        =     0.000 kg/m²
Root Chord       =     1.676 m
MAC              =     1.460 m
TipTwist         =    -1.500°
Aspect Ratio     =     7.598
Taper Ratio      =     1.375
Root-Tip Sweep   =    -1.190°
XNP = d(XCp.Cl)/dCl =      0.343 m
Mesh elements    =   494
```

```
         V =   62.39 m/s
     Alpha =    3.000°
      Beta =    0.000°
        CL =    0.386
        CD =    0.013
Efficiency =    0.997
     CL/CD =   28.794
        Cm =   -0.027
        Cl =   -0.000
        Cn =    0.000
      X_CP =    0.563 m
      X_CG =    0.464 m
```

Discussion:

- At the tip the distribution is closer to the elliptical load distribution which is different from the analysis in problem 1
- The parts in the proximity of the root is rather smaller than the elliptical load distribution which is also different from the previous analysis

Since from the OpPoint >> properties we know at alpha that ICD = 0.00627 and CL = 0.38615

Thus

$$e = 1.6035$$

Discussion:

- This is substantially larger than the previous analysis where we did not have a twist at the angle of attack for the maximum lift-to-drag ratio

$$ICD/CD = 0.00627/0.01341 = 0.4676$$

Discussion:

- This is not that different from the previous analysis, which indicates that both the total and induced drags were reduced with the effect of the twist

To find the angle of attack corresponding to CL = 0.3830, we obtain from interpolation

Alpha = 2.9643

Next for this interpolated angle we will find the ICD and CD for this twisted wing with interpolation

ICD = 0.0084

CD = 0.0156

Thus,

ICD/CD = 0.5347

Discussion:

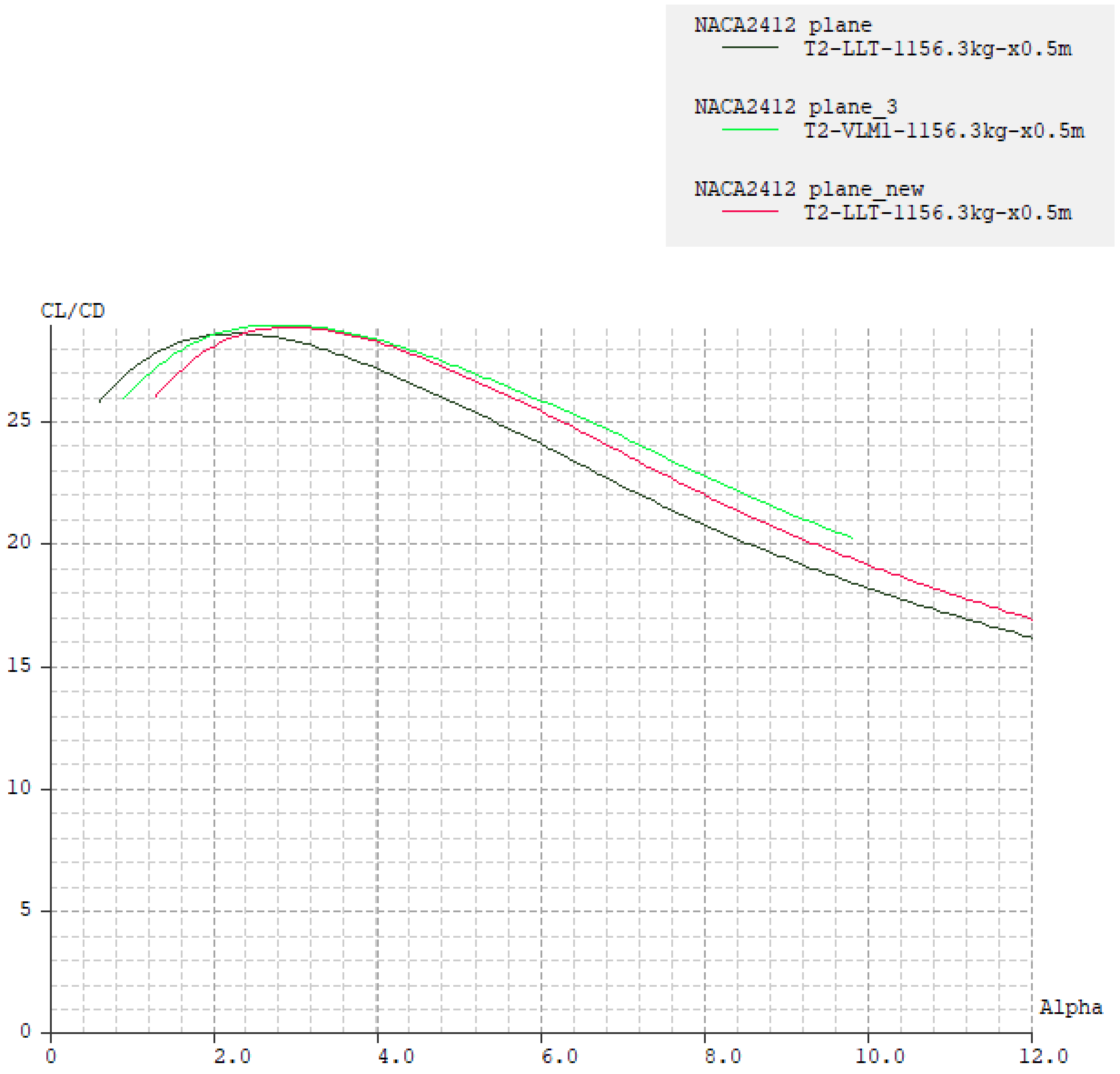
- Compared to the value in problem 1 which is 0.4786 we can see that the for this analysis the ratio is larger and the induced drag accounting for the total drag is overall larger. This is understandable because the angle of attack at the corresponding CL increases with the twist due to the additional absolute angle of attack.

3. Create a new wing without twist but with rearward wing sweep. Do this by defining the offset values of the wing sections so that the leading edge is swept back by 30 ∘ . Since LLT does not account for wing sweep, choose to run the analysis with the vortex lattice method (use VLM1). Do the same tasks and answer the same questions as in problem 2, and discuss the differences rearward wing sweep makes on the results.

Set the offset as Cr – Ct = 0.457 m

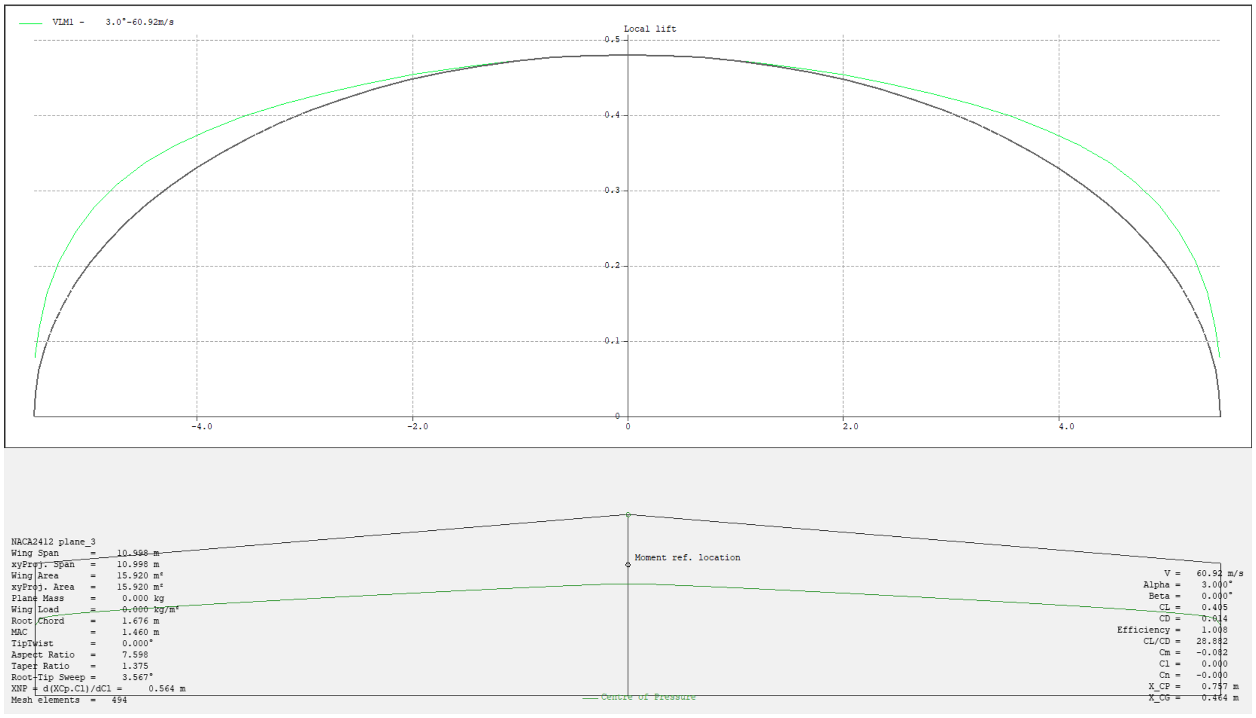
Then do the analysis

We get the following plots for the lift-to-drag ratio



Discussion:

- The maximum lift-to-drag ratio becomes 28.903464
- This occurs at the angle of attack of 2.8 degrees
- This makes sense since reverse swept wings cause the wing to flex which adds more angle of attack



```
NACA2412 plane_3
Wing Span       =    10.998 m
xyProj. Span    =    10.998 m
Wing Area       =    15.920 m²
xyProj. Area    =    15.920 m²
Plane Mass      =     0.000 kg
Wing Load       =     0.000 kg/m²
Root Chord      =     1.676 m
MAC             =     1.460 m
TipTwist        =     0.000°
Aspect Ratio    =     7.598
Taper Ratio     =     1.375
Root-Tip Sweep  =     3.567°
XNP = d(XCp.Cl)/dCl =     0.564 m
Mesh elements   =   494
```

```
         V =   60.92 m/s
     Alpha =    3.000°
      Beta =    0.000°
        CL =    0.405
        CD =    0.014
Efficiency =    1.008
     CL/CD =   28.882
        Cm =   -0.082
        Cl =    0.000
        Cn =   -0.000
      X_CP =    0.757 m
      X_CG =    0.464 m
```

Discussion:

- For this reversely swept wing the load distribution is constantly larger than the elliptical load distribution. This is due to the reaction force to the upwash occurring due to reversely swept wings

Since from the OpPoint >> properties we know at alpha that ICD = 0.00682 and CL = 0.40511

Thus

$$e = 1.6625$$

Discussion:

- The span efficiency is higher than both problem 1 and problem 2. This means that the lift coefficient is higher compared to the two previous problems because of the upwash

$$ICD/CD = 0.00682/0.01403 = 0.4676$$

Discussion:

- Interestingly this is the same result of problem 2

To find the angle of attack corresponding to CL = 0.3830, we obtain from interpolation

$$Alpha = 2.7271$$

Next for this interpolated angle we will find the ICD and CD for this twisted wing with interpolation

$$ICD = 0.0061$$

$$CD = 0.0133$$

Thus,

$$ICD/CD = 0.4603$$

Discussion:

- Compared to the value in problem 1 which is 0.4786 we can see that the values of ICD and CD barely changed for this analysis. From this we can deduce that the direction of the swept applied to the wing does not have that much effect on the drag despite the fact that lift coefficient is influenced greatly.

APPENDIX

AAE 334 HW6 MATLAB CODE

```
clear all; close all; clc;
fdir = 'C:\Users\Tomo\Desktop\studies\2020-Spring\AAE334\MATLAB\outputs\HW6';
set(groot, 'defaulttextinterpreter',"latex");
set(groot, 'defaultAxesTickLabelInterpreter',"latex");
set(groot, 'defaultLegendInterpreter',"latex");
```

1 - setup

```
% Define plane dimensions
b = inch2m(36*12+1);  % [m]
c_r = inch2m(66);  % [m]
b_cs = inch2m(16*12+9);  % [m]
b_os = b - b_cs;  % [m]
c_tip = inch2m(4*12);  % [m]
u_a = knots2m_s(124);  % [m/s]
W = lbf2newtons(2550);  % [N]
m = W/9.81
h_alt = inch2m(8500*12);  % [m]
gamma = 1.4;
R = 287.95;
% Obtaining the atmospheric conditions at the given altitude
[rho_alt,a_alt,T_alt,P_alt,nu_alt,h,sigma] = atmos(h_alt,'units','SI');
% Calculating the viscosity at the given altitude using Sutherland's
% formula
myu = sutherland_visc_calc(T_alt,"air")
nu = myu/rho_alt
% Reynolds number range
Re_range1 = 1.0:0.5:7.5;
Re_legendCell = cellstr(num2str(Re_range1', 'Re=%-f'));
```

1 - a

```
% wing area
S1 = c_r*b_cs;
S2 = (c_r + c_tip)*b_os;
S = S1 + S2;
% wing aspect ratio
AR = b^2/S;
% mach number
Ma = u_a/sqrt(gamma*R*T_alt);
% lift coefficient
C_L = calc_lift_coeff(W,rho_alt,u_a,S);
```

```
% Reynolds number for the wing root section
Re_root = calc_reynolds_number(rho_alt,u_a,c_r,myu);
% Reynolds number for the wing tip section
Re_tip = calc_reynolds_number(rho_alt,u_a,c_tip,myu);
```

1 - b

```
% Loading data
cl_alpha_1b = readmatrix('inputs\hw6\1b_cl_alpha.csv');
cd_alpha_1b = readmatrix('inputs\hw6\1b_cd_alpha.csv');
clcd_alpha_1b = readmatrix("inputs\hw6\1b_clcd_alpha.csv");


[alpha1_b1, cl_b1] = read_batch_analysis_data(cl_alpha_1b);
[alpha2_b1, cd_b1] = read_batch_analysis_data(cd_alpha_1b);
[alpha3_b1, clcd_b1] = read_batch_analysis_data(clcd_alpha_1b);
```

```
% Plotting
% cl coeffs
fig1 = figure('Renderer',"painters");
    hold on
    plot_array(alpha1_b1,cl_b1)
    hold off
    title('$C_L$ vs $\alpha$ P1(a) - By: Tomoki Koike')
    xlabel('$\alpha$ [deg]')
    ylabel('$C_L$')
    legend(Re_legendCell, "Location","northwest","FontSize",8)
    grid on; grid minor; box on;
saveas(fig1, fullfile(fdir, 'Cl_vs_alpha_1b.png'));
```

```
% cd coeffs
fig2 = figure('Renderer',"painters");
    hold on
    plot_array(alpha2_b1,cd_b1)
    hold off
    title('$C_D$ vs $\alpha$ P1(a) - By: Tomoki Koike')
    xlabel('$\alpha$ [deg]')
    ylabel('$C_D$')
    legend(Re_legendCell, "Location","best","FontSize",8)
    grid on; grid minor; box on;
saveas(fig2, fullfile(fdir, 'Cd_vs_alpha_1b.png'));
```

```
% cd coeffs
fig3 = figure('Renderer',"painters");
```

```
    hold on
    plot_array(alpha3_b1,clcd_b1)
    hold off
    title('$C_L$/$C_D$ vs $\alpha$ P1(a) - By: Tomoki Koike')
    xlabel('$\alpha$ [deg]')
    ylabel('$C_L$/$C_D$')
    legend(Re_legendCell, "Location","best","FontSize",8)
    grid on; grid minor; box on;
saveas(fig3, fullfile(fdir, 'ClCd_vs_alpha_1b.png'));
```

```
% Computations
max_clcd_b1 = max(max(clcd_b1));
```

1 - d

```
% Loading data
cl_alpha_1d = readmatrix('inputs\hw6\1d_cl_alpha.csv');
cl_cd_1d = readmatrix('inputs\hw6\1d_cl_cd.csv');
clcd_alpha_1d = readmatrix("inputs\hw6\1d_clcd_alpha.csv");
alpha1_1d = cl_alpha_1d(:,1);
cl1_1d = cl_alpha_1d(:,2);
cd_1d = cl_cd_1d(:,1);
cl2_1d = cl_cd_1d(:,2);
alpha2_1d = clcd_alpha_1d(:,1);
clcd_1d = clcd_alpha_1d(:,2);
```

```
% Plotting
fig4 = figure("Renderer","painters");
    plot(alpha1_1d, cl1_1d)
    title('$C_L$ vs $\alpha$ P1(d) - By: Tomoki Koike')
    xlabel('$\alpha$ [deg]')
    ylabel('$C_L$')
    grid on; grid minor; box on
saveas(fig4, fullfile(fdir, 'cl_alpha_1d.png'));
```

```
fig5 = figure("Renderer","painters");
    plot(cd_1d, cl2_1d)
    title('$C_L$ vs $C_d$ P1(d) - By: Tomoki Koike')
    xlabel('$C_D$')
    ylabel('$C_L$')
    grid on; grid minor; box on
saveas(fig5, fullfile(fdir, 'cl_cd_1d.png'));
```

```
fig6 = figure("Renderer","painters");
    plot(alpha2_1d, clcd_1d)
    title('$C_L$/$C_d$ vs $\alpha$ P1(d) - By: Tomoki Koike')
    xlabel('$\alpha$ [deg]')
    ylabel('$C_L$/$C_D$')
    grid on; grid minor; box on
saveas(fig6, fullfile(fdir, 'clcd_alpha_1d.png'));
```

```
% Computations
max_clcd_1d = max(clcd_1d);
clcd_pdiff_1b_1d = max_clcd_b1 - max_clcd_1d;


% Double the drag coefficient to be more realistic
clcd_1d_real = cl1_1d./(2*cd_1d);
fig7 = figure("Renderer","painters");
    plot(alpha2_1d, clcd_1d_real)
    title('$C_L$/$C_d$ vs $\alpha$ with Double Drag P1(d) - By: Tomoki Koike')
    xlabel('$\alpha$ [deg]')
    ylabel('$C_L$/$C_D$')
    grid on; grid minor; box on
saveas(fig7, fullfile(fdir, 'clcd_alpha_1d_real.png'));


[max_clcd_1d_real, idx] = max(clcd_1d_real)
max_alpha_1d = alpha1_1d(idx)
cl_at_max_alpha = cl1_1d(idx)
```

```
slope_1d = (cl1_1d(idx+1)-cl1_1d(idx-1))/deg2rad(alpha1_1d(idx+1)-alpha1_1d(idx-
1))
th_slope_1d = 6.588/(1 + 6.588/pi/AR)
pdiff = (slope_1d - th_slope_1d) / th_slope_1d * 100
```

1 - e

```
% Creating table
data_array = [2.30 0.01342 0.00714 0.00629 -0.03504 0.38296; 10.00 0.05710 0.0108
0.04627 -0.00386 1.03421];
data_T = array2table(data_array, "VariableNames",{'Alpha', 'CD', 'VCD', 'ICD',
'Cm', 'CL'});
```

```
% e factor
e = (data_T.CL).^2/pi/AR./data_T.CD
```

```
data_T = addvars(data_T, e, 'after', "CL")
% total drag and induced drag ratio
Dtot_Di_ratio = data_T.ICD./data_T.CD


% Calculate the drag
Vinf = [62.652; 38.125];
D = 0.5 * rho_alt * u_a^2 .* data_T.CD * S;
data_T = addvars(data_T, D, 'after', 'e')
data_T = addvars(data_T, Vinf, 'before', 'D')


bourget = data_T.CL./data_T.CD.*data_T.Vinf
```

1 - f

```
data_1f = readmatrix("inputs\hw6\1f_lcl_lift.csv");
b_llt = data_1f(:,1);
llt = data_1f(:,2);
b_e_1f = data_1f(:,3);
llt_e_1f = data_1f(:,4);


% Plotting
fig8 = figure("Renderer","painters");
    plot(b_llt, llt)
    title('Spanwise Local Lift Distribution $\alpha$=2.3 P1(f) - By: Tomoki
Koike')
    xlabel('Span')
    ylabel('local lift Coeff')
    hold on
    plot(b_e_1f, llt_e_1f, '.-')
    hold off
    legend('LLT', 'Elliptical Load')
    grid on; grid minor; box on
saveas(fig8, fullfile(fdir, 'llt_elliptical.png'));
```

1 - g

```
% Calculate the area under LLT
Ltot_1g = trapz(rmmissing(b_llt), rmmissing(llt))
Ltot_1g = Ltot_1g * 0.5 * rho_alt * u_a^2 * S;
X_CP = 0.593;
M_LE = X_CP*Ltot_1g/b
d = inch2m(15*12+6) - X_CP
L_tail = M_LE / d
```

```
data_1f2 = readmatrix("inputs\hw6\1f_lcl_lift10.csv");
b_llt2 = data_1f2(:,1);
llt2 = data_1f2(:,2);
b_e_1f2 = data_1f2(:,3);
llt_e_1f2 = data_1f2(:,4);


% Plotting
fig9 = figure("Renderer","painters");
    plot(b_llt2, llt2)
    title('Spanwise Local Lift Distribution $\alpha$=10 P1(f) - By: Tomoki
Koike')
    xlabel('Span')
    ylabel('local lift Coeff')
    hold on
    plot(b_e_1f2, llt_e_1f2, '.-')
    hold off
    legend('LLT', 'Elliptical Load')
    grid on; grid minor; box on
saveas(fig9, fullfile(fdir, 'llt_elliptical10.png'));


% Calculate the area under LLT
Ltot_1g2 = trapz(rmmissing(b_llt2), rmmissing(llt2))
Ltot_1g2 = Ltot_1g2 * 0.5 * rho_alt * u_a^2 * S;
X_CP2 = 0.456;
M_LE2 = X_CP2*Ltot_1g2/b;
d2 = inch2m(15*12+6) - X_CP2;
L_tail = M_LE2 / d2
```

2

```
ICD2 = 0.00627;
CL2 = 0.38615;
e = CL2^2/pi/ICD2/AR
ICDrat = 0.00627/0.01341
```

```
data2 = readmatrix("inputs\hw6\CL3.csv");
alpha2 = rmmissing(data2(:,1));
cl2 = rmmissing(data2(:,2));
alpha_interp = interp1(cl2, alpha2, 0.3830);


data21 = readmatrix("inputs\hw6\ICD2.csv");
data22 = readmatrix("inputs\hw6\CD2.csv");
a21 = rmmissing(data21(:,1));
ICD21 = rmmissing(data21(:,2));
```

```
a22 = rmmissing(data22(:,1));
CD22 = rmmissing(data22(:,2));


ICD_interp = interp1(a21, ICD21, alpha_interp)
CD_interp = interp1(a22, CD22, alpha_interp)
ratio_interp = ICD_interp/CD_interp
```

3

```
ICD3 = 0.00682;
CL3 = 0.40511;
e = CL3^2/pi/ICD3/AR
ICDrat3 = 0.00627/0.01341
```

```
data3 = readmatrix("inputs\hw6\CL3.csv");
alpha3 = rmmissing(data3(:,1));
cl3 = rmmissing(data3(:,2));
alpha_interp3 = interp1(cl3, alpha3, 0.3830)


data31 = readmatrix("inputs\hw6\ICD3.csv");
data32 = readmatrix("inputs\hw6\CD3.csv");
a31 = rmmissing(data31(:,1));
ICD31 = rmmissing(data31(:,2));
a32 = rmmissing(data32(:,1));
CD32 = rmmissing(data32(:,2));


ICD_interp3 = interp1(a31, ICD31, alpha_interp3)
CD_interp3 = interp1(a32, CD32, alpha_interp3)
ratio_interp3 = ICD_interp3/CD_interp3
```

FUNCTIONS

```
function C_L = calc_lift_coeff(L,rho,v,S)
    C_L = 2*L/rho/v^2/S;
end


function Re = calc_reynolds_number(rho, v, c, myu)
    Re = rho * v * c / myu;
end


function [t, y] = read_batch_analysis_data(data)
    t = [];
    y = [];
    ct = 1;
    for i = 1:3:40
        t(:,ct) = data(:,i);
        y(:,ct) = data(:,i+1);
        ct = ct + 1;
    end
end


function plot_array(t_array, y_array)
    for i = 1:size(t_array, 2)
        plot(t_array(:,i), y_array(:,i), '-')
    end
end
```

```
function f = lbf2newtons(w)
    f = w * 4.44822;
end

function u = knots2m_s(v)
    u = v * 0.514444;
end

function myu = sutherland_visc_calc(T, gas)
    if gas == "air"
        C = 120;
    elseif gas == "NH3"
        C = 370;
    elseif gas == "CO2"
        C = 240;
    elseif gas == "CO"
        C = 118;
    elseif gas == "H2"
        C = 72;
    elseif gas == "N2"
        C = 111;
    elseif gas == "O2"
        C = 127;
    elseif gas == "SO2"
        C = 416;
    else
        disp("Error. Choose an appropriate gas.")
    end
    % Calculation
    T_ref = 288.15;  % [K]
    myu_ref = 1.789 * 10^(-5);  % [Pa-s]
    myu = myu_ref * (T / T_ref)^(1.5) * (T_ref + C) / (T + C);
end
```

FILE EXCHANGE

```
function varargout = atmos(h,varargin)
%  ATMOS  Find gas properties in the 1976 Standard Atmosphere.
%   [rho,a,T,P,nu,z,sigma] = ATMOS(h,varargin)
%
%   ATMOS by itself gives atmospheric properties at sea level on a standard day.
%
%   ATMOS(h) returns the properties of the 1976 Standard Atmosphere at
%   geopotential altitude h, where h is a scalar, vector, matrix, or ND array.
%
%   The input h can be followed by parameter/value pairs for further control of
%   ATMOS. Possible parameters are:
%     tOffset      - Returns properties when the temperature is tOffset degrees
%                    above or below standand conditions. h and tOffset must be
%                    the same size or else one must be a scalar. Default is no
%                    offset. Note that this is an offset, so when converting
%                    between Celsius and Fahrenheit, use only the scaling factor
%                    (dC/dF = dK/dR = 5/9).
%     tAbsolute    - Similar to tOffest, but an absolute air temperature is
%                    provided (°K or °R) instead of an offset from the standard
%                    temperature. Supersedes tOffset if both are provided.
%     altType      - Specify type of input altitude, either 'geopotential' (h)
%                    or 'geometric' (z). Default altType = 'geopotential'.
%     structOutput - When set, ATMOS produces a single struct output with fields
%                    rho, a, T, P, nu, and either z or h (whichever complements
%                    input altType). Default structOutput = false.
%     units        - String for units of inputs and outpus, either 'SI'
%                    (default) or 'US'. This is ignored if the provided input h
%                    is a DimVar, in which case all outputs are also DimVars and
%                    expected tOffset is either a DimVar or in °C/°K.
%                                Description:         SI:           US:
%                     Input:     --------------       -----         -----
%                       h | z    Altitude or height   m             ft
%                       tOffset  Temp. offset         °C/°K         °F/°R
%                     Output:    --------------       -----         -----
%                       rho      Density              kg/m^3        slug/ft^3
%                       a        Speed of sound       m/s           ft/s
%                       T        Temperature          °K            °R
%                       P        Pressure             Pa            lbf/ft^2
%                       nu       Kinem. viscosity     m^2/s         ft^2/s
%                       z | h    Height or altitude   m             ft
%                       sigma    Density ratio        -             -
%
%   ATMOS returns properties the same size as h and/or tOffset (P does not vary
%   with temperature offset and is always the size of h).
%
%   Example 1: Find atmospheric properties at every 100 m of geometric height
%   for an off-standard atmosphere with temperature offset varying +/- 25°C
%   sinusoidally with a period of 4 km.
%       z = 0:100:86000;
%       [rho,a,T,P,nu,h,sigma] = atmos(z,'tOffset',25*sin(pi*z/2000),...
%                                     'altType','geometric');
%       semilogx(sigma,h/1000)
%       title('Density variation with sinusoidal off-standard atmosphere')
%       xlabel('Density ratio, \sigma'); ylabel('Geopotential altitude (km)')
%
%   Example 2: Create tables of atmospheric properties up to 30,000 ft for a
%   cold (-20°C), standard, and hot (+20°C) day with columns
%   [h(ft) z(ft) rho(slug/ft³) sigma a(ft/s) T(R) P(psf) µ(slug/ft-s) nu(ft²/s)]
%   leveraging n-dimensional array capability.
%       [~,h,dT] = meshgrid(0,-5000:1000:30000,[-20 0 20]);
%       [rho,a,T,P,nu,z,sigma] = atmos(h,'tOffset',dT*9/5,'units','US');
%       t = [h z rho sigma a T P nu.*rho nu];
%       format short e
%       varNames = {'h' 'z' 'rho' 'sigma' 'a' 'T' 'P' 'mu' 'nu'};
%       ColdTable       = array2table(t(:,:,1),'VariableNames',varNames)
%       StandardTable   = array2table(t(:,:,2),'VariableNames',varNames)
%       HotTable        = array2table(t(:,:,3),'VariableNames',varNames)
%
%   Example 3: Use the unit consistency enforced by the DimVar class to find the
%   SI dynamic pressure, Mach number, Reynolds number, and stagnation
%   temperature of an aircraft flying at flight level FL500 (50000 ft) with
%   speed 500 knots and characteristic length of 80 inches.
%       V = 500*u.kts; c = 80*u.in;
%       o = atmos(50*u.kft,'structOutput',true);
%       Dyn_Press = 1/2*o.rho*V^2;
%       M = V/o.a;
%       Re = V*c/o.nu;
%       T0 = o.T*(1+(1.4-1)/2*M^2);
%
%   This model is not recommended for use at altitudes above 86 km geometric
%   height (84852 m / 278386 ft geopotential) but will attempt to extrapolate
%   above 86 km (with a lapse rate of 0°/km) and below 0.
%
%   See also ATMOSISA, ATMOSNONSTD, TROPOS,
%     DENSITYALT - http://www.mathworks.com/matlabcentral/fileexchange/39325,
%     UNITS      - http://www.mathworks.com/matlabcentral/fileexchange/38977.
%
%   [rho,a,T,P,nu,z,sigma] = ATMOS(h,varargin)


%   Copyright 2015 Sky Sartorius
%   www.mathworks.com/matlabcentral/fileexchange/authors/101715
%
%   References: ESDU 77022; www.pdas.com/atmos.html


%% User-customizable defaults:
defaultUnits = 'SI'; % Alternate: 'US'
```

```matlab
defaultStructOutput = false;


%% Parse inputs:
if nargin == 0
    h = 0;
end
if nargin <= 1 && ~nnz(h)
    % Quick return of sea level conditions.
    rho = 1.225;
    a = 340.2940;
    temp = 288.15;
    press = 101325;
    kvisc = 1.4607186e-05;
    ZorH = 0;
    if isa(h,'DimVar')
        rho = rho*u.kg/(u.m^3);
        if nargout == 1
            varargout = {rho};
            return
        end
        a = a*u.m/u.s;
        temp = temp*u.K;
        press = press*u.Pa;
        kvisc = kvisc*u.m^2/u.s;
        ZorH = ZorH*u.m;
    end


    varargout = {rho,a,temp,press,kvisc,ZorH,1};
    return
end

% validateattributes(h,{'DimVar' 'numeric'},{'finite' 'real'});


p = inputParser;
addParameter(p,'tOffset',0);
addParameter(p,'tAbsolute',[]);
addParameter(p,'units',defaultUnits);
addParameter(p,'altType','geopotential');
addParameter(p,'structOutput',defaultStructOutput);
parse(p,varargin{:});


tOffset = p.Results.tOffset;
tAbsolute = p.Results.tAbsolute;


if strcmpi(p.Results.units,'SI')
```

```
    convertUnits = false;
elseif strcmpi(p.Results.units,'US')
    convertUnits = true;
    % Flag if I need to convert to/from SI.
else
    error('Invalid units. Expected: ''SI'' or ''US''.')
end


if strcmpi(p.Results.altType,'geopotential')
    geomFlag = false;
elseif strcmpi(p.Results.altType,'geometric')
    geomFlag = true;
    % Flag specifying z provided as input.
else
    error('Invalid altType. Expected: ''geopotential'' or ''geometric''.')
end


structOutput = p.Results.structOutput;


%% Deal with different input types:
dimVarOut = false;
if isa(h,'DimVar')
    h = h/u.m;
    dimVarOut = true;
    convertUnits = false; % Trumps specified units.
end
if isa(tOffset,'DimVar')
    tOffset = tOffset/u.K;
    % It is allowed to mix DimVar h_in and double tOffset (or reverse).
end
if isa(tAbsolute,'DimVar')
    tAbsolute = tAbsolute/u.K;
end


if convertUnits
    h = h * 0.3048;
    tOffset   = tOffset   * 5/9;
    tAbsolute = tAbsolute * 5/9;
end




%% Constants, etc.:
```

```
%  Lapse rate Base Temp       Base Geop. Alt    Base Pressure
%   Ki (°C/m) Ti (°K)         Hi (m)            P (Pa)
D =[-0.0065   288.15          0                 101325           % Troposphere
    0         216.65          11000             22632.04059693474 % Tropopause
    0.001     216.65          20000             5474.877660660026 % Stratosph. 1
    0.0028    228.65          32000             868.0158377493657 % Stratosph. 2
    0         270.65          47000             110.9057845539146 % Stratopause
    -0.0028   270.65          51000             66.938535373039073% Mesosphere 1
    -0.002    214.65          71000             3.956392754582863 % Mesosphere 2
    0         186.94590831019 84852.04584490575 .373377242877530];% Mesopause


% Constants:
rho0 = 1.225;   % Sea level density, kg/m^3
gamma = 1.4;
g0 = 9.80665;   %m/sec^2
RE = 6356766;   %Radius of the Earth, m
Bs = 1.458e-6;  %N-s/m2 K1/2
S = 110.4;      %K


K = D(:,1);  %°K/m
T = D(:,2);  %°K
H = D(:,3);  %m
P = D(:,4);  %Pa


R = P(1)/T(1)/rho0; %N-m/kg-K
% Ref:
%   287.05287 N-m/kg-K: value from ESDU 77022
%   287.0531 N-m/kg-K:  value used by MATLAB aerospace toolbox ATMOSISA




%% Convert from geometric altitude to geopotental altitude, if necessary.
if geomFlag
    hGeop = (RE*h) ./ (RE + h);
else
    hGeop = h;
end


%% Calculate temperature and pressure:
% Pre-allocate.
temp = zeros(size(h));
press = temp;


nSpheres = size(D,1);
```

```
for i = 1:nSpheres
    % Put inputs into the right altitude bins:
    if i == 1 % Extrapolate below first defined atmosphere.
        n = hGeop <= H(2);
    elseif i == nSpheres % Capture all above top of defined atmosphere.
        n = hGeop > H(nSpheres);
    else
        n = hGeop <= H(i+1) & hGeop > H(i);
    end


    if K(i) == 0 % No temperature lapse.
        temp(n) = T(i);
        press(n) = P(i) * exp(-g0*(hGeop(n)-H(i))/(T(i)*R));
    else
        TonTi = 1 + K(i)*(hGeop(n) - H(i))/T(i);
        temp(n) = TonTi*T(i);
        press(n) = P(i) * TonTi.^(-g0/(K(i)*R)); % Undefined for K = 0.
    end
end


%% Switch between using standard temp and provided absolute temp.
if isempty(tAbsolute)
    % No absolute temperature provided - use tOffset.
    temp = temp + tOffset;
else
    temp = tAbsolute;
end


%% Populate the rest of the parameters:
rho = press./temp/R;
sigma = rho/rho0;


a = sqrt(gamma * R * temp);
kvisc = (Bs * temp.^1.5 ./ (temp + S)) ./ rho; %m2/s
if geomFlag % Geometric in, ZorH is geopotential altitude (H)
    ZorH = hGeop;
else % Geop in, find Z
    ZorH = RE*hGeop./(RE-hGeop);
end


%% Process outputs:
if dimVarOut
    rho = rho*u.kg/(u.m^3);
    a = a*u.m/u.s;
    temp = temp*u.K;
    press = press*u.Pa;
    kvisc = kvisc*u.m^2/u.s;
    ZorH = ZorH*u.m;
elseif convertUnits
    rho = rho / 515.3788;
    a = a / 0.3048;
    temp = temp * 1.8;
    press = press / 47.88026;
    kvisc = kvisc / 0.09290304;
    ZorH = ZorH / 0.3048;
end


varargout = {rho,a,temp,press,kvisc,ZorH,sigma};


if structOutput
    if geomFlag
        ZorHname = 'h';
    else
        ZorHname = 'z';
    end
    names = {'rho' 'a' 'T' 'P' 'nu' ZorHname 'sigma'};
    varargout = {cell2struct(varargout,names,2)};


end


end
```