Foundations of Aircraft Design Lab

Maklen Estrada, AJ Lauffer, Fabrizio Roberts, Linus Schmitz, Nate Sanchez, and Nathan Tonella
The University of Colorado Boulder, Boulder, Colorado, United States of America
January 31, 2021

I. Lift Curve Comparison (C_L vs. α)

A. Comparison of Data

For both the Boeing 747 (Fig 1.) and the Tempest (Fig 2.) we see that the slope for the 2-D airfoil has a higher Coefficient of Lift for its respective angle of attack. Additionally, the 3-D finite wing is completely linear for both because equation 1 and 2, only calculates the linear portion of the coefficient of lift for the 3-D finite wing. Both the Boeing and the Tempest experience similar effects of a decrease in slope when dealing with a 3-D finite wing compared to that of their respective 2-D airfoil slope.

$$a = \frac{d_{C_L}}{d\alpha} = \frac{a_o}{1 + \frac{57.3 \cdot a_o}{\pi \cdot e \cdot AR}} \qquad (1)$$

$$C_L = a \cdot (\alpha - \alpha_{L=0}) \qquad (2)$$

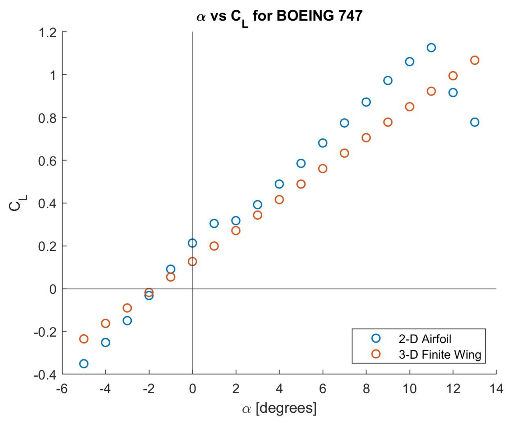


Fig. 1 Boeing Angle of Attack compared to the C_L

B. Explanation of Data

When dealing with a 3-D finite wing, we have a circulation of high-pressure air below the wing and low-pressure air above the wing near the wingtip. This leads to wingtip vertices which reduce the effecting angle of attack and cause downwash. In a 2-D infinite wing, we do not deal with induced drag this, in turn, results in a steeper slope for a 2-D infinite wing and a less steep slope for a 3-D finite wing.

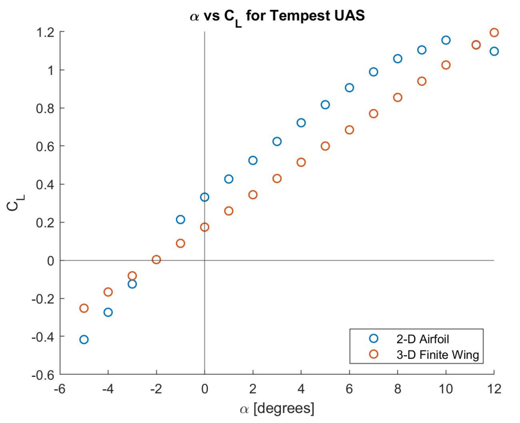


Fig. 2 Tempest Angle of Attack compared to the C_L

II. Drag Polar Comparison (C_D vs. C_L)

A. Comparison of Data

For the Boeing 747-200 it can be seen that the wing drag polar (red circles in Fig. 3) is almost directly on the truth data provided, which is confirmation that our model and methodology for the calculated whole aircraft drag polar is fairly accurate. The difference between the wing drag polar and the whole aircraft drag polar is in line with the knowledge that for a larger aircraft such as the 747 there is considerable additional drag on the aircraft due to the fuselage. The rapid increase in the wing drag polar (red circles in Fig. 3) at roughly 0.9 coefficient of lift can possibly be explained by a stall occurring at that point.

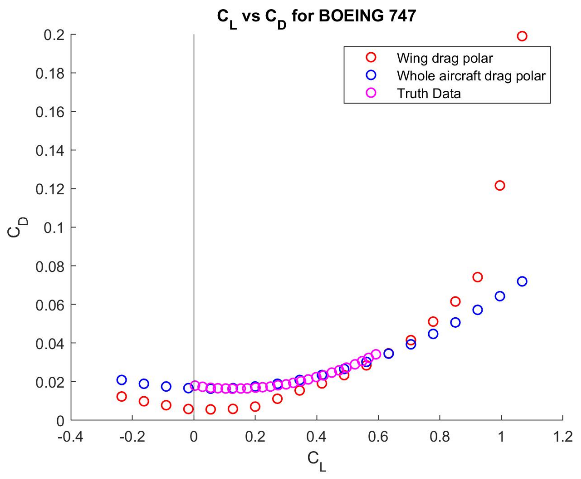


Fig. 3 Boeing C_L compared to C_D.

The discrepancy between our calculated model and the truth data can be explained by our estimations in some values used for our calculations. In particular the wetted area of the 747-200 was taken from a low quality plot (Fig. 4) which while not outdated was not very accurate either. The estimated error on that figure is somewhere in the range of $\pm$ 1000 ft^2. Additionally we made the assumption that the entire wing was only a singular airfoil (the BACJ airfoil) for simplicity sake, when in reality it is two different airfoils at the root of the wing vs the tip of the wing. The two together form a geometric twist which changes the aerodynamic performance of the wing across the full length.

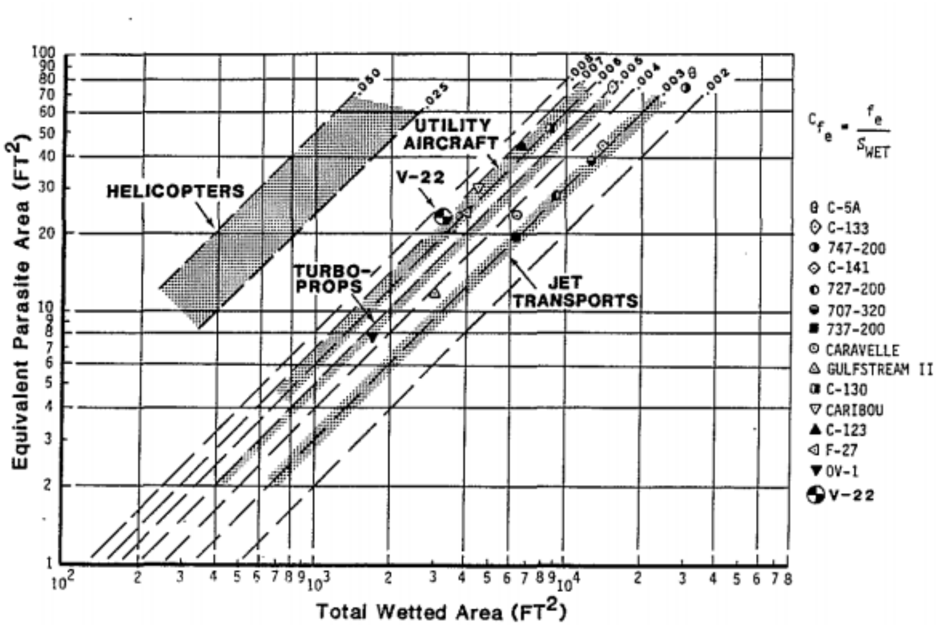


Fig. 4

When examining the different methods for calculating the Oswalds efficiency factor used in equation 3, we determined that all applicable methods gave roughly the same e_o value. As a result we chose not to plot or analyze the other methods of calculating e_o for the 747 analysis.

$$k_1 = \frac{1}{\pi \cdot e_o \cdot AR} \qquad (3)$$

The Tempest drag polar can be calculated with two different methods, each with a different Oswald's efficiency factor. The first method uses equation 4 from U.S. navy research and the second uses equation 5 from research done by Ed Obert. Unlike the Boeing 747-200 drag polar, the calculated drag polar (blue circles in Fig. 5 and Fig. 6) of the Tempest Aircraft did not align with the truth data (pink circles in Fig. 5 and Fig. 6). In Fig. 5 and Fig. 6, the modeled aircraft drag polar is much less than the true data, and closely aligns with the 3-D wing drag polar. Since the model produced accurate values for the 747, there is an error in the calculations and assumptions, not in the model itself.

$$e_o = 1.78(1 - 0.045AR^{.68}) - 0.64 \qquad (4)$$

$$e_o = \frac{1}{1.05 + 0.007\pi A} \qquad (5)$$

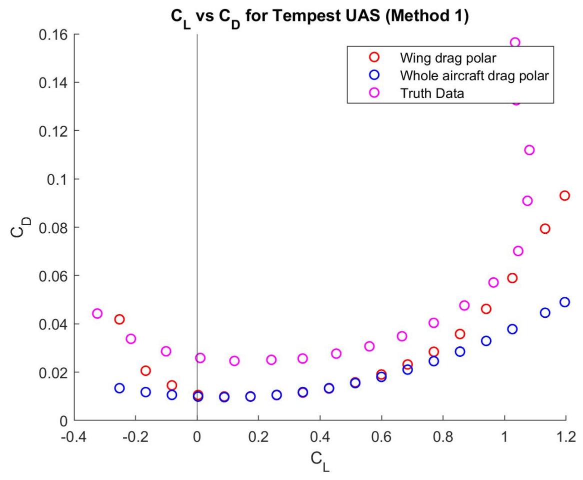


Fig. 5 Tempest C_L compared to C_D using Method 1.

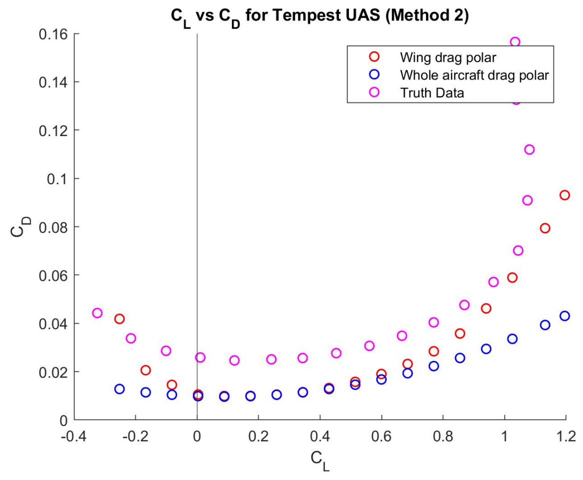


Fig. 6 Tempest C_L compared to C_D using Method 2.

B. Errors associated with Data

The Tempest analysis had more sources of uncertainty. One source of error are the calculations to find the value of $C_{min,d}$. $C_{min,d}$ relates to skin friction and the ratio of the wetted area and planform area. In particular, the calculation of the wetted area ignored key aspects of the aircraft geometry. The fuselage is assumed to be perfectly symmetrical when it is not, and the surface area analysis ignores the tapering of the aircraft towards the tail. Additionally, the surface area of the wing requires time consuming, complicated calculation using airfoil data, and so the wetted area analysis assumes that the wing surface area is two times the wing planform area. Therefore, the approximate wetted area is likely an underestimate of the true wetted area, and so the estimate for $C_{min,d}$ is less than the true value of $C_{min,d}$. Furthermore, there is some error in measuring the dimensions of the aircraft that were not given to us. Unknown dimensions could be determined using a scale drawing with some uncertainty. Each unknown dimension has an error of approximately

0.015 m. An error analysis of the wetted area of the tail section showed the uncertainty of the tail wetted area is 0.012 m^2, while the fractional error is 0.07 or 7. The analysis shows that the resultant error in Swet for the tail is small but not insignificant. The error of the entire wetted area is uncertain since the CAD software used to find the surface area did not perform any uncertainty analysis, but the error analysis of the tail shows that the uncertainty in the wetted area could be a source of the discrepancy between the model and the truth data.

III. Performance Comparisons

A. Performance Comparison

Glide Range	Calculated Velocity (m/s)	True Velocity (m/s)
Tempest	15.64	16.81
Boeing 747	79.45	86.59

Powered Range	Calculated Velocity (m/s)	True Velocity (m/s)
Tempest	15.64	16.81
Boeing 747	101.46	108.64

Powered Endurance	Calculated Velocity (m/s)	True Velocity (m/s)
Tempest	12.55	14.72
Boeing 747	79.45	86.59

B. Discussion

As can be seen above, the calculated values are always slightly lower than the true velocities, this is partly due to the error associated with the calculation of the drag coefficient and Oswald's efficiency factor discussed earlier. It is also worth noting that the velocities are the same depending on which lift/drag ratio was used. For example, the powered endurance and glide range of the Boeing 747 are the same due to both requiring the maximized $\frac{L}{D}$ ratio.

IV. Summary

A. Modifications and Analysis

From comparing the whole aircraft drag polar to the truth data, it can be seen that the aerodynamic models used were much more accurate when modeling the larger Boeing 747 aircraft than the Tempest. When designing a smaller glider in a future lab, it is clear that modifications should be made to the models in order to obtain more accurate results. One key modification that should be made is utilizing an alternative approach for calculating Oswald's efficiency factor which will have a significant effect on the drag polar due to lift. The approach provided in the lab document seemed fairly accurate for larger scale aircraft, such as the Boeing, for which empirical data had been collected, although for the purposes of this lab, the geometric twist of the Boeing airfoil was neglected. This geometric twist of the airfoil can have some contribution to the drag polar and is considered with other methods for calculating the efficiency factor. Because the results seemed fairly accurate when compared to the truth data, it can be inferred that neglecting the geometric twist did not create too large of a negative impact. It was necessary, however, to research an alternative method when calculating the efficiency factor in order to correct for the differing aircraft geometry of the Tempest. The method that was obtained took into account the viscous and inviscid effects on the drag polar of an aircraft. The viscous portion is caused by certain wing parameters such as camber, sweep, and thickness ratio that will affect the skin friction. The inviscid portion is caused by vortex drag.This method can account for the unique geometry of an aircraft by taking a ratio between the diameter of the fuselage and the wingspan. This approach would be much more useful with the design of a glider that will have unique geometric properties when compared to larger civilian aircraft. This alternative method provides given constants obtained experimentally to the terms governed by the inviscid and viscous effects which made the calculation of the efficiency factor simpler, but again failed to consider the geometries of the glider.

Furthermore, another modification that could be made to the aerodynamic models would be to simply treat the aircraft as a series of flat plates when calculating the skin friction drag coefficient. This method proved to be ineffective when modeling the drag polar for the Boeing and the Tempest as there are many factors that could contribute to parasite drag that would be neglected when using this approach. The fuselage for the Boeing, for example, could account for a large portion of the total skin friction drag. It could be more plausible, however, to apply this approach when modeling the effect of skin friction on a smaller aircraft such as a glider.

V. References

Books

[1]John D. Anderson Jr. Introduction to Flight Eighth Edition. McGraw-Hill Education, 2016.

Class Resources

[2] ASEN 2004 Aero Lab:Foundations of Aircraft Design Lab Manuel

[3] John Mah's ASEN 2004 Lectures

Reports, Theses, and Individual Papers

[4]M. Nita, D. Scholz, 2012. Estimating the Oswald Factor From Basic Aircraft Geometrical Parameters. Hamburg University of Applied Sciences Aero.

[5]Rosenstein, H. and Clark, R., 1986. [online] Dspace-erf.nlr.nl. Available at: ¡https://dspace-erf.nlr.nl/xmlui/bitstream/handle/20.500.11881/2975/ERF

Table of Contents

```
% ASEN 2004 Lab 1 Group 013-09
% Maklen Estrada, AJ Lauffer, Fabrizio Roberts, Nate Sanchez,
% Linus Schmitz, and Nathan Tonella

% January 30, 2021

% Finding the coefficient of lift for a 3-D finite wing and comparing
 it to
% 2-D coefficient of lift for an airfoil.
% Finding whole air craft and wing drag polar and comparing both
 values to
% given truth data
% Calculating and comparing various performance values between our
% aerodynamic model and the truth data provided


close all
clear
clc
```

2-D Airfoil Data

Note: all data came from the excel file provided on the ASEN 2004 Canvas page and was separated into their own files

```
% read in the airfoil data for both models
Temp = xlsread('Tempest_MH32_Airfoil_Data.xlsx'); % Temp = Tempest UAS
BO = xlsread('BOEING_BACJ_Airfoil_Data.xlsx'); % BO = BOEING 747

% extract coefficient of lift data (2-D)
CLTemp2D = Temp(:,2);
CLBO2D = BO(:,2);

% extract coeifficeint of drag data (2-D)
Cd_Temp = Temp(:,3);
Cd_BO = BO(:,3);
```

```
% extract angle of attack for each data point
AoATemp = Temp(:,1);
AoABO = BO(:,1);
```

Hard Coded Values

```
e = 0.9; % span efficiency factor

% aspect ratio
AR_Temp = 16.5; % found in "Tempest Characteristic"
AR_BO = 7; % found in "BOEING 747-200 Characteristics"
```

Find a0

```
% caluclate the slope of the 2-D airfoil using equation m=(y1-y2)/(x1-
x2)
a0Temp = (CLTemp2D(7)-CLTemp2D(6))/(AoATemp(7)-AoATemp(6));
a0BO = (CLBO2D(7)-CLBO2D(6))/(AoABO(7)-AoABO(6));
```

Find AoA at L=0

```
% calculate the slope with same form as above using one index above
 the
% x axis and one index below the x axis
mTemp = (CLTemp2D(4)-CLTemp2D(3))/(AoATemp(4)-AoATemp(3));
mBO = (CLBO2D(5)-CLBO2D(4))/(AoABO(5)-AoABO(4));

% make a large vector of AoA values to use in estimating AoA at L=0
xTemp = linspace(AoATemp(1),AoATemp(end),1000);
xBO = linspace(AoABO(1),AoABO(end),1000);

% use point slope formula to find respective CL values
yTemp = CLTemp2D(4) + mTemp*(xTemp - AoATemp(4));
yBO = CLBO2D(4) + mBO*(xBO - AoABO(4));

% find the index closest to CL = 0
for i = 1:1000
   if yTemp(i) <= 0
       idxTemp = i;
   end

   if yBO(i) <= 0
      idxBO = i;
   end

end

% use the index found to assign AoA when L=0
AoATemp_L0 = xTemp(idxTemp);
AoABO_L0 = xBO(idxBO);
```

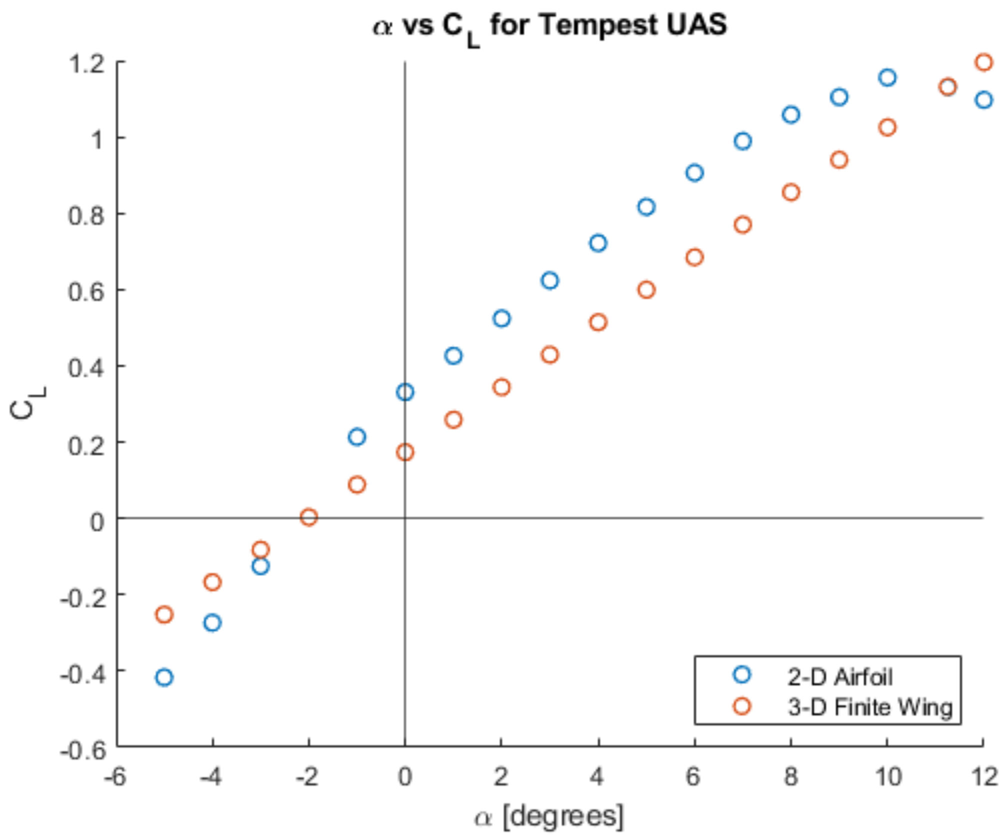
Find 3-D Coefficient of Lift

Note: all equations came directly from the lab document given

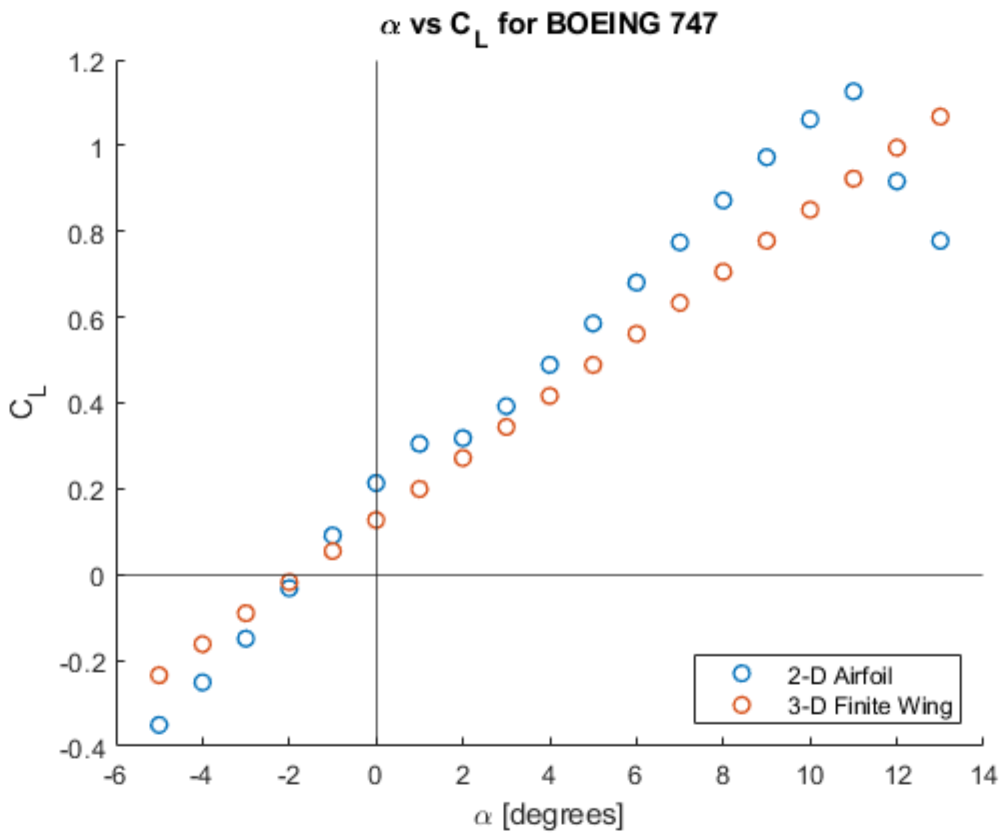
```
% lift curve slope equation (3-D) (eq. 1)
aTemp = a0Temp/(1+((57.3*a0Temp)/(pi*e*AR_Temp)));
aBO = a0BO/(1+((57.3*a0BO)/(pi*e*AR_BO)));

% 3-D coefficent of lift equation (eq. 2)
CLTemp3D = aTemp*(AoATemp-AoATemp_L0);
CLBO3D = aBO*(AoABO-AoABO_L0);
```

Plot 2-D Airfoil vs Approximation 3-D Finite Wing

```
% plot for the Tempest UAS
figure(1)
hold on
plot(AoATemp,CLTemp2D,'o','Linewidth',1); % AoA vs 2-D airfoil
plot(AoATemp,CLTemp3D,'o','Linewidth',1); % AoA vs 3-D finite wing
xline(0); % y axis
yline(0); % x axis
% labels
title('\alpha vs C_L for Tempest UAS');
ylabel('C_L');
xlabel('\alpha [degrees]');
legend('2-D Airfoil','3-D Finite Wing','Location','southeast');
hold off

% plot for the BOEING 747
figure(2)
hold on
plot(AoABO,CLBO2D,'o','Linewidth',1); % AoA vs 2-D airfoil
plot(AoABO,CLBO3D,'o','Linewidth',1); % AoA vs 3-D finite wing
xline(0); % y axis
yline(0); % x axis
% labels
title('\alpha vs C_L for BOEING 747');
ylabel('C_L');
xlabel('\alpha [degrees]');
legend('2-D Airfoil','3-D Finite Wing','Location','southeast');
hold off
```



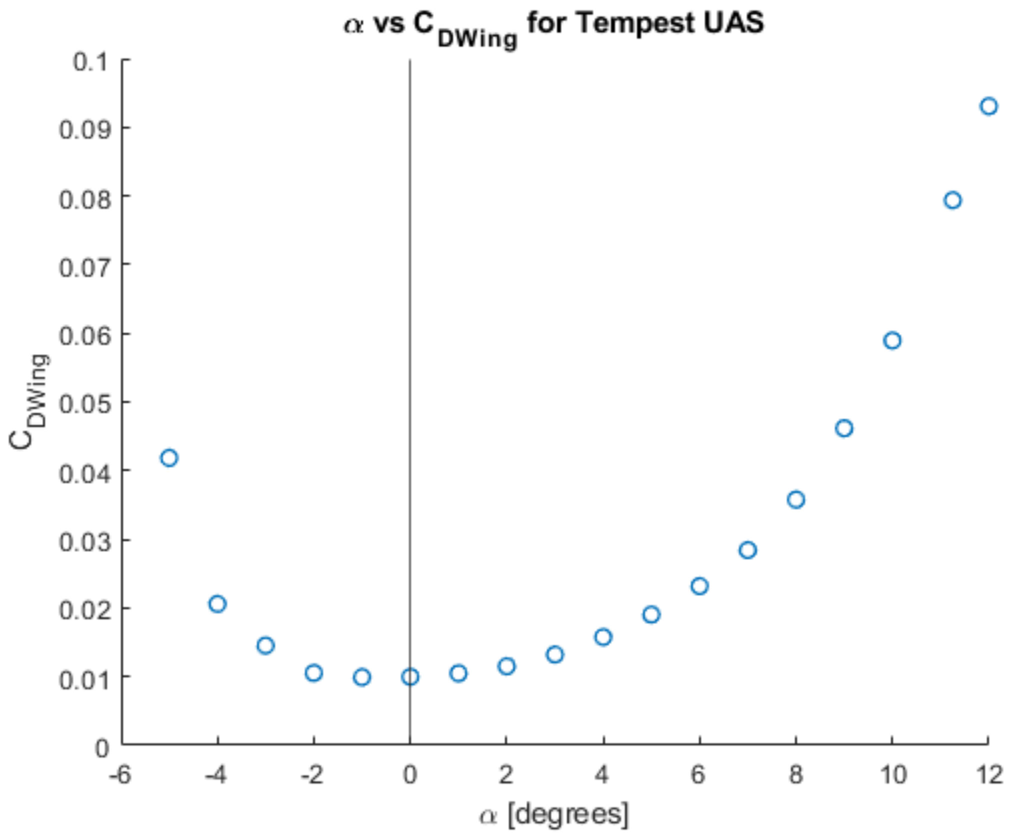
α vs C_L for Tempest UAS
C_L
α [degrees]
2-D Airfoil
3-D Finite Wing



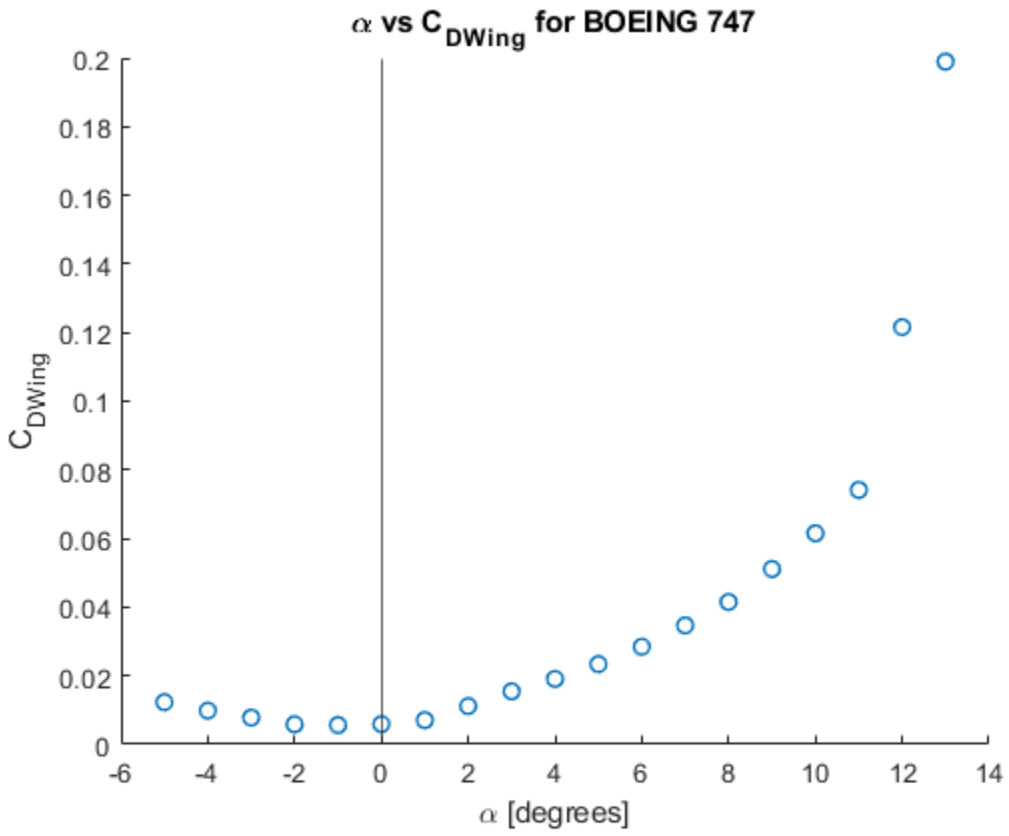
α vs C_L for BOEING 747
C_L
α [degrees]
2-D Airfoil
3-D Finite Wing

Find Wing Drag Polar and Coefficient of Lift at Minimum Drag

```
% 3-D wing drag polar (eq. 3)
CDwing_Temp = Cd_Temp + ((CLTemp3D.^2)./(pi*e*AR_Temp));
CDwing_BO = Cd_BO + ((CLBO3D.^2)./(pi*e*AR_BO));

% finding the index where wing drag polar is minimizes
[~,idxMinTemp] = min(CDwing_Temp);
[~,idxMinBO] = min(CDwing_BO);

% applying said index to find coefficient of lift at minimum drag
CLminD_Temp = CLTemp3D(idxMinTemp);
CLminD_BO = CLBO3D(idxMinBO);

% plot of wing drag polar vs AoA for Tempest UAS
figure(3)
hold on
plot(AoATemp,CDwing_Temp,'o','Linewidth',1);
xline(0); % y axis
yline(0); % x axis
% labels
title('\alpha vs C_D_W_i_n_g for Tempest UAS');
ylabel('C_D_W_i_n_g');
xlabel('\alpha [degrees]');
hold off

% plot of wing drag polar vs AoA for BOEING 747
figure(4)
hold on
plot(AoABO,CDwing_BO,'o','Linewidth',1);
xline(0); % y axis
yline(0); % x axis
% labels
title('\alpha vs C_D_W_i_n_g for BOEING 747');
ylabel('C_D_W_i_n_g');
xlabel('\alpha [degrees]');
hold off

% displaying coefficent of lift at minimum drag
%fprintf("Tempest: %f \nBOEING: %f \n",CLminD_Temp,CLminD_BO);
```



α vs C_{DWing} for Tempest UAS
C_{DWing}
α [degrees]



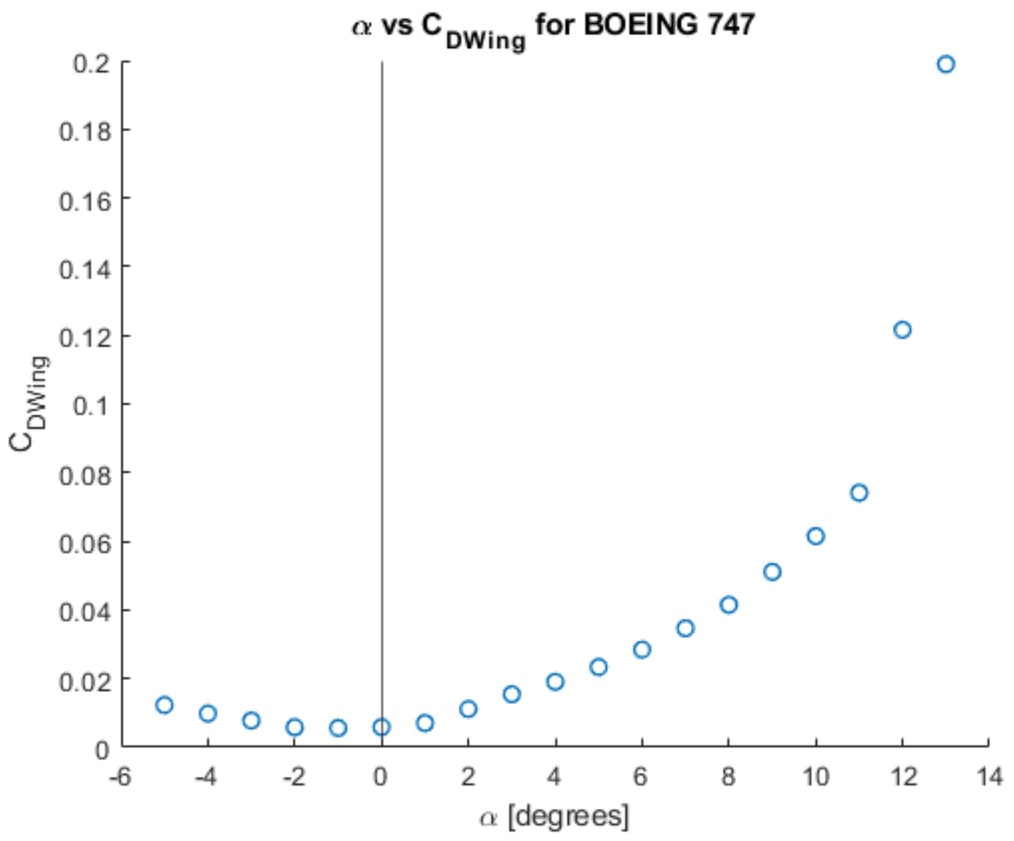
α vs C_{DWing} for BOEING 747
C_{DWing}
α [degrees]

Drag vs Lift Plots

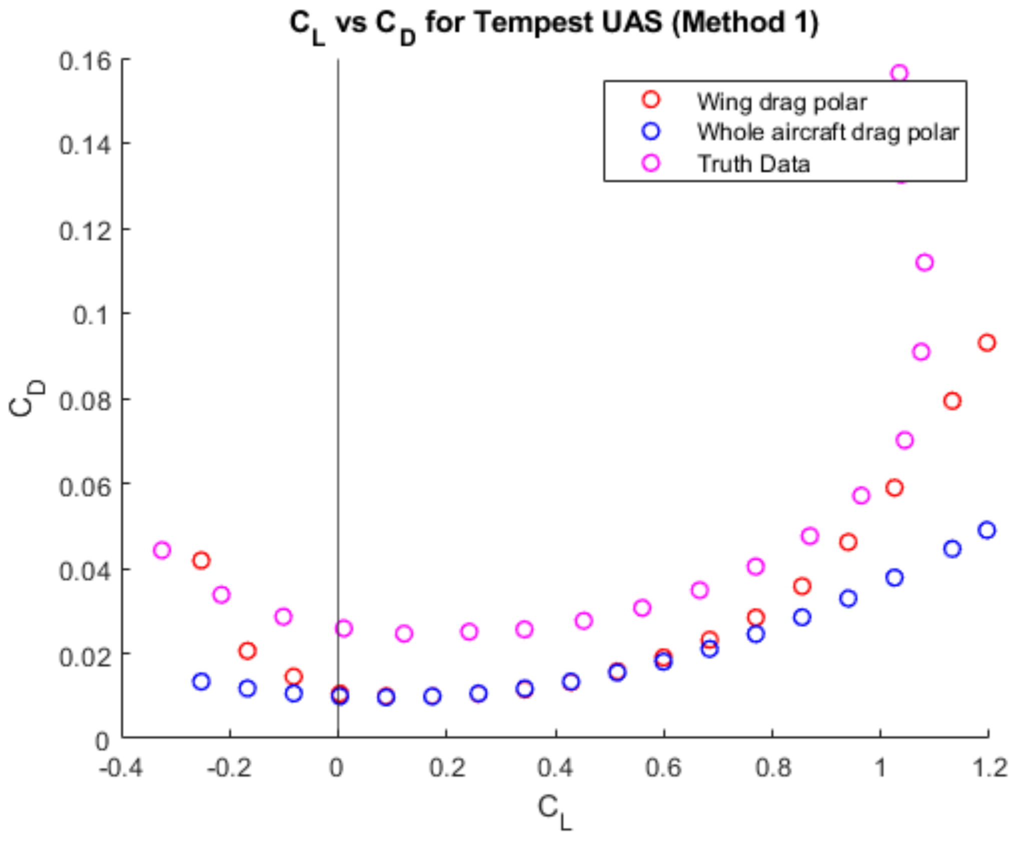
BOEING 747 data

```
% read in truth data for both models
TempTruth = xlsread('Tempest_CFD_Drag_Polar.xlsx');
BOTruth = xlsread('Boeing_747_Drag_Polar_(Exp).xlsx');

% extract coefficient of lift (3-D)
CLtruthTemp = TempTruth(:,2);
CLtruthBO = BOTruth(:,1);

% extract coefficient of drag (3-D)
CDtruthTemp = TempTruth(:,3);
CDtruthBO = BOTruth(:,2);

% Hard coded Values
Cfe_Temp = 0.003; % equivalent skin friction coefficient
Swet_Temp = 2.027; % [m^2] wetted area (Tempest)
Sref_Temp = 0.63; % [m^2] projected area of the wing (Tempest)

Cfe_BO = 0.003; % equivalent skin friction coefficient
Swet_BO = 30000; % [m^2] wetted area (BOEING)
Sref_BO = 5500; % [m^2] projected area of the wing (BOEING

% minimum drag (eq. 10)
CDminTemp = Cfe_Temp*Swet_Temp/Sref_Temp;
CDminBO = Cfe_BO*Swet_BO/Sref_BO;

% Oswald's efficiency factor (eq. 12)
e0_Temp = 1.78*(1-0.045*(AR_Temp^0.68))-0.64; % method 1
e0_BO = 1.78*(1-0.045*(AR_BO^0.68))-0.64;

% method 2 for finding Oswald's Efficiency Factor
Q = 1.05;
P = 0.007;

e0_Temp_Method2 = 1/(Q+P*pi*AR_Temp);

% arbitrary constant (eq. 5)
k1_Temp = 1/(pi*e0_Temp*AR_Temp);
k1_BO = 1/(pi*e0_BO*AR_BO);

k1_Temp_Method2 = 1/(pi*e0_Temp_Method2*AR_Temp);

% whole aircraft drag (eq. 6)
CD_Temp = CDminTemp + k1_Temp*(CLTemp3D - CLminD_Temp).^2;
CD_BO = CDminBO + k1_BO*(CLBO3D - CLminD_BO).^2;

CD_Temp_Method2 = CDminTemp + k1_Temp_Method2*(CLTemp3D -
 CLminD_Temp).^2;

% plot for Tempest UAS Method 1
```

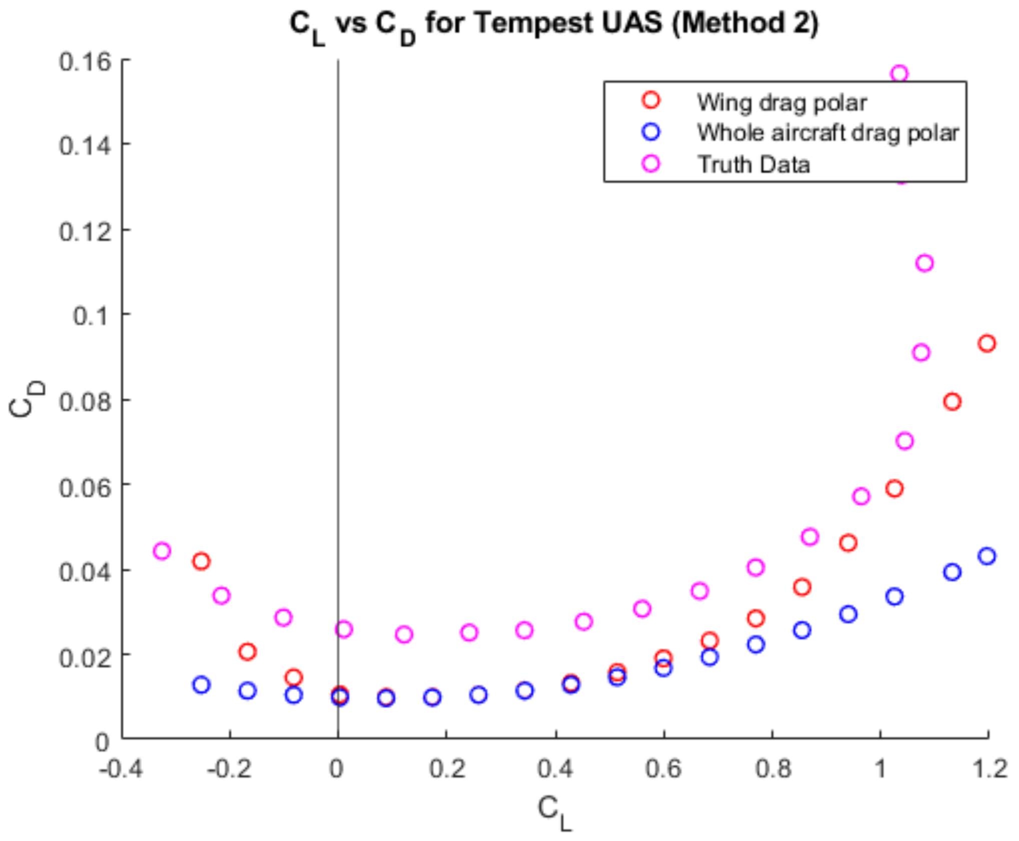
```
figure(5)
hold on
plot(CLTemp3D,CDwing_Temp,'ro','LineWidth',1); % 3-D finite wing drag
 polar
plot(CLTemp3D,CD_Temp,'bo','LineWidth',1); % whole aircraft drag polar
plot(CLtruthTemp,CDtruthTemp,'mo','LineWidth',1); % truth data drag
 polar provided
xline(0); % y axis
yline(0); % x axis
% labels
title('C_L vs C_D for Tempest UAS (Method 1)');
ylabel('C_D');
xlabel('C_L');
legend('Wing drag polar','Whole aircraft drag polar','Truth Data');
hold off

% plot for Tempest UAS Method 2
figure(6)
hold on
plot(CLTemp3D,CDwing_Temp,'ro','LineWidth',1); % 3-D finite wing drag
 polar
plot(CLTemp3D,CD_Temp_Method2,'bo','LineWidth',1); % whole aircraft
 drag polar
plot(CLtruthTemp,CDtruthTemp,'mo','LineWidth',1); % truth data drag
 polar provided
xline(0); % y axis
yline(0); % x axis
% labels
title('C_L vs C_D for Tempest UAS (Method 2)');
ylabel('C_D');
xlabel('C_L');
legend('Wing drag polar','Whole aircraft drag polar','Truth Data');
hold off

% plot for BOEING 747
figure(7)
hold on
plot(CLBO3D,CDwing_BO,'ro','LineWidth',1); % 3-D finite wing drag
 polar
plot(CLBO3D,CD_BO,'bo','LineWidth',1); % whole aircraft drag polar
plot(CLtruthBO,CDtruthBO,'mo','LineWidth',1); % truth data drag polar
 provided
xline(0); % y axis
yline(0); % x axis
% labels
title('C_L vs C_D for BOEING 747');
ylabel('C_D');
xlabel('C_L');
legend('Wing drag polar','Whole aircraft drag polar','Truth Data');
hold off
```

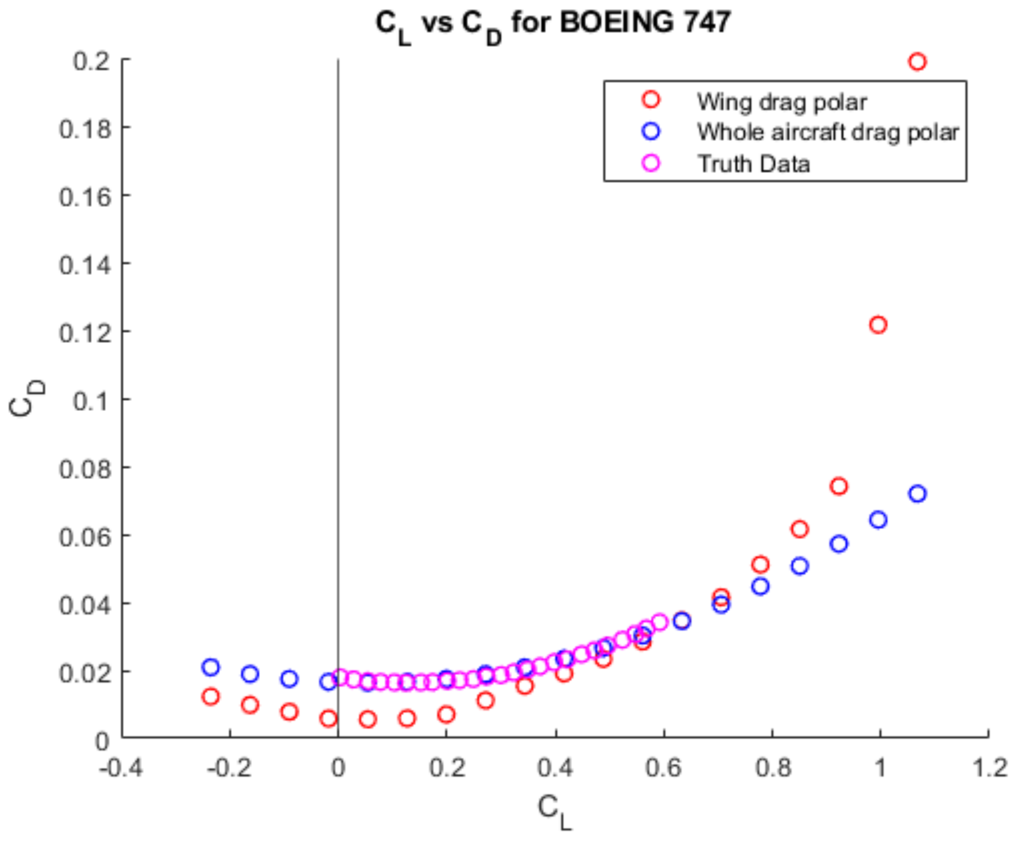


α vs C_{DWing} for BOEING 747
C_{DWing}
α [degrees]



C_L vs C_D for Tempest UAS (Method 1)
Wing drag polar
Whole aircraft drag polar
Truth Data
C_D
C_L


C_L vs C_D for Tempest UAS (Method 2)
Wing drag polar
Whole aircraft drag polar
Truth Data
C_D
C_L
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
1.2


C_L vs C_D for BOEING 747
Wing drag polar
Whole aircraft drag polar
Truth Data
C_D
C_L
0.2
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
1.2

Comparisons

```
%Glide Range
% Solve Cl from drag
[ClCdBOMAX,iGRBO] = max(CLBO3D./CD_BO);
[ClCdTempMAX,iGRTemp] = max(CLTemp3D./CD_Temp);

[ClCdBOMAXTru,iGRBOTru] = max(CLtruthBO./CDtruthBO);
[ClCdTempMAXTru,iGRTempTru] = max(CLtruthTemp./CDtruthTemp);

hTemp = 1500; %m
hBO = 10668; %m

wBO = 3705368; %N
%wBO = 3559000; %N
wTemp = 62.78; %N

pinfBO = 0.38034956; %kg/m^3
%pinfBO = 0.38857; %kg/m^3
pinfTemp = 1.0581; %kg/m^3

VinfBO = sqrt((2*wBO)/(CLBO3D(iGRBO)*Sref_BO*pinfBO));
VinfTemp = sqrt((2*wTemp)/(CLTemp3D(iGRTemp)*Sref_Temp*pinfTemp));

VinfBOTru = sqrt((2*wBO)/(CLtruthBO(iGRBOTru)*Sref_BO*pinfBO));
VinfTempTru = sqrt((2*wTemp)/
(CLtruthTemp(iGRTempTru)*Sref_Temp*pinfTemp));

MaxGlideRange = [VinfBO VinfBOTru; VinfTemp VinfTempTru];

%Powered Endurance Propeller (Tempest)
%   largest possible propeller efficieency
%   lowest sfc
%   largest weight fraction of fuel
%   largest L^(3/2)/D
%   largest pinf (sea level)

[~,iETemp] = max((CLTemp3D.^(1.5))./CD_Temp);
[~,iETempTru] = max((CLtruthTemp.^(1.5))./CDtruthTemp);

VinfTemp = sqrt((2*wTemp)/(CLTemp3D(iETemp)*Sref_Temp*pinfTemp));
VinfTempTru = sqrt((2*wTemp)/
(CLtruthTemp(iETempTru)*Sref_Temp*pinfTemp));

%Powered Endurance Jet (747)
%   lowest tsfc
%   largest weight fraction of fuel
%   largest L/D

VinfBO = sqrt((2*wBO)/(CLBO3D(iGRBO)*Sref_BO*pinfBO));
VinfBOTru = sqrt((2*wBO)/(CLtruthBO(iGRBOTru)*Sref_BO*pinfBO));

MaxPoweredEndurance = [VinfBO VinfBOTru; VinfTemp VinfTempTru];
```

```
%Powered Range Propeller (Tempest)
%   largest possible propeller efficieency
%   lowest sfc
%   largest weight fraction of fuel
%   largest L/D
[~,iPRTemp] = max(CLTemp3D./CD_Temp);
[~,iPRTempTru] = max(CLtruthTemp./CDtruthTemp);

VinfTemp = sqrt((2*wTemp)/(CLTemp3D(iPRTemp)*Sref_Temp*pinfTemp));
VinfTempTru = sqrt((2*wTemp)/
(CLtruthTemp(iPRTempTru)*Sref_Temp*pinfTemp));

%Powered Range Jet (747)
%   lowest sfc
%   largest weight fraction of fuel
%   largest L^(1/2)/D
%   lowest pinf (high altittude)
[~,iPRBO] = max((CLBO3D.^(1/2))./CD_BO);
[~,iPRBOTru] = max((CLtruthBO.^(1/2))./CDtruthBO);
VinfBO = sqrt((2*wBO)/(CLBO3D(iPRBO)*Sref_BO*pinfBO));
VinfBOTru = sqrt((2*wBO)/(CLtruthBO(iPRBOTru)*Sref_BO*pinfBO));

MaxPoweredRange = [VinfBO VinfBOTru; VinfTemp VinfTempTru];
```

Published with MATLAB® R2019a