Drag Buildup Tool Report

Claire Griffith, Chandler Kuhn, and Max Whiton
California Polytechnic State University, San Luis Obispo, California, 93410

This report outlines the development of the drag buildup tool and a user manual for its use.

I. Nomenclature

κ	=	Surface Roughness
μ	=	Air Viscosity
ρ	=	Air Density
Δ_f	=	Fuselage Tail Upsweep Angle
AR	=	Aspect Ratio
b	=	Wingspan
c	=	Chord Length
C_d	=	Coefficient of Drag, airfoil
C_l	=	Coefficient of Lift, airfoil
C_D	=	Total Coefficient of Drag
C_{Di}	=	Coefficient of Induced Drag
C_{Do}	=	Coefficient of Parasitic Drag
C_L	=	Coefficient of Lift
C_f	=	Skin Friction Coefficient
D	=	Fuselage Maximum Diameter
e	=	Oswald Efficiency Factor
f	=	Fineness Ratio
FF	=	Form Factor
K	=	Tail Upsweep Angle Factor
L	=	Length
Re	=	Reynold's number
S	=	Wing Area
$S_{exposed}$	=	Exposed Wing Area
S_{fuse}	=	Surface Area of the Fuselage

S_{tail}	=	Surface Area of Vertical or Horizontal Tail
S_{wet}	=	Wetted Area
T	=	Temperature
W	=	Aircraft Maximum Takeoff Weight
v	=	Airspeed
NACA	=	National Advisory Committee for Aeronautics

II. Introduction

This project aimed to understand and produce and code that is able to predict the total drag on an aircraft. In the goal of beating the Rutan Voyager, an accurate and reliable drag estimation is extremely important due to the high lift to drag ratio needed to accomplish the mission. Additionally, the geometries capable of completing this mission tend to be unique. The Voyager and Global Flyer both had three fuselages and multiple vertical stabilizers. This meant that the code also had to be able to accommodate unique geometries. In order to accomplish this task a drag build up method was used. The main script and all functions were written in MATLAB as well as a user interface.

III. Methodology

A component build up method was used in this tool. The total drag on the aircraft is a combination of the parasitic and induced drags on each of the components, as well as the interference drag between each of them. An additional miscellaneous drag component accounts for the drag from additional drag-contributing components that are not calculated individually. Equation 1 describes the drag on the aircraft.

$$C_D = C_{D,Wing} + C_{D,Empennage} + C_{D,Fuselage} + C_{D,Nacelle} + C_{D,Interference} + C_{D,Miscellaneous} \tag{1}$$

A. Wing Drag

Both the parasitic and induced drag on the wing are calculated. The total drag coefficient on the wing is the sum of the drag coefficients, as seen in equation 2.

$$C_{Dwing} = C_{Do,wing} + C_{Di,wing} \tag{2}$$

1. Parasitic Drag

The parasitic drag on the wing was calculated using existing parasitic drag data for NACA 6-series airfoils[1]. For each airfoil, the coefficients of lift and drag at a Reynolds number of $Re = 10^6$ were recorded. Each airfoil is defined by its % thickness and its design C_l. To calculate the drag on any given airfoil thickness or design C_l, the tool linearly interpolates between four airfoils - two encapsulating the thickness and two encapsulating the design C_l. The result is a new interpolated parabola. Additionally, each drag polar for a 6-series airfoil has a "drag bucket" centered around its design C_l, the width of which is related to the airfoil's % thickness, and the depth of which is related to the airfoil's design C_l. A relationship between these variables was found and used to linearly interpolate the characteristics of the drag bucket for any airfoil based on % thickness and design C_l. The result is a discontinuity in the interpolated drag polar. An example of this interpolation is shown in Fig. 1.

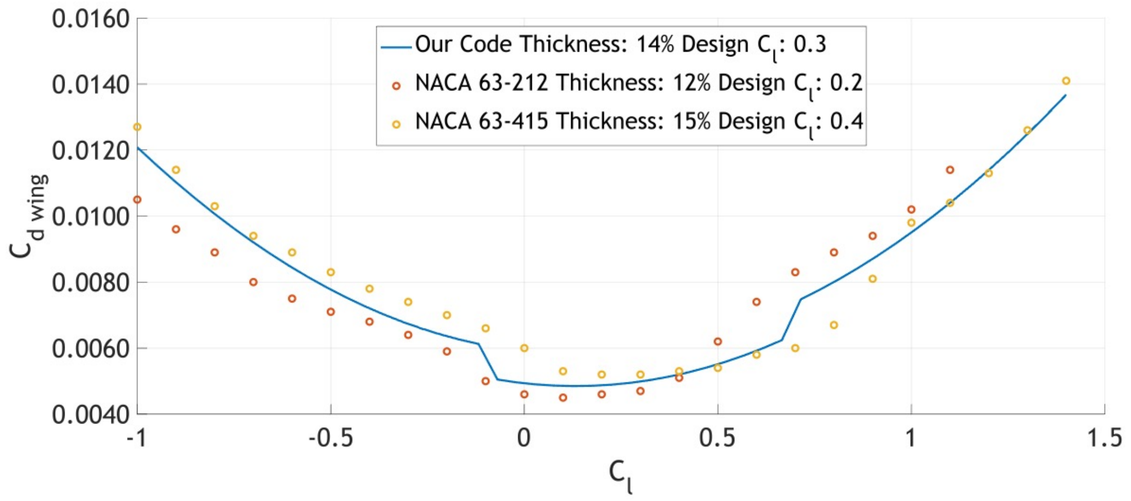


Fig. 1 Interpolated airfoil drag polar

This interpolation method allows for the calculation of parasitic drag for various combinations of % thickness and design C_l. The tool outputs a drag polar based on these input characteristics at the range of lift coefficients that correspond to the range of input airspeeds, a relationship described by equation 3. In this case, S is the wing area including the section through the fuselage, shown in Fig. 2. There is a limitation on the combinations of these variables that can be accommodated, a result of the data available which limits the interpolation that is possible. The specific testable ranges are described in Section IV.

$$C_L = \frac{2W}{\rho S v^2} \tag{3}$$

2. Induced Drag

The induced drag on the wing is calculated using equation 4, where the aspect ratio is calculated as in equation 5[3]. The exposed area of the wing, $S_{exposed}$, is the area of the wing that can be seen in a top down view of the aircraft, excluding the fuselage. This is shown in Fig. 3.

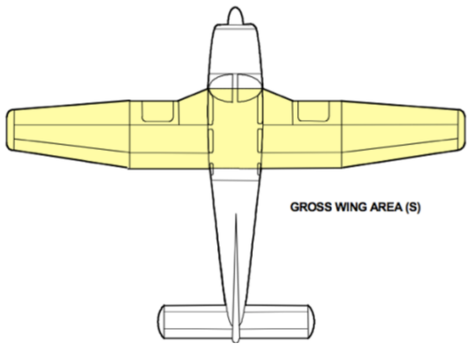


Fig. 2 Wing area [2]

$$C_{Di,wing} = \frac{C_l^2}{\pi ARe} \tag{4}$$

$$AR = \frac{b^2}{S_{exposed}} \tag{5}$$

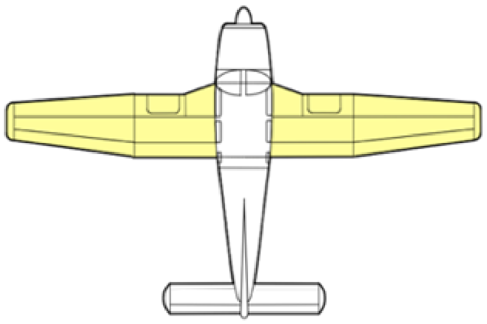


Fig. 3 Exposed wing area [2]

The Oswald efficiency factor, e, is assumed to be 0.8. The induced drag coefficient is calculated at each C_l over the range of user inputted airspeeds.

B. Empennage Drag

The methodology for calculating the coefficient of drag on the empennage is very similar to that of the wing. This method accommodates vertical and horizontal tails, as well as v-tails.

1. Parasitic Drag

To calculate the parasitic drag on a vertical or horizontal, the same airfoil method used to calculate parasitic drag on a wing is used. The same assumptions apply.

2. *Induced Drag*

The induced drag on the tail surfaces is ignored for the purposes of this tool.

C. Fuselage and Nacelle Drag

The method for calculating drag coefficient is the same for fuselages and nacelles.

1. *Parasitic Drag*

The parasitic drag on the fuselage/nacelle is calculated by modeling the components as bodies of revolution[3]. The geometry of the fuselage is simplified into the shapes shown in Fig. 4 - a paraboloid, cylinder, and cone. The tool is able to accommodate multiple engine configurations. If there is an engine on the back of a fuselage or nacelle, the cone is changed to a paraboloid. The surface area of this shape is calculated using equation 11.

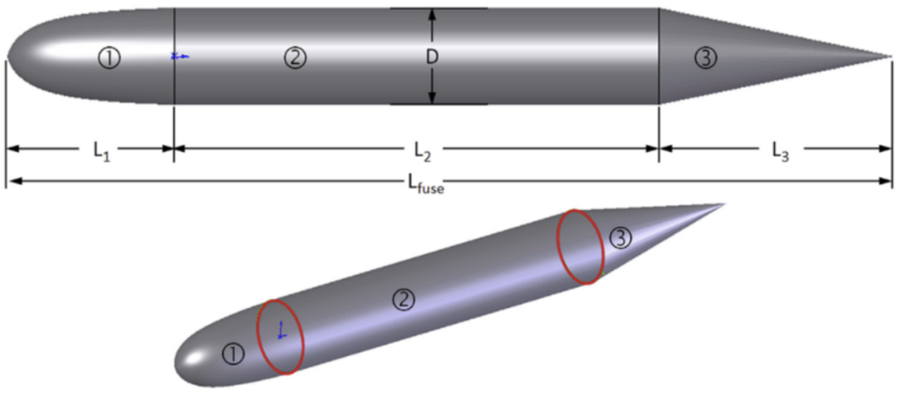


Fig. 4 Exposed wing area [3]

$$S_{fuse} = \left\{ \frac{\pi D}{12 L_1^2} \left[\left(4L_1^2 + \frac{D^2}{4} \right)^{1.5} - \frac{D^3}{8} \right] - \frac{\pi D^2}{4} \right\} + \left\{ \pi D L^2 \right\} + \left\{ \frac{\pi D}{2} \sqrt{L_3^2 + \frac{D^2}{4}} \right\} \tag{6}$$

Equation 7 describes the parasitic drag coefficient on the fuselage[4].

$$C_{Do,fuselage} = C_{Do_1} + C_{Do_2} + C_{Do_3} + C_{Do,\Lambda_f} \tag{7}$$

Equations 8, 9, and 10 are the parasitic drag coefficients on each section of the fuselage.

$$C_{Do,1} = \frac{C_{f\,fuse} \cdot FF_{fuse} \cdot S_{wet,1}}{S_{fuse}} \tag{8}$$

$$C_{Do,2} = \frac{C_{f\,fuse} \cdot S_{wet,2}}{S_{fuse}} \tag{9}$$

$$C_{D_{o,3}} = \frac{C_{f_{fuse}} \cdot FF_{fuse} \cdot S_{wet,3}}{S_{fuse}} \tag{10}$$

The wetted areas for sections 1, 2, and 3 of the fuselage body of revolution correspond to the three terms in 6. The fuselage area, described by equation 11, is the area of a cross section of the body of revolution.

$$S_{fuse} = \left(\frac{4}{3}\right) L_1 D + L_2 D + \left(\frac{1}{3}\right) L_3 D \tag{11}$$

If the back of the fuselage or nacelle has another engine, both equations 6 and 11 are modified, making the third terms the surface area and cross sectional area of a paraboloid.

It is necessary to define a fuselage form factor, FF_{fuse} - an empirical equation based on the fuselage geometry. This tool uses Raymer's form factor for a generic fuselage, shown in equation 12.

$$FF_{fuse} = 1 + \frac{60}{(f_{fuse})^3} + \frac{f_{fuse}}{400} \tag{12}$$

The fuselage fineness ratio, f_{fuse}, relates the length of the fuselage to the maximum diameter.

$$f_{fuse} = \frac{L_{fuse}}{D} \tag{13}$$

It is also necessary to calculate a skin friction coefficient on the fuselage, described by equation 14[3].

$$C_f = \frac{0.074}{Re^{0.2}} \left(1 - \left(\frac{X_{tr} - X_o}{L_{fuse}}\right)\right)^{0.8} \tag{14}$$

The Reynold's number in equation 14 is either the Reynold's number based on the inputted flight conditions (Eq. 15) or the Reynold's number based on the skin friction on the fuselage, called the cutoff Reynold's number (Eq. 17)[3]. To determine which one is used, both are calculated and compared to each other and the lowest Reynold's number is used.

$$Re_{cond} = \frac{\rho v L_{fuse}}{\mu} \tag{15}$$

μ in equation 15 is calculated based on the input flight altitude using equation 16. In this equation, the temperature is found based on the flight altitude using a standard atmosphere calculator.

$$\mu = (3.170x10^{-11}) T^{1.5} \left(\frac{734.7}{T + 216}\right) \tag{16}$$

$$Re_{cutoff} = 38.21 \left(\frac{L_{fuse}}{\kappa}\right)^{1.053} \tag{17}$$

κ in equation 17 is the skin roughness of the fuselage surface. Because the tool is designed for highly aerodynamic aircraft, the tool holds κ at a constant $9.843x10^-5$ ft, the value for high quality paint.

To estimate the locations of the start of the turbulent boundary layer and the transition point on the fuselage, it assumed there is mixed laminar-turbulent flow. The location of the start of the turbulent boundary layer is calculated using equation 18[4]. This relates the boundary layer location to the geometry of the fuselage.

$$X_o = (0.374)L_{fuse} + (0.533)L_1 \tag{18}$$

The transition point along the fuselage is calculated using equation [3].

$$\left(\frac{X_o}{L_{fuse}}\right) = 36.9\left(\frac{X_{tr}}{L_{fuse}}\right)^{0.625}\left(\frac{1}{Re}\right)^{0.375} \tag{19}$$

The final term in equation 7 is the parasitic drag resulting from any tail upsweep of the fuselage (Eq. 20).

$$C_{Do,\Lambda_f} = (7)\left(\frac{K}{100}\right)C_{Do,3} \tag{20}$$

The tail upsweep angle, Λ_f is the angle between the horizontal reference line of the fuselage and the trailing edge of the tail, shown in Fig. 5. K is a factor based on the tail upsweep angle. The relationship between these variables is shown in Fig. 6.

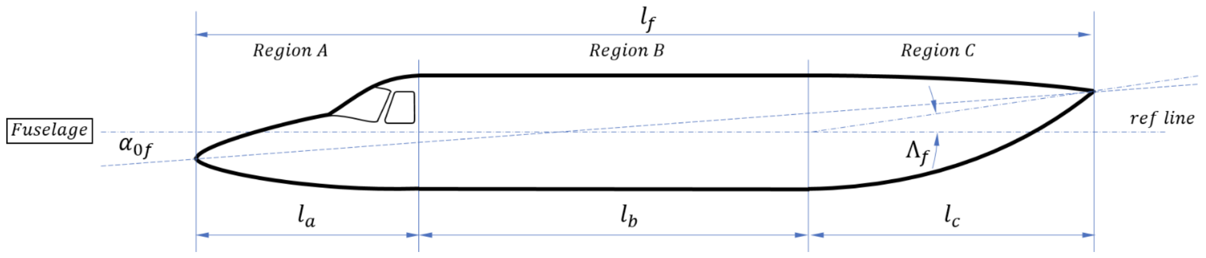


Fig. 5 Tail upsweep angle [4]

2. Induced Drag

The induced drag on a fuselage or nacelle is ignored.

D. Interference Drag

Interference drag is the drag resulting from the interferences between components. For this tool, the wing-fuselage, wing-nacelle, and tail-fuselage interferences were considered[4].

The wing-fuselage interference was estimated as 5% of the total fuselage drag. This basic calculation is described by equation 21.

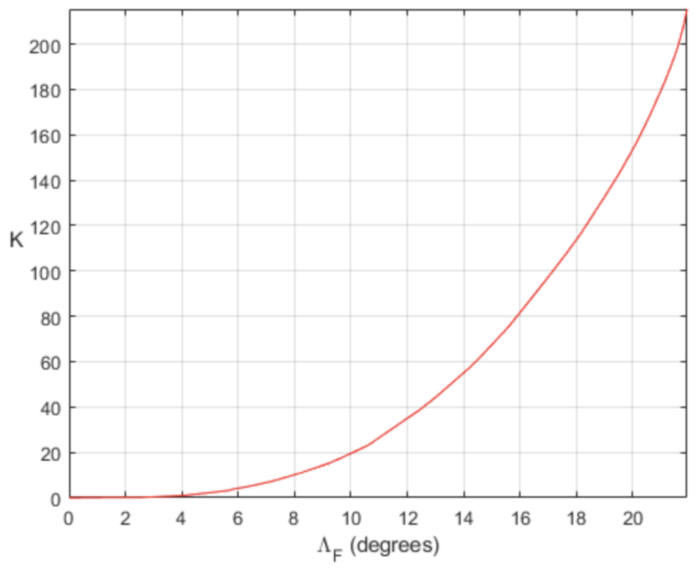


Fig. 6 Tail upsweep angle factor v Λ_f [4]

$$\Delta C_{Dwf} = 0.05 \cdot C_{D fuse} \tag{21}$$

The same 5% estimate was used for the wing-fuselage interference (Eq. 22). n is the number of nacelles.

$$\Delta C_{Dwn} = 0.05 \cdot C_{Dnacelle} \cdot n \tag{22}$$

The interference between the tail and the fuselage is approximated by considering the number of corners between the tail and the fuselage, n_c. Equation 23 describes the additional interference drag between a tail and the fuselage. Equation 24 describes the additional interference drag between two tails. (t/c) is the % thickness of the airfoils used for the tails. c_j is the chord of the tail at the junction between the tail and the fuselage. The equations are applicable to both horizontal and vertical tails.

$$\Delta C_{Dtf} = n_c \left[0.8(t/c)^3 - 0.0005\right] \frac{c_j^2}{S_{tail}} \tag{23}$$

$$\Delta C_{Dtf} = \frac{n_c}{2} \left[17(t/c)^4 - 0.05(t/c)^2\right] \frac{c_j^2}{S_{tail}} \tag{24}$$

E. Miscellaneous Drag

An additional amount of parasitic drag is added to the final drag coefficient of the aircraft to account for additional drag contributing components that are not included in the component build up[3]. The amount of miscellaneous drag added is a somewhat subjective estimation based on typical values for drag of these components. Four common additional components were considered.

First, the additional drag from aircraft canopies is considered to be 5% of the total aircraft parasitic drag (Eq. 25).

$$C_{Do,canopy} = 0.05 \cdot C_{Do} \tag{25}$$

The additional drag from cooling ducts is also considered to be 5% of the total aircraft parasitic drag (Eq. 26).

$$C_{Do,cooling} = 0.05 \cdot C_{Do} \tag{26}$$

The additional drag from fixed landing gear is considered to be 10% of the total aircraft parasitic drag (Eq. 27).

$$C_{Do,landinggear} = 0.1 \cdot C_{Do} \tag{27}$$

Finally, the additional drag due to a more rectangular fuselage than the body of revolution used to calculate fuselage drag is considered to be 5 % of the total aircraft parasitic drag (Eq. 28).

$$C_{Do,rect.fuse} = 0.05 \cdot C_{Do} \tag{28}$$

If these components are present on the configuration being tested, the additional drag is added to the total aircraft drag.

F. Normalization

The drag coefficients on each component are normalized to the aircraft reference area. The surface area of each component is divided by the aircraft reference area (wing area) to scale the drag coefficient appropriately. A generalized form of this is shown in equation 29.

$$C_{Dcomp,norm} = C_{Dcomp} \cdot \frac{S_{comp}}{S} \tag{29}$$

IV. Assumptions & Limitations

Many assumptions should be known when using the drag build-up tool. Going in order of the methodology sections these will be explored in this sections. Some of the overarching assumptions of the code are that it is designed to predict drag in the cruise condition. Additionally, the aircraft must be a high aspect ratio aircraft, flying between 0 and 65000 feet, and be subsonic. Due to the lack of fidelity with the function and for a margin on the reliability of the prediction a miscellaneous drag factor is added to all calculations. This factor is a percent of the total parasitic drag added in at the end of the calculation.

A. Wing Drag

The first assumption associated with the wing drag function is the interpolation method. As stated above the wing drag function uses an interpolation to predict airfoil properties for NACA 63 series airfoils. This interpolation method is linear in the function, however, this is not perfect. There could be different characteristics observed in a real airfoil of the desired parameters. This is related to the limitation of the function in that it only predicts the behavior of NACA 63 series airfoils. If any other airfoil is desired the drag polar could be put in manually but is not available through the function. Another limitation in the functions is that there are only a certain range of thicknesses and design lift coefficients it will accurately predict. Table 1 shows these combinations. Anything outside of these ranges will either break or give an inaccurate prediction of drag characteristics of the airfoil.

% Thickness	4-9	10-12	13-20
Design C_l	0-0.3	0-0.5	0-0.7

Table 1 Airfoil % thickness and C_l combinations.

The final assumption that should be assumed is that the function only works for higher aspect ratios. This is due to the assumption that induced drag is estimated using aspect ratio and Oswald efficiency. Which leads to the final limitation of the function which is that the user must calculate their own estimated Oswald efficiency.

B. Fuselage Drag

The fuselage drag, as explain in the previous section, is estimated using an equivalent body of revolution and estimating the form drag on this object. In this process, the skin friction coefficient must be calculated. The fuselage drag function assumes that the surface has a smooth paint job. The main limitation of this simplification of the shape of a fuselage is that no real fuselages are shaped this way, which leaves some guess work to the user to determine how long each of the three sections is. Giving more length to the middle section and less to the front or back will result in different amounts of drag. Another limitation of the function is that it does not calculate the induced drag produced by the fuselage in flight. This induced drag is on the order of drag counts but at higher angles of attack/lift coefficient it can become more significant leading to less accurate results at higher lift conditions.

C. Empennage Drag

The main assumption in the empennage drag function is that the horizontal stabilizer does not produce any induced drag. The drag polar code was developed to estimate the drag of an aircraft in the cruise condition. This means that a well designed aircraft's tail should produce very little lift and therefore very little induced drag. Additionally, the function has the same limitations as the wing drag function mentioned above as it uses the same database of NACA 63 series airfoils and interpolation method. The final assumption in the function is that it assumes no side slipping. Again there is no induced drag calculated for the tail surfaces, so any force produced by the vertical stabilizer is not captured.

D. Nacelle Drag

The assumptions made for the nacelle drag are the same as fuselage. The function estimates the equivalent body of revolution using a parabolic, cylinder, and cone. This is a very simplified geometry and will not capture different or special contours. The code also assumes a skin friction coefficient with a smooth paint job. Then the function, like the fuselage drag function, does not take into account the lift produced by the nacelle. This will be more of a limitation at high angles of attack and high lift coefficients.

E. Interference Drag

For interference drag the first assumption made was that the drag produced by the intersection of the wing with the fuselage was 5%. This is a conservative assumption and the actual value of drag will be different. The next assumption was that the interference drag with the empennage was related only to the thickness ratio and the reference around of the horizontal and vertical stabilizer. This is a good assumption however it does not capture any turbulent or disturbed flow coming from the fuselage and wing wings.

F. Miscellaneous Drag

The values chosen for the additional miscellaneous drag are somewhat subjective, which limits their fidelity. Guidelines are in place to ensure standardization of the miscellaneous drag across configurations.

V. Validity

In order to validate the code a selection of aircraft were taken with drag data and compared to the prediction made by the code. These aircraft ranged from gliders, to general aviation aircraft, to unmanned vehicles. The first example shown below is a comparison to the Cal Poly Van's RV-7A.

This aircraft does not meet a lot the assumptions the code makes, therefore a miscellaneous drag factor of 25% was included to the parasitic drag. This is per the guidelines outlined above. To restate the factors again though, 10% was added for the fixed landing gear, 5% was added for the cooling ducts, 5% was added for the more square fuselage, and 5% was added for the canopy bubble. As can be seen from the chart is that they are close but not exact. This is most likely due to the effects that are not captured. While miscellaneous drag helps correct the model is not perfect and will result in a miscalculation of parasitic drag. Furthermore, the only component producing induced drag is the wing when in reality the fuselage and horizontal stabilizer also contribute to this for of drag.

Moving on to an example of an aircraft that has some strange geometry, similar to something that may be seen on

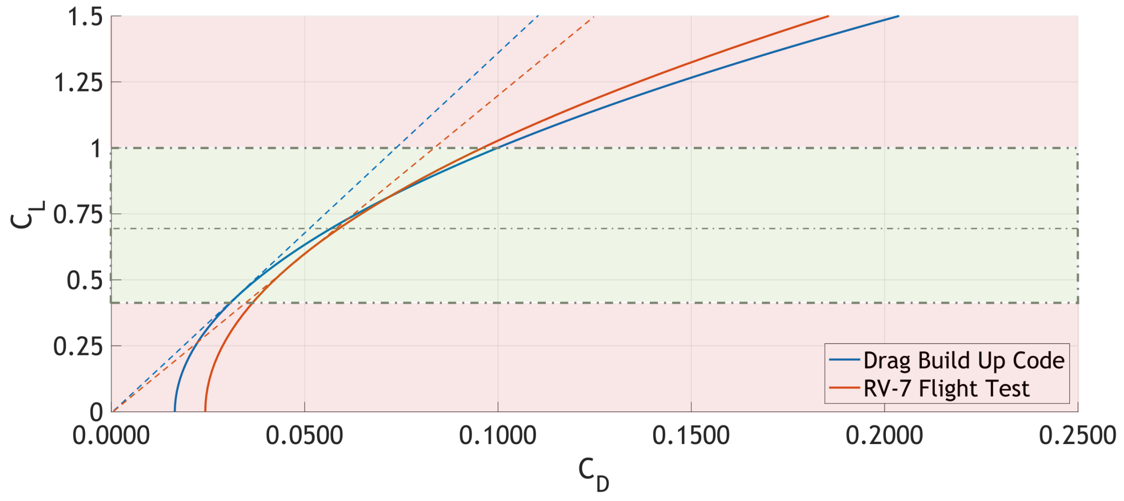


Fig. 7 Cal Poly RV-7A data plotted with results produced by drag build up tool.

our aircraft, the ANCE UAV was compared to the code. This is an unmanned vehicle designed and tested in a subsonic, closed-circuit wind tunnel by researchers at the Simón Bolívar University in Venezuela.

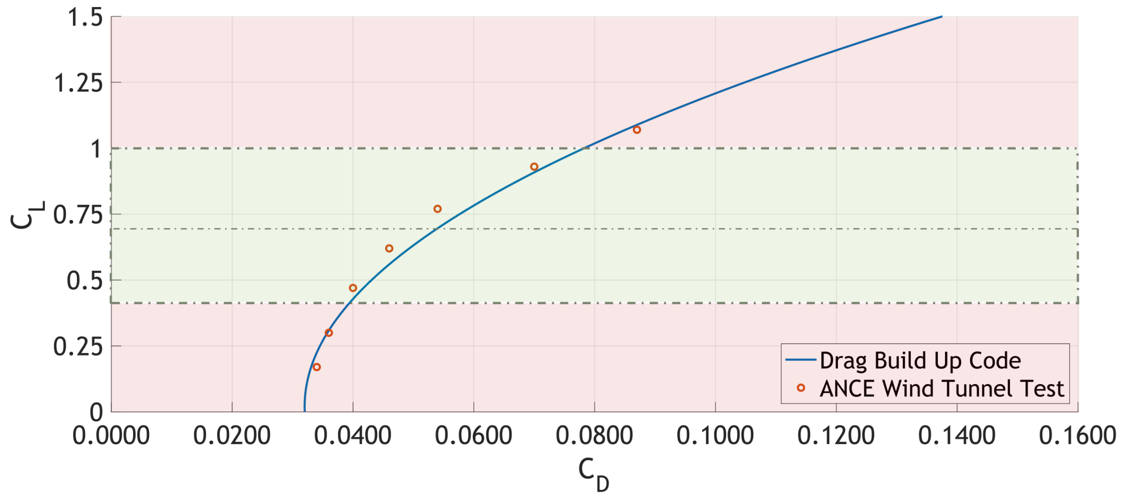


Fig. 8 Simón Bolívar ANCE UAV data [5] plotted with results produced by drag build up tool.

The final test case was the the T-6 glider. This glider was a one of kind purpose build sailplane that has a reputation for being one of the most flight-tested gliders and published data. The miscellaneous drag added to this was 5% for the canopy, and sleek design that was designed to be highly aerodynamic.

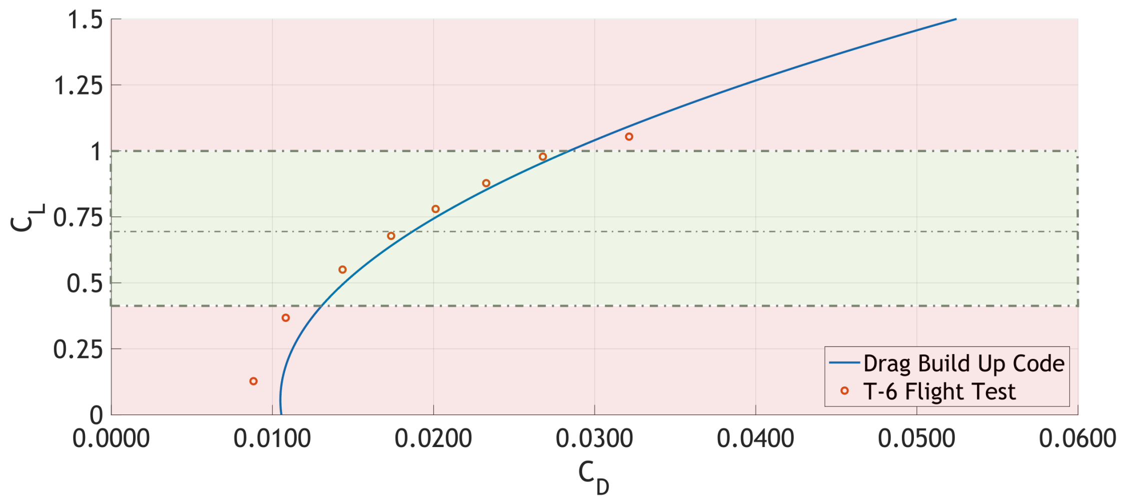


Fig. 9 Published T-6 Glider data [6] plotted with results produced by drag build up tool.

To validate modification results, we ran multiple cases with modified parameters of the 1st Design Teams configuration. These consisted of baseline comparisons to increased aspect ratio, increased fuselage length, and zero percent miscellaneous drag, all considered independently. The baseline miscellaneous drag assigned was 10%, determined using the logic above earlier, and the average difference was 32 drag counts across the range of C_L's for this test case. This represents roughly ten percent or more of total drag at C_L's less than 1.

When the aspect ratio was increased, expected results were obtained, with roughly equal parasitic drag and decreased induced drag, which is driven by the increased wing lift, and overall improvements in aircraft efficiency. Alternatively, increasing the length of the fuselage resulted in increased parasitic drag and almost equal induced drag, influenced likely by skin friction effects and altered form drag. It is important again to note that this is also due to the tool neglecting fuselage lift effects.

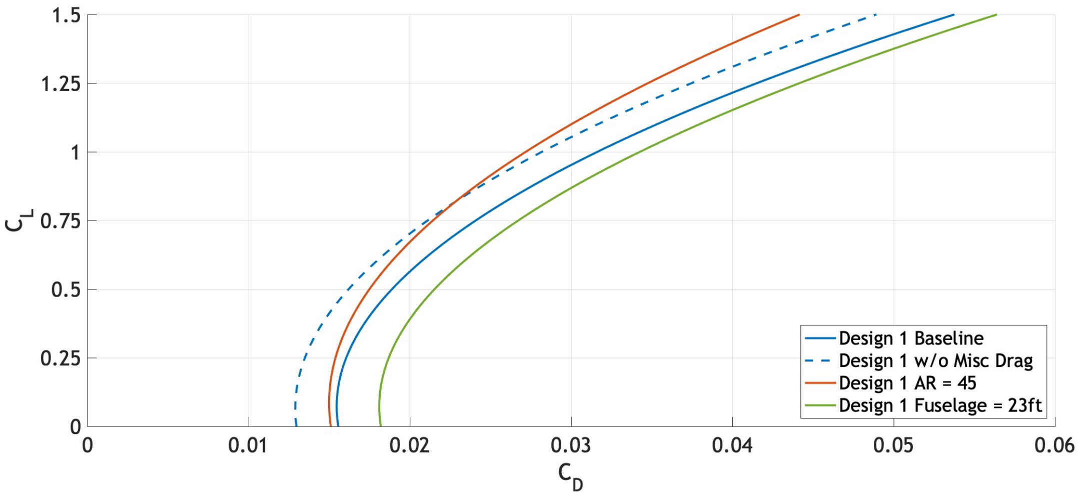
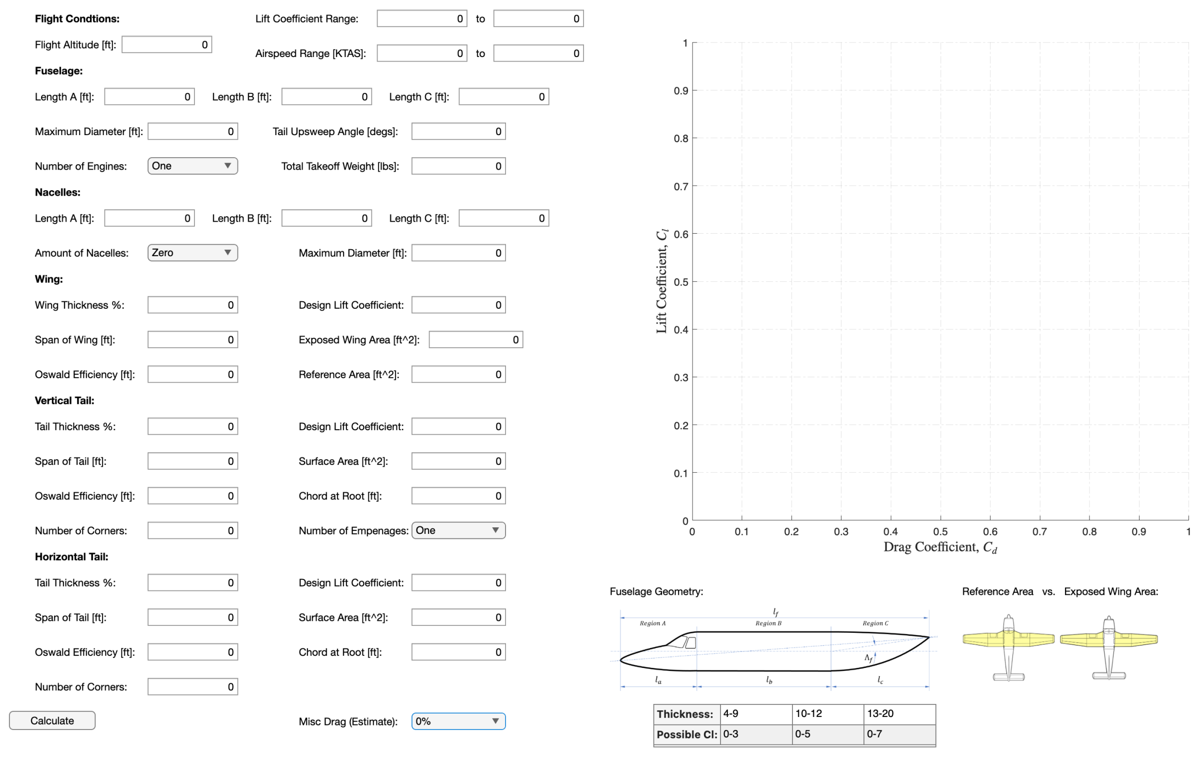


Fig. 10 Design Team 1 baseline configuration with two modification cases.

VI. User Manual

Starting with the `DragPolarDashboard.m` file opened in `MATLAB`, click the Run button in the top toolbar, which should proceed to launch the user interface shown in the screenshot below:



Flight Condtions:

Flight Altitude [ft]: 0

Lift Coefficient Range: 0 to 0

Airspeed Range [KTAS]: 0 to 0

Fuselage:

Length A [ft]: 0 Length B [ft]: 0 Length C [ft]: 0

Maximum Diameter [ft]: 0 Tail Upsweep Angle [degs]: 0

Number of Engines: One Total Takeoff Weight [lbs]: 0

Nacelles:

Length A [ft]: 0 Length B [ft]: 0 Length C [ft]: 0

Amount of Nacelles: Zero Maximum Diameter [ft]: 0

Wing:

Wing Thickness %: 0 Design Lift Coefficient: 0

Span of Wing [ft]: 0 Exposed Wing Area [ft^2]: 0

Oswald Efficiency [ft]: 0 Reference Area [ft^2]: 0

Vertical Tail:

Tail Thickness %: 0 Design Lift Coefficient: 0

Span of Tail [ft]: 0 Surface Area [ft^2]: 0

Oswald Efficiency [ft]: 0 Chord at Root [ft]: 0

Number of Corners: 0 Number of Empenages: One

Horizontal Tail:

Tail Thickness %: 0 Design Lift Coefficient: 0

Span of Tail [ft]: 0 Surface Area [ft^2]: 0

Oswald Efficiency [ft]: 0 Chord at Root [ft]: 0

Number of Corners: 0

Misc Drag (Estimate): 0%

Thickness:	4-9	10-12	13-20
Possible Cl:	0-3	0-5	0-7

Fig. 11 Blank Drag Polar Dashboard Display.

Using the type fields and dropdown boxes for each drag buildup category, input all necessary geometry and flight condition based parameters, or closest estimates if unknown. The inputs should reflect the aircraft's projected measurements as accurately as possible to ensure the validity of the calculation. Below is a guide on each dropdown/box, which should be utilized for more specific direction on each input. Keep in mind the tools limitations, such as neglecting induced drag on the fuselage & nacelles, as well as underestimation of the drag drag caused by landing gear, cooling systems, or windscreen/canopies.

NOTE: it is important to ensure that all seven necessary functions (namely: `fuselageDrag.m, wingDrag.m, tailDrag.m, nacelleDrag.m, interferenceDrag.m, standatm.m, iterpol.m`) are all installed in the same folder/path as the `DragPolarDashboard.m` file.

1) Flight Conditions

- **Flight Altitude:** Input the average altitude above sea level at which the aircraft will cruise during the flight.

  Units: ft

- **Airspeed Range:** *Range* of true airspeeds the aircraft will fly throughout the design mission.

  Units: KTAS

- **Lift Coefficient Range:** *Range* of lift coefficients the aircraft will fly throughout its design mission.

  Units: n/a

2) Fuselage

- **Number of Engines:** Dropdown separating a single or double engine configuration.
- **Length:** Using Fig. 12, insert the *three* respective sectional lengths of the fuselage.

  Units: ft

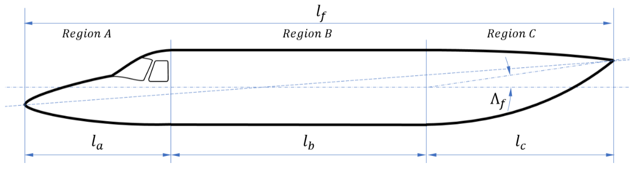


Fig. 12 Fuselage Geometry Graphic.[4]

- **Max Diameter:** The largest diameter of the fuselage cross-section, derived along section B.

  Units: ft

- **Tail Upsweep Angle:** The angle between the fuselage's bottom and the tail's upsweep.

  Units: degs

- **Total Takeoff Weight:** Maximum expected gross takeoff weight expected according to design mission.

  Units: lbs

3) Nacelles

NOTE: If there are no nacelles, leave the dropdown at 'Zero' and ignore all applicable input fields.

- **Amount of Nacelles:** Dropdown for the number of nacelles associated with aircraft configuration.
- **Length:** Using Fig. 12, insert the *three* respective sectional lengths of any nacelle(s).

  Units: ft
- **Max Diameter:** The largest diameter of the nacelles cross-section, derived along section B.

  Units: ft

4) Wing

- **Wing Thickness:** The greatest thickness of the wing as a percentage of the chord length.

  Units: %

  NOTE: ensure wing thickness and design C_L are within the bounds of Table 1 for wing/tail surfaces.
- **Design C_l:** The design lift coefficient of the wing producing the least drag, multiplied by *ten*.

  Units: n/a
- **Span of Wing:** The wingspan length measured from one tip of the wing to the other.

  Units: ft
- **Exposed Wing Area:** According to Fig. 13, the total wing area exposed & independent of the fuselage.

  Units: ft^2

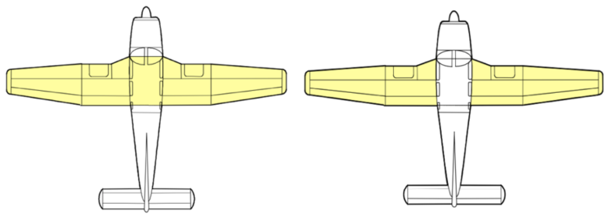


Fig. 13 Wing Areas Geometry Graphic.[3]

- **Oswald Efficiency:** Wing property Oswald Efficiency Factor, *e*, between 0 and 1.

  Units: n/a
- **Reference Area:** According to Fig. 13, the total wing area *including* that through the fuselage.

  Units: ft^2

5) Vertical Tail

- **Number of Empennages:** Dropdown allowing for a multiple tail configuration or flying wing.
- **Tail Thickness:** The greatest thickness of the vertical stabilizer as a percentage of the chord length.

  Units: % (*x100*)
- **Design C_l:** The design lift coefficient of the vertical stabilizer, multiplied by *ten*.

Units: n/a

- **Span of Tail:** The height of the vertical stabilizer measured from root to tip.

  Units: ft

- **Surface Area:** The total vertical stabilizer area exposed & independent of the fuselage.

  Units: ft^2

- **Oswald Efficiency:** Wing property Oswald Efficiency Factor, *e*, between 0 and 1.

  Units: n/a

- **Chord at Root:** The vertical tail chord at the root; maximum exposed chord length.

  Units: ft

- **Number of Corners:** The amount of corners/intersection points with the fuselage or horizontal stabilizer.

  Units: n/a

6) Horizontal Tail

- **Tail Thickness:** The greatest thickness of the horizontal stabilizer as a percentage of the chord length.

  Units: % (*x100*)

- **Design C$_l$:** The design lift coefficient of the horizontal stabilizer, multiplied by *ten*.

  Units: n/a

- **Span of Tail:** The span of the horizontal stabilizer measured from tip of one end to the other.

  Units: ft

- **Surface Area:** The total horizontal stabilizer area exposed & independent of the fuselage.

  Units: ft^2

- **Oswald Efficiency:** Measure of the wing's efficiency Oswald Efficiency Factor, *e*, between 0 and 1.

  Units: n/a

- **Chord at Root:** The horizontal tail chord at the root; maximum exposed chord length.

  Units: ft

- **Number of Corners:** The amount of corners/intersection points with the fuselage or vertical stabilizer.

  Units: n/a

7) Miscellaneous Drag

- **Misc Drag:** Dropdown accounting for estimated additional drag; measure of overall aerodynamic efficiency.

  Units: %

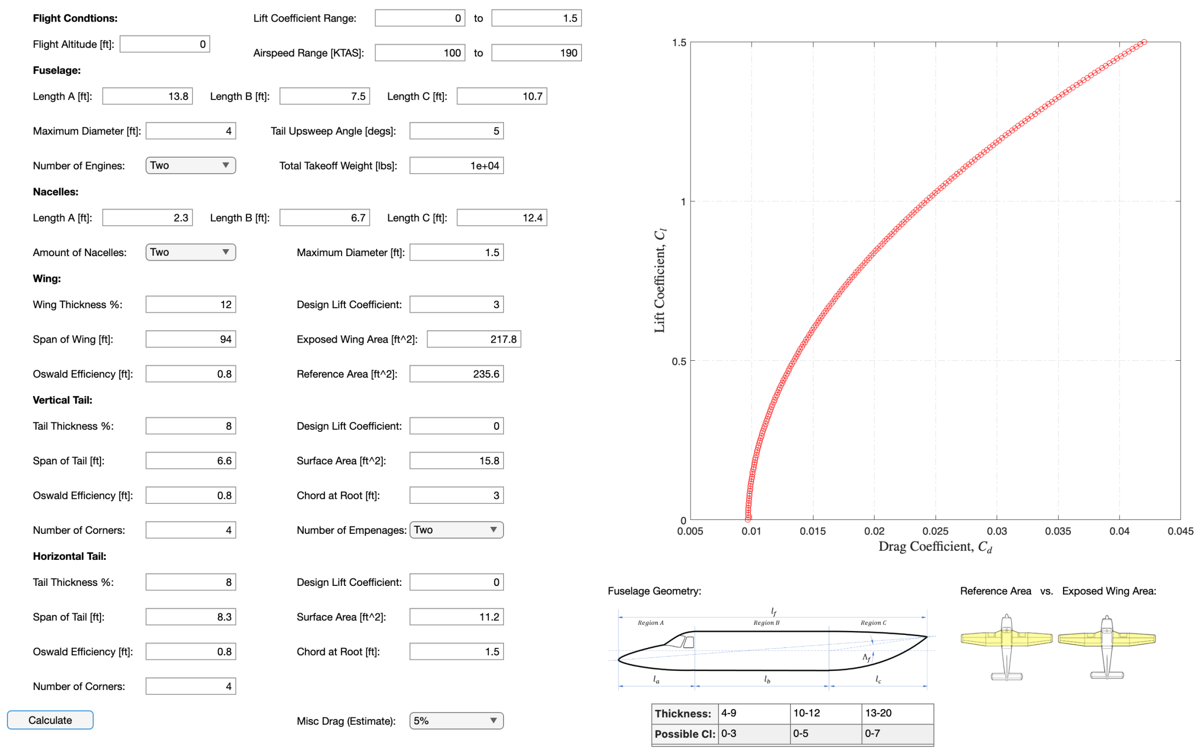
  NOTE: estimated based on the following

  - Cooling Ducts - 5%
  - Aircraft Canopy - 5%
  - Fixed Landing Gear - 10%

– Rectangular Fuselage - 5%

After entering all the necessary inputs click Calculate to generate a drag polar chart displaying the relationship between the drag coefficient (C_D) and the lift coefficient (C_L) on the horizontal and vertical axes respectively. Use the drag polar data to assess and refine the design, as a feedback loop, to achieve desired aerodynamic efficiency.

e.g. The following figure is an example of an output drag polar with all fields filled out and calculated:



Flight Condtions:
Flight Altitude [ft]: 0
Lift Coefficient Range: 0 to 1.5
Airspeed Range [KTAS]: 100 to 190

Fuselage:
Length A [ft]: 13.8
Length B [ft]: 7.5
Length C [ft]: 10.7
Maximum Diameter [ft]: 4
Tail Upsweep Angle [degs]: 5
Number of Engines: Two
Total Takeoff Weight [lbs]: 1e+04

Nacelles:
Length A [ft]: 2.3
Length B [ft]: 6.7
Length C [ft]: 12.4
Amount of Nacelles: Two
Maximum Diameter [ft]: 1.5

Wing:
Wing Thickness %: 12
Design Lift Coefficient: 3
Span of Wing [ft]: 94
Exposed Wing Area [ft^2]: 217.8
Oswald Efficiency [ft]: 0.8
Reference Area [ft^2]: 235.6

Vertical Tail:
Tail Thickness %: 8
Design Lift Coefficient: 0
Span of Tail [ft]: 6.6
Surface Area [ft^2]: 15.8
Oswald Efficiency [ft]: 0.8
Chord at Root [ft]: 3
Number of Corners: 4
Number of Empenages: Two

Horizontal Tail:
Tail Thickness %: 8
Design Lift Coefficient: 0
Span of Tail [ft]: 8.3
Surface Area [ft^2]: 11.2
Oswald Efficiency [ft]: 0.8
Chord at Root [ft]: 1.5
Number of Corners: 4

Misc Drag (Estimate): 5%

Fuselage Geometry:

Reference Area vs. Exposed Wing Area:

Thickness:	4-9	10-12	13-20
Possible Cl:	0-3	0-5	0-7

Fig. 14 Example Display of Drag Polar Dashboard Calculated.

VII. Appendix

A. Source Code

```
function DragPolarDashboard()
drawnow
fig = uifigure('Name', 'Aircraft Design Drag Polar Dashboard','Position', [20
    0 1400 900], 'WindowState','maximized'); % 'Position', [100 100 600 400],
fig.WindowState = 'maximized';
set(fig,'color','w')

%===========================Flight Conditions==============================
head1 = uilabel(fig, 'Position', [50 860 150 20], 'Text', 'Flight Condtions:')
    ;
head1.FontWeight = 'bold';

uilabel(fig, 'Position', [50 830 150 20], 'Text', 'Flight Altitude [ft]:');
condAlt = uieditfield(fig, 'numeric', 'Position', [150 830 100 20]);

uilabel(fig, 'Position', [300 820 150 20], 'Text', 'Airspeed Range [KTAS]:');
condVmin = uieditfield(fig, 'numeric', 'Position', [440 820 100 20]);
uilabel(fig, 'Position', [550 820 150 20], 'Text', 'to');
condVmax = uieditfield(fig, 'numeric', 'Position', [570 820 100 20]);

uilabel(fig, 'Position', [300 860 150 20], 'Text', 'Lift Coefficient Range:');
condClmin = uieditfield(fig, 'numeric', 'Position', [440 860 100 20]);
uilabel(fig, 'Position', [550 860 150 20], 'Text', 'to');
condClmax = uieditfield(fig, 'numeric', 'Position', [570 860 100 20]);

%========================Fuselage Drag Inputs=============================
head2 = uilabel(fig, 'Position', [50 800 150 20], 'Text', 'Fuselage:');
head2.FontWeight = 'bold';

% Dropdown for Engine Selection
```

```matlab
uilabel(fig, 'Position', [50 690 150 20], 'Text', 'Number of Engines:');
ddEngine = uidropdown(fig, 'Position', [180 690 100 20], 'Items', {'One', 'Two'});

% Input Fields for Fuselage Geometric Parameters
uilabel(fig, 'Position', [50 770 100 20], 'Text', 'Length A [ft]:');
fuseL(1) = uieditfield(fig, 'numeric', 'Position', [130 770 100 20]);

uilabel(fig, 'Position', [250 770 100 20], 'Text', 'Length B [ft]:');
fuseL(2) = uieditfield(fig, 'numeric', 'Position', [330 770 100 20]);

uilabel(fig, 'Position', [450 770 100 20], 'Text', 'Length C [ft]:');
fuseL(3) = uieditfield(fig, 'numeric', 'Position', [530 770 100 20]);

uilabel(fig, 'Position', [50 730 150 20], 'Text', 'Maximum Diameter [ft]:');
fuseD = uieditfield(fig, 'numeric', 'Position', [180 730 100 20]);

uilabel(fig, 'Position', [320 730 150 20], 'Text', 'Tail Upsweep Angle [degs]:');
fuseTS = uieditfield(fig, 'numeric', 'Position', [480 730 100 20]);

uilabel(fig, 'Position', [330 690 150 20], 'Text', 'Total Takeoff Weight [lbs]:');
fuseTW = uieditfield(fig, 'numeric', 'Position', [480 690 100 20]);

% Dropdown for Misc
uilabel(fig, 'Position', [350 50 150 20], 'Text', 'Misc Drag (Estimate):');
ddMisc = uidropdown(fig, 'Position', [480 50 100 20], 'Items', {'0%', '5%', '10%', '15%', '20%', '25%'});

%========================Nacelle Drag Inputs==============================
head3 = uilabel(fig, 'Position', [50 660 150 20], 'Text', 'Nacelles:');
```

```
head3.FontWeight = 'bold';

% Dropdown for Extra Nacelle(s)
uilabel(fig, 'Position', [50 590 150 20], 'Text', 'Amount of Nacelles:');
ddNacelle = uidropdown(fig, 'Position', [180 590 100 20], 'Items', {'Zero', '
    One', 'Two', 'Three'});

% Input Fields for Fuselage Geometric Parameters
uilabel(fig, 'Position', [50 630 100 20], 'Text', 'Length A [ft]:');
nacL(1) = uieditfield(fig, 'numeric', 'Position', [130 630 100 20]);

uilabel(fig, 'Position', [250 630 100 20], 'Text', 'Length B [ft]:');
nacL(2) = uieditfield(fig, 'numeric', 'Position', [330 630 100 20]);

uilabel(fig, 'Position', [450 630 100 20], 'Text', 'Length C [ft]:');
nacL(3) = uieditfield(fig, 'numeric', 'Position', [530 630 100 20]);

uilabel(fig, 'Position', [350 590 150 20], 'Text', 'Maximum Diameter [ft]:');
nacD = uieditfield(fig, 'numeric', 'Position', [480 590 100 20]);

%===========================Wing Drag Inputs================================
head4 = uilabel(fig, 'Position', [50 560 150 20], 'Text', 'Wing:');
head4.FontWeight = 'bold';

uilabel(fig, 'Position', [50 530 150 20], 'Text', 'Wing Thickness %:');
wingT = uieditfield(fig, 'numeric', 'Position', [180 530 100 20]);

uilabel(fig, 'Position', [350 530 150 20], 'Text', 'Design Lift Coefficient:')
    ;
wingCl = uieditfield(fig, 'numeric', 'Position', [480 530 100 20]);

uilabel(fig, 'Position', [50 490 150 20], 'Text', 'Span of Wing [ft]:');
```

```
wingB = uieditfield(fig, 'numeric', 'Position', [180 490 100 20]);

uilabel(fig, 'Position', [350 490 150 20], 'Text', 'Exposed Wing Area [ft^2]:'
    );
wingS = uieditfield(fig, 'numeric', 'Position', [500 490 100 20]);

uilabel(fig, 'Position', [50 450 150 20], 'Text', 'Oswald Efficiency [ft]:');
wingE = uieditfield(fig, 'numeric', 'Position', [180 450 100 20]);

uilabel(fig, 'Position', [350 450 150 20], 'Text', 'Reference Area [ft^2]:');
condSref = uieditfield(fig, 'numeric', 'Position', [480 450 100 20]);

%==========================Tail Drag Inputs===============================
head5 = uilabel(fig, 'Position', [50 420 150 20], 'Text', 'Vertical Tail:');
head5.FontWeight = 'bold';

uilabel(fig, 'Position', [50 390 150 20], 'Text', 'Tail Thickness %:');
vtailThick = uieditfield(fig, 'numeric', 'Position', [180 390 100 20]);

uilabel(fig, 'Position', [350 390 150 20], 'Text', 'Design Lift Coefficient:')
    ;
vtailCl = uieditfield(fig, 'numeric', 'Position', [480 390 100 20]);

uilabel(fig, 'Position', [50 350 150 20], 'Text', 'Span of Tail [ft]:');
vtailB = uieditfield(fig, 'numeric', 'Position', [180 350 100 20]);

uilabel(fig, 'Position', [350 350 150 20], 'Text', 'Surface Area [ft^2]:');
vtailS = uieditfield(fig, 'numeric', 'Position', [480 350 100 20]);

uilabel(fig, 'Position', [50 310 150 20], 'Text', 'Oswald Efficiency [ft]:');
vtailE = uieditfield(fig, 'numeric', 'Position', [180 310 100 20]);

```

```matlab
uilabel(fig, 'Position', [350 310 150 20], 'Text', 'Chord at Root [ft]:');
vtailcor = uieditfield(fig, 'numeric', 'Position', [480 310 100 20]);

uilabel(fig, 'Position', [50 270 150 20], 'Text', 'Number of Corners:');
vtailnum = uieditfield(fig, 'numeric', 'Position', [180 270 100 20]);

% Dropdown for Empanage
uilabel(fig, 'Position', [350 270 150 20], 'Text', 'Number of Empenages:');
ddEmp = uidropdown(fig, 'Position', [480 270 100 20], 'Items', {'One', 'Two'})
    ;

head6 = uilabel(fig, 'Position', [50 240 150 20], 'Text', 'Horizontal Tail:');
head6.FontWeight = 'bold';

uilabel(fig, 'Position', [50 210 150 20], 'Text', 'Tail Thickness %:');
htailThick = uieditfield(fig, 'numeric', 'Position', [180 210 100 20]);

uilabel(fig, 'Position', [350 210 150 20], 'Text', 'Design Lift Coefficient:')
    ;
htailCl = uieditfield(fig, 'numeric', 'Position', [480 210 100 20]);

uilabel(fig, 'Position', [50 170 150 20], 'Text', 'Span of Tail [ft]:');
htailB = uieditfield(fig, 'numeric', 'Position', [180 170 100 20]);

uilabel(fig, 'Position', [350 170 150 20], 'Text', 'Surface Area [ft^2]:');
htailS = uieditfield(fig, 'numeric', 'Position', [480 170 100 20]);

uilabel(fig, 'Position', [50 130 150 20], 'Text', 'Oswald Efficiency [ft]:');
htailE = uieditfield(fig, 'numeric', 'Position', [180 130 100 20]);

uilabel(fig, 'Position', [350 130 150 20], 'Text', 'Chord at Root [ft]:');
htailcor = uieditfield(fig, 'numeric', 'Position', [480 130 100 20]);
```

```

uilabel(fig, 'Position', [50 90 150 20], 'Text', 'Number of Corners:');
htailnum = uieditfield(fig, 'numeric', 'Position', [180 90 100 20]);

% Plot Area
ax = uiaxes(fig, 'Position', [750 250 600 600]);
ylabel(ax, 'Lift Coefficient, $C_l$', 'Interpreter','latex', 'FontSize',16);
xlabel(ax, 'Drag Coefficient, $C_d$', 'Interpreter','latex', 'FontSize',16);
ax.XGrid = 'on';
ax.YGrid = 'on';
ax.GridLineStyle = '-.';

% Thickness Table
uitable(fig,'RowName',{'Thickness:';'Possible Cl:'},'Data',{'4-9','10-12','
    13-20';'0-3','0-5','0-7'},'ColumnName',{},'Position', [750 30 300 50]);

% Images
uilabel(fig, 'Position', [700 200 150 20], 'Text', 'Fuselage Geometry:');
uiimage(fig,'ImageSource','fusegeo.png','Position', [700 95 378 100]); % 567
uilabel(fig, 'Position', [1100 200 235 20], 'Text', '  Reference Area   vs.
    Exposed Wing Area:');
uiimage(fig,'ImageSource','wingareas.png','Position', [1100 95 235 100]);

% Button for Calculation
btnCalculate = uibutton(fig, 'push', 'Position', [20 50 100 22], 'Text', '
    Calculate');
btnCalculate.ButtonPushedFcn = @(btn,event) updatePlot(fig, ax, ...
     ddEngine.Value, ddNacelle.Value, ddEmp.Value, ddMisc.Value, ...
     [condAlt.Value, condVmin.Value, condVmax.Value condSref.Value condClmin.
         Value condClmax.Value], ...
     [fuseL(1).Value, fuseL(2).Value, fuseL(3).Value, fuseD.Value, fuseTS.Value
         , fuseTW.Value], ...
    [nacL(1).Value, nacL(2).Value, nacL(3).Value, nacD.Value], ...
    [wingT.Value, wingCl.Value, wingB.Value, wingS.Value, wingE.Value], ...
    [vtailThick.Value, vtailCl.Value, vtailB.Value, vtailS.Value, vtailE.Value
        , vtailcor.Value, vtailnum.Value], ...
    [htailThick.Value, htailCl.Value, htailB.Value, htailS.Value, htailE.Value
        , htailcor.Value, htailnum.Value]);
end

function updatePlot(fig, ax, ddEngine, ddNacelle, ddEmp, ddMisc, conditions,
   fuse, nac, wing, vtail, htail)

cla(ax,'reset')
clc
clear lbl1 lbl2 lbl3

if ddEngine == "One"
    engine = 1;
else
    engine = 2;
end

if ddEmp == "One"
    n_emp = 1;
else
    n_emp = 2;
end

CD_misc = 0;

if ddMisc == "0%"
    CD_misc = 0;
elseif ddMisc == "5%"
```

```
    CD_misc = .05;
elseif ddMisc == "10%"
    CD_misc = .1;
elseif ddMisc == "15%"
    CD_misc = .15;
elseif ddMisc == "20%"
    CD_misc = .2;
elseif ddMisc == "25%"
    CD_misc = .25;
end

if ddNacelle == "One"
    n_nac = 1;
elseif ddNacelle == "Two"
    n_nac = 2;
elseif ddNacelle == "Three"
    n_nac = 3;
elseif ddNacelle == "Zero"
    n_nac = 0;
end

KTAS = linspace(conditions(2), conditions(3), 200); % Knots True Airspeed
Clrange = linspace(conditions(5), conditions(6), 200);
Cond = [conditions(1), KTAS];
tratio = [htail(1)/100 vtail(1)/100];
c = [htail(6) vtail(6)];
nc = [htail(7) vtail(7)];
S = [htail(4) vtail(4)];
S_ref = conditions(4);

% Wing Calculations
for r = 1:length(KTAS)
```

```
[Clw(r), Cd0w(r), Cdiw(r)] = wingDrag(wing(1), wing(2), wing(5), wing(4), wing
    (3), fuse(6), KTAS(r), conditions(1), Clrange(r));
CD_Wing(r) = (Cd0w(r) + Cdiw(r))*(wing(4)/S_ref);
end

% Tail Calculations
[~, Cd0hs, ~] = tailDrag(htail(1), htail(2), htail(5), htail(4), htail(3));
CD_HS = Cd0hs*(htail(4)/S_ref);

[~, Cd0vs, ~] = tailDrag(vtail(1), vtail(2), vtail(5), vtail(4), vtail(3));
CD_VS = Cd0vs*(vtail(4)/S_ref);

% Fuselage Calculations
[Cd0f, S_Fuse] = fuselageDrag([fuse(1) fuse(2) fuse(3)], fuse(4), Cond, engine
    , fuse(5));
CD_Fuse = Cd0f*(S_Fuse/S_ref);

% Nacelle Calculations
[Cd0n, S_Nac] = nacelleDrag([nac(1) nac(2) nac(3)], nac(4), Cond, n_nac);
CD_Nac = Cd0n*(S_Nac/S_ref);

CD0FN = [CD_Fuse CD_Nac];

% Interference Calculations
[CD_int] = interferenceDrag(CD0FN, n_nac, tratio, c, nc, S);

CD = (CD_Wing + CD_Fuse + CD_HS*n_emp + CD_VS*n_emp + CD_int)*(1+CD_misc);

P = polyfit(Clw, CD, 2);
pv = polyval(P, Clw);

minimumDrag = P(3);
```

```matlab

plot(ax, pv, Clw,'o-r')
ylabel(ax, 'Lift Coefficient, $C_l$', 'Interpreter','latex', 'FontSize',16);
xlabel(ax, 'Drag Coefficient, $C_d$', 'Interpreter','latex', 'FontSize',16);
ax.XGrid = 'on';
ax.YGrid = 'on';
ax.GridLineStyle = '-.';

k = 1/((wing(3)^2/wing(4))*pi*wing(5));

LDmax = sqrt(minimumDrag/k)/(2*minimumDrag);

disp(['The minimum Drag Coefficient is ', num2str(minimumDrag)])
disp(['The Drag Polar Equation is ', num2str(P(1)),'x^2 + ', num2str(P(2)), 'x
    + ', num2str(P(3))])
disp(['The Max L/D is ', num2str(LDmax)])
end
```

```matlab
function [Cd0, Sref] = fuselageDrag(L, D, Cond, eng, t_upsweep)

%==========================================================================
% INPUTS
% l array of lengths of the equivalent bodies of revolution (in ft)
% d largest diameter of the fuselage (in ft)
% Cond flight conditions [alt(ft), KTAS]

% FOR eng == 1 (Two+ engines) ASSUMES two parabolic ends with a cylindrical
    center shape
% FOR eng == 0 (One engine) ASSUMES a parabolic & a conical end with a
    cylindrical center shape
%==========================================================================

% standard atmospheric conditons from input flight altitude and speed
```

```
[Tstd, ~, rho, ~] = standatm(Cond(1), 0, "IMP");

% kelvin to rankine
T = Tstd*1.8; % R

% fuselage fineness ratio
f = sum(L)/D;

% fuselage form factor - Hoerner's method
FF = 1 + (60/f^3) + (f/400);

% location of start of turbulent boundary layer - relation from Iscold's
% instructure
x0 = 0.374*sum(L) + 0.533*L(1); % ft

% viscosity - equation from gudmundsson
mu = 3.17*10^(-11)*T^1.5*(734.7/(T+216)); %lb*s/ft^2

% reynolds number from flight conditions
ReF = (rho*Cond(2)*1.688*sum(L))/mu;

% skin roughness factor - for high quality paint from Iscold's instructure
k = 30*10^-6*3.281; %ft

% cutoff reynolds number based on skin roughness
ReC = 38.21*(sum(L)/k)^1.053;

% compare reynolds number, use lower one
if ReF > ReC
    Re = ReC;
else
    Re = ReF;
```

```
end

% location of laminar-turbulent transition - from gudmundsson
xtr = (((x0/sum(L))/(1/Re)^(0.375))/36.9)^(1/0.625); %ft

% skin friction coeff. for mixed laminar-turbulent flow - gudmundsson
Cf = (0.074/Re^0.2)*(1-((xtr-x0)/sum(L)))^0.8;

% calculate wetted areas
% for two engines
if eng == 2
Sref = pi*D/4*(1/(3*L(1)^2)*((4*L(1)^2+D^2/4)^1.5-D^3/8)-D+4*L(2)+2*1/(3*L(1)
    ^2)*((4*L(1)^2+D^2/4)^1.5-D^3/8));
Swet = (4/3)*L(1)*D + L(2)*D + (4/3)*L(3)*D;

Scom(1) = pi*D/(12*L(1)^2)*((4*L(1)^2+D^2/4)^1.5-D^3/8)-pi*D^2/4; % parabaloid
Scom(2) = pi*D*L(2); % cylinder
Scom(3) = pi*D/(12*L(3)^2)*((4*L(3)^2+D^2/4)^1.5-D^3/8)-pi*D^2/4; % parabaloid

% if one engine
elseif eng == 1
Sref = pi*D/4*(1/(3*L(1)^2)*((4*L(1)^2+D^2/4)^1.5-D^3/8)-D+4*L(2)+2*sqrt(L(3)
    ^2+D^2/4));
Swet = (4/3)*L(1)*D + L(2)*D + (1/3)*L(3)*D;

Scom(1) = pi*D/(12*L(1)^2)*((4*L(1)^2+D^2/4)^1.5-D^3/8)-pi*D^2/4; % parabaloid
Scom(2) = pi*D*L(2); % cylinder
Scom(3) = pi*D/2*sqrt(L(3)^2+D^2/4); % cone

else
    error('Input proper engine flag: two+ engines == 2, one engine == 1')
end
```

```matlab

% tail upsweep angle drag

% equation fit from graph data on iscold's instructure
K = @(t_upsweep) (0.019251*(t_upsweep^3)) - (0.0024779*(t_upsweep^2)) +
    (0.0189825*t_upsweep) - 0.4070;

% upsweep factor at tail upsweep angle in degrees
K_fuse = K(t_upsweep);


% drag on components of fuselage
Cd0A = (Cf*FF*Scom(1))/Swet; % nose section
Cd0B = (Cf*Scom(2))/Swet; % center section
Cd0C = (Cf*FF*Scom(3))/Swet; % tail section

Cd0_up = (K_fuse / 100) * Cd0C; % additional drag from tail upsweep

% add all fuselage components to get total parasitic drag
Cd0 = Cd0A + Cd0B + Cd0C + Cd0_up;
```

```matlab
function [Cl, Cd0, Cdi] = wingDrag(t, c, e, S, b, W, speed, alt, Cl)

%==========================================================================
% INPUTS
% t = Maximum Thickness in %
% c = Design Cl multiplied by 10
% e = oswald efficiency
% S = wing planform area
% b = wing span
% W = aircraft weight (which one?)
% speeds = vector of speeds to find Cd at
% alt = operating altitude
% t and c combinations are only found for combination lying near availible
% data
%=========================================================================

% Load in All the TOWS Data
load('NACA6data.mat');
NACA63 = rmfield(NACA63, "N6210");

fn1 = fieldnames(NACA63);
c1 = 1;
c2 = 1;

%Constant for how much to drop the function and how wide the drop it
width = [6,9,12,15,18,21; 0.15, 0.28, 0.3, 0.44, 0.5, 0.6]';  %Thickness
    Dependant

%Find all the airfoils within the range of thicknesses (+/- 2%)
for i = 1:length(fn1)
    if NACA63.(fn1{i}).thick >= t-2 && NACA63.(fn1{i}).thick <= t+2
        x(c1) = NACA63.(fn1{i});
        c1 = c1+1;
    end
end

%Out of those airfoils find the ones within the right designed Cl (+/- 0.2)
for j = 1:length(x)
    if x(j).camb >= c-1 && x(j).camb <= c+1
        y(c2) = x(j);
        c2 = c2+1;
    end
end

```

```
%Make a data table of all the data from the selected airfoils
for k = 1:length(y)
    fd(:,k) = y(k).data{:,2};
end

numFoils = size(fd);

%Depending on how many airfoils are found, interpolate or not
if numFoils(2) == 1

    yfit = fd;
    xfit = y.data{:,1};

elseif numFoils(2) == 2

    % Find the weight to put on the interpolation
    if y(1).thick == y(2).thick

        wt = 1/((y(1).camb+y(2).camb)/c);

    elseif y(1).camb == y(2).camb

        wt = 1/((y(1).thick+y(2).thick)/c);

    end

    xfit = y(1).data{:,1};
    yfit = iterpol(fd, wt, 0)';

elseif numFoils(2) == 3

    xfit = y(2).data{:,1};
    yfit = y(2).data{:,2};

elseif numFoils(2) == 4

    if y(1).thick == y(2).thick

        wt1 = 1/((y(1).camb+y(2).camb)/c);

    elseif y(1).camb == y(2).camb

        wt1 = 1/((y(1).thick+y(2).thick)/c);

    end

    if y(3).thick == y(4).thick

        wt2 = 1/((y(3).camb+y(4).camb)/c);

    elseif y(3).camb == y(4).camb

        wt2 = 1/((y(3).thick+y(4).thick)/c);

    end

    y1 = iterpol(fd(:,1:2), wt1, 0)';
    y2 = iterpol(fd(:,3:4), wt2, 0)';
    ydat = [y1, y2];
    wt = mean([wt1,wt2]);
    yfit = iterpol(ydat, wt, 0)';
    xfit = y(1).data{:,1};

end
```

```matlab

%Interpolate the width and depth
wid = iterpol(width, t, 1);
range = [(c/10)-wid (c/10)+wid];

c2 = 1;

% Sort Data for NaN and for the width and dpeth
for p = 1:length(yfit)
    if (xfit(p) < range(1) || xfit(p) > range(2)) && isnan(yfit(p)) == 0
        xpoly(c2) = xfit(p);
        ypoly(c2) = yfit(p);
        c2 = c2+1;
    end
end

n = 200;

% Poly fit
p = polyfit(xpoly, ypoly, 2);
x = linspace(min(Cl), max(Cl), n);
y = polyval(p, x);
dep = min(y)-min(yfit);

% Add in the drag drop
for o = 1:length(y)
    if x(o) > range(1) && x(o) < range(2)
        y(o) = y(o)-dep;
    end
end


```

```
%Cl = x;
% standard atmospheric conditons from input flight altitude and speed

r = 1;
while x(r) < Cl
    r = r+1;
end

Cd0 = y(r);

AR = b^2/S;

Cdi = Cl.^2./(pi*AR*e);
```

```
function [Cl, Cd0, Cdi] = tailDrag(t, c, e, S, b)

%==========================================================================
% INPUTS
% t = Maximum Thickness in %
% c = Design Cl multiplied by 10
% e = oswald efficiency
% S = wing planform area?
% b = wing span
% W = aircraft weight (which one?)
% speeds = vector of speeds to find Cd at
% alt = operating altitude
% t and c combinations are only found for combination lying near availible
% data
%==========================================================================

% Load in All the TOWS Data
load('NACA6data.mat');
NACA63 = rmfield(NACA63, "N6210");
```

```

fn1 = fieldnames(NACA63);
c1 = 1;
c2 = 1;

%Constant for how much to drop the function and how wide the drop it
width = [6,9,12,15,18,21; 0.15, 0.28, 0.3, 0.44, 0.5, 0.6]';  %Thickness
    Dependant

%Find all the airfoils within the range of thicknesses (+/- 2%)
for i = 1:length(fn1)
    if NACA63.(fn1{i}).thick >= t-2 && NACA63.(fn1{i}).thick <= t+2
        x(c1) = NACA63.(fn1{i});
        c1 = c1+1;
    end
end

%Out of those airfoils find the ones within the right designed Cl (+/- 0.2)
for j = 1:length(x)
    if x(j).camb >= c-1 && x(j).camb <= c+1
        y(c2) = x(j);
        c2 = c2+1;
    end
end

%Make a data table of all the data from the selected airfoils
for k = 1:length(y)
    fd(:,k) = y(k).data{:,2};
end

numFoils = size(fd);

```

```
%Depending on how many airfoils are found, interpolate or not
if numFoils(2) == 1

    yfit = fd;
    xfit = y.data{:,1};

elseif numFoils(2) == 2

    % Find the weight to put on the interpolation
    if y(1).thick == y(2).thick

        wt = 1/((y(1).camb+y(2).camb)/c);

    elseif y(1).camb == y(2).camb

        wt = 1/((y(1).thick+y(2).thick)/c);

    end

    xfit = y(1).data{:,1};
    yfit = iterpol(fd, wt, 0)';

elseif numFoils(2) == 3

    xfit = y(2).data{:,1};
    yfit = y(2).data{:,2};

elseif numFoils(2) == 4

    if y(1).thick == y(2).thick

        wt1 = 1/((y(1).camb+y(2).camb)/c);

    elseif y(1).camb == y(2).camb

        wt1 = 1/((y(1).thick+y(2).thick)/c);

    end

    if y(3).thick == y(4).thick

        wt2 = 1/((y(3).camb+y(4).camb)/c);

    elseif y(3).camb == y(4).camb

        wt2 = 1/((y(3).thick+y(4).thick)/c);

    end

    y1 = iterpol(fd(:,1:2), wt1, 0)';
    y2 = iterpol(fd(:,3:4), wt2, 0)';
    ydat = [y1, y2];
    wt = mean([wt1,wt2]);
    yfit = iterpol(ydat, wt, 0)';
    xfit = y(1).data{:,1};

end

%Interpolate the width and depth
wid = iterpol(width, t, 1);
range = [(c/10)-wid (c/10)+wid];

c2 = 1;

```

```
% Sort Data for NaN and for the width and dpeth
for p = 1:length(yfit)
    if (xfit(p) < range(1) || xfit(p) > range(2)) && isnan(yfit(p)) == 0
        xpoly(c2) = xfit(p);
        ypoly(c2) = yfit(p);
        c2 = c2+1;
    end
end

% Poly fit
p = polyfit(xpoly, ypoly, 2);
x = linspace(-1.3, 1.3, 50);
y = polyval(p, x);
dep = min(y)-min(yfit);

% Add in the drag drop
for o = 1:length(y)
    if x(o) > range(1) && x(o) < range(2)
        y(o) = y(o)-dep;
    end
end

Cl = 0;

r = 1;

while x(r) < Cl
    r = r+1;
end

Cd0 = y(r);

```

```
AR = b^2/S;

Cdi = Cl.^2./(pi*AR*e);
```

```
function [Cd0, Swet] = nacelleDrag(L, D, Cond, n)

%==========================================================================
% INPUTS
% l array of lengths of the equivalent bodies of revolution (in ft)
% d largest diameter of the fuselage (in ft)
% Cond flight conditions [alt(ft), KTAS]

% ASSUMES two conical ends with a cylindrical center shape
%==========================================================================

if n == 0
    Cd0 = 0;
    Swet = 0;
    return
end

[Tstd, ~, rho, ~] = standatm(Cond(1), 0, "IMP");

T = Tstd*1.8;

f = sum(L)/D;

FF = 1 + (0.35/f);

x0 = 0.374*sum(L) + 0.533*L(1); %ft

mu = 3.17*10^(-11)*T^1.5*(734.7/(T+216)); %lb*s/ft^2

```

```
ReF = (rho*Cond(2)*1.688*sum(L))/mu;

k = 30*10^-6*3.281; %ft

ReC = 38.21*(sum(L)/k)^1.053;

if ReF > ReC
    Re = ReC;
else
    Re = ReF;
end

xtr = (((x0/sum(L))*(1/Re)^-0.375)/36.9)^(1/0.625); %ft

Cf = (0.074/Re^0.2)*(1-((xtr-x0)/sum(L)))^0.8;

Swet = pi*D/4*((2*sqrt(L(3)^2+D^2/4))-D+4*L(2)+2*sqrt(L(3)^2+D^2/4));

Scom(1) = pi*D/2*sqrt(L(3)^2+D^2/4);
Scom(2) = pi*D*L(2);
Scom(3) = pi*D/2*sqrt(L(3)^2+D^2/4);

Cd0A = (Cf*FF*Scom(1))/Swet;
Cd0B = (Cf*Scom(2))/Swet;
Cd0C = (Cf*FF*Scom(3))/Swet;

Cd0 = Cd0A+Cd0B+Cd0C;
```

```
function [Cd0] = interferenceDrag(Cd0fn, nNac, tratio, c, nc, S)

%=========================================================================
% INPUTS
% Cd0fn - [Cd0 of Fuselage, Cd0 of Nacelle]
```

```matlab
% nNac - number of nacelles
% tratio - [Thickness ratio of Horizontal, Thickness Ratio of Vertical]
% c - [Chord of Horizontal, Chord of Vertical]
% nc - [Number of Corners Horizontal, Number of Corner Vertical]
% S - [Reference Area of Horizontal, Reference Area of Vertical]
%=======================================================================

Cdfuse = 0.05*Cd0fn(1);

CdNac = 0.05*Cd0fn(2)*nNac;

Cdhs = nc(1)*(0.8*tratio(1)^3 - 0.0005)*(c(1)^2/S(1));

Cdvs = nc(2)*(0.8*tratio(2)^3 - 0.0005)*(c(2)^2/S(2));

Cd0 = Cdfuse+CdNac+Cdhs+Cdvs;
```

```matlab
function [Tstd, P, rhostd, rho] = standatm(alt, T, unit)

if unit == "IMP" %Temperature in K, Altitude in Feet

    if 0<=alt && alt<=36089.2

        Tstd = 288.15-1.9812e-3*alt; %K
        P = (4.2927085-29.514885e-6*alt)^5.2558797; %lbf/ft^2
        rhostd = (0.24179285 - 1.6624675e-6*alt)^4.2558797; %slug/ft^3
        rho = rhostd/(1 + ((T - Tstd)/Tstd)); %slug/ft^3

    elseif 36089.2 < alt && alt <= 65616.8

        Tstd = 216.65; %K
        P = 2678.442*exp(-48.063462e-6*alt); %lbf/ft^2
        rhostd = 4.0012122e-3*exp(-48.063462e-6*alt); %slug/ft^3
        rho = rhostd/(1 + ((T - Tstd)/Tstd)); %slug/ft^3

    else

        Tstd = "Error";
        P = "Error";
        rho = "Error";

    end

elseif unit == "MET" %Temperature in K, Altitude in Meters

    if 0 <= alt && alt<=11000

        Tstd = 288.15 - 6.5e-3*alt; %K
        P = (8.9619638-0.20216125e-3*alt)^5.2558797; %N/m^2
        rhostd = (1.04884 - 23.659414e-6*alt)^4.2558797; %kg/m^3
        rho = rhostd/(1 + ((T - Tstd)/T)); %kg/m^3

    elseif 11000<alt && alt<=20000

        Tstd = 216.65; %K
        P = 128244.5*exp(-0.15768852e-3*alt); %N/m^2
        rhostd = 2.06214*exp(-0.15768852e-3*alt); %kg/m^3
        rho = rhostd/(1 + ((T - Tstd)/T)); %kg/m^3

    else

        Tstd = "Error";
        P = "Error";
        rho = "Error";

    end
else

    Tstd = "Error";
    P = "Error";
    rho = "Error";

end
```

```
function [xdata] = iterpol(mat, wt, flag)

% Flag = 0 need a weight
% Flag = 1 have a value to interpolate to

if flag == 0
    for i = 1:length(mat)

        y1 = mat(i,2);
        y0 = mat(i,1);

        xdata(i) = wt * y1 + (1-wt) * y0;

    end

elseif flag == 1
    c=2;
    while mat(c,1) < wt
        c = c+1;
    end

    y1 = mat(c,2);
    y0 = mat(c-1,2);
    x1 = mat(c,1);
    x0 = mat(c-1,1);

    xdata = (((y1-y0)/(x1-x0))*(wt-x0)) + y0;
end
```

References

[1] Abbott, I. H., and Von Doenhoff, A. E., *Theory of Wing Sections*, 2nd ed., Dover Publications, New York, 1959, Chap. appx. IV.

[2] Mohammed, N., "Important Terminology - Aircraft Technical," , 2014. URL `https://aviationclass.wordpress.com/2014/12/22/importants-terminologies-aircraft-technical/`.

[3] Gudmundsson, S., *General Aviation Aircraft Design: Applied Methods and Procedures*, Elsevier Science Technology, Oxford, 2014, Chaps. 12, 15.

[4] Iscold, P., "Aerodynamic Calculations," , 2020. URL `https://lor.instructure.com/resources/0abd9c028fff4d2b829780a17f277bc3`.

[5] Boschetti, P., Amerio, A., and Cárdenas, E., "Aerodynamic Performance as a Function of Local Twist in an Unmanned Airplane," 2009, pp. 2, 7. https://doi.org/10.2514/6.2009-1481.

[6] Bikle, P., "Sailplane Performance Measured in Flight," , 1970. URL https://journals.sfu.ca/ts/index.php/ts/article/view/1173/1122, 12th OSTIV Congress.