AE6343-A/Q Fixed Wing Design I Fall 2024 Project Part 1

Adam P. Benabou

The document describes the creation of a sizing and synthesis aircraft design tool in Python. The tool was built using primarily equations and references from the course materials and a selection of aircraft design textbooks [1, 2]. The tool is comprised of two principal components. The initial phase, constraint analysis, entails identifying all constraints that may impact the aircraft and its certification process. The second component, designated "Mission Analysis," entails the calculation of the aircraft's weight at each phase of its operational profile. The objective of this tool is to identify a viable aircraft design that adheres to all constraints. The user may enter inputs into designated files and run the tool using Python. Despite the tool's parametric design, a comprehensive example (A320neo) is provided and will be utilized as a reference throughout this report. The results demonstrate that the aircraft designed by the tool exhibits a comparable maximum takeoff weight and thrust-to-weight ratio to the A320neo. However, the wing loading values exhibit slight discrepancies, resulting in a difference in the calculated wing area.

Nomenclature

W_{TO}	=	Takeoff Weight (lbf)
W_{ramp}	=	Ramp Weight (lbf)
W_P	=	Payload Weight (lbf)
W_E	=	Empty Weight (lbf)
W_C	=	Crew Weight (lbf)
W_f	=	Fuel weight (lbf)
β_i	=	Weight fraction at Mission Phase i
WSR	=	Wing Loading ($\mathrm{lbf/ft^2}$)
EAS	=	Equivalent Airspeed (kts)
TAS	=	True Airspeed (kts)
ROC	=	Rate of climb (ft/min)
TSFC	=	Thrust-specific fuel consumption ($\mathrm{lb/(lbf * h)}$)
α	=	refers to the thrust lapse at a given phase

β	=	refers to the weight fraction at a given phase
γ	=	flight path angle (deg)

Summary

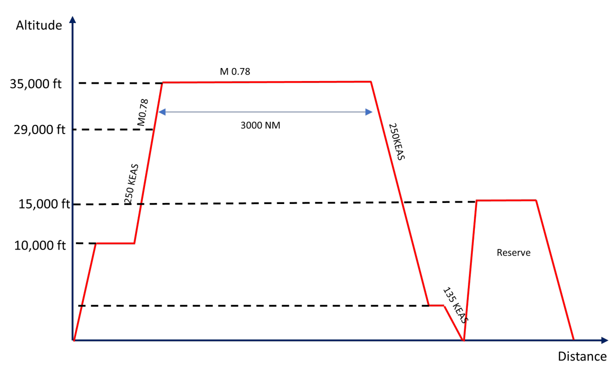
I. Introduction

THIS document aims to summarize the work done for the Aircraft Design Project Part 1. The tool is available **here*** on github † The goal of this first part is to create a sizing and synthesis tool to design an aircraft while following specific requirements. The tool has been designed using Python to size an Airbus A320 but requirements can be modified by the user to design other legacy commercial aircraft (see below to further explanation on the user interface). The tool will design an aircraft by computing a Constraint and Mission Analysis. Table 1 and Figure 1 summarize the top level requirements of the Aircraft as well as the Mission profile used to size the Aircraft. The complete Mission profile can be found in the appendix IX.C *It should be noted that ramp weight is used during the sizing process and not takeoff weight, Therefore, W_{TO} refers to the weight before taxi and not the weight before takeoff. It has been assumed that before taxi, $\beta = 1$*

Table 1 Top level requirements of the Aircraft

NPAX	150
Cruise Mach	0.78
Cruise range	3000 NM

Fig. 1 Mission Profile of the Aircraft



II. Formulation of Sizing and Synthesis Approach

In order to design our Aircraft, the process has been divided into main part. Constraint Analysis and Mission Analysis.

*The tool was a private repository until the submission deadline

†Readme File is write in Markdown, It is therefore easier to use github to read it. However, It is totally understandable in a text file

Step 1 : Convergence of the weight fractions

First, a Thrust to Weight Ratio ($\frac{T_{SL}}{W_{TO}}$) and a Wing Loading ($\frac{W_{TO}}{S}$) value will be assumed. From this point , The Mission Analysis will be conducted and will return values of weight fractions at each phase of the flight (value at the beginning of the phase) $\beta_i = \frac{W_i}{W_{ramp}}$. *It should be noted that ramp weight is used during the sizing process and not takeoff weight, Therefore, from now on,* W_{TO} *actually refers to the weight before taxi and not the weight before takeoff (i.e on the runway). Thus, before taxi,* $\beta = 1$. It will be mentioned clearly if W_{TO} is corresponding to the takeoff weight

Once these values are obtained, constraint analysis is performed to obtain new values for the thrust to weight ratio and wing loading, which are re-injected into the mission analysis. The process is considered complete when the differences between the beta values for each segment are below a given tolerance. This will provide a converged design point ($\frac{T_{SL}}{W_{TO}}$ and $\frac{W_{TO}}{S}$) as well as a final weight fraction.

Step 2 : Computation of the Takeoff Weight

The next step is to calculate the Takeoff Weight. To do so, we need to breakdown the weights of the Aircraft

$$W_{\mathrm{TO}} = W_{\mathrm{E}} + W_{\mathrm{f}} + W_{\mathrm{C}} + W_{\mathrm{PL}} \tag{1}$$

The crew weight is a given requirement from the user. For this example, there are 2 pilots and 3 flight attendants with their baggage, totaling 1050 lbs Payload weight is also requirements. User can specify the number of passengers Npax. Assuming that each passenger weights 190 lbs and carries a bag weighing 30lbs such as

$W_P = \text{Npax} * (\text{Bag Weight} + \text{Passenger Weight})$

Empty weight can be computed using historical data An empirical empty weight fraction approximation created for the A320neo is provided below

$$\Gamma = \frac{W_E}{WTO} = k_{WE} * W_{TO}^{-0.06} \tag{2}$$

where $W_{TO}$is maximum ramp weight in lbs. In Mattingly [1], $K_{WE} = 1.02$, but in this project, K_{WE} has been set to 1.15 for more accurate results. Step 1 has provided $\beta_{\text{final}} = \frac{W_{\text{final}}}{W_{TO}}$ which is the Weight fraction at the end of the flight such as $\frac{W_{\mathrm{f}}}{W_{TO}} = 1.06(1 - \beta_{\text{final}})$. 1.06 factor comes from the fact that they are some fuel reserves in the aircraft that are not used during the flight , they represents usually 6% of the Zero Fuel Weight

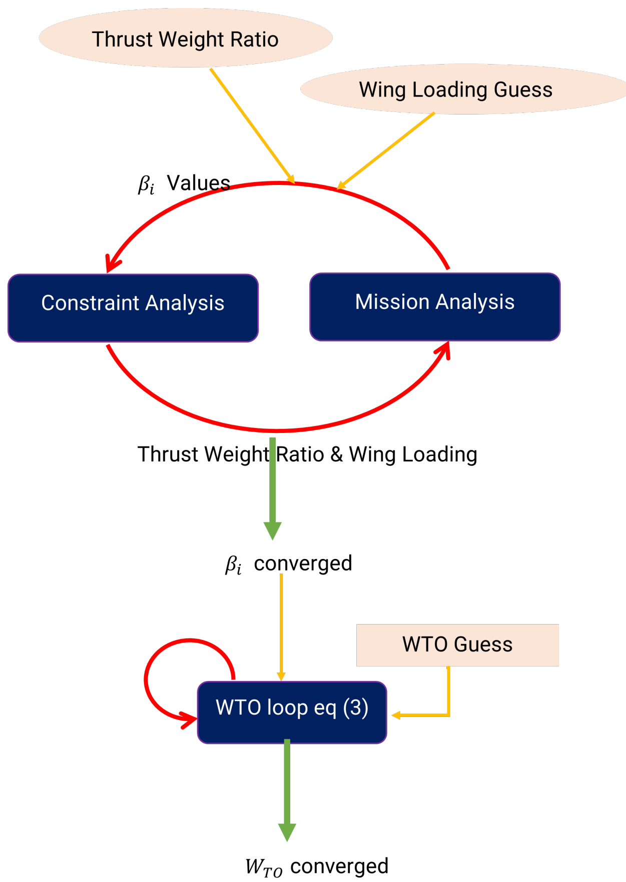
$$W_{TO} = \frac{W_C + W_P}{1 - 1.06\,(1 - \beta_{\text{final}}) - k_{WE} * W_{TO}^{-0.06}} \tag{3}$$

A first W_{TO} is guessed and injected into this equation. After a few iteration of equation 3, a final W_{TO} is computed

Step 3 : Computation of other design variables

Once the gross weight is computed, others design variables can be computed such as the sea level thrust or the wing area

Fig. 2 Approach of the Sizing and Synthetis tool



III. Description of the tool

This section aims to describe how the tool works and how it is structured. The goal of this section is not to give a physical explanation of the methods (that will be done in the next sections V and VI), but to describe how these functions work together. The installation process of the tool can be found in the README file and will not be discussed here.

A. Tool structure

The tool is structured in different sub folders, each one having a specific purpose (see figure 3)

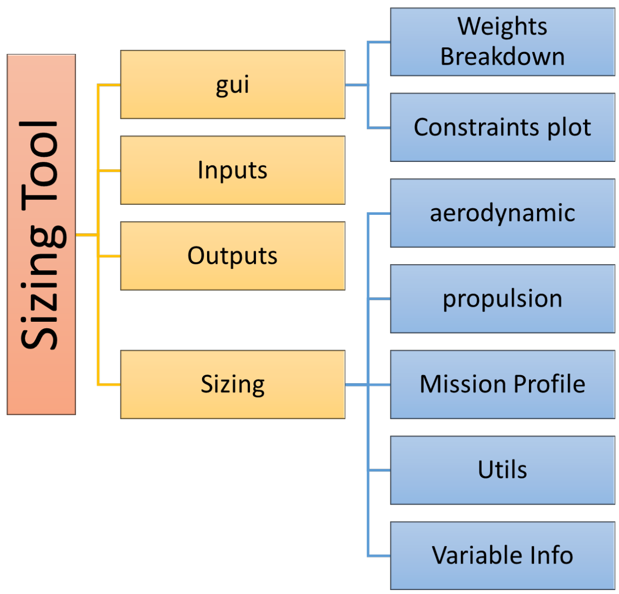
Inputs: Folder where the user can enter its requirements and mission profile in data format files (json file)

Outputs After the tool is run, all outputs are generated in html files that are located in this folder. User can access them by opening these files in his folder

gui: Graphical User Interface folder. This folder contains methods used to plot results of the tool once the aircraft is designed (Weight Breakdown , weights during each phases, $\frac{T_{SL}}{W_{TO}}$ and $\frac{W_{TO}}{S}$, constraints plot ...).It contains three main files. The first is used to plot all the graphs related to the weights (weight breakdown and weight fractions for each phase). The second is used to plot the constraints and the instantaneous T/W and W/S during the mission. The third one is used to plot aerodynamics and propulsion characteristics (tsfc, lift to drag ratios, lift coefficients, Thrust available ...)

Sizing: This is the main folder where the motions equations are to compute Mission and Constraint Analysis.

Fig. 3 Tool tree diagram



B. Sizing Folder

Because the Sizing folder is the main one of the tool. We will look at it in much detail. This folder is also decomposed in subfolders.

1. Assumptions : Aerodynamic and Propulsion

First, **aerodynamic and propulsion assumptions** are located in this folder in the Aerodynamic and Propulsion subdirectory respectively. These assumptions are necessary for the computation of the constraints and the mission fuel weight. These assumptions can be find in the next section V

2. Utils sub folder

These folder contains all the function used for the tool that are not related directly to the design of the aircraft and that are used in every file. For example, this includes every functions used to convert units. Specifically, this folder contains also functions which convert different speed. As the matter of fact, in aviation , they are several speeds that are used and they don't mean the same thing. For instance, the mission requirements(1) refers to Equivalent Airspeed (EAS) which is not the true Airspeed of the aircraft. EAS is the airspeed at sea level in the International Standard Atmosphere at which the dynamic pressure is the same as the dynamic pressure at the true airspeed (TAS) and altitude at which the aircraft is flying[2].

$$EAS = TAS * \sqrt{\frac{\rho}{\rho_0}} = a_0 * M * \sqrt{\frac{P}{P_0}} \tag{4}$$

where ρ is the actual air density and ρ_0 is the standard sea level density (1.225 kg/m3 or 0.00237 slug/ft3). $\frac{P}{P_0}$ is the pressure ratio at a given altitude. M the Maxh number and a_0 the speed of the sound a sea level

```python
def KEAS_to_TAS(KEAS, altitude, meter=False):
    atm = Atmosphere(altitude, meter)
    return KEAS / np.sqrt(atm.density_ratio.value)
```

Listing 1 Example of a function used to convert speed

One can notice that this function has the keyword Atmosphere. Indeed, the utils folder contains also a atmosphere class used to provide every useful variable at a given altitude (Density, Pressure, Temperature). This class has been created using a external package named USSA1976 [3] which is based on the US standard Atmosphere model provided by the National Aeronautics and Space Administration [4]

```python
import ussa
class Atmosphere:
    def __init__(self, altitude, meter=False):
        self.meter = meter
        self.altitude = altitude

    @property
    def temperature(self):
        """
        This function calculates the temperature of the atmosphere based on the altitude.
        """
        temperature = ussa.compute(
            variables=["t"],
            z=np.array([self.altitude.value]),
        )
        return Variable(
            "temperature", temperature["t"].values, "K", "Temperature of the atmosphere"
        )
```

Listing 2 A snippet of the implementation of the atmosphere class

3. Variable info subfolder

This subfolder contains a Variable class that is used to represent variables throughout the tool. Each variable is an object with a name, a value unit and a description. This helps to understand the code. The variables.py files provide some variables for the aircraft that will be used as final variables. The use of the class is only to structure the code and make it easier to search for a variable. In the example below, TOW is a variable object whose name attribute is "TOW", value is 1, unit is lbf and description is "Aircraft ramp weight". Value of the variable doesn't matter as it is updated at the end of the design process.

```python
from Variable_info import variable

class Aircraft:
    class Design:
        TOW = Variable(name="TOW", value=1, unit="lbf", description="Ramp Weight")
        WING_LOADING = Variable(
            "wing_loading",
            value=1,
            unit="lbf/ft^2",
            description="Wing loading of the aircraft",
        )
```

Listing 3 Snippet of the Aircraft class

4. Mission profile subfolder

The user should be aware that a segment is not a phase. In fact, the mission profile provided 1 contains 18 phases but less segments. Each phase is represented as a segment, but some phases are represented by the same segment but with different attributes (e.g. phase 4 and 5, see appendixIX.C to see the phases number)

Each segment is represented by an abstract class called segments (even if it's plural, each class segments represents one segment...). Each segment has common attributes like weight fraction (at the beginning the segment), name, phase number. Also each segment has abstract methods like wf_wi which calculates the weight fraction from this segment only $\frac{W_{end}}{W_{start}}$ (used for mission analysis) or Thrust_Weight_Ratio(WSR) which calculates the thrust to weight ratio for the segment (used for constraint analysis). Thanks to this structure, the main loop will only consist of computing weight fractions and constraints for each phase represented by a segment (a for loop in a list of *segments* objects)

```python
class segments:
    def __init__(self, type, phase_number, weight_fraction=1, name=None):
        self.type = type #Type of segment, climb, takeoff, landing
        self.phase_number = phase_number
```

```python
        self.weight_fraction = Variable(
            "weight_fraction",
            weight_fraction,
            "",
            "Weight fraction (beta) at the end of the segment",
        )  # Weight fraction (beta)
        self.name = name

    @abstractmethod
    def wf_wi(self, wing_loading, TWR):
        print("error, this method should be implemented in the subclass")
        return 1

    @abstractmethod
    def Thrust_Weight_Ratio(self, WSR):
        print("error, this method should be implemented in the subclass")
        pass
```

Listing 4   snippet of the segments class

Then a subclass was created for specific segments with specific attributes. This is where the abstract methods seen earlier are implemented. The Segments folder contains a file for each specific segment. Currently there are seven segment types available.

- Taxi
- Takeoff
- climb (or descent if Rate of Climb of flight path angle is negative)
- acceleration (or deceleration if start speed is less than end speed)
- cruise
- loiter (in the cruise.py file)
- approach
- landing

The user can add, modify or remove any segments to create its own mission profile in the Inputs folder in the *Mission_Profile.json* file

5. Mission_analysis folder

This is where the Mission analysis is computed. For each phase, weight fraction is computed. A more detail explanation of this file is given in the sectionV

6. Constraint Analysis folder

For each phase, the thrust to weight ratio is calculated (if relevant) and a design point (TWR,WSR) is found that minimizes the thrust to weight ratio while still being in the feasible design space. One may notice that there is also an additional file called Additional Constraint in this folder. This file is used to calculate the additional constraints specified in the project requirements. For now, the user cannot easily change them, as they must go into this file to change the parameter. A future enhancement of the tool could be the implementation of an input file for these additional constraints.

To conclude this section, the following table represents which segment type is associated to which phase for the given mission profile (see IX.C)

Table 2 Caption

Phase number	Segment type in the Python Tool
1	Taxi
2	Takeoff
3	Climb
4	Acceleration
5	Climb
6	Climb
7	Cruise
8	Climb (ROC < 0)
9	Acceleration (Start Speed < End speed)
10	Approach
11	Climb
12	Cruise
13	Loiter
14	Climb (ROC < 0)
15	Acceleration (Start Speed < End speed)
16	Approach
17	Landing
18	Taxi

IV. Assumptions

A. Propulsions

In order to compute the Mission and the constraints analysis. Several assumptions have been made.

The engine model provided below approximates the performance of the actual engine of the A320 WV055: PW 1100G-JM as described in the A320 Airport Planning Document and the PW1100G-JM Series Engines Type Certificate Data Sheet[5]. Some helpful equations include:

1) **Thrust lapse (from FWD lecture notes):**

$$\alpha = \left[0.568 + 0.25(1.2 - M_\infty)^3\right] \sigma^{0.6} \tag{5}$$

where σ is the ambient density ratio relative to standard sea level condition.

2) **TSFC (from Mattingly[1] Equation 3.54):**

$$TSFC = k_{TSFC}(0.45 + 0.54M_\infty)\sqrt{\theta} \tag{6}$$

where θ is the ambient temperature ratio relative to standard sea level condition. k_{TSFC} is a technology factor for fuel flow applied on Mattingly's equation (refer to the footnote). In this project, k_{TSFC} can be assumed at 0.64.

These equations are located in the propulsion folder in Sizing

B. Aerodynamics

Using the provided equations from [1]. The following equations can be used for the Aerodynamics

1)

$$C_D = C_{D_0} + K_1 C_L^2 + K_2 C_L \tag{7}$$

Where $K_1 = 0.0556$ and $K_2 = -0.0197$

2)

$$C_{D_0} = 0.0311 \left(\frac{1}{\sqrt{1 - M_\infty^2}} - 1.273 \right)^2 - \frac{0.0027}{\sqrt{1 - M_\infty^2}} + 7.86 \times 10^{-8} \cdot h + 0.0215 \tag{8}$$

where h is altitude.

C. Takeoff and Landing

For takeoff and Landing, the following assumptions have been made using mainly sources from Mattingly[1] and from A320 open sources data[6]

- Takeoff configuration maximum Cl $C_{L_{TO}} = 2.56$
- C_{DR} due to non-clean configuration $C_{DR} = 0.07$
- Takeoff speed safety factor $K_{TO} = 1.2$
- Ground roll friction coefficient $\mu = 0.05$
- Landing configuration maximum Cl $C_{L_{Land}} = 3$
- Approach speed safety factor $k_{LD} = 1.3$

V. Mission Analysis

This section details how the mission analysis was calculated for each segment (not each phase, as some phases use the same function). To see what method has been applied for each phase, refer to the table 2 above. For each phase i , the goal is to compute the following weight fraction $\Pi_i = \frac{W_{Start}}{W_{End}}$

A. Taxi

For this segment , It has been assumed that 10% of takeoff full power fuel flow is used for 20 minutes. Fuel flow is defined such as

$$\frac{dW}{dt} = -TSFC * T \tag{9}$$

Assuming that $T_{Taxi} = k_{taxi} * T$ where $k_{taxi} = 10\%$, (9) can be rewritten as

$$\frac{dW}{W} = -\frac{\alpha}{\beta} TSFC * k_{taxi} \frac{T_{SL}}{W_{TO}} dt = A * dt \tag{10}$$

where α is the Thrust Lapse during taxi and β the weight fraction a the begining of the taxi i.e $\beta = 1$. During Taxi, we can assume that A is almost constant and close to zero (almost no thrust is used). Therefore, by integrating (10)

$$\Pi_{Taxi} = 1 - \frac{\alpha}{\beta} TSFC(V, h) * k_{taxi} \frac{T_{SL}}{W_{TO}} \tag{11}$$

where TSFC can be computed using (6) by assuming that taxi speed V is 15 knots and altitude is 0

B. Takeoff

Takeoff segment will be divided into two segment, the acceleration and the rotation.

During acceleration, specific Power P_s is no longer zero. Using that $\frac{T}{W} dt = \frac{T ds}{WV} = \frac{d\left(h + V^2/2g_0\right)}{1-u}$ where $u = \frac{D+R}{T}$ This results to the type A equation

$$\frac{dW}{W} = -\frac{TSFC}{V(1-u)} * d\left(h + \frac{V^2}{2g_0}\right) \tag{12}$$

This equation will be used for several segments.

For Takeoff acceleration , $\frac{dh}{dt} = 0$. u can be calculated using Mattingly[1]

$$u = \frac{D+R}{T} = \frac{qS * (C_D + C_{DR} - \mu_{TO}) + \mu_{TO} * \beta * W_{TO}}{\alpha T_{SL}} = \left[\xi_{TO}\left(\frac{q}{\beta}\right)\left(\frac{S}{W_{TO}}\right) + \mu_{TO}\right] \frac{\beta}{\alpha}\left(\frac{W_{TO}}{T_{SL}}\right) \tag{13}$$

Where $\xi_{TO} = (C_D + C_{DR} - \mu_{TO})$. Furthermore, $\left(\frac{q}{\beta}\right)\left(\frac{S}{W_{TO}}\right) = \frac{1}{Cl_{TO}} = \frac{k_{TO}^2}{Cl_{max}}$ Therefore, the following equation is used for takeoff acceleration:

$$\mathbf{\Pi = exp}\left(-\frac{\boldsymbol{TSFC}}{\boldsymbol{g_0}} * \left[\frac{\boldsymbol{V_{T0}}}{\mathbf{1}-\boldsymbol{u}}\right]\right) \tag{14}$$

where V_{TO} can be found using lift equation $L = W$ such as

$$V_{TO} = k_{TO} * \sqrt{\frac{2}{\rho Cl_{max}} * \frac{W_{TO}}{S} * \beta} \tag{15}$$

and

$$u = \left[\xi_{TO}\frac{k_{TO}^2}{Cl_{max}} + \mu_{TO}\right]\frac{\beta}{\alpha}\left(\frac{W_{TO}}{T_{SL}}\right) \tag{16}$$

C. Climb

For the climb segment, R can be assumed zero as the aircraft is not on the ground anymore, then equation (12) can be used and $u = \frac{D}{T} = \frac{C_D}{C_L}\frac{\beta}{\alpha} * \frac{W_{TO}}{T_{SL}}$ (it has been assumed that the aircraft is climbing with a flight path angle $\gamma \simeq 0$ such as $\cos\gamma = 1$ C_L is determined using lift equation as speed climb is a given parameter

$$C_L = \frac{\beta}{0.5 * \rho_0 * EAS^2} * \frac{W_{TO}}{S} \tag{17}$$

Cd can be determined using (7) and (8). It should be noticed that some variables are not constant during the climb. Therefore, this segment has been divided into smaller segments where we can assume that variables such as C_D $TSFC$ and u are constant. We can then integrate (12)

$$\Pi = \Pi_{0ft}^{100ft} * \Pi_{100ft}^{200ft} \cdots \tag{18}$$

where in each subsegment:

$$\Pi = \exp\left[\frac{TSFC}{V(1-u)} * \Delta\left(h + \frac{V^2}{2g_0}\right)\right] \tag{19}$$

The subdivision is done in the Compute_Beta_Climb method

D. acceleration

For acceleration, using a similar process as climb, we can use equation(19) where $\Delta(h) = 0$. Acceleration is also subdivided into smaller segments as $TSFC$ and u are not constant during this phase. The subdivision is done in the Compute_Beta_Approach method

E. cruise

For cruise, $P_s = 0$. Speed and altitude are constant. Therefore, eq (9) can be used knowing that $dt = ds/V$

$$\frac{dW}{W} = -TSFC\frac{T}{V * W}ds \tag{20}$$

where $T = D$ and $L = W$. Therefore,

$$\frac{dW}{W} = -\frac{TSFC}{V}\frac{C_D}{C_L}ds \tag{21}$$

C_L and C_D can determined using equation (17) and (7). Cruise is also subdivided into smaller steps as C_L varies through the segment (because β does). We can then integrate (21)

$$\Pi = \Pi_{0\text{NM}}^{100\text{NM}} * \Pi_{100\text{NM}}^{200\text{NM}} \cdots \tag{22}$$

where in each subsegment:

$$\Pi = \exp\left[\frac{TSFC}{V}\frac{C_D}{C_L}\Delta s\right] \tag{23}$$

The subdivision is done in the Compute_Beta_Cruise method

F. Loiter

Loiter is a cruise segment, the only difference will be the speed parameter. Mission requirements mentioned that speed during loiter should be the best endurance speed which is the speed which maximizes lift to drag ratio for a jet aircraft. Therefore, we are trying to minimize $\frac{L}{D} = \frac{C_L}{C_D}$. By solving $\frac{d}{dC_L}\left(\frac{C_L}{C_D}\right) = 0$. Best Endurance speed is achieved for $\frac{C_L}{C_D} = \sqrt{4C_{D_0}K_1} + K_2$. Therefore, weight fraction is for this segment is

$$\Pi = \exp\left[-TSFC\frac{C_L}{C_D}\Delta t\right] = \exp\left[\left(-TSFC\sqrt{4C_{D_0}K_1} + K_2\right)\Delta t\right] \tag{24}$$

G. Descent,Deceleration and Landing

for theses segments, we assume that there is no fuel flow, therefore $\Pi_{\text{Descent}} = \Pi_{\text{Deceleration}} = 1$

H. Approach

Approach is done at a given flight path and constant speed. Therefore, the time of the phase can be estimated using that $\frac{dh}{dt} = V * sin(\gamma)$. As for the climb, we subdivide this segment into smaller steps and then using equation (24)

$$\Pi = \exp\left[-TSFC\frac{C_L}{C_D}\frac{\Delta h}{V \sin\gamma}\right] \tag{25}$$

Once the mission analysis has provided values for the weight fractions at each stage, we can use these values to calculate the constraint analysis.

VI. Constraint Analysis

For the Constraint Analysis, the Master equation has been used. Proof of this equation can be found in Mattingly [1]

$$\frac{T_{SL}}{W_{TO}} = \frac{\beta}{\alpha}\left\{\frac{qS}{\beta W_{TO}}\left[K_1\left(\frac{n\beta W_{TO}}{qS}\right)^2 + K_2\left(\frac{n\beta W_{TO}}{qS}\right) + C_{D_o} + \frac{R}{qS}\right] + \frac{1}{V}\frac{d}{dt}\left(h + \frac{V^2}{2g_o}\right)\right\} \tag{26}$$

A. Taxi

It has been assumed that no constraint is required for this segment as Thrust required is almost null for taxi

B. Takeoff

For this segment, it has been assumed that thrust overcomes drag during the roll (which can be discussed in this situation, but it simplifies the equation). The takeoff was divided into three phases.

- Ground roll distance: s_g
- Rotation distance s_{tr}
- Clearance of an obstacle during Transition s_{obs}

Total takeoff distance is $s_{TO} = s_g + s_{tr} + s_{obs}$

1. Ground Roll

. For this phase, It is assumed that $T >> D$. Therefore, The master equation can be simplified after eliminating the drag related terms and rearranging the acceleration term, equation (26) is reduced to:

$$\frac{T_{SL}}{W_{TO}} = \frac{\beta}{\alpha g_o}\frac{dV}{dt} = \frac{\beta}{\alpha g_o}\frac{dV}{ds/V} \tag{27}$$

Then by integrating this equation between 0 and s_g and using (15)

$$\frac{T_{SL}}{W_{TO}} = \frac{\beta^2}{\alpha}\frac{k_{TO}^2}{s_G \rho g_o C_{L_{\max}}}\left(\frac{W_{TO}}{S}\right) \tag{28}$$

2. Rotation

Rotation distance can be expressed as

$$s_r = t_r * V_{TO} \tag{29}$$

where t_r is the rotation time and has been assumed to be 3s

3. Obstacle clearance

s_{obs} is the distance from the end of rotation to the point where the height h_{obs} is attained

$$s_{obs} = R_c * \sin\theta_{obs} = \frac{V_{TO}^2 \sin\theta_{obs}}{g_0 * (0.8k_{TO} - 1)} \tag{30}$$

where $\theta_{obs} = \cos^{-1}\left(1 - \frac{h_{obs}}{R_C}\right)$

From those three distances, we can isolate $\frac{T_{SL}}{W_{TO}}$ to express it as function of $\frac{W_{TO}}{S}$

C. climb

We assumed for this section that we climb at a constant speed (KEAS speed) such as $\frac{dV}{dt} = 0$ and n=1 (Lift approximately equals Weight). The Master equation (26) can be simplified such as:

$$\frac{T_{SL}}{W_{TO}} = \frac{\beta}{\alpha}\left\{K_1\left(\frac{\beta W_{TO}}{qS}\right) + K_2 + \frac{C_{D_0}}{\frac{\beta}{q}\frac{W_{TO}}{S}} + \frac{1}{V}\frac{dh}{dt}\right\} \tag{31}$$

Some variables such as C_{D_0}or α actually vary during these phases. The average value between starting and ending point of the climbing will be taken

D. Acceleration

For this segment, we assumed that laod factor is almost 1 and that $\frac{dh}{dt} = 0$. From Master Equation (26), we obtained

$$\frac{T_{SL}}{W_{TO}} = \frac{\beta}{\alpha}\left\{K_1\left(\frac{\beta W_{TO}}{qS}\right) + K_2 + \frac{C_{D_0}}{\frac{\beta}{q}\frac{W_{TO}}{S}} + \frac{1}{g_0}\frac{dV}{dt}\right\} \tag{32}$$

As for the climb segment, values that are not constant during this phase will be averaged (q,α)

E. cruise & loiter

Cruise is done at a constant speed and altitude, Therefore:

$$\frac{T_{SL}}{W_{TO}} = \frac{\beta}{\alpha}\left\{K_1\left(\frac{\beta W_{TO}}{qS}\right) + K_2 + \frac{C_{D_0}}{\frac{\beta}{q}\frac{W_{TO}}{S}}\right\} \tag{33}$$

Loiter is defined as the best endurance speed. Using the same method as (24) from mission analysis. Best endurance can be calculated such as

$$V_{ed} = \sqrt{\frac{2}{\rho_\infty}\left(\frac{W}{S}\right)\sqrt{\frac{K}{C_{D0}}}} \tag{34}$$

Where C_{D0} can be found using (8). As we may notice C_{D0} is function of V. Therefore, the tool is using an iterative approach to find the best lift-to-drag speed by adjusting the Mach number until convergence is achieved within a specified tolerance. (Usually tolerance is achieved in two of three iterations)

F. Descent and Deceleration

We have assumed that no thrust is required for these phases (as in the mission analysis part, we have assumed that the descent and deceleration are done without thrust). Therefore, there are no constraints for these segments.

G. Approach

The approach constraint used the same equation as the climb constraint, but used the flight path angle instead of the rate of climb (which is negative). The mission requirements mention that only 20% of the full power takeoff fuel flow is used during this phase. This can be interpreted as the thrust lapse α being 20%. Furthermore, this constraint is computed with a weight fraction of 0.85 ($\beta_{\text{Approach}} = 0.85$

$$\frac{T_{SL}}{W_{TO}} = \frac{\beta}{\alpha}\left\{K_1\left(\frac{\beta W_{TO}}{qS}\right) + K_2 + \frac{C_{D_0}}{\frac{\beta}{q}\frac{W_{TO}}{S}} - \sin(\gamma)\right\} \tag{35}$$

H. Landing

The landing constraint is one the most important. Indeed, the goal is to land at a certain speed (usually very low) without stalling. Therefore the lift equation (17) is used such as

$$\left(\frac{W_{TO}}{S}\right)_{\text{Land}} = \frac{C_{L_{\text{Max}}} * q}{\beta * C_{L_{Land}}^2} \tag{36}$$

However, the mission requirements mentioned that Landing should be done at a 5% fuel reserve. Therefore, this constraint should be done with 5% of the fuel. If β_L is the weight fraction of the aircraft during landing and β_c is the weight fraction used to compute the constraint analysis.

$$\frac{W_{\text{fuel\_constraint}}}{W_{TO}} = 95\% \frac{W_{\text{fuel\_aircraft}}}{W_{TO}}$$
$$1 - \beta_c = 95\%(1 - \beta_L)$$
$$\beta_c = 5\% + 95\%\beta_L$$

I. Additional Constraints

5 additional constraints have been given as an input for this design

1. One engine climb

This constraint consists of maintaining a gradient of climb no less than 5% with one engine inoperative after clearing the 35-ft obstacle during takeoff. To compute this constraint , eq (31) has been used and multiplied by 2 to simulate the loss of one engine. This emergency has been simulated at a speed after Takeoff such as $V_2 = k_{\text{safety}} * V_{TO}$ where $k_{\text{safety}} \approx 1.2$knots[2]. The speed at which the aircraft may safely climb with one engine inoperative is called V2 speed and is defined by the FAA. Furthermore, this constraint simulates a climb until 500 ft which will be the minimum altitude the aircraft will keep flying with one engine. This altitude is crucial as it has a direct impact on this constraint.

2. Service ceiling

This constraints consists in maintaining a rate of climb no less than 300 f t/min with two engines at a service ceiling of 41,000 ft with cruise Mach number (0.78) and the weight at the top of climb (35,000ft). For this constraint, equation (31) is used.

3. Maximum max number

This constraints consists in achieving a maximum Mach number of 0.82 in level flight at cruise altitude with the weight at the top of climb. For this constraint, a cruise segment has been used (equation (33)

4. Steep turn

the goal is to sustain a 45-deg banked steep turn at 39,000 ft with cruise Mach number This constraints can be computed directly from the master equations knowing that load factor is directly related to bank angle such as

$n = \dfrac{1}{\cos\phi} \approx 1,41$. This constraint has been evaluated at the top of climb weight.

$$\frac{T_{SL}}{W_{TO}} = \frac{\beta}{\alpha}\left\{K_1 n^2 \left(\frac{\beta W_{TO}}{qS}\right) + K_2 n + \frac{C_{D_0}}{\dfrac{\beta}{q}\dfrac{W_{TO}}{S}}\right\} \tag{37}$$

5. Approach constraint

Approach constraint described in VI.G has been evaluated with a weight fraction of 0.85

J. Find the design space and the design point

To find the design point of the constraint plot, the goal is to find the minimum Thrust to Weight ratio that satisfies all constraints. First , for each wing loading, we will pick the maximum thrust to weight ratios among all constraints. Then, the minimum of this list will be picked. This will provide a first design point. The wing loading associated must be compared to the wing loading for the landing (calculated in (36)). Next, the calculated wing loading for this minimum thrust-to-weight ratio is compared to the wing loading required by the landing constraint. If the wing loading required for the calculated thrust-to-weight ratio exceeds the wing loading needed for landing, the design is adjusted to prioritize the landing constraint. In this case, the wing loading for landing is used, and the corresponding thrust-to-weight ratio is recalculated based on this landing requirement. If the landing constraint does not dominate, the initially calculated values for wing loading and thrust-to-weight ratio are kept.

VII. Use of the tool

This section briefly summarize how to use the tool To use the tool, you need inputs. Inputs are located in the `Inputs` folder in JSON files.

A. Use the existing example

A set of example input files are provided in the `Inputs` folder. These files contain the necessary parameters for the sizing calculations for the mission given in the project for an A320. You can use these files to test the tool and see how it works.

To run the tool with the provided example inputs, simply run the main script:

```
python Main.py
```

Run time can take up to 5 minutes depending on the computer's performance. Please be patient. A progress bar will be displayed in the console to show the progress of the calculations. Usually, the number of iterations is 3.

B. Input your own parameters

If you want to use your own parameters, you can create your own input files. You need to provide input files in JSON format. These input files should be placed in the `Inputs` folder. Each JSON file should contain the necessary parameters for the sizing calculations.

Here is an example of what an input JSON file might look like:

```
{
        "name": null, ### Default name is type + phase number but user can enter its own
        "type": "taxi", ## Type of segment
        "weight_fraction": 1, ## Initial Weight fraction will be updated during run
        "phase_number": 1, ## Number of the phase
        "time": 20, ## Time of the phase, if not relevant , can put null
        "percent_fuel_flow": 0.1, ## Percent fuel flow used during phase
        "speed": 15, ## EAS speed
        "altitude": 0
},
```

Listing 5 Snippet of a json file used to input mission

Make sure to adjust the parameters and values according to your specific requirements.

Five input files are provided in the `Inputs` folder. You can use them as a reference to create your own input files:

- `aerodynamics.json`: This file contains the aerodynamics constants K_1, K_2.
- `propulsion.json`: This file contains the propulsion constants.
- `structural.json`: This file contains the structural constants such as k_{WE}.

These three files cannot be renamed or modified; the program will not run correctly if you do so.

- `Mission_Profile.json`: This file contains the mission profile segments. Feel free to modify any phases or add new ones using this example.

Careful: You need specific parameters for each phase. Check the provided `Mission_Profile.json` file to see which parameters are required for each phase so the program can run correctly.

Here is a list of the available phases you can add:

- "taxi"
- "takeoff"
- "climb"
- "acceleration"
- "cruise"
- "loiter" (in the `cruise.py` file)

- "deceleration" (acceleration with start speed inferior to end speed)
- "descent" (climb with negative rate or flight path angle)
- "approach"
- "Landing"

To run the tool with your own input files, be careful to modify the name of the inputs file in the `Main.py` and simply run the main script:

```
python Main.py < My_Mission_Profile.json >
```

VIII. Outputs

After running the tool, the main results will be printed in a console, and graphs will be generated:

- The first graph shows the constraints analysis plot.
- The second graph shows the Weight Breakdown and the weight decomposition.
- The third graph shows some aerodynamics and propulsion characteristics. A dropdown menu is available to select the parameter to display.

If the graphs are not displayed, please refresh the page (F5) where the graphs are displayed. (Sometimes Plotly has issues with displaying the graphs.)

A. Output folder

All the results will be saved in the `outputs` folder as HTML files. You can open them in your browser.

- ***aero_and_prop_characteristics.html*** This graph plots several aerodynamics and propulsion characteristics
- ***combined_weight_plot.html*** This graphs shows the converged weight breakdown and plots the weight of the aircraft at each phase
- ***Constraint_analysis.html*** This is the constraint analysis plot
- ***final_design_results.html*** This file summarizes the value of the main variables computed by the tool (WSR,TWR,Wingspan, Wing Area, TOW ...)
- ***TWR_and_WSR_per_phase.html*** Thi file shows instantaneous W/S and T/W at each phase

IX. Results

This section summarizes the main results for the given Mission and Performance requirements once the Aircraft has been designed

A. Mission Analysis

The following table summarizes the weight breakdown for the converged design and for the A320 neo [2, 6, 7]. An histogram is also available on the output folder of the tool (html files)

Table 3 Weight Breakdown (Values are rounded

Weight Type	Designed Aircraft	A320neo
Takeoff Weight	169,700 lbf	175 000 lbf
Empty Weight	94,700 lbf	90 000 lbf
Fuel Weight	40,800 lbf	29 762 lbf
Payload Weight	33 000lbf	31 400lbf

We can see that the takeoff weight and the empty weight are very similar. However, the fuel weight of the designed aircraft is higher than that of the A320 neo. This could be explained either by the engine assumptionIV, which could be wrong since the engine installed on the aircraft might not be the same as the one used to design the aircraft (PW 1100G-JM). It should be noted that the added value of the A320 neo is its ability to burn less fuel than a regular A320 (neo means new engine option). Therefore, using an A320 engine to design an A320 neo could lead to some fuel calculation errors. For some segments, such as taxiing or approach, assumptions have been made about the consumption of the aircraft, which could also explain this difference. Finally, the payload is slightly different, but the payload is directly related to the mission requirements, so it is difficult to compare these values. The value found for the A320 is a design payload. It would be more accurate to find a payload weight for this exact mission for the A320neo.

The tool also provide a instantaneous weight, aerodynamic and propulsive characteristics, and instantaneous T/W and W/S values for each phase in the mission (see output folder).The following table summarizes these variables for key point in the mission (each data is computed at the beginning of the segment)

Table 4 Mission analysis characteristics

Phase	T/W	W/S ($\mathrm{lbf/ft^2}$)	TSFC (1/h)	Lift Drag Ratio
start	0.03	110.71	0.296	0
Takeoff	0.242	110.4	0.3586	11
begin of cruise	0.088	108.57	0.486	18.03
end of cruise	0.173	90.16	0.453	18.4
begin to loiter	0.195	87.78	0.437	20.8
end of loiter	0.219	86.49	0.429	19.34
landing	0.311	85.86	0.3585	11

We can see that some values actually fit with their phases. For instance, during loiter , lift to drag ratio should be optimized and fuel consumptions minimized (low TSFC and high lift drag ratio)

The following figure 4 shows the TSFC at each phase. As expected, the TSFC is maximized during climb, acceleration and cruise (phases that consume a lot of fuel). We can also see that during loiter, the TSFC tends to decrease because loiter phase is done at a speed that should minimize fuel consumption at a given altitude.

Fig. 4 TSFC throughout the mission

B. Constraint Analysis

The following figure depicts the constraint plot and feasible design space for the most constraining cases. The full constraint plot is available in appendix and in the output folder.

Fig. 6 Thrust available at each phases

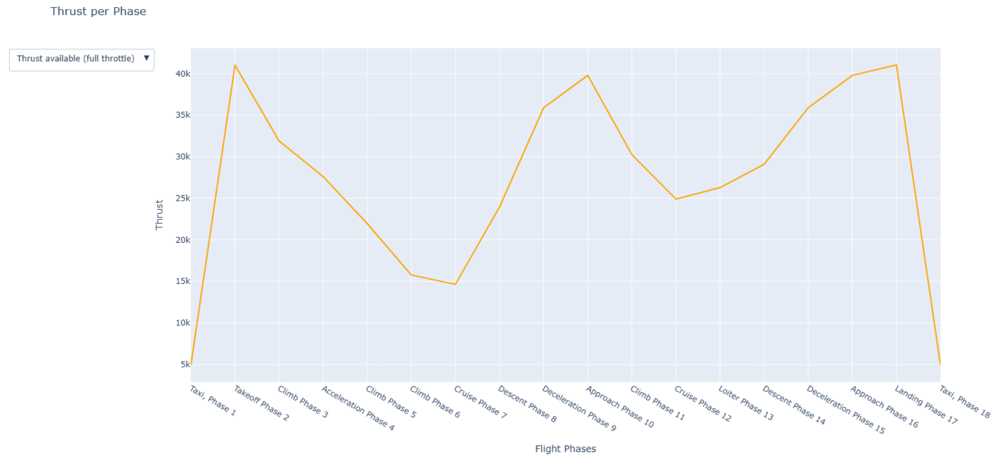
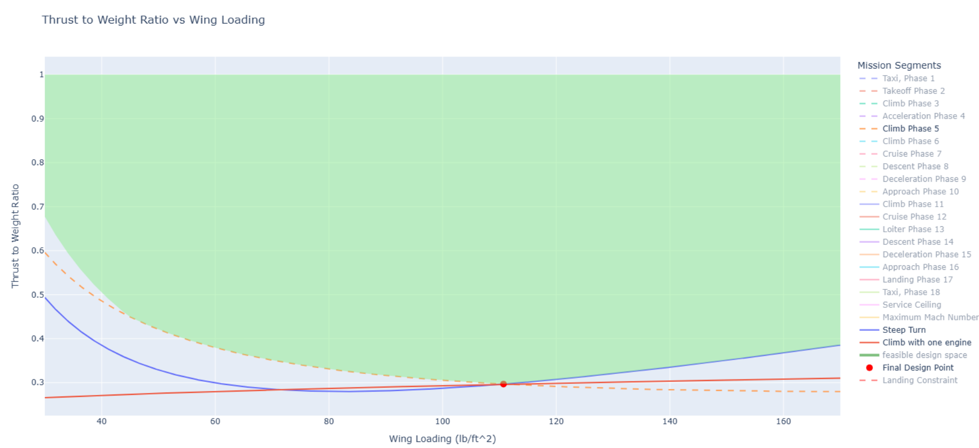


Fig. 5 Constraint Plot for the designed Aircraft (Only phase 5 , steep tun and one engine climb



As the figure shows, the limiting constraint is the climb at 290 KEAS until mach 0.78 at a rate of climb of 3000ft/min (phase 5 see IX.C. This mainly due because the cruise mach is achieved at a very high altitude. This transition between airspeed and Mach occurs at a point called the "crossover altitude" (usually between FL250 and FL300 depending on the aircraft type). When the aircraft climbs to the crossover altitude at a constant IAS, Mach increases. For our example , crossover altitude is 28700 ft. Therefore, keeping a climb rate of 3000ft/min is very difficult for the aircraft. Thrust available is very low at high altitude (see figure 6) (This is why after reaching the crossover altitude, climb rate drops to 1500 ft/min).

Another limitation is the one engine climb after takeoff.In fact, because the airplane's speed is low (about 150 knots at V2, see VI.I.1, it is very difficult for the airplane to climb with only one engine (available thrust is divided by 2) and to maintain a certain flight path angle. Without this additional constraint. The next constraint is the take-off constraint (linked to the phase 6 constraint, which is the climb to cruise), which would have given a value of wing loading and therefore wing area closer to that of the A320. It should also be noted that the takeoff roll assumption 28 could also play an important role as it underestimates the takeoff constraint.

Finally, steep turn constraints also play a role in the constraint plot. This is mainly due to the fact that the load factor at 45° bank angle increases the thrust to weight ratio. Also, it was assumed for this phase that $\beta = \beta_{\text{Top of climb}}$, which doesn't have the same effect as if the weight fraction had been chosen after cruise.

C. Conceptual Design Point

This last subsection summarizes the main variables for the design point found for the designed aircraft. The aspect ratio is not a value calculated by the tool, but is taken directly from the A320neo data. [7] It has been used to compute the wing span using the formula $AR = \dfrac{\text{Wing Span}^2}{S}$

Table 5 Main Desin Point Variables

Variable	Computed value	A320neo value
TOW (lbf)	169,700	175 000
Wing Loading (lbf/ft²)	110.715	123
Thrust-to-Weight Ratio	0.2968	0.3084
Wing Area (ft²)	1,532.66	1,313,2
Sea Level Thrust (lbf)	50,364.51	50000
Span (ft)	119.96	117.45
Aspect Ratio	9.39	9.39

As the table 5 shows, many values are actually close to the real ones (about 10% error). The main difference comes from the wing loading, which is designed to be lower in the tool. This is directly related to the constraint analysis plot 5. In fact, the constraint that consists of climbing to crossover altitude is very difficult to achieve and lowers the Wing Loading value. This also affects the wing area and span. The aircraft designed by the tool is longer with a larger area, which makes it heavier (that's why the calculated empty weight is heavier). Ultimately, the aircraft will burn more fuel. This problem could also be due to the way the climb constraint is computed (see (31)).While the climb segment has been subdivided for mission analysis. This was not the case for the constraint analysis. Therefore some values were averaged between start and end altitude (sometimes 15 000 ft difference!).

It should be noted that this tool is a first conceptual design process to design an aircraft. Therefore, the results are still acceptable at this stage in the design of the aircraft. To determine if the results are acceptable, an engineer could compare these results to other aircraft that share the market with the A320neo (such as the Boeing 737-800). The aim is not to create a copy of an existing aircraft, but to test the accuracy of the tool by comparing the results with well-known aircraft.

To improve the accuracy, an external analysis tool such as OpenMDAO could be used to perform multidisciplinary analysis. Furthermore, it could be interesting to use integration methods to see the difference with the current tool. Finally, each discipline (aerodynamics, geometry, propulsion) could use a specific tool to better model variables such as TSFC or aerodynamic coefficient (for example, using NPSS to model the engine of the aircraft).

However, the tool is very limited. For example, the stability of the aircraft is not calculated, although it plays an important role in the design of the aircraft. An extension of this tool could be the implementation of a module that calculates the center of gravity of the aircraft as well as the wing position and the vertical and horizontal stabilizer geometry. Computing the entire geometry of the aircraft would help the user to visualize the vehicle shape.

As mentioned above, the tool is designed to work on an A320neo. Therefore many equations are empirical and will only work on this aircraft (or very similar like B737 family or A320 family). Building a new type of aircraft is very limited with this tool, as models have to be redefined. For example, building an aircraft with a different type of propulsion will require new propulsion assumptions. As another example, the empty weight fraction is based on historical data; building a new aircraft will also require finding a new way to estimate the empty weight fraction.

Finally, the main limitation of this tool is that it was designed assuming that the aircraft is a point mass, which is not true. As mentioned earlier, the stability of the vehicle is not taken into account (for example, the moments of inertia are not calculated). This would limit the constraint analysis as it would affect the aerodynamics and the weight and balance of the aircraft. Weight and balance are critical for an airplane because they determine the maneuverability and stability of the vehicle. The structure of the aircraft is also not modeled when it is considered as a point mass. For example, wing geometry and volume are necessary to know how much fuel the aircraft can carry, since fuel is usually located in the wings.

As mentioned above, this tool is only in its preliminary design and will be enhanced throughout the year to improve the accuracy of the model and the user interface. People are welcome to contribute to improve its performance.

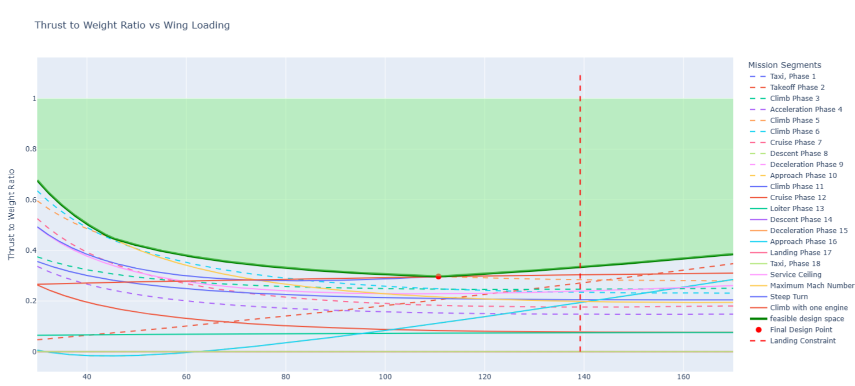
Appendix

Mission Profile

Table 6 Design Mission and Performance Constraints

Phase	Description
1	Taxi out for 20 min, assuming 10% of takeoff full power fuel flow
2	Takeoff and clear a 35-ft obstacle in no greater than 5500 ft with two engines operating at sea level and standard day. Assume takeoff rate of climb at 3000 ft/min
3	Climb to 10,000 ft at 250 KEAS with a rate of climb of 3000 ft/min
4	Accelerate from 250 KEAS to 290 KEAS at 10,000 ft within 1 minute
5	Climb at 290 KEAS with a rate of climb of 3000 ft/min until reaching Mach 0.78
6	Continue climb to 35,000 ft at Mach 0.78 with a rate of climb of 1500 ft/min
7	Cruise at 35,000 ft for 3000 nautical miles at Mach 0.78
8	Descend from cruise altitude to 3000 ft at 250 KEAS with a rate of descent of 1500 ft/min
9	Decelerate from 250 KEAS to an approach speed of 135 KEAS at 3000 ft, assuming no fuel flow
10	Approach at 135 KEAS with a descent angle of 3 degrees until sea level, assuming 20% of takeoff full power fuel flow
11	Execute missed approach (full power go-around) and climb from sea level to 15,000 ft at 250 KEAS with a rate of climb of 3000 ft/min
12	Cruise at 15,000 ft for 200 nautical miles at 250 KEAS to an alternate airport
13	Loiter at 15,000 ft for 45 minutes at best endurance speed
14	Descend to 3000 ft at 250 KEAS with a rate of descent of 1500 ft/min
15	Decelerate from 250 KEAS to an approach speed of 135 KEAS at 3000 ft, assuming no fuel flow
16	Approach at 135 KEAS with a descent angle of 3 degrees, assuming 20% of takeoff full power fuel flow
17	Land at the alternate airport with a 5% fuel reserve
18	Taxi in for 20 min, assuming 10% of takeoff full power fuel flow

Fig. 7 Constraints plot for all phases and additional constraints



References

[1] Mattingly, J., *Aircraft Engine Design*, AIAA Education Series, American Institute of Aeronautics & Astronautics, 2002. URL https://books.google.com/books?id=2Wy5rpdm3DMC.

[2] John D. Anderson, J., *Fundamentals of Aerodynamics*, 4$^{\text{th}}$ ed., McGraw-Hill, New York, USA, 2007.

[3] Nollet, Y., "USSA1976," Tech. rep., 2023. URL https://ussa1976.readthedocs.io/en/latest/#.

[4] NASA, N., and USAF, "U.S. Standard Atmosphere, 1976," Tech. Rep. NASA-TM-X-74335, National Aeronautics and Space Administration, 1976. URL https://ntrs.nasa.gov/search.jsp?R=19770009539, visited on 2019-10-25.

[5] EASA, "PW1100GJM Series Engines Type Certificate Data Sheet:," Tech. rep., European Union Aviation Safety Agency, 2022. URL https://www.easa.europa.eu/en/document-library/type-certificates/engine-cs-e/easaime093-pw1100g-jm-series-engines.

[6] Lloyd R. Jenkinson, D. R., Paul Simpkin, *Civil jet aircraft Design*, elsevier, 1999.

[7] AIRBUS, "A320neo overall charasteristics," *Airbus Website*, 2024. URL https://aircraft.airbus.com/en/aircraft/a320-the-most-successful-aircraft-family-ever/a320neo.