POLITECNICO DI TORINO

Department of Mechanical and Aerospace Engineering

Master Degree Course in Aerospace Engineering

Master Degree Thesis

Evaluation and trade-off of aircraft variants with FAST-OAD-GA

Supervisors :

Prof. Frulla Giacomo

External Supervisors :

Mr. Jézégou Joël

Mr. Lutz Florent

Candidate :

Rémond Lucas

ACADEMIC YEAR 2020-2021

This thesis is based on part of the work achieved in a 6-month internship performed at ISAE-SUPAERO. More detail about this French research center can be found in the introduction of the thesis.

The external supervisors who closely monitored the progress of the internship are Mr. Joël Jézégou along with Mr. Florent Lutz, both engineers at ISAE-SUPAERO. The accomplishment of the project related to the internship led to work with Mr. Aurélien Reysset, Mr. Scott Delbecq and Mr. Christophe David, all engineers at ISAE-SUPAERO at the time of writing.

An industrial collaborator is the aircraft manufacturer DAHER, with whom an interaction during the whole internship allowed to obtain confidential data about some of their aircrafts. Due to their sensitive nature they will not be disclosed in this report.

Summary

List of Figures

List of Tables

List of Symbols and Abbreviations (SI units followed by British units)

OAD	Overall Aircraft Design
CS-23	Certification Specifications 23
$FASTOAD$	Future Aircraft Sizing Tool – Overall Aircraft Design
$FASTOADGA$	Future Aircraft Sizing Tool – Overall Aircraft Design – General Aviation
MDA	Multidisciplinary Design Analysis
$TLARs$	Top Line Aircraft Requirements
$MTOW$	Maximum TakeOff Weight, kg or lb
OWE	Overall Weight Empty, kg or lb
MAC	Mean Aerodynamic Chord, m or ft
HTP	Horizontal Tail
VTP	Vertical Tail
CG	Center of Gravity
n	sizing load factor
n_{ult}	ultimate sizing load factor
l_{fuse}	fuselage length, m or ft
w_{fuse}	fuselage width, m or ft
h_{fuse}	fuselage height, m or ft
R_{fuse}	fuselage radius, m or ft
S_{fuse}	fuselage wetted area, m**2 or ft**2
l_{cabin}	cabin length, m or ft
V_{cabin}	cabin volume, m**3 or ft**3
l_{nose}	nose length, m or ft
V_{nose}	nose volume, m**3 or ft**3
l_{rear}	rear (tail cone) length, m or ft
V_{rear}	rear (tail cone) volume, m**3 or ft**3
λ_{cone}	taper ratio of the tail cone's enclosed area
t_{fuse_skin}	fuselage skin thickness, m or ft
$V_{fuse\ skin}$	fuselage skin volume, m**3 or ft**3
$W_{fuse\ skin}$	fuselage skin mass, kg or lb
$W_{stringers}$	stringers weight, kg or lb
W_{frames}	frames weight, kg or lb
t_{fuse_shell}	fuselage shell thickness, m or ft
$W_{fuse\ shell}$	fuselage shell weight, kg or lb
ρ_{skin}	fuselage skin material mass density, kg/m**3 or lb/ft**3
k_{λ}	factor allowing for the influence of the fuselage slenderness ratio
$l_{wing-HTP}$	distance between 25% of htp MAC and 25% of wing MAC, m or ft
$V_{\max\_SL}$	maximum speed at sea level, m/s or kn
V_D	design dive speed, m/s or kn
q_{cruise}	cruise dynamic pressure, N/m**2 or psf
q_{NE}	never-exceed dynamic pressure, N/m**2 or psf
ΔP	maximum sizing pressure differential of the cabin, Pa or psi

$L_{h_{max}}$	maximum lift generated by the horizontal tail, N or lbf
$L_{v_{max}}$	maximum lift generated by the vertical tail, N or lbf
S_{HTP}	horizontal tail wetted area, m**2 or ft**2
S_{VTP}	vertical tail wetted area, m**2 or ft**2
$C_{Lh_{max}}$	maximum lift coefficient of the horizontal tail
$C_{Lv_{max}}$	maximum lift coefficient of the vertical tail
x_{tail}	x-coordinate of the mass lumped point of the tail group, m or ft
W_{HTP}	horizontal tail weight, kg or lb
$x_{cg_{HTP}}$	x-coordinate of the horiontal tail barycenter, m or ft
W_{VTP}	vertical tail weight, kg or lb
$x_{cg_{VTP}}$	x-coordinate of the vertical tail barycenter, m or ft
$x_{cone_{beginning}}$	x-coordinate of the beginning of the tail cone, m or ft
$x_{cone_{end}}$	x-coordinate of the end of the tail cone, m or ft
$W_{tail\ cone}$	weight of the tail cone, kg or ft
r_{Mh}	horizontal tail inertia-relief factor
r_{Mv}	vertical tail inertia-relief factor
M_h	horizontal bending moment, N*m or lbf*ft
M_v	vertical bending moment, N*m or lbf*ft
L_{wing}	lift generated by the wing, N or lbf
$M_{lift\ compensation}$	bending moment generated by the wing lift, N*m or lbf*ft
$W_{cabin_{shell}}$	cabin shell weight, kg or lb
$W_{airframe\_additional\_components}$	fuselage airframe extra components weight, kg or lb
$W_{payload}$	payload weight, kg or lb
$W_{cabin_{furnishing}}$	cabin furnishing and equipments weight, kg or lb
$W_{cabin\ tot}$	total weight distributed on the cabin, kg or lb
x_{wing}	x-coordinate of the wing mass centroid, m or ft
W_{nose}	nose weight, kg or lb
W_{engine}	engine powerplant weight, kg or lb
x_{engine}	x-coordinate of the engine barycenter, m or ft
g	local gravitational field of Earth, m/s**2 or ft/s**2
$x_{ratio\ front}$	ratio of the cabin distributed weight applied on the front bending moment distribution
$x_{ratio\ rear}$	ratio of the cabin distributed weight applied on the rear bending moment distribution

Chapter 1

Introduction and general context of the thesis

To face the increasing environmental footprint of commercial aviation, industrial and research efforts have been focusing on exploring unconventional configurations and new propulsion paradigms, such as distributed propulsion or electric propulsion. Overall Aircraft Design (OAD), the conceptual or preliminary level design process of an aircraft, requires integrating models representing the inherently multidisciplinary properties of such complex systems. The goal is to search for the best configuration that fulfills a given set of TLARs (Top Level Aircraft Requirements). These TLARs define the design mission by specifying the speed for several phases of the flight, the payload mass and the range. Overall Aircraft Design's purpose is then to return an aircraft that responds to this set of TLARs and that can be optimized to exactly fit the requirements. In addition to commercial tools, universities or research institutes contribute to the development of tools or methods for OAD. At Linköping University a design framework is being developed to support the initial conceptual design phase of new aircraft and has been first presented in [1]. The University of Stanford develops SUAVE, a conceptual level aircraft design environment developed in Python to analyze and optimize both conventional and unconventional designs [2]. At the University of Michigan, the MDOLab collaborates with NASA in the development of OpenMDAO [3], a framework to facilitate the application of Multidisciplinary Design Optimization (MDO). The MDOLab also provides a lightweight wing aero-structural optimization code [4]. Finally, ISAE-SUPAERO and ONERA developed FAST-OAD for aircraft sizing analysis and optimization that aims at user friendliness and modularity [5].

ISAE-SUPAERO is a French school that delivers aerospace science and engineering education. It also hosts some top level research units with departments that cover the wide range of aeronautics and aerospace topics, including non primarily related fields such as data science, structure analysis, electromagnetism... ONERA is a French aerospace laboratory. Both of these laboratories are located in Toulouse, which is the city that has be chosen to be the heart of the aerospace research and industry in France.

This thesis is related to the Department of Aerospace vehicles Design and Control (DCAS) of ISAE-SUPAERO, which develops methods, simulation tools and experimental platforms for the design and the control of aerospace vehicles. The department also manages a fleet of nine aircrafts for research purposes. The DCAS includes 35 permanent staff members and around 45 non-permanent members. The research activities are supported through six industrial chairs.

The project that motivated this thesis is part of the ISAAR chair, which stands for Innovative Solutions for Aviation Architecture and Regulation. It is a 5-year long contract that was initiated in July 2019. Its main objective is the analysis of the design and the certification of innovative CS-23 aircrafts architecture. It was funded by the French

aircraft manufacturer DAHER. It is a multi-disciplinary research subject that involves all the scientific fields related to aircraft sizing.

DAHER was created in 1863. It designs, manufactures and supports the TBM, a family of single-turbopropeller aircrafts. One of the latests aircrafts of this family is the TBM 930. This aircraft illustrated by the following picture is strongly related to this thesis purpose.

Figure 1 - Picture of a TBM 930

The CS-23 certification category covers all the aircrafts with a Maximum TakeOff Weight (MTOW) under 5670 kg and that can accomodate at most nine passengers (without the crew). It also extends to the bi-motor propeller commuters with a Maximum TakeOff Weight under 8618 kg and that can accomodate at most 19 passengers (without the crew). The family of DAHER aircrafts studied in this chair (i.e., the TBM 930) is certified in this aircraft category. This certification category is also known as General Aviation (GA). It is mainly opposed to the CS-25 category (certification specifications for large aeroplanes).

One of the main principles of Overall Aircraft Design is the will to achieve rather low computational times in order to match the conceptual design phase typical fast estimations. The aim of preliminary sizing is indeed to test several configurations and to determine which ones prove to be feasible. The computation of an aircraft should last only a few tens of seconds. That is why a lot of physical models that are part of an OAD code make compromises and strong hypothesis to save computational time. Statistical models extracted from aircraft preliminary design books tend to be extremely useful for this reason. They don't require over-detailed data to be computed and they are also often profoundly reliable for conventional aircrafts. Historically Torenbeek establishes a strong basis of such methods for subsonic general aviation and transport aircrafts [6]. Gudmundsson proposes and resumes preliminary aircraft design approaches and statistical models specifically for general aviation [7]. Other authors such as Roskam or Raymer adopted a similar perspective while covering all the certification categories of aircrafts [8, 9]. With the Flight Optimization Design System Weights Estimation Method (FLOPS) the NASA developed a new methodology to compute the mass breakdown of an aircraft which makes the distinction between general aviation and the other aircraft categories [10]. Nevertheless, for innovative architectures and special configurations it might be necessary to reconsider using these models that could no longer be appropriated. If the conceived aircraft configuration is too much remote from the aircrafts that were used

to obtain the database which shapes the statistical models then the outputs from these models might lose in accuracy. In that case a new approach has to be defined and more precise models can be developed (analytical or semi-analytical models). The CS-23 norm also defines a lot of data and of constraints that a general aviation aircraft has to meet. The 4th amendment of the CS-23 has been the main source of documentation about the regulation for this thesis [11]. Even if the designer is theoretically free to innovate on all the aspects of the aeroplane architecture, it is still mandatory for the resulting aircraft to pass through the certification criteria. The same reasoning obviously applies for an aircraft sizing tool. What's more some steps of the design of the aircraft are strongly related to the certification, such as the computation of the flight envelope. Finally it is important to know the global level of accuracy required to perform preliminary design. The whole code must meet this accuracy level, and it might be irrelevant to implement models that are too refined in one section of the code. It is the engineer's work to keep this point in mind and to be aware of the current state of the code in order to optimize its performance and relevance.

The following reference axes mutually perpendicular and originating at the center of gravity is used in the whole thesis.

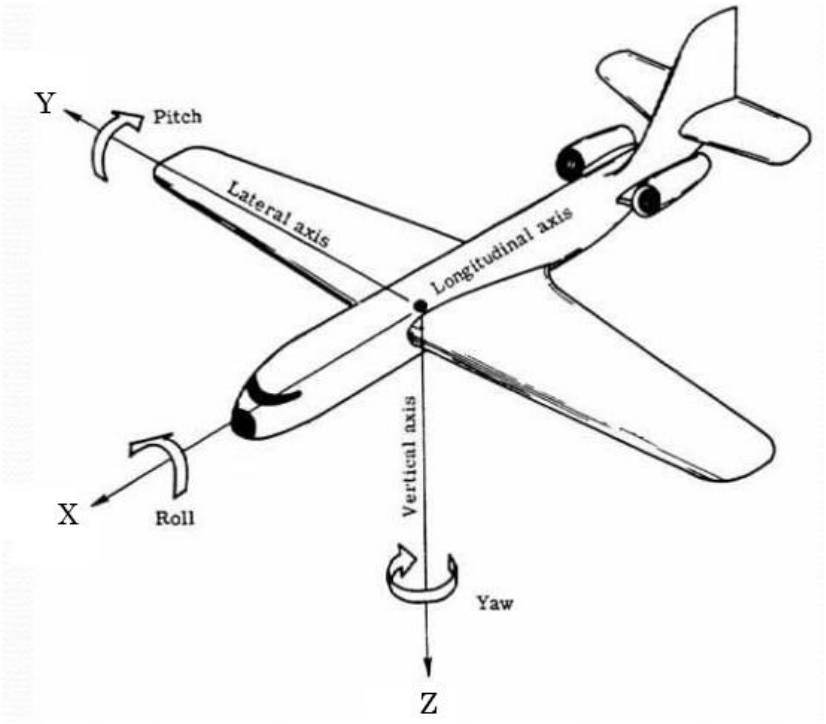

Figure 2 - Airplane reference axes

Chapter 2

FAST-OAD-GA

2.1 Presentation of the current aircraft sizing tool

OpenMDAO (where MDAO stands for Multidisciplinary Design Analysis and Optimization) is an open-source optimization framework and a platform to building new analysis tools with analytic derivatives. OpenMDAO allows to analyze and solve MDAO problems which are represented by a system of objects called components. These components have input and output attributes and perform calculation when they are executed. The inputs and outputs of one component can be connected to those of other components, allowing data to be passed between the components. This feature proves to be extremely interesting since it links the several scientific disciplines that rule aircraft sizing. For instance if a component generates the geometry of the aircraft, its outputs can be converted as inputs for another component, let's say the aerodynamic characteristics computation component . Finally the outputs of this aerodynamic component can be used to resize the aircraft geometry. This reasoning is the foundation of aircraft sizing software.

Future Aircraft Sizing Tool – Overall Aircraft Design (FAST-OAD) is an open source preliminary aircraft sizing tool. It is a framework for performing rapid Overall Aircraft Design on aircrafts from the CS-25 certification. It proposes multi-disciplinary analysis and optimisation by relying on the OpenMDAO framework. It has been created by ISAE-SUPAERO in collaboration with ONERA. Its development was achieved using entirely Python environments.

FAST-OAD-GA is an open source preliminary aircraft sizing tool derived from FAST-OAD. It stands for Future Aircraft Sizing Tool – Overall Aircraft Design – General Aviation. It was created by ISAE-SUPAERO and ONERA as part of the ISAAR chair and is an independent extension of FAST-OAD. This sizing tool is designed exclusively for the CS-23 aircrafts. It has the same structure and basic functionalities as FAST-OAD, but it has changed in order to account for the differences between CS-25 and CS-23 aircrafts. Typically, what changes between the two sizing tools are the scientific models in the code, especially the semi-statistical ones. The code is indeed an assembly of multidisciplinary modules whose inputs and outputs intersect to compute an aircraft. Each of these modules uses models to estimate some quantities, and these models are either fully analytical, semi-analytical, or statistical. FAST-OAD-GA also aims to meet growing needs which are a merge between an ambition expressed by DAHER and an intent to broaden the code features. In particular, this thesis aimed to add a new functionality to FAST-OAD-GA, called the analysis mode. At first the structure of the code will be presented. Then will follow the introduction of the need expressed by DAHER which results in the creation of a new functionality in the program. How this need was met in this thesis will be the subject of the remaining sections of this report.

This figure presents the structure of the main analysis type of FAST-OAD-GA : the Multidisciplinary Design Analysis, often written as MDA. This process uses linear and

nonlinear solvers in order to compute an aircraft which is sized for a set of TLARs that are chosen by the user as inputs of the program. The inputs also define some of the aircraft basic characteristics among which its rough geometry, its Maximum TakeOff Weight first estimation, its propulsive system parameters. The code also uses some correction factors to scale and rectify the weight models of the airframe. The output of a MDA is a fully modelled aircraft that has passed through all the models implemented in the code.

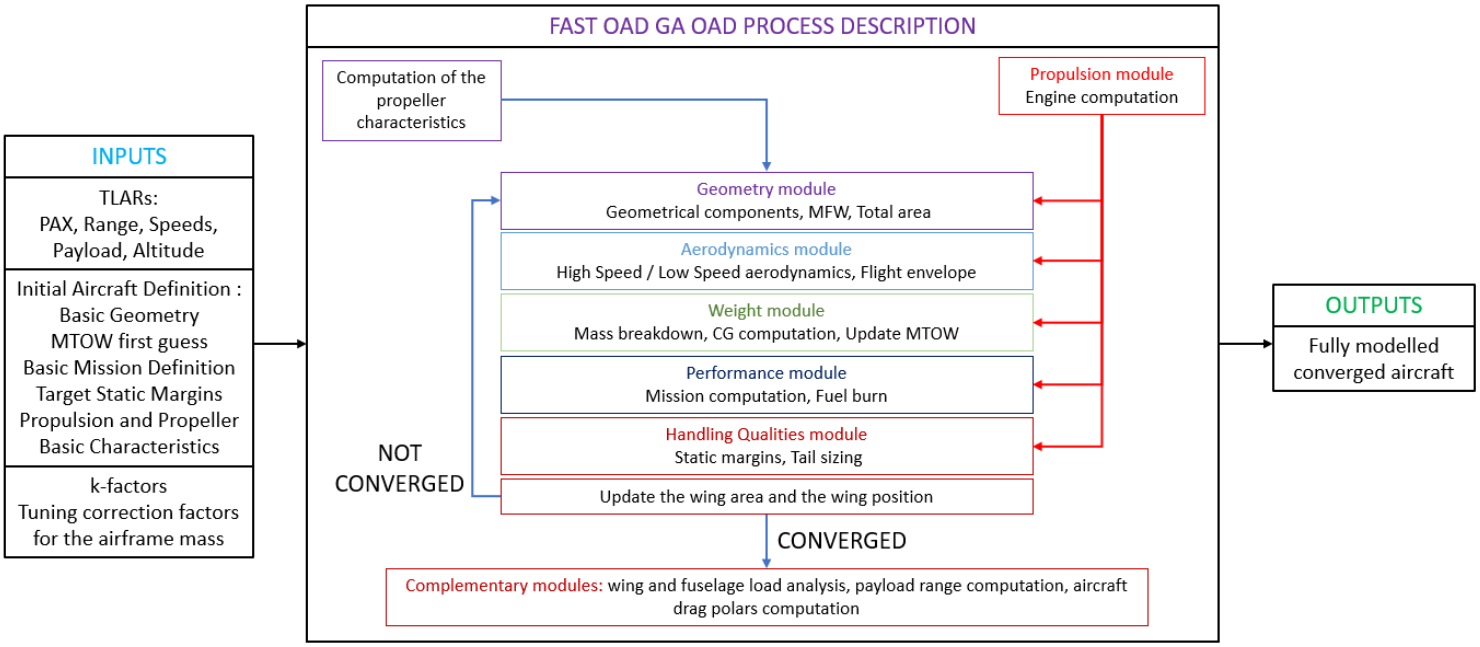

Figure 3 – Overview of a Multidisciplinary Design Analysis

The unique code structure highlighted by this figure is described in detail in the next section.

2.2 Description of FAST-OAD-GA modules

This section aims to briefly go over all the code modules that are executed in a Multidisciplinary Design Analysis and to highlight the analysis logic and structure. FAST-OAD-GA is entirely coded in an openMDAO framework, and its classes are all clustered into the appropriated modules. The MDA input file is a .yml file that defines the different sizing loops. The conventional file structure results in an analysis identical to the one presented in the current section. The user can also define other types of analyses, by adding or removing classes, modules or even loops in the process. Moreover the .yml file is the place where the modules parameters are specified. The module descriptions below are supported by screenshots of the .yml file, which contain the module ids and their parameters.

```
propeller:
    id: fastga.aerodynamics.propeller
    vectors_length: 7
```

The code starts with the propeller computation, that is computing the propeller aerodynamic properties (thrust, limit thrust, efficiency and speed) based on the propeller geometry defined in the inputs of the analysis. These quantities will then be used by the propulsion module to compute the engine performance by taking in account the propeller efficiency and performance. Since the geometry of the propeller is fixed for the analysis, this computation is not part of the main loop and is only performed once in the beginning of the analysis.

Once the propeller computation is done, the code can pass through the code different modules. The figure 2 lists all of them and also highlights their main functions. These modules are named after the physical field or sizing step that they represent.

```
geometry:
    id: fastga.geometry.legacy
    propulsion_id: fastga.wrapper.propulsion.basicIC_engine
```

The geometry module computes the main aircraft geometrical components that are the fuselage, the wing, the horizontal tail (commonly reffered as htp), the vertical tail (commonly reffered as vtp), the nacelle and the landing gears. The majority of the models used to obtain these components is analytical, since simple geometrical formulas allow to obtain the full geometry if certain quantities are already known. Such quantities are defined in the initial file and serve for the first iteration as a first guess. To give a clear idea of how it works, the entire wing geometry is calculated based on its aspect ratio, its taper ratio , its mean thickness/chord ratio, its sweep angle at a certain point and its area. For the first iteration these quantities are directly extracted from the input file and some of them such as the wing area are then updated throughout the analysis.

```
aerodynamics_lowspeed:
    id: fastga.aerodynamics.lowspeed.legacy
    propulsion_id: fastga.wrapper.propulsion.basicIC_engine
    wing_airfoil : naca63_415.af
    result_folder_path : D:/tmp
    compute_slipstream : false
aerodynamics_highspeed:
    id: fastga.aerodynamics.highspeed.legacy
    propulsion_id: fastga.wrapper.propulsion.basicIC_engine
    wing_airfoil: naca63_415.af
    result_folder_path: D:/tmp
    compute_mach_interpolation: false
    compute_slipstream: false
```

The aerodynamics module computes the aerodynamic properties of the aeroplane : global and local lift and drag coefficients, lift coefficient distribution, aerodynamic efficiencies, global and local Reynolds number, finesse, global and local Mach numbers, and the several derivatives of the lift coefficients of the aircraft. These quantities are computed for two distinct cases : the first one is equivalent to the cruise conditions reffered to as high speed aerodynamics with the use of the cruise speed and altitude and certain control surface parameters. The other case is reffered to as low speed aerodynamics and is related to the approach conditions. The approach speed and atmosphere are used along with the associated control surfaces parameters. The code uses XFOIL to read and exploit the wing and HTP airfoil polars. It proposes two approaches to model the 3D aerodynamic effects : the use of the open-source software OpenVSP or the use of a homemade Vortex Lattice

Method (VLM) code. OpenVSP is more complete and more easily adaptable than the VLM code but it requires more computational resources and time. The characteristic speeds and load factors that constitute the flight envelope and that are needed by some models of the code are also computed in this module.

```
weight:
    id: fastga.weight.legacy
    propulsion_id: fastga.wrapper.propulsion.basicIC_engine
```

The weight module has two main functions. First of all it performs the mass breakdown of the aircraft, that is estimating the mass of all the single aircraft components and summing them to obtain the Overall Weight Empty of the aircraft (reffered as OWE). These mass components include the airframe, the propulsion systems, the power systems, the navigation systems, the passenger seats, and the life-support systems (internal lighting, air conditioning, security kits,…). These models are mainly statistical. The Maximum TakeOff Weight is then computed with the following definition :

$$MTOW = OWE + payload + fuel$$

The payload is defined as the sum of the passengers, the crew and the freight (which is entirely made of luggages for CS-23 aircrafts). The added mass of fuel is the necessary mass to achieve the mission defined by the TLARs and in the input file. Since this fuel weight is estimated at each iteration in the performance module that follows the weight module, the weight module takes the value from the last iteration. For the first one an initial guess is extracted from the inputs of the analysis. The same reasoning applies to the MTOW. Indeed some of the statistical models of the mass breakdown use the MTOW as input so a first value, even a rough guess has to be written in the input file for the first iteration.

The weight module also performs a parallel analysis which is the computation of the center of gravity of the aircraft. The approach is quite the same as for the mass breakdown. The barycenter of each of the components of the aircraft is computed, thanks to their position and their mass that was computed in the previous section. Then the center of gravity of the empty aircraft is computed. Adding the payload and the fuel, all the possible load cases and scenarios are defined (on ground and in flight) and the resulting range of barycenters is returned.

```
performance:
    id: fastga.performances.mission
    propulsion_id: fastga.wrapper.propulsion.basicIC_engine
```

The performance module aims to compute the several phases that constitute the mission. The one that can be computed in the code is very conventional. It is formed by the successive simulations of the following phases : taxi out, takeoff, climb, cruise, descent, landing, and taxi in. Extra fuel for the reserve is also taken in account, the duration of which is user-defined (for CS-23 aircrafts the typical value would be around 1 hour of reserve). The fuel consumed during each phase is returned. The progressive aircraft mass reduction and the CG shifting as the fuel burns is modelled. By adding the fuel consumed for all the phases the total sizing design fuel is obtained.

```
hq:
    tail_sizing:
        id: fastga.handling_qualities.tail_sizing
        propulsion_id: fastga.wrapper.propulsion.basicIC_engine
    static_margin:
        id: fastga.handling_qualities.static_margin
```

The handling qualities module first of all consists in sizing the tail areas depending on the results obtained from the previous modules. The horizontal tail is sized to make sure the aircraft has enough rotational power to support the takeoff and landing phases. The vertical tail is sized to ensure that the aircraft has enough controllability in flight and that it can maintain a straight flight path during a crosswind landing. For bi-motor aircrafts, the impact of the loss of an engine is also considered for takeoff, landing and for a linear trajectory at a limited altitude of 5000ft. In that case the vertical tail area is sized accordingly. The certification defines all of these scenarios.

The other objective of the handling qualities module is to compute the static margins of the aircraft, stick fixed and stick free.

```
wing_position:
    id: fastga.loop.wing_position
wing_area:
    id: fastga.loop.wing_area
```

The last step of each iteration of the code is to resize and update the wing geometry and position. The wing area is sized in order to generate enough lift at required approach speed and to be able to stock the fuel consumed by the design mission. The wing longitudinal position is also updated in order to meet an input target value for the stick-fixed static margin of the aircraft.

The propulsion module is a bit different from the other modules. It basically gathers all the informations and classes related to the propulsive system. The main feature of this module is the computation of flight points. These points are defined as a Mach number, an altitude, and a required thrust (or a thrust rate). The code then simulates the engine behavior in these particular conditions and returns the Thrust Specific Fuel Consumption and the actual thrust. The module also hosts classes about the engine geometry and weight. It is not executed independently during the process, but is called by all the other modules when a simulation of the engine characteristics is needed. The above screenshots of the .yml file clearly show that the majority of FAST-OAD-GA modules have the propulsion module as an input parameter. For example the performance module calls the propulsion module to achieve the mission computation, and the geometry module calls the propulsion module in order to get the dimensions of the engine and to properly size the nacelle geometry based on that information.

After the wing has been updated the solvers compare the evolution of all the computed quantities with the previous iteration. Based on the relative tolerance that the user defines the code states that the analysis has converged or not. If it has not converged then the whole process is done again, starting from the geometry computation in the geometry module. Since the wings and the tails have been updated the output of this module will change and it will affect all the other modules. The update of the MTOW and of the mission sizing fuel will also modify the results. When the convergence is reached, the code has

properly generated all the characteristics and the properties of the aircraft that responds to the input TLARs and settings.

Extra loops using solvers internal to some modules are also implemented in the code to ensure a proper process. These loops are not represented in figure 2. For example there is an internal loop in the computation of the mission in the performance module. It is necessary since in the code the distance of the descent phase is an input of the cruise phase which is simulated before the descent. The use of such solvers allows to get converged quantities.

The final step of the MDA is the computation of complementary modules which are mostly related to postprocessing. Among others, we can find the estimation of the payload range diagram, the calculation of the aircraft drag polars (equilibrated and non-equilibrated) and the structural analysis of the wings and of the fuselage. Executing these modules before the convergence of the aircraft would be tantamount to unnecessarily increasing the computational time since the outputs of these modules are not used in the sizing loops.

As mentioned during the introduction of preliminary aircraft sizing, the Multidisciplinary Design Analysis does not require by definition a lot of computational resources, and its completion is a matter of seconds, perhaps minutes. Some modules propose different ways to compute their results, and the use of external softwares (in the aerodynamics module) can consequently increase the time required by the analysis. This way the user has a certain freedom of choice about the accuracy and speed of the sizing, while always remaining in the domain of preliminary aircraft design. Regarding the convergence speed, a MDA for a conventional aircraft will always be successful after less than a dozen iterations (the typical number would be 7 or 8 iterations). These results typically depend on the chosen TLARs, the geometry and the propulsive system of the processed aircraft.

The .yml input file which defines the analysis cannot be run on its own. As described with figure 2, the code also needs to be provided the TLARs associated with a certain amount of quantities that primarily describe the aircraft. All these values are defined in a .xml file, which is a format that allows to stock variables readable by the code. Each quantity possesses a name, a unit, a value and a short description for user-friendliness. The following figure is an example of what contains a .xml file.

```
<data>
    <TLAR>
        <NPAX_design>4.0<!--top-level requirement: number of passengers for the design mission, assuming a classic eco/business class repartition--></NPAX_design>
        <luggage_mass_design units="kg">40.<!--top-level requirement: luggage mass per passenger for the design mission--></luggage_mass_design>
        <range units="NM">1730.0</range>
        <v_approach units="kn">128.0<!--top-level requirement: approach speed--></v_approach>
        <v_cruise units="kn">252.0<!--top-level requirement: cruise speed--></v_cruise>
        <v_limit units="kn">392.0<!--top-level requirement: limit speed--></v_limit>
        <v_max_sl units="kn">380.0<!--top-level requirement: sea level maximum speed--></v_max_sl>
        <level>2.0</level> # 0 to 1 pax : 1.0, 2 to 6 pax : 2.0, 7 to 9 pax : 3.0, 10 to 19 pax : 4.0
        <category>3.0</category> # Aerobatic = 1.0, Utility = 2.0, Normal = 3.0, Commuter = 4.0
    </TLAR>
    <geometry>
        <flap>
            <chord_ratio>0.2<!--mean value of (flap chord)/(section chord)--></chord_ratio>
            <span_ratio>0.67<!--ratio (width of flaps)/(total span)--></span_ratio>
        </flap>
```

Figure 4 – Overview of a .xml file

In fact the .xml file has a more critical role than just memorizing the input values of the analysis. Each time that a class is computed, its inputs are extracted from the .xml file

and its outputs are added to the file. For the classes that update already existing values, the file is overwritten. At the end of the analysis the .xml file contains all the computed values of the converged aircraft, and it is ready to be used by the postprocessing utilities. The code efficiently manages the unit conversion of the .xml file quantities. Indeed there are some models that require the quantities to be expressed in British units (length in feet, mass in pounds, speed in knots) while other models are written in the International System of Units (length in meters, mass in kilograms and speed in meters per second). In each class the units of the input and output parameters are defined and the code automatically converts the units from one class to the other, while the .xml file serves as a depository for the current values and units.

The basic diagram below sums up the code structure and the different files used in a Multidisciplinary Design Analysis.

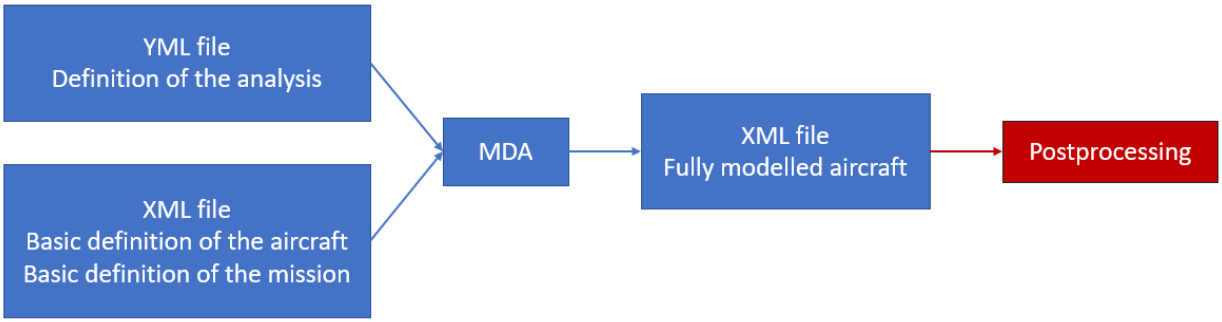

Figure 5 – Transfer of data between files in a MDA

Thanks to the structure of the code and of the use of OpenMDAO components, the code has achieved a satisfying modularity. This is a prerequisite for any open source code and it results in remarkably valuable features for an external user that decides to download and test FAST-OAD-GA. For example, new propulsion types can be implemented if the user respects the existing structure defined in the propulsion module. For the purpose of the thesis a basic model of a turbopropeller engine for the TBM 930 has been implemented in the code, and the only main task to achieve was to get the scientific models ready. Then to choose the propulsive model that has to be run in an analysis, the user simply has to modify the input .yml file by specifying the ID of the desired model. What's more in theory an infinite amount of aircrafts can be analyzed and computed by the code, as long as their specificities are included in the code (typically, not all tail configurations have been implemented in the code).

The analyses can be manually processed in any python integrated environment, but the use of Jupyter Notebook is recommended. This tool aims to allow the user to run the code without necessarily having to go deep into the code. The most interesting feature of such notebooks is the ability to implement vizualisation widgets to improve the user-friendliness of the code and to ease the postprocessing part of the analyses. At the end of a MDA the code has generated around 500 physical quantities that entirely describe the computed aircraft. Considering this huge number and the format of the .xml files, it would be extremely arduous to manually scroll through the output files to locate the desired quantity. A very useful widget called VariableViewer has been implemented in the earlier

versions of FAST-OAD. It lists in an understandible way all the variables of the .xml file by module and sub-categories.

--ALL-- | data | geometry | wing | --ALL--

	Name	Value	Unit	Description	I/O
104	data:geometry:wing:area	11.805	m**2	wing reference area	IN
105	data:geometry:wing:aspect_ratio	9.149		wing aspect ratio	IN
106	data:geometry:wing:b_50	10.388	m	actual length between root and tip along 50% of chord	IN
107	data:geometry:wing:outer_area	10.058	m**2	wing area outside of fuselage	IN
108	data:geometry:wing:span	10.384	m	wing span	IN
109	data:geometry:wing:sweep_0	0.027	rad	sweep angle at leading edge of wing	IN
110	data:geometry:wing:sweep_100_inner	-0.080	rad	sweep angle at trailing edge of wing (inner side of the kink)	IN
111	data:geometry:wing:sweep_100_outer	-0.080	rad	sweep angle at trailing edge of wing (outer side of the kink)	IN
112	data:geometry:wing:sweep_25	0.000	deg	sweep angle at 25% chord of wing	IN
113	data:geometry:wing:taper_ratio	0.640		taper ratio of wing	IN
114	data:geometry:wing:thickness_ratio	0.150		mean thickness ratio of wing	IN
115	data:geometry:wing:wet_area	21.523	m**2	wet area of wing	IN
116	data:geometry:wing:MAC:length	1.156	m	length of mean aerodynamic chord of wing	IN
117	data:geometry:wing:MAC:y	2.394	m	Y-position of mean aerodynamic chord of wing	IN
118	data:geometry:wing:MAC:at25percent:x	4.272	m	X-position of the 25% of mean aerodynamic chord of wing w.r.t. aircraft nose (drives position of wing along fuselage)	IN

Figure 6 – Overview of the Variable Viewer

This widget is the cornerstone of the postprocessing part of the code. Nevertheless it only displays the value of the computed quantities. For a design engineer this is clearly not enough. FAST-OAD-GA fortunately hosts an appreciable number of vizualisation tools, that plot all sorts of diagrams and tables when called in jupyter notebooks. New functions are continually implemented in order to cover the wide range of figures that are of interest for aircraft preliminary design and help decision-making.

Reference aircrafts are a list of aeroplanes that have been modelled through .xml files and for which a sufficient amount of data is known. These reference aircrafts are used to ensure that the code is functional. They are very useful to test, calibrate and validate the models and the postprocessing functions. By definition running a Multidisciplinary Design Analysis with a reference aircraft should return a .xml file that has the same characteristics as the real aircraft (still taking in account the order of accuracy of preliminary aircraft sizing). The main reference aircrafts are the Beechcraft Model 76 Duchess and the Cirrus SR22. The Beechcraft posseses retractable landing gears, a T-tail and has two motors positioned on the wings. The Cirrus has non-rectractable landing gears and possesses one motor which is placed on the aircraft nose. Both of these aircrafts have internal combustion engines and their geometry is quite conventional.

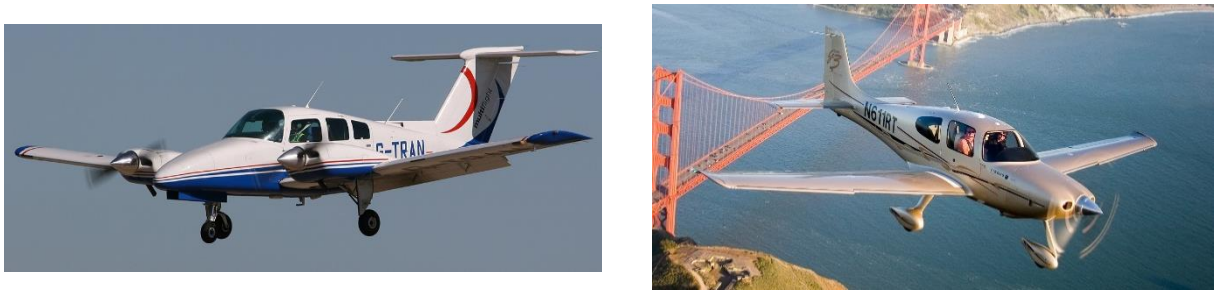

Figure 7 - Beechcraft Model 76 Duchess (on the left) and Cirrus SR22 (on the right)

Obviously the reference aircrafts have to respond to every technology implemented in the code. The TBM 930 has been studied in detail in this thesis to be a new reference aircraft of the code in order to test the turbopropeller propulsion model in FAST-OAD-GA. It also allows to validate the code statistical models for a technologically advanced and modern aircraft. The TBM 930 was introduced in 2016, whereas the Beechcraft 76 has been produced mainly between 1978 and 1983 and the first model of the Cirrus SR22 was manufactured in 2001. The TBM 930 also prepares the way for new modules and models, such as winglets. Being a private business jet, it also possesses some components that lighter aircrafts do not have, such as lavatories.

One of the main ideas behind FAST-OAD-GA is that the potential user already possesses good knowledge about aircraft sizing and has python programming skills. Since the code aims to be modulable and easily adaptable for new aircraft technologies and configurations, the user can theoretically modify all the sections of it. Very few warning and errors messages have been implemented. It is supposed that the user will be aware of the impact of his decisions and that physically consistent choices will be made. The code is permissive and relies on the awareness of the user.

For this thesis work a duplicate of the code has been created thanks to a version control tool called GitHub. As a consequence the changes in the code have been local, but the updates of the main branch of FAST-OAD-GA have also been continuously implemented in the duplicate version. In that way the progress achieved by the rest of the engineer's team has been taken in account and the models developed in the thesis are assured to fit into the code main structure.

Chapter 3

Analysis mode

3.1 Definition of the need

The aim of this new mode is to implement and evaluate design alternatives for general aviation aircrafts. It is interesting for manufacturers and for designers to be able to determine the impact of an incremental modification on a reference architecture. The developed methodology shall allow comparison of different aircraft variants defined by the user and will be evaluated through different use-cases of aircraft modification defined during the thesis. In practice this results in the implementation of a new type of analysis in FAST-OAD-GA, which will be commonly named the analysis mode further in the thesis. This mode has several objectives that are defined in the following section :

- Define an approach to modify the architecture of a reference aircraft. In order to achieve this task, the structure of the code and the idea behind it have to be perfectly known. By doing so new models will have to be implemented and some existing one will have to be refined. The modified aircraft must have the same geometry as the reference one, except for the modification selected by the user.

- Define an approach to effectively compare the resulting modified aircraft and the reference one. This includes defining the values of interest for an aeroplane designer, implementing ways to analyze them and providing visual tools for the user. The reference aircraft architecture also has to be frozen in order to perform the comparison with the modified aircraft.

- Define the global integration of this mode in the whole code in order to avoid conflicts with MDAs and with the work in progress of the other engineers of the team.

- Take in account the objective of modularity of FAST-OAD-GA and define this mode according to this point of view. An external user that would want to code his own aircraft modifications should be able to do so easily without having to modify the whole structure of the analysis mode.

The analysis mode also has to respond to the good programming practices that are an obligation for an open source programme.

3.2 Presentation of the Analysis Mode implemented in this thesis

The aircraft modifications that are considered in the analysis mode have been defined by communicating with DAHER and by thinking of the modifications that would generally interest designers. The resulting range of modifications has been implemented in this

thesis. The computation of a single modification is called a use case. Two main use cases have been defined :

- Fuselage Cabin Stretch use case
- Wing Span Increase use case

As this list illustrates, only geometrical modifications are handled by the analysis mode. They are indeed the most common modifications a designer would be willing to implement, as they greatly impact all the characteristics of the aircraft. Another variation of interest would be to modify the propulsive system of the aircraft (to study a change of technology, of engine performance, of propeller geometry,…). These two use cases will be described in detail in this document. The overall structure of the analysis mode and a small section about postprocessing will however be presented prior to reach that point.

In addition to these use cases, a global analysis can be performed. It aims to study the impact of adding a row of seats in a reference aircraft, while modifying the wing span and the wing longitudinal position in order to remain in the envelopes of the reference aircraft (flight and gust envelope) and to keep the static margins of the modified aircraft equal to the ones of the reference one. This global analysis calls the fuselage cabin stretch use case to add the fuselage length equivalent to a row of seats and it calls the wing span increase use case to stretch the wing in the second part of the analysis.

The following figure schematizes the way the analysis mode is processed. The figure format is analog to the one of figure 2, which allows to easily comprehend the difference between the developed analysis mode and the Multidisciplinary Design Analysis.

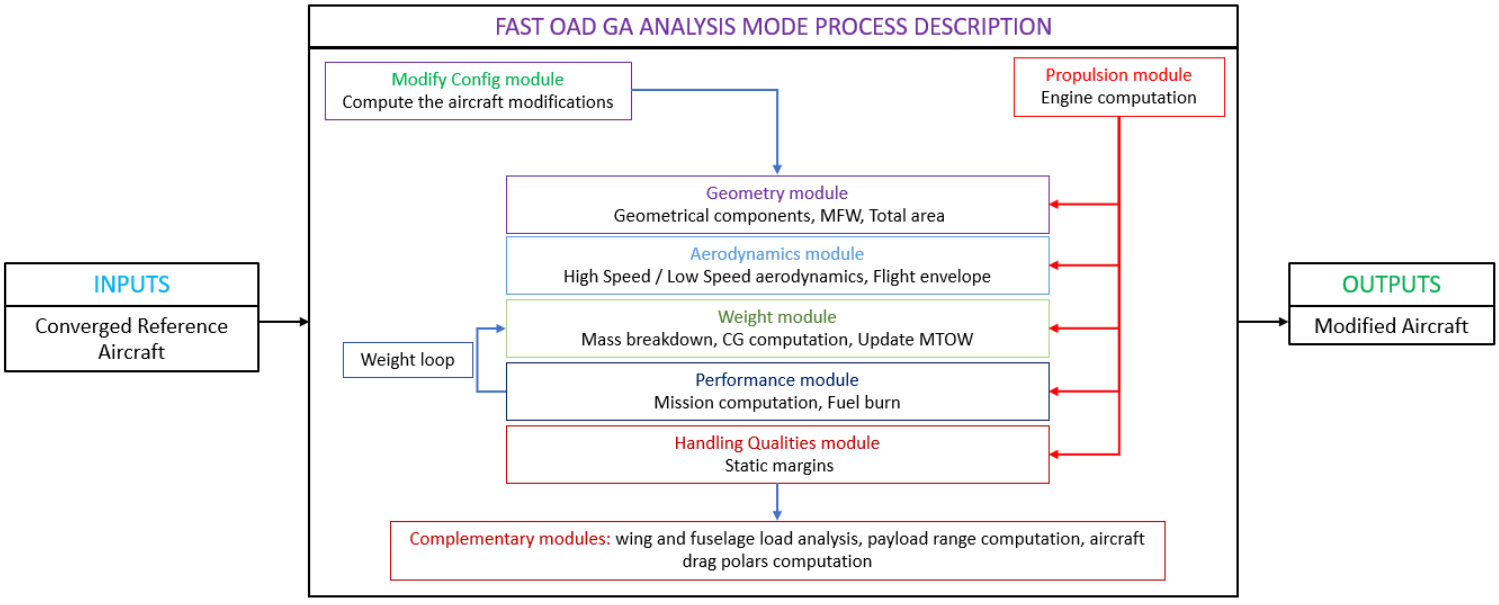

Figure 8 – Overview of the analysis mode structure

First of all the main property of the analysis mode is that it is not an iterative process. With respect to the MDA, at the end of the execution of the modules of the code there is no update of the wings and of the tails geometry. Indeed this process does not aim to generate an aircraft optimized to respond to a set of TLARs but simply to compute the impact of a geometrical modification on an already converged aircraft. The input of the analysis mode is the .xml file of the reference aircraft. This aircraft is the one that will

receive the modifications that are defined by the user. Outputs from MDAs (that is to say converged aircrafts) are preferably used. This means that by computing the geometrical modification on the reference aircraft and by running once all the code modules the output file will correspond to a converged modified aircraft.

To fully understand the analysis mode it is necessary to introduce an Application Programming Interface (API) function of FAST-OAD-GA called generate_block_analysis. This function basically allows to execute a single component of FAST-OAD-GA independently. The only constraint is that the executed component has to be a class defined in the code. This class can very well be an openMDAO group, that is a class that gathers other classes. This way even whole modules can be executed with generate_block_analysis. The diagram below graphically presents the way the function works.

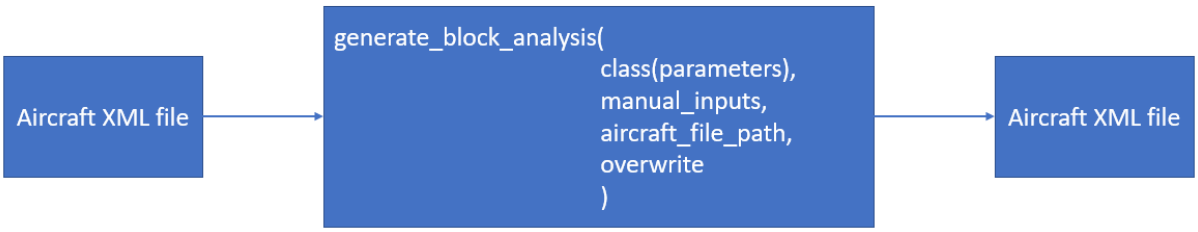

Figure 9 - Overview of the generate_block_analysis function use

The main input and the output of the function is a .xml file that models an aircraft. The parameters of generate_block_analysis are presented below.

- class(parameters) is the class executed by generate_block_analysis. Eventual class parameters like the propulsion ID have to be specified here.

- manual_inputs is a dictionary where the user can choose to specify the value of certain inputs of the class computed in generate_block_analysis. In that case the value specified in the dictionary will override the value of the .xml file. For example if the user wants to manually compute the wing geometry class with generate_block_analysis, he can choose to impose the value of the wing aspect ratio with this functionality and the value chosen will be the one used to size the wing.

- aircraft_file_path is the path of the .xml file that will be used to generate the inputs of the class that is computed in generate_block_analysis.

- overwrite is a boolean. If True, the output .xml file will possess the same path as the input file. In other words the outputs of the class computed with generate_block_analysis will be written on the input .xml file, and if the quantities already exist they will be updated with the new ones. If False, then a new .xml file will be created and will serve as the output file.

It is worth noting that generate_block_analysis cannot be used if the class or the group of class computed uses a quantity that is both an input and an output. This is purely due to the way openMDAO components interact together and it is a limitation of the function. Thus it prevents from executing a class that updates a quantity while also taking this

quantity as an input (for example, a class that would update the airframe weight by multiplying its value by a fudge factor).

Now that generate_block_analysis has been presented the process of the analysis mode will be described in detail in the following part. First of all the reference aircraft file is duplicated. One version of the file will pass through the analysis mode while the other remains untouched. This allows to compare both architectures at the end of the process.

The aircraft modifications are then defined and computed. This is achieved by calling the module specifically created for the analysis mode : Modify Config. This module hosts the classes that compute (ComputeConfigMod) and update (UpdateXML) the quantities of the modified architecture. Depending on the use case, the user has the possibility to go deep into the definition of the modification. This is done thanks to the use of the parameters of the class ComputeConfigMod when calling it with generate_block_analysis. For example in the wing span increase use case for an aircraft with motors under the wings the user can choose to conserve the spanwise position of the engines or to conserve their span-ratio. In the first case the engine position will remain unchanged for the modified architecture and in the second one it will be updated by the code. The geometrical quantities that are updated by the module Modify Config are only the ones that serve as input of the Geometry module. Indeed by definition and as explained in the dedicated section the whole aircraft geometry is computed thanks to a narrow set of quantities. After the Modify Config module has been executed the user disposes of the .xml file of the reference aircraft (unchanged) and of the .xml file of the modified aircraft that has some geometrical quantities updated. These quantities entirely describe the modifications.

The next step of the process is to compute the impact of the geometrical modifications on the aircraft's characteristics. In other words, the .xml file containing the modifications has to pass through all of FAST-OAD-GA modules. The geometry module is logically executed first with generate_block_analysis. At this point the updated quantities that are the input of the geometry module have been read and used by the dedicated classes to estimate the new aircraft geometry. The aerodynamics module is computed next.

The treatment of the weight and the performance modules is a bit special. As explained in the presentation of the MDAs, the weight quantities are defined in this order in the structure of the code at iteration i :

$$weight\ of\ the\ aircraft\ components_i(MTOW_{i-1}) \rightarrow OWE_i$$

$$MTOW_i = OWE_i + payload + fuel_{i-1}$$

$$fuel_i\ is\ obtained\ by\ simulating\ the\ mission\ with\ MTOW_i$$

In a MDA the successive iterations result in the convergence of these quantities, but if this process is only performed once the values will not be consistent. Let's say that the cabin length is increased by 20%. In that case the fuselage airframe mass would be increased by a non-negligible amount. By definition the Overall Empty Weight would also be increased by the same delta of mass. The code would then compute the MTOW of the modified aircraft by taking in account the increase in OWE mass but adding it to the fuel mass of the reference aircraft, since the computation of the mission and of the fuel burn is performed in the performance module that has to follow the weight module. This fuel mass would correspond to the exact mass that allows the reference aircraft to achieve its design mission, but obviously it would not be enough for the heaviest modified aircraft. There is

thus a need to loop on these two modules (weight and performance) in order to get the proper value of the fuel burned in the mission and the resulting MTOW value. This internal process is done with a dedicated .yml file that only calls the weight and the performance module.

```
model:
    nonlinear_solver: om.NonlinearBlockGS(maxiter=50, iprint=2, rtol=1e-7, debug_print=True, reraise_child_analysiserror=True)
    linear_solver: om.DirectSolver()
    weight:
        id: fastga.weight.legacy
#       propulsion_id: fastga.wrapper.propulsion.basicIC_engine
        propulsion_id: fastga.wrapper.propulsion.basic_turboprop
    mtow:
        id: fastga.loop.mtow
    performance:
        id: fastga.performances.mission
#       propulsion_id: fastga.wrapper.propulsion.basicIC_engine
        propulsion_id: fastga.wrapper.propulsion.basic_turboprop
```

Figure 10 – .yml file for the weight and performance modules

After the weight related quantities have converged, the remaining modules of the code can be executed with generate_block_analysis. It is worth noting that only half of the handling qualities module is computed. Indeed in this type of analysis there is no need to size the tails and to update their geometry as the modified aircraft must have the same geometry as the reference one except for the chosen modifications. Modifications of the tail geometry are not part of the implemented use cases. The remaining modules to compute are the ones reffered as the complementary ones, that is to say the load analysis module, the calculation of the payload range and of the aircraft drag polars.

The propulsion module has the same role as in a Multidisciplinary Design Analysis.

There is no need to compute the propeller characteristics since is has already been done in the MDA which gives the reference aircraft file, and the propeller remains unchanged for the modified aircraft.

Regarding the files management the analysis mode does not use yml files (except for the internal loop for the weight and performances modules). The tool that describes the order of the modules to be computed and that calls generate_block_analysis is Jupyter notebook. This allows to simply choose the reference aircrafts and to clearly define the modifications. The following diagram represents the way the files interact with each other.

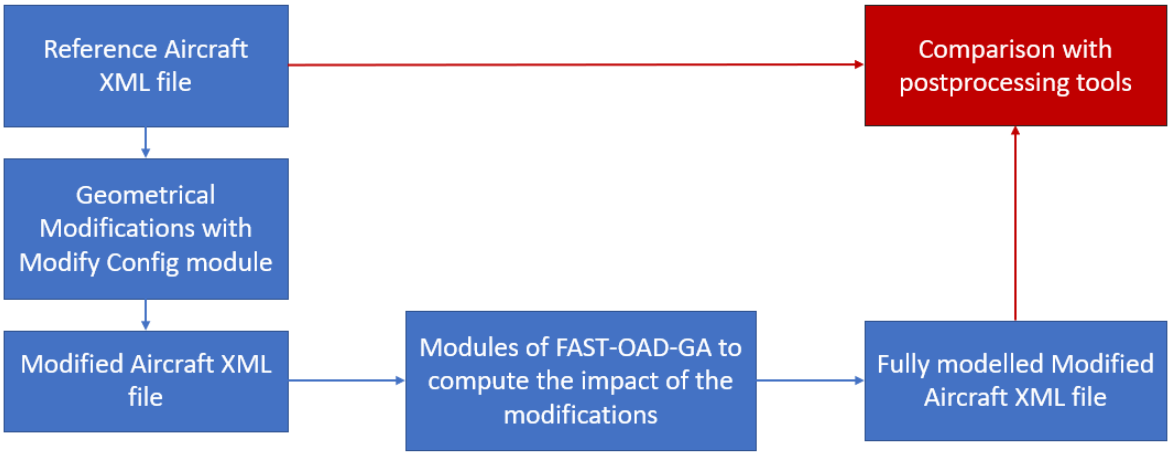

Figure 11 – Transfer of data between files in the analysis mode

At the end of a use case computation, the user disposes of the fully modelled modified aircraft .xml file in addition to the untouched reference aircraft .xml file. These two files can now be exploited by the postprocessing utilities in order to effectively compare the aircrafts.

3.3 Discussion on the limitations of the analysis mode

The main limitation of the analysis mode is the fact that the only architecture modifications that can be implemented are the one that have been defined and coded beforehand in the Modify Config module. The user cannot simply choose one quantity of the .xml file, modify its value and apply this modification. This feature in fact already exists in the code since this process can be done by manually setting the value of some input quantities of a class with generate_block_analysis through the dictionary manual_inputs. The disadvantage of such an approach is that the user needs to be perfectly aware of the whole code structure. For example if the user chooses to modify the value of the tip chord of the wing but forgets to adjust the taper ratio accordingly, the root chord will also be modified and the user may not take notice of it immediately. In the analysis mode the modifications related to the use cases are very well defined, studied and implemented, meaning that the user has a real knowledge about the consequences of his choices. For each use case the user has access to several parameters the value of which affect how the code will treat the use case computation. Thus the user by just switching on or off a boolean parameter can implicitly modify a lot of quantities of the .xml file. This process cannot be as easily done if the user has to manually vary all the quantities related to a single change.

On the other hand the analysis mode has been specifically studied to be user-friendly and to potentially host new use cases. A user wanting to implemented its own use case just has to follow the existing structure of the existing use cases. The only module he would have to modify is Modify Config, meaning that the analysis mode update and modification does not have any negative impact on the other components of the code.

Chapter 4

Post Processing

This short section aims to briefly introduce the postprocessing part of FAST-OAD-GA. It was part of the thesis' work to develop and implement new visualization tools to properly analyze the outputs of the code. Sometimes the quantities that need to be analyzed have already been computed by a code module and only the function extracting and shaping them has to be coded. But most of the time the related physical models have to be developed. That is the case for the computation of the aircraft drag polars or the computation of the payload range diagram. Both of these postprocessing tools didn't exist prior to the thesis and their implementation required to study and adapt their models to take in account general aviation and the structure of the code. The plots that have been added during the thesis result from a brainstorming in the engineer's team. The aircraft sizing books also assemble a great amount of such figures. The idea was to adopt the point of view of an aircraft designer and to try to think about the tools that would facilitate decision-making. All the postprocessing functions are gathered into a dedicated jupyter notebook.

The following diagrams have been adapted from FAST-OAD or were already in FAST-OAD-GA prior to this thesis and have been simply updated to match eventual evolving needs.

- Plot of the aircraft geometry (top view) with highlighting of the engine and nacelle positions. There is the possibility to superpose the geometry of multiple aircrafts on the same plot.

- Flight envelope and gust envelope of the aircraft. There is the possibility to superpose the envelopes of multiple aircrafts on the same plot.

- Bar plot of the global mass breakdown of the aircraft (MTOW, OWE, fuel, payload). There is the possibility to superpose the mass breakdowns of multiple aircrafts on the same plot.

- Sun plot of the OWE mass breakdown of the aircraft.

The following diagrams have been fully conceived and implemented in this thesis :

- Lift coefficient distribution along the span of the wing. This diagram only has comparisons objectives. It needs two aircraft files in input in order to be generated. It plots two figures. The first one is the superposition of the lift coefficient distribution along the span of the two input aircrafts. The second figure is the distribution of the absolute difference in magnitude of the two distributions along the span.

- Lateral view of the aircraft with a highlight of the barycenter position of the empty aircraft (without payload and without fuel). The aft and fwd extreme positions of the CG which depend on the ground and the flight load cases are also plotted. These

three quantities when linked form the range of the aircraft barycenter. Finally the diagram displays the value of the static margins (stick fixed and stick free) of the aircraft. There is the possibility to superpose the lateral geometry and the barycenters of multiple aircrafts.

- Payload Range diagram. There is the possibility to superpose the payload range diagram of multiple aircrafts.

- Aircraft drag polars. The non equilibrated (without the influence of the horizontal tail) polar and the equilibrated one (trimmed aircraft) can both be displayed. Regardless of the polar type plotted, two figures are displayed : the first one corresponds to the cruise conditions (cruise speed and altitude, control surfaces with low angle of activation). The second figure is the display of the drag polar in low speed conditions (approach speed, flaps activated). There is the possibility to superpose the drag polars of multiple aircrafts.

- Sun plot of the drag breakdown of the aircraft. Both the induced drag and the parasite drag are displayed and the contribution of the aircraft components to the parasite drag can be visualized in detail. As for the aircraft drag polars, two diagrams are plotted which are related to the cruise flight point and to the low speed flight point.

Chapter 5

Use Cases : Fuselage Cabin Stretch

5.1 Use Case Description

The use case that is going to be described in detail in this section is the stretch of the fuselage cabin. The need that motivated the design of this use case is the will to increase the length of the fuselage cabin, while maintaining the wings and the tails geometries constant. The total fuselage length would also increase. The impact of such a change concerns mostly the weight and balance part of the aircraft, with a great variation of the static margins. The use case presented in this section is the answer to this need that has been achieved in this thesis.

As described in the analysis mode presentation the user has to manually define the modification of the aircraft in the notebook dedicated to the use case. Below is a screenshot of all the settings the user can play with for the stretch of the cabin.

```
# Cabin length increase parameters
added_length = 0
added_section_x_ratio_front = 1
added_section_x_ratio_rear = 0
added_pax = 2
added_luggage = 10

fuselage_mod_parameters = [added_length, added_section_x_ratio_front, added_section_x_ratio_rear, added_pax, added_luggage]
```

Figure 12 – Parameters of the fuselage cabin stretch use case

added_length [m] : This is the main parameter of the use case. It basically specifies the length in meters by which the cabin will be stretched. The user also has the possibility to set this parameter's value to 0, which is the default value in the jupyter notebook. In that case the length of a row of seats will be taken (this length is defined in the .xml file of each aircraft and will vary based on the aircraft analyzed) and the code will also automatically increase the maximum number of passengers of the aircraft (the value is increased by the number of seats contained by a row which is also a data of the .xml file). It allows to analyze with ease the impact of the addition of a row of seats without having to look for the corresponding length in the .xml file and having to manually change the other related quantities. It is worthy to note that if the user chooses to specify his own length (for example added_length = 0.5 m) then the code will never increase the number of maximum passengers, nor the maximum freight mass. The added section will simply be an empty space in the cabin behind the freight.

added_section_x_ratio_front : Float between 0 and 1 defining the percentage of the added fuselage section that will be placed ahead of the x-position of the 25% of the wing Mean Aerodynamic Chord (MAC). The x-coordinate refers to the axis going from the nose of the aircraft to its tail.

added_section_x_ratio_rear : Float between 0 and 1 defining the percentage of the added fuselage section that will be placed at the back of the x-position of the 25% of the wing Mean Aerodynamic Chord. By definition the sum of added_section_x_ratio_front and added_section_x_ratio_rear must be equal to one.

added_pax : Integer stating the number of passengers added to the design mission. For this parameter the user has to take care not to exceed the maximum number of passengers the aircraft can hold. This value is defined in the .xml file of the aircraft. As described above, if the user chooses to specify his own length then the maximum number of passengers will not increase, but the user can still choose to add passengers to the mission if in the reference mission the aircraft was not filled at maximum capacity. If the user instead adds a row of seats, the maximum number of passengers is increased but the code will not automatically consider that the new seats are occupied by passengers. In that way the user can decide not to add extra passengers to the design mission, an action that has an impact on the mass of the payload of the mission and on the mass distribution along the cabin.

added_luggage [kg] : Float stating the mass of luggage added to the design mission. This parameter has to be dealt with the same way as the added_pax parameter, in the sense that the user has to take care not to exceed the maximum freight capacity of the aircraft (quantity also defined in the .xml file). In the use case increasing the cabin length will not result in a bigger maximum freight weight so this quantity remains invariant with respect to the reference aircraft. Nevertheless it allows the user to study the impact of a variation of payload.

There is no limit in the code for the increase in cabin length (if the user sets a value for the parameter added_length), but some modules will have problems running with an extreme value. A cabin length reduction can also be theorically computed by specifying a negative length, but the consequences on the code have not been studied.

As can be seen on figure 11, these parameters are all clustered in an array called fuselage_mod_parameters. This array will be provided to the Modify Config class as its parameter when it is executed by the generate_block_analysis function.

The two parameters added_section_x_ratio_front and added_section_x_ratio_rear allow the user to choose and to compare multiple configurations of the added section position with respect to the x-position of the 25% of the Mean Aerodynamic Chord of the wing. Three configurations are possible and are presented below. The following three figures present the output of the postprocessing function aircraft_geometry_plot without displaying the nacelles. The reference aircraft is the Beechcraft Model 76 Duchess. The cabin length is increased by the length of a row of seats in all three cases (parameter added_length = 0). For this particular aeroplane this length is equal to 0.8 m. The figures display the superposition of the geometry of the reference aircraft in blue and of the resulting modified aircraft in red. Please note that from now on the term wing MAC 25% will be used to refer to the x-position of the 25% of the mean aerodynamic chord of the wing.

1. The new section of the fuselage is fully added ahead of the wing MAC 25% (added_section_x_ratio_front = 1, added_section_x_ratio_rear = 0). In that case the wings are moved back by the length of a row of seats with respect to the reference aircraft.

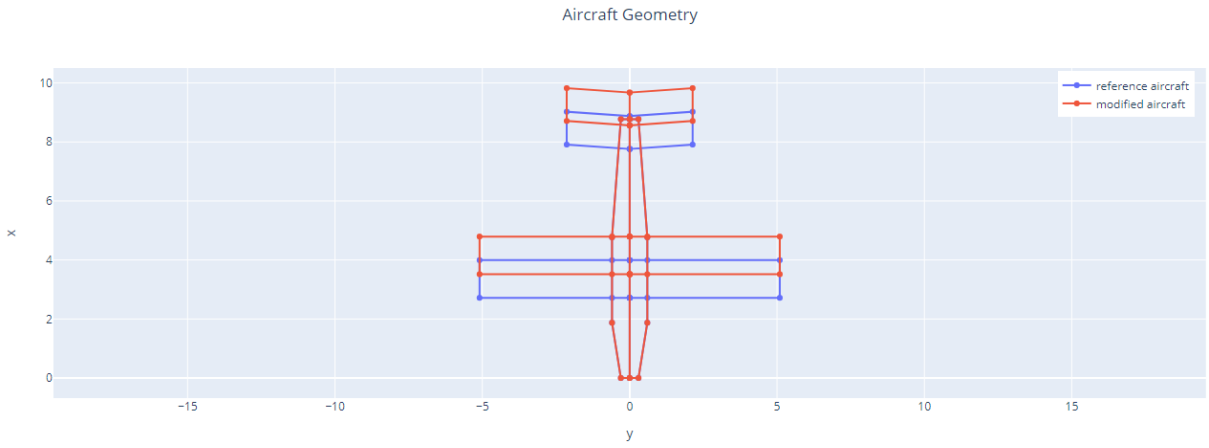

Figure 13 - Geometry Plot of the cabin stretch with new section ahead of the wings

2. The added section of the fuselage is fully behind the wing MAC 25%. (added_section_x_ratio_front = 0, added_section_x_ratio_rear = 1). The wings of both aircrafts perfectly superimpose.

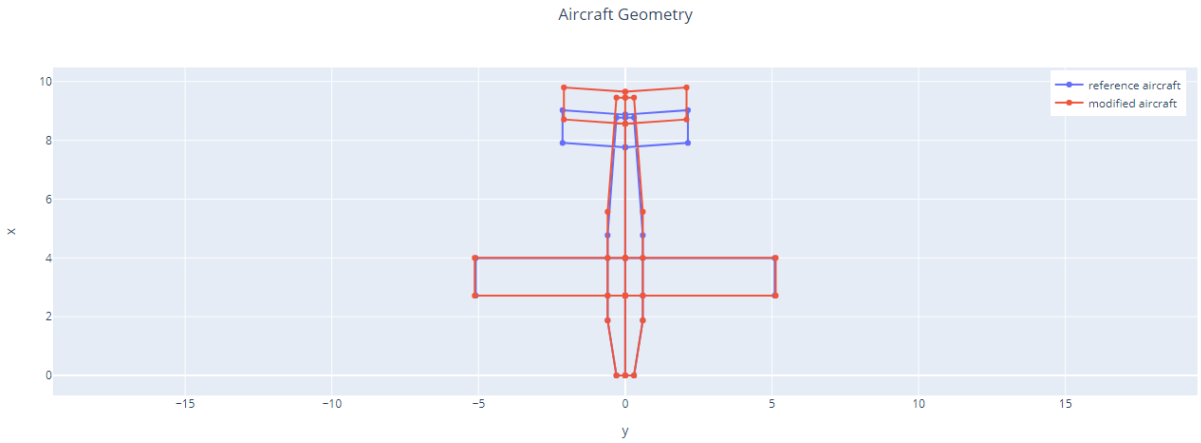

Figure 14 - Geometry Plot of the cabin stretch with new section at the back of the wings

3. Part of the added section is ahead of the wing 25% and part of it is behind the wing MAC 25% (added_section_x_ratio_front = x, added_section_x_ratio_rear = 1-x where $0 < x < 1$). The figure below has added_section_x_ratio_front = 0.3 and added_section_x_ratio_rear = 0.7. So 30% of the added fuselage section is ahead of the wing MAC 25% and 70% is behind it.

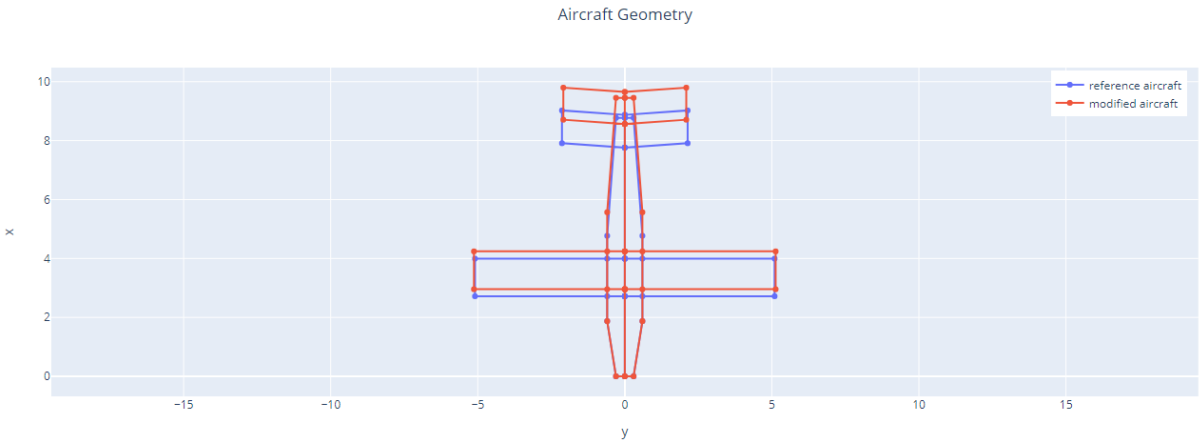

Figure 14 - Geometry Plot of the cabin stretch with new section 30% ahead of the wings and 70 % beyond them

As can be seen from these figures the total fuselage length is the same for all three configurations, so all three modified aircrafts will have more or less the same weight. What varies considerably with the choice of the configuration is the balancing of the aircraft and the x-position of the center of gravity. The structural sollicitation of the fuselage is also very likely to increase. However prior to the thesis the code did not perform any load analysis of the fuselage. A model has therefore been developed in order to be able to perform this type of analysis. While developing this method, it appeared that the fuselage airframe mass model did not give enough detail and thus arose the need for a refined mass model of the airframe of the fuselage. This use case is a perfect example of how considering a new approach allows to discover the areas for improvement of the code. The two models are described in the next section.

5.2 Models to add to the code

As just mentioned, two models related to the fuselage have been developed in this thesis :

- A semi-analytical weight model of the fuselage airframe. This model aims to replace the currently used statistical model whose accuracy is questioned for aircrafts with modern and innovative architectures.

- An analytical model of the bending moment distribution over the length of the fuselage. There was no equivalent approach in the code before the work of the thesis. This model constitutes the main feature of the load / structural analysis of the fuselage, which is part of the load analysis module. As a consequence, this model will not be used to size the aircraft in a Multidisciplinary Design Analysis but to compute additional quantities to analyze the aircraft in the postprocessing part of the code.

Both of these models will now be presented in detail.

5.2.1 Fuselage airframe weight model

The airframe of the fuselage is primarily constituted by the fuselage skin. This thin aluminum envelope cannot stand all the fuselage loads by its own. Instead of increasing its thickness, method that would result in a great increase of the fuselage weight, the skin is reinforced and stiffened by longitudinal elements (stringers and longerons) and circular elements (frames). The terminology fuselage shell is used to talk about the skin and its reinforcements. This particular assembly is very useful for structural analyses, since it allows to make a couple of assumptions that are close to the reality. For example a regular assumption is that the shear and normal stresses due to pressurization are taken by the skin only, or that the frames do not contribute to bending. It amounts to assuming a typical aluminum fuselage structure where the stringers are contiguous and solidly riveted to the skin, but the frames are either offset from the skin or have cutouts for the stringers which interrupt the frames circumferential loads (also called hoop loads). In the following section the term stringers refers to both stringers and longerons.

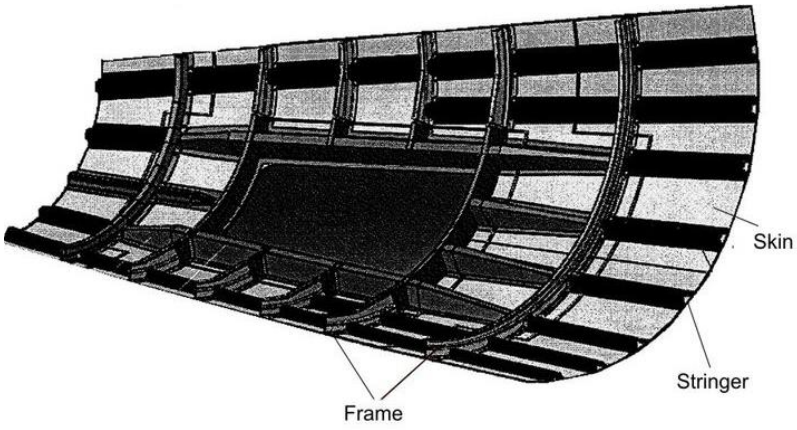

Figure 15 – Main structural components of the fuselage airframe

In the code the fuselage airframe also includes the components added to the fuselage shell to make it fully functional for flight, such as the aircraft doors, windows, hatches, potential pressure bulkheads and the floor. Depending on the aircraft configuration, local reinforcements to support the wing structure or the fuselage mounted engine are also considered in the airframe of the fuselage. The weight of the passenger seats is not covered by this section of the code. There is indeed a dedicated class of the mass breakdown which handles them.

There are two statistical models implemented in the code that predict the airframe fuselage weight for general aviation. The first one has been proposed by Nicolai and is illustrated by Gudmundsson in [7] and the second one has been developed by Raymer in [9]. The Raymer model takes in account an potential pressurization of the aircraft. British units are used in both of these models.

$$W_{fuselage\ airframe\ Nicolai} = 200\left(\left(\frac{MTOW * n_{ult}}{10^5}\right)^{0.286}\left(\frac{l_{fuse}}{10}\right)^{0.857}\left(\frac{w_{fuse}+h_{fuse}}{10}\right)\left(\frac{V_{\max\_SL}}{100}\right)^{0.338}\right)^{1.1}$$

Where $V_{\max\_SL}$ is the aircraft maximum speed at sea level in kn. l_{fuse}, w_{fuse}, h_{fuse} are respectively the length, the width and the maximum height of the fuselage in feet.

$$W_{fuselage\ airframe\ Raymer} = 0.052\, S_{fuse}^{1.086}(MTOW * n_{ult})^{0.177} l_{wing-HTP}^{-0.051} \left(\frac{l_{cabin}}{h_{fuse}}\right)^{-0.072} q^{0.241} + 11.9(V_{cabin}\Delta P)^{0.271}$$

Where S_{fuse} is the fuselage wetted area in ft**2, $l_{\text{wing-htp}}$ is the distance between the 25% of the horizontal tail mean aerodynamic chord and the 25% of the wing mean aerodynamic chord in feet. q is the cruise dynamic pressure in lb/ft**2, V_{cabin} is the volume of the cabin in ft**3 and ΔP is the maximum sizing pressure differential of the aircraft in psi.

The default model used by FAST-OAD-GA is the Nicolai one. These models are quite accurate for conventional aircrafts but they obviously need some refinement for more modern aircrafts or unconventional configurations. The need for a more precise model has also been enhanced by the will to perform basic structural analysis of the fuselage.

The model developed during this thesis is based on two complementary approaches :

- The sizing of the fuselage main structure with a computation of the required skin thickness based on load cases derived from the CS-23 certification for pressurized aircrafts, and based on industrial constraints and data for unpressurized aircrafts. The secondary structural reinforcements such as stringers and frames are modelled with statistical models from Torenbeek in [6].

- The weight penalty method introduced by Torenbeek in [6] is used to implement the additional essential components of the fuselage such as the doors and the windows. It also takes in account the weight of the local reinforcements needed to install these components as well as the fuselage skin material removed by their fitting. Each of these models takes in account the possible pressurization of the cabin.

5.2.1.1 The sizing of the fuselage skin

The fuselage of a general aviation pressurized aircraft is modeled as a single bubble pressure vessel whose cross-section is supposed to have a uniform thickness t_{skin}. The cabin is assumed as cylindrical, the nose of the aircraft is modeled as a partial ellipse and the rear part of the fuselage is modeled as a cone and is commonly denominated the tail cone. Two flat pressure bulkheads separate the cabin from the nose and the tail cone. This results in the following scheme :

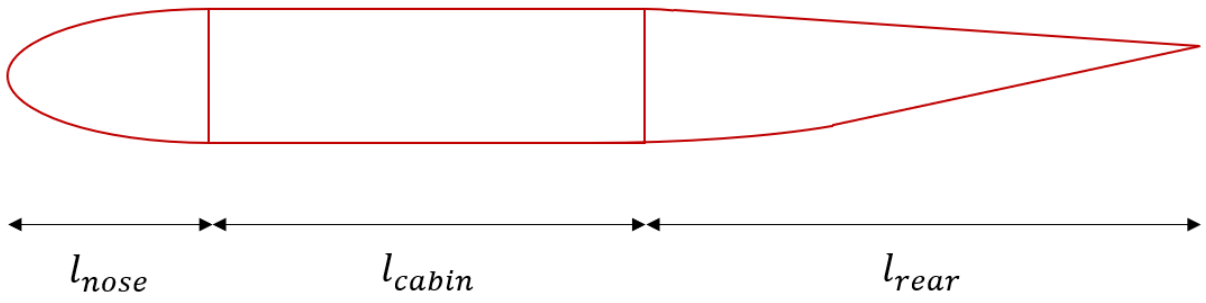

Figure 16 – Assumed layout of the fuselage

This geometry and in particular the tail cone is inspired by the fuselage description made by the TASOPT in [12] for large commercial aircrafts. It has been adapted in this thesis to fit general aviation aircrafts. The conventional shape of GA aircrafts (rear length close to cabin length) is recognizable in this figure. For unpressurized aircrafts the geometry of the fuselage is the same but the bulkheads are removed.

The pressurization load generated by the pressure differential between the cabin air and the outside air produces axial and circumferential (or hoop) stresses in the fuselage skin, with the assumption that the stringers share the axial loads, but the frames do not share the hoop loads. These two loads are well defined in the literature for cylindrical sections with constant thickness. The TASOPT restates their expression :

$$\sigma_{axial} = \frac{\Delta p}{2}\frac{R_{fuse}}{t_{shell}}$$

$$\sigma_{hoop} = \Delta p \frac{R_{fuse}}{t_{skin}}$$

Where Δp is the maximum supported pressure differential in Pa, R_{fuse} is the fuselage radius in meters, t_{skin} is the thickness of the fuselage skin in meters, and t_{shell} is the equivalent thickness of the fuselage shell in meters that is to say the assembly of the skin, the stringers and the frame.

The fuselage skin thickness is sized by the larger load value of σ_{hoop}. The CS-23 certification has to be considered at this point. The following extracts directly come from the 4th amendment of the CS-23 [11].

« Strength requirements are specified in terms of limit loads (the maximum loads to be expected in service) and ultimate loads (limit loads multiplied by prescribed factors of safety). […] Unless otherwise provided, a factor of safety of 1·5 must be used.» (European Aviation Safety Agency, 15 July 2015)

« The aeroplane structure must be strong enough to withstand the pressure differential loads corresponding to the maximum relief valve setting multiplied by a factor of 1·33, omitting other loads. » (European Aviation Safety Agency, 15 July 2015)

About the value of the stress that has to be taken to size the fuselage skin thickness here is what the norm states :

« The structure must be able to support limit loads without detrimental, permanent deformation. At any load up to limit loads, the deformation may not interfere with safe operation. » (European Aviation Safety Agency, 15 July 2015)

« The structure must be able to support ultimate loads without failure for at least three seconds. » (European Aviation Safety Agency, 15 July 2015)

Thus for the limit loads the maximum pressure differential will be multiplied by 1.33 and the appropriated stress is the elasticity limit of the material. For the ultimate loads an additional factor of safety of 1.5 will be applied on the limit loads. It may be noted that this analysis does not provide any way to verify that the structure is effectively able to support the ultimate loads for at least three seconds.

Consequently two sizing equations are obtained for the fuselage skin thickness calculation.

$$t_{skin} = 1.33\,\Delta p \frac{R_{fuse}}{\sigma_{02}}$$

$$t_{skin} = 1.5 * 1.33\,\Delta p \frac{R_{fuse}}{\sigma_{02}}$$

Both results are computed in the code and by definition the most constraining one is chosen as the definitive skin thickness value. Obviously in this case the ultimate load formula will be chosen. Manufacturing limitations and constraints are also taken in account in this analysis. Indeed fuselage sheet manufacturers also have to respond to certain challenges and obligations, which render them unable to produce sheets under a limit value of thickness. A study of general aviation manufacturers websites and data demonstrated that a technological minimum fuselage skin thickness of 0.5 mm is appropriate. Thsi value is locally implemented in the model of the code so it does not depend on the aircraft analyzed. In the end the code selects the highest skin thickness value between the technological one and the one given by the structural analysis and the CS-23 certification requirements. For unpressurized aircrafts the technological value is automatically selected by the code.

The material chosen for the fuselage skin, the stringers and the frames is common aluminum whose mechanical properties are presented below. The data used comes from an analysis made on aluminum sheet manufacturers :

- Density ρ: 2700 kg/m3
- Young Modulus E : 69.5 GPa
- Elasticity limit of the material σ_{02} : 110 MPa

Once the fuselage uniform skin thickness has been obtained the volume of the entire fuselage is computed. The fuselage is cut into three separate sections already described :

- The elliptical nose volume estimation uses Cantrell's approximation for the surface area of an ellipsoid :

$$V_{nose} = t_{skin} * 2\pi R_{fuse}^2 \left[\frac{1}{3} + \frac{2}{3}\left(\frac{l_{nose}}{R_{fuse}}\right)^{\frac{8}{5}}\right]^{\frac{5}{8}}$$

Where l_{nose} is the length of the fuselage nose in meters.

- The volume of the cylindrical cabin delimited by two flat bulheads is trivial to obtain :

$$V_{cabin} = t_{skin} * 2\pi R_{fuse} * l_{cabin}$$

Where l_{cabin} is the length of the cabin in meters.

- The volume of the tail cone is computed following its definition in the TASOPT :

$$V_{tail\ cone} = t_{skin} * \pi R_{fuse} * l_{rear}\,(1 + \lambda_{cone}^2)^2$$

Where λ_{cone} is the taper ratio of the cone's enclosed area and l_{rear} is the length of the rear part of the aircraft hosting the tail cone in meters.

The total volume of the fuselage is obtained by simply summing its components :

$$V_{fuselage\ skin} = V_{nose} + V_{cabin} + V_{tail\ cone}$$

The final step to get the fuselage airframe skin mass is to multiplying the fuselage skin volume with the density of the material.

$$W_{skin} = V_{fuse\ skin} * \rho_{skin}$$

The mass of the skin reinforcements ie the stringers and the frames is computed with statistical models proposed by Torenbeek in [6]. These models do not depend of the pressurization of the aircraft.

$$W_{stringers} = 0.0117 * k_{\lambda} * S_{fuse}^{1.45} * V_D^{0.39} * n_{ult}^{0.316}$$

Where k_{λ} is a factor allowing for the influence of the fuselage slenderness ratio. It is defined in detail in the book and its computation will not be the object of the current section. S_{fuse} is the wetted area of the fuselage in m**2, V_D is the aircraft design diving speed in m/s and n_{ult} is the sizing ultimate load factor that the aircraft is designed to encounter.

The model giving the mass of the frames depends on the mass of the fuselage skin and stringers obtained above.

$$\begin{cases} W_{skin} + W_{stringers} > 286\ kg \rightarrow W_{frames} = 0.19\ (W_{skin} + W_{stringers}) \\ W_{skin} + W_{stringers} \leq 286\ kg \rightarrow W_{frames} = 0.0911\ \left(W_{skin} + W_{stringers}\right)^{1.13} \end{cases}$$

The gross shell weight which is the sum of all the fuselage airframe main structural components can now be computed. This weight is the base of the analysis done in the next section.

$$W_{shell} = W_{skin} + W_{frames} + W_{stringers}$$

The specific gross shell weight is also defined. It is the ratio of the weight of the shell on its wetted area (which corresponds to the wetted area of the skin).

5.2.1.2 Presentation and adaptation of the Torenbeek weight penalty method

The Torenbeek method is applied on the estimation of the fuselage gross shell weight. It considers a wide set of extra components that need to be installed on the fuselage airframe by cutting or removing skin material. Each component is processed in the following way :

1. The weight of the component and of its eventual surroundings is computed using the statistical models proposed by Torenbeek. For the majority of the components the model depends on the pressurization of the aircraft and on the assumed geometry of the component (mainly its wetted area). This area is estimated according to statistical analysis of the reference aircrafts of the code.

2. The wetted area of the component that has been used to compute its mass in the first point is also equivalent to the area of shell to remove to install the component. Thus the weight of removed shell material is obtained by multiplying the specific gross shell weight by the wetted area of the opening.

3. The total resulting mass is then : $W_{to\ add\ to\ shell} = W_{single\ component} - W_{shell\ removed}$

This procedure is performed for the following components : passenger and crew doors, nose landing gears door, cabin windows and cockpit window glazing. The resulting weight for each of these components has been separately studied to determine if the models were realistic enough for the reference aircrafts of FAST-OAD-GA. It turned out that the statistical model computing the weight of the cockpit window glazing way overestimates actual manufacturer data. As a consequence the choice has been made to ignore the Torenbeek model for this particular component, and to compute it in another way. A simple geometrical and analytical approach has been investigated. The typical thickness of a cockpit window glazing has been estimated by analyzing manufacturers data. This analysis also allowed to estimate the average mass density of the material used to produce these special windows. By estimating that such a window occupies half the fuselage maximum height and occupies the whole fuselage width the following model has been implemented in the code :

$$W_{cockpit\ glazing\ window} = height_{window} * thickness_{window} * width_{window} * density_{material}$$

The surroundings of this component are then computed using the dedicated Torenbeek model and the total weight of the component is obtained by considering the removed material of the shell due to the cutout made in the fuselage.

Torenbeek also proposes several statistical models that represent airframe components or support structures that do not require to remove material from the shell : the floor, potential pressure bulkheads, potential engine support structure for a nose or rear mounted engine configuration and the support stucture for the wing / fuselage connection. The statistical models related to these components are not going to be presented here, but they have been analyzed prior to being included in the code.

The final fuselage airframe mass is obtained by summing the shell and all its components. The default aircraft configuration implemented in the code is the following one :

$$W_{fuselage\ airframe} = W_{shell} + 4 * W_{window} + 2 * W_{door} + W_{nlg\ door} + W_{cockpit\ window}$$
$$+ W_{floor} + W_{wing\ connection} + 2 * W_{bulkhead} + W_{engine\ support}$$

The two quantities in red are optional weights that depend on the aircraft configuration. FAST-OAD-GA is perfectly able to autonomously determine if the analyzed aircraft requires to compute them or not by reading the .xml file.

5.2.1.3 Results

The following table sums up the main results obtained by computing and comparing the proposed mass model with the Raymer and Nicolai models. The developed model uses the above-mentioned default fuselage configuration for the three aircrafts. The TBM 930 is

the only pressurized aircraft of the list, so the code only computes the bulkheads mass for it. Similarly, the TBM 930 and the Cirrus SR22 both have a nose mounted engine so in their case the code computes the weight of the engine support structure. For the TBM 930 the real value of the fuselage airframe mass has been communicated by DAHER. Nevertheless due to the sensitive nature of this information the value will not be written in this report. It can serve as a reference point for the validation of the models. For the Beechcraft Duchess 76 and the Cirrus SR22 this data is unfortunately unknown. All the masses in this table are expressed in kg.

	Beechcraft Duchess 76	Cirrus SR22	TBM 930
Nicolai	156.6	145.6	250.2
Raymer	160.7	154.9	233.4
Model Proposed	152.8	174.1	315.2
Aircraft Data	Unknown	Unknown	Known

Table 1 – Comparison of the models with aircraft data

First of all it can be seen that Nicolai and Raymer statistical models give pretty close results, that is in the relative accuracy of preliminary aircraft sizing. No reference point is available for the Beechcraft and the Cirrus, but since these two aircrafts are conventional it can be fairly assumed that the real mass is close to the ones that Nicolai and Raymer estimate. On the other hand, the output of these models shows a lack of accuracy for the TBM 930 case. Since there are very few aircrafts similar to the TBM 930, it seems logical that it does not fit well with the rest of the aircrafts in the statistical models database. Comparing with the value indicated by DAHER it seems that the developed model shows a much better accuracy than the statistical models for the TBM 930. Interestingly, it is also very close to the output of the statistical models for the Beechcraft Duchess Model 76. The developed model nevertheless deviates a bit from Nicolai and Raymer for the Cirrus SR22. The main reason is the fact that some parts of the airframe of the Cirrus SR22 are made of composite, and that the developed model always estimates the weight of the airframe based on an Aluminum structure. It is possible to find fudge factors for composite structures in the literature that are specifically studied for aircraft conceptual design. Gudmundsson states that applying a correcting factor of 0.85 gives satisfying results for a composite airframe [7]. In that case the result obtained with the Cirrus SR22 is 147.9 kg and the developed model fits well between Nicolai and Raymer results. The first conclusions of this analysis are that the developed model has the ability to adapt well to the type and to the technology level present in the aircraft. The main limitation of this model is that it relies on several statistical models the accuracy of which cannot be proven. Still it is convenient for the following reasons :

- Since the model is a bit more profound and detailed than a single statistical model, its results and its errors can be explained and analyzed more easily because they don't rely solely on the chosen database of aircrafts.

- The user can adjust the number of components of the fuselage to match a specific aircraft.

- The user controls every parameter of the model such as the fuselage skin material, the industrial constraint about the skin thickness, the number of components, etc. This flexibility (impossible to achieve with the statistical models of Raymer and Nicolai) allows to easily test several fuselage configurations without having to redefine new models or to go deep into the code.

- As already mentioned the weight of the components of the fuselage is mainly obtained throught statistical models. However the code structure allows to modify and update these models with more precise ones if the need for an improved accuracy is expressed.

- The inherent advantage of the Raymer and Nicolai models is the very fast computational time. Switching between a statistical model and an analytical one often results in a loss of computational efficiency and of speed. In this case however the developed model does not take much time than the current models to execute since it does not include any resource-consuming algorithms or loops.

These points make the developed model more valuable for the code than the Raymer and Nicolai ones. It is thus validated.

5.2.2 Load Analysis of the fuselage model

The load analysis module has been developed a few months prior to the beginning of this thesis. The main effort was focused on the structural analysis of the wings. However it seemed necessary to extend the spectrum of similar analyses for other structural components of a general aviation aircraft. It was decided that the fuselage should be the priority. The creation of the fuselage stretch use case motivated the parallel study of the fuselage loads. Indeed the designers and the engineers would greatly benefit from a model that would show the impact of a fuselage stretch on the way it is sollicitated. Even if the load analysis module is not part of the primary loop of the Multidisciplinary Design Analysis that sizes the aircrafts, its results cannot be neglected for design choices. A load analysis model fit for preliminary aircraft sizing and for Overall Aircraft Design needs to remain relatively basic. The level of detail in the available and computed aircraft data does not allow to perform an in-depth study involving external software or sophisticated methods. The output of the analysis must however be sufficiently detailed to be reliable and useful. It has been decided that an appropriate model would be the determination of the bending moment distribution over the fuselage length. This distribution would be rough but would anyway take into account the heaviest components of the fuselage and consider different ground and flight conditions. The work accomplished in this thesis and portrayed in this section is the answer to this need.

The model developed in FAST-OAD-GA is derived from the TASOPT [12]. In the paper the proposed model aims to compute the bending moment over the whole fuselage length of a CS-25 aircraft. The following figure schematizes and sums up the main features of this

model. As the model implemented in the code draws a lot from the TASOPT approach, its introduction allows to easily understand the work done.

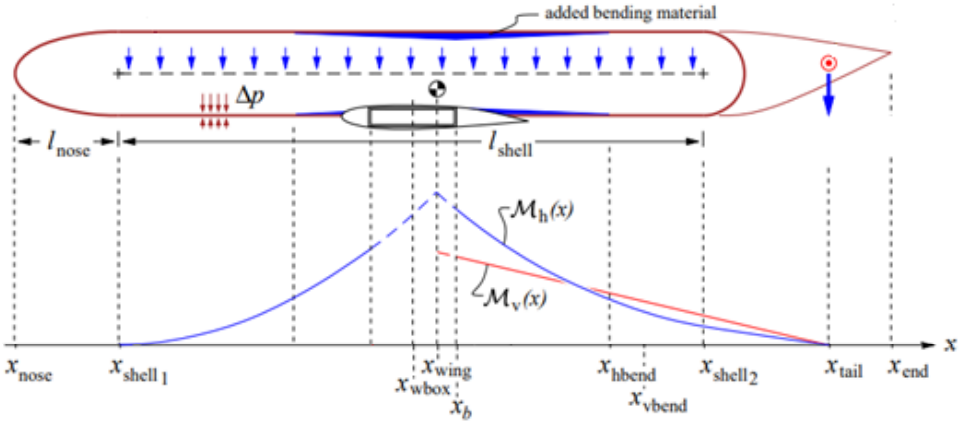

Figure 17 – Overview of the TASOPT model and of its results [12]

The upper part of the figure is the representation of the aircraft layout. It features the fuselage, the wings and the tails. As can be seen the general shape of the fuselage matches the one assumed in the fuselage airframe weight model, but in this case the geometry clearly fits a large aircraft. The fuselage shell thickness is sized exactly like in the fuselage airframe weight model : the loads generated by the cabin pressurization are used to size the fuselage skin thickness. The fuselage airframe is then reinforced by local elements (stringers / longerons and frames). The horizontal tail, the vertical tail, the tail cone and any other auxiliary system mounted at the rear of the aircraft (in this report none are considered) are converted into a single equivalent mass point which is at the position x_{tail}. This mass centroid lumped point is defined using the masses and the CG positions of the tail components :

$$x_{tail} = \frac{W_{htp}x_{cg_{htp}} + W_{vtp}x_{cg_{vtp}} + 0.5(x_{cone_{beginning}} + x_{cone_{end}})}{W_{htp} + W_{vtp} + W_{tail\,cone}}$$

The sum of the htp mass, the vtp mass and the tail cone airframe mass gives the equivalent mass of the tail group named $W_{tail\,group}$ in the upper part of the figure. The loads that act on the fuselage and that generate bending moments are displayed on the figure. There are two types of bending loads sollicitating the fuselage structure :

- The loads caused by the weight of the fuselage components. These loads are constant and depending on the component they are expressed as point masses or are distributed over a certain length. In the TASOPT all the components except for the tail group are supposed distributed uniformly over the fuselage cabin (which is also denominated as the shell in the TASOPT). This explains the parabolic shape of the bending moment distribution of the figure. The weight of the components is obtained by multiplying its mass with the local gravitational field of Earth. The weight loads are scaled up by the load factor of the aircraft in the bending moment computation.

- The aerodynamic loads of the tail group. An impulsive load on the horizontal or the vertical tail will indeed produce a bending load on the fuselage. This static load is expressed as a force acting on the tail group center of mass point x_{tail}. Physically

the force is equivalent to a lift. The maximum value of these forces is the quantity used in the model to represent the most extreme cases of aero-loads. As such in the expression of the lifts the maximum dynamic pressure and the maximum tail lift coefficients are computed.

$$L_{h_{max}} = q_{NE} S_h C_{Lh_{max}}$$
$$L_{v_{max}} = q_{NE} S_v C_{Lv_{max}}$$

Where q_{NE} is the never-exceed dynamic pressure, related to the design diving speed and the cruise altitude. S_h and S_v are respectively the horizontal tail wetted area and the vertical tail wetted area, while $C_{Lh_{max}}$ and $C_{Lv_{max}}$ are the corresponding maximum lift coefficients. The units of these quantities are related to the international system of units. The impulsive tail loads will also result in angular acceleration of the aircraft, whose inertia will tend to alleviate the static bending loads of the tail. This effect is modelled through the inertial-relief factors r_{Mh} and r_{Mv} (coefficients with value inferior to 1). These factors are multiplied by the lift forces in the bending moment computation. Their estimated value is directly given by the TASOPT.

The superposition of these loads results in the bending moment distribution displayed in the middle plot of the figure. The bending moment is expressed as a function of the x-coordinate that roams through the axis going from the nose to the tail of the aircraft. As can be seen the bending moment is separated in two independent components. The horizontal axis bending moment distribution $M_h(x)$ is displayed with the blue curve while the vertical axis bending moment distribution $M_v(x)$ is related to the red one. $M_h(x)$ is related to the loads applied normal to the x axis and resulting in a fuselage deformation in the xz plane. $M_v(x)$ is related to the loads applied normal to the y axis and resulting in a fuselage deformation in the xy plane.

A distinction is also made between the distribution at the front and at the rear of the aeroplane. The front distribution goes from the nose to the point x_{wing}, while the rear distribution goes from this point to the tail cone. The point x_{wing} corresponds to the wing's net lift-weight centroid. It varies theoretically in flight with the fuel fraction in the wings and the flap settings and is assumed constant for this model. For simplicity it is approximated as the wing's area centroid. Both distributions (front and rear) are assumed to match at this point. In the TASOPT only the rear bending moment distribution is computed for simplicity. Owing to the layout of the fuselage and its loading the horizontal-axis bending moment is indeed assumed to be roughly symmetric about the wing's center of lift. The horizontal bending moment front distribution is therefore generated by reversing the rear bending moment distribution and by matching its maximum amplitude with the one obtained at the point x_{wing}. About the vertical bending moment the assumption is made that the front part of the aircraft does not undergo any load that would result in vertical bending. With the lift generated by the wing the vertical bending distribution is logically annuled at x_{wing}.

The TASOPT model has been adapted for general aviation aircrafts and for FAST-OAD-GA. First of all the horizontal bending moment distribution is computed on the front part and on the rear part of the fuselage and these front and rear distributions are assumed to match at the wing's area centroid. In general aviation the aircrafts are lighter than in CS-

25 certification. As a consequence each component that weighs a lot has an impact on the horizontal bending moments encountered by the aircraft, and the hypothesis made by the TASOPT that the distributions are symmetrical can no longer be guaranteed. By analytically computing the front distribution along with the rear distribution the model gains in accuracy and it allows to consider specific cases, such as the analysis of the impact of a nose-mounted engine with respect to a rear or wing mounted engine. The front distribution covers the nose of the aeroplane along with the cabin that is before the point x_{wing}. The nose is loaded by its own weight that is distributed uniformly over its length. In the case of a nose mounted engine a point mass modelling the powerplant is also considered. However the front and the rear bending moment distribution are still assumed to match at x_{wing}. This hypothesis is made possible by considering the lift generated by the wing that alleviates the bending moment and inverts its tendency at x_{wing}. The approach chosen is to compute both distributions separately. In practice it appears that the maximum amplitude of the front distribution is way lower than the maximum amplitude of the rear distribution (both distributions reach their maximum value at x_{wing}). Then the maximum bending moment is extracted from the rear distribution (it is its value at x_{wing}). The lift required to get the front distribution to match this maximum amplitude is then computed and applied to the front distribution. Since the lift is modelled as a force acting on a single point of mass, the resulting horizontal bending moment term is a linear term. Regarding the vertical bending moment it appears that the only load sollicitating the fuselage is the aero-load imparted by the vertical tail. The wing inertia results in a null front distribution and the vertical bending moment distribution is therefore similar to the one computed in TASOPT. The following section mainly presents the computation of the horizontal bending moment. The vertical bending moment will also be adressed.

The following weights are assumed to be applied over the whole cabin length :

- $W_{cabin_{shell}}$: the weight of the cabin airframe shell (skin + stringers + frames).

- $W_{airframe\_additional\_components}$: the weight of the fuselage airframe extra components (windows, doors, floor,…) that have been computed in the previous section.

- $W_{payload}$: the payload mass (which consists of the sum of the crew, the passengers and the freight).

- $W_{cabin_{furnishing}}$: the additional systems and furnishings inside the cabin, such as the internal lighting, the air conditioning, the insulation, the seats, the security kits,… All these components masses are computed in the weight module.

This assumption allows to simplify the bending moment distribution by clustering the low weight elements together while including them in the model anyway. The distributed weight is then partially applied to the rear and the front distributions by taking in account the portion of cabin length that is prior to x_{wing} and the portion that is after. This results in the quantities $x_{ratio_{rear}}$ and $x_{ratio_{front}}$ which scale the weight related load in the bending moment distributions. Like in the TASOPT the tail group weight is computed on the lumped point x_{tail}. An additional point mass can be considered at some point in the rear length if the aircraft has a rear mounted engine. Taking in account the aero-loads that are

also applied at x_{tail} results in the following load diagram where the fuselage is schematized as a beam for simplicity. It is discretized along its x coordinate from the nose to the tail group lumped point x_{tail}. The total length of the fuselage is displayed with the dotted line.

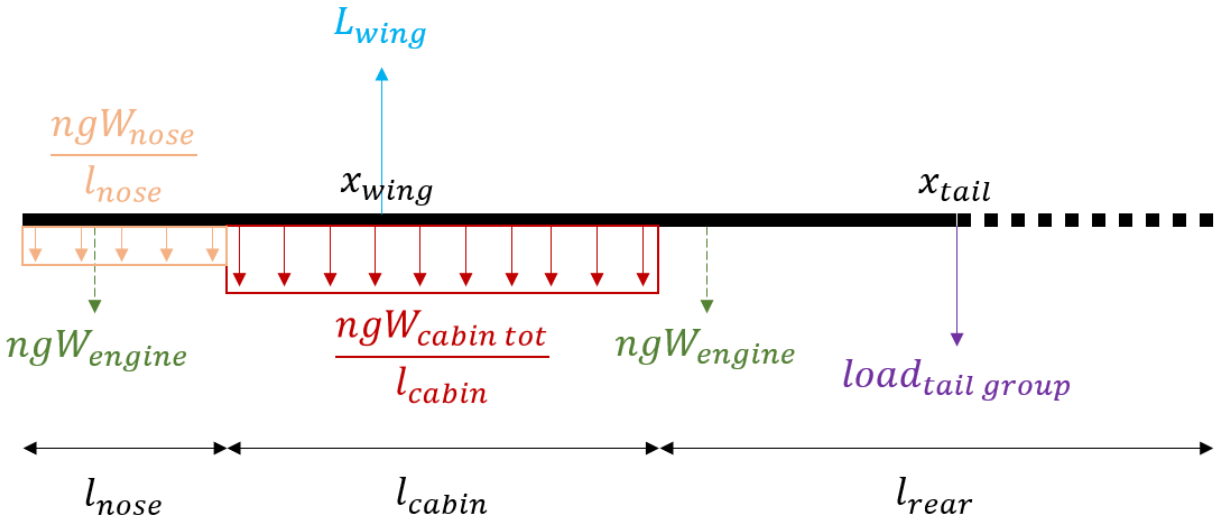

Figure 18 – Loads sollicitating the fuselage

Where $W_{cabin\ tot} = W_{cabin\ shell} + W_{airframe\ additional\ components} + W_{payload} + W_{cabin\ furnishing}$ is the cumulative weight in kg of the cabin. The weights are multiplied by g the local gravitational field of Earth to obtain the related load in N. The loads are then scaled up by the load factor n to take in account the CS-23 regulation and the different load cases the aircraft will encounter. The load of the tail group is not written in its entirety for visual clarity reasons. It is the assembly of the aero-loads of the horizontal tail plus the weight load of the tail group and is written in N.

$$load_{tail\ group} = ngW_{tail\ group} + r_{Mh}L_{h_{max}}$$

The dotted arrows represent potential fuselage-mounted propulsive systems. As of now a nose-mounted engine of weight W_{engine} will be considered. The x coordinate of the engine barycenter is named x_{engine}. L_{wing} is the lift generated by the wing. It acts opposite to all the other loads that the fuselage undergoes. The qualitative resulting bending moment diagram is presented below. The rear distribution is computed in blue. The front distribution is displayed in green, while the lift corrected front distribution is displayed in black.

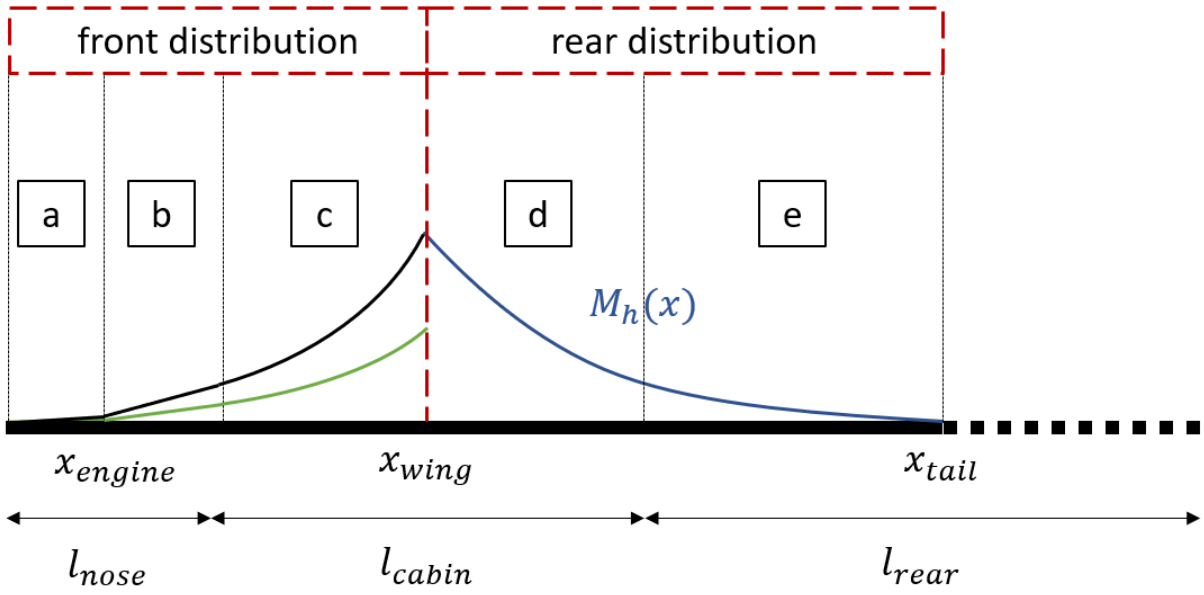

Figure 19 – Horizontal bending moment distribution along the fuselage

As can be seen the bending moment distribution can be separated in five zones. As a reminder the assumption is made that the front and the rear distributions match at x_{wing}, and that the wing lift is used to exactly adjust the front distribution so that it goes at the same maximum amplitude than the rear distribution. The front distribution is computed autonomously from the rear distribution. By definition the bending moment compensated by the lift is the difference between the distributions at this point. It is therefore necessary to first formulate the front and the rear distribution without taking in account the lift moment. Then once the difference value at x_{wing} is computed the correcting moment is included in the front distribution and the two distributions will match. The next part presents the value of the front and rear bending moment distribution by taking in account this independence between the front and the rear distributions. The lift linear term is also written in the front distribution as a red term. If this additional term is neglected then the resulting front distribution will be the one represented by the green curve.

Zone a (related to the front distribution)

Except for the lift which outputs a linear term, the bending moment is only generated by the distributed weight of the nose on its length. The resulting term is quadratic.

$$M_{h\,front}(x) = n\frac{gW_{nose}}{l_{nose}}x^2 + \frac{M_{lift\ compensation}}{x_{wing}}x$$

Zone b (related to the front distribution)

A linear term modelling the impact of the nose-mounted engine is superposed to the quadratic one.

$$M_{h\,front}(x) = n\frac{gW_{nose}}{l_{nose}}x^2 + ngW_{engine}(x - x_{engine}) + \frac{M_{lift\ compensation}}{x_{wing}}x$$

Zone c (related to the front distribution)

The quadratic term modelling the load of the nose weight becomes a linear term as the x coordinate is past the nose length. The engine weight is still linear. The cabin distributed load results in a quadratic term.

$$M_{h\,front}(x) = nx_{ratio_{front}} \frac{gW_{cabin\,tot}}{l_{cabin}}(x - l_{nose})^2 + ngW_{nose}x + ngW_{engine}(x - x_{engine}) + \frac{M_{lift\,compensation}}{x_{wing}}x$$

Zone d (related to the rear distribution)

It is the superposition of a linear and a quadratic term. The linear term is the weight of the tail group alongside the horizontal tail aero-loads both applied to the lumped point of mass, while the quadratic term is generated by the distributed weights over the cabin.

$$M_{h\,rear}(x) = nx_{ratio_{rear}} \frac{gW_{cabin\,tot}}{l_{cabin}} * (l_{nose} + l_{cabin} - x)^2 + (ngW_{tail\,group} + r_{Mh}L_{h_{max}}) * (x_{tail} - x)$$

The maximum lift generated by the horizontal tail elevator activation depends on the value of its maximum lift coefficient. This quantity is roughly estimated and is assumed global for all the aircrafts of the code (ie not an input of the aircraft .xml file but a local variable of the class). For the horizontal tail the selected maximum lift coefficient is 1.2. The value of the relief-inertia factor r_{Mh} is 0.4. It is directly extracted from the TASOPT. Using the TASOPT value means making the assumption that the horizontal tails of CS-25 aircrafts have the same inertia properties as the horizontal tails of general aviation aircrafts. An equivalent reasoning is followed about the aero-loads of the vertical tail.

Zone e (related to the rear distribution)

The tail group fully governs the bending moment.

$$M_{h\,rear}(x) = (ngW_{tail\,group} + r_{Mh}L_{h_{max}}) * (x_{tail} - x)$$

The bending moment generated by the lift is computed with the difference between the rear and front distributions at the point where they match that is to say x_{wing}.

$$M_{lift\,compensation} = M_{h\,rear}(x_{wing}) - M_{h\,front}(x_{wing})$$

As described in the current section the vertical bending moment is only computed at the rear of the aircraft. The only load is the aero-load induced by the vertical tail rudder activation. It results in a linear term that starts at x_{tail}.

$$M_v(x) = r_{Mv}L_{v_{max}}(x_{tail} - x)$$

The inertia-relief factor r_{Mv} takes the value of 0.7 given by the TASOPT. It means that 30% of the vertical aero-load is balanced by the inertia of the aircraft. The maximum lift coefficient of the vertical tail is assumed to be equal to 0.55. The following figure displays the corresponding simple bending moment distribution.

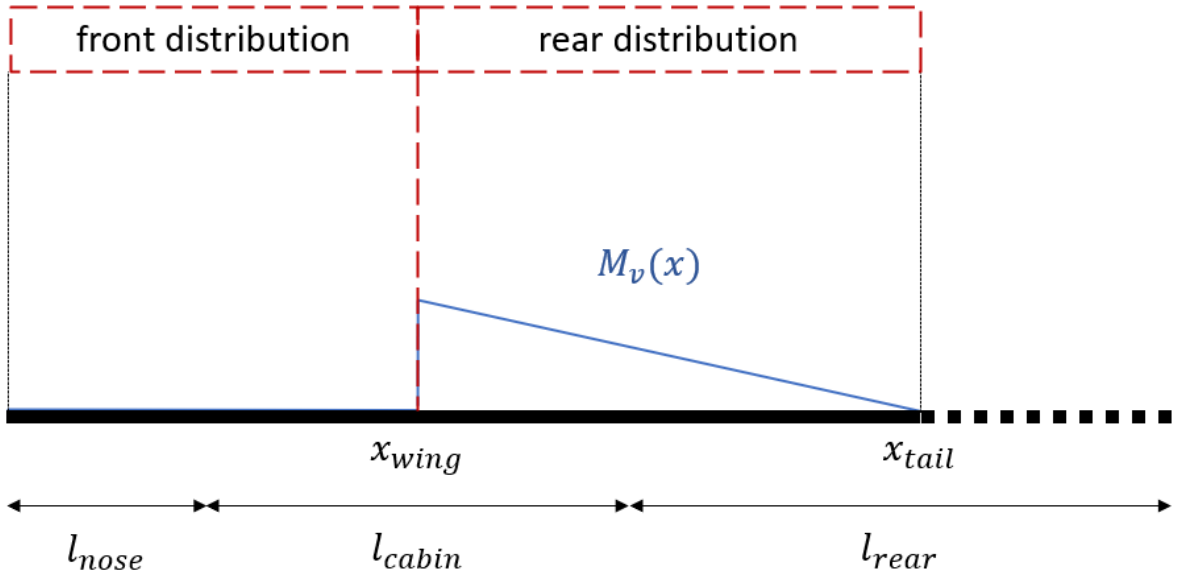

Figure 20 – Vertical bending moment distribution along the fuselage

The loads that the aircraft is subject to change depending on the ground or flight conditions. The different situations the aeroplane can encounter are expressed in the code through load factor variation and through variation of the aerodynamic forces on the aircraft tails. The CS-23 certification helps to define these loads. After analysis of certification documents and of manufacturers data the two following load cases have been retained for the estimation of the horizontal bending moment distribution. They are considered as the most extreme conditions the aircraft is likely to face and the implicit assumption made here is that there is no real need to define and compute other use cases since they would result in less sollicitating bending moments.

1. Emergency landing. For this load case the CS-23 norm allows for a maximum downward load factor of 6.0g to be experienced by the occupants of the cabin. The horizontal tail control surfaces are not activated for this use case, so the corresponding tail aero-loads will be equal to zero.

2. Cruise flight conditions with the impulsive full activation of the elevator. For this use case the conventional sizing ultimate load factor of the CS-23 norm is used. Its value is equal to 3.8. The elevator activation generates the maximum available aero lift.

Both load cases will result in different bending moment distribution and maximum amplitude. From the engineer's point of view, the sizing load case will be the one generating the maximum bending moment amplitude.

About the vertical bending moment distribution the only load case retained is the impulsive activation of the rudder in cruise conditions (load factor equal to 3.8 like for the horizontal tail). The related aero-load is the one defined with the never-exceed dynamic pressure and the maximum vertical tail lift coefficient.

The obtained results are arrays that represent the x-coordinate discretization of the fuselage length and the associated horizontal and vertical bending moment in Nm. A postprocessing function has been created that displays these arrays. Please note that it

only displays the fuselage length from the nose of the aircraft (x=0) to the lumped mass point of the tail group x_{tail}.

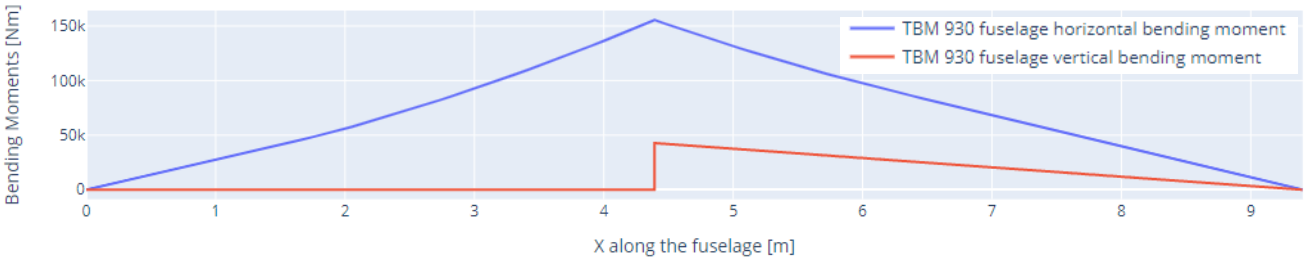

Figure 21 – Results of the developed model

In this particular case (TBM 930 computed by a MDA) the x-coordinate of the end of the cabin is 6.634m. It is at this point that the rear horizontal bending moment distribution loses its quadratic term and becomes fully linear. On the figure this transition is not clearly visible. Unfortunately no data about the reference aircrafts allows to validate the results of the model.

The main limitation of the model revolves around the simplicity of the mass distribution along the fuselage. This distribution could be refined especially in the cabin by modelling the payload, the passengers and their seats as mass points instead of dispatching their weight along the whole cabin length. However this decision needs to be pondered. The developed model indeed only aims to give an estimation of the bending moment distribution that would fit preliminary aircraft sizing. What's more refining the distribution would complexify the equations and the code would become a bit harder to comprehend, for a gain in precision that would probably be quite low. The model would also benefit from a more advanced definition of the cruise conditions load case since in practice the value of the load factor in flight heavily depends on the type of aircraft, the atmospheric conditions, the presence of gusts, and other key factors.

5.3 Application of the use case to a reference aircraft : TBM 930

For this section and the equivalent section about the wing span increase use case the TBM 930 is used as a reference aircraft which undergoes some variations in order to effectively measure and discuss the impacts of the use case modifications on an aircraft performance. As described in the analysis mode presentation, the use cases take the converged .xml file of the reference aircraft as input. In that respect the TBM 930 has to pass through a Multidisciplinary Design Analysis. The TLARs that have been chosen for this analysis are the following :

- Range : 1730 NMI
- Cruise Speed : 252 kn
- Altitude : 31000 ft
- Number of passengers : 0
- Freight mass : 150 kg (max available mass : 50 kg front storage and 100 kg rear storage)

The said TLARs have been chosen to be identical to the long range mission described in the specification documents of DAHER. Modelling a reference mission for which data is available allows to analyze the output of the MDA by directly comparing the results with the real aircraft data.

For the analysis presented in this report the fuselage of the TBM 930 will be stretched in order to host a new row of seats. This quantity is equal to 1 meter for this aircraft. The new cabin section will be fully placed behind the 25% of the wing mean aerodynamic center. No extra passengers or freight are added to the design mission. In that way the changes in the CG and balance of the aircraft will be entirely caused by the new fuselage section. Note that in this particular case the user could decide to add 6 passengers to fill the aircraft at max capacity, but the freight max capacity is already reached in the design mission so the user must not add luggage mass in the use case. These parameters are expressed in the appropriated section of the jupyter notebook as described in the presentation of the use case.

```
# Cabin length increase parameters
added_length = 0
added_section_x_ratio_front = 0
added_section_x_ratio_rear = 1
added_pax = 0
added_luggage = 0

fuselage_mod_parameters = [added_length, added_section_x_ratio_front, added_section_x_ratio_rear, added_pax, added_luggage]
```

Figure 22 – Parameters of the use case selected for this analysis

The analysis mode is then computed and the stretched fuselage aircraft is saved to perform the postprocessing comparison with the reference aircraft. Please note that the reference aircraft as well as the modified one are both the output of FAST-OAD-GA which is a conceptual design code and that the quantities of the reference aircraft are not equal to the real quantities of the analyzed aircraft.

First of all the geometry of the new aircraft is displayed :

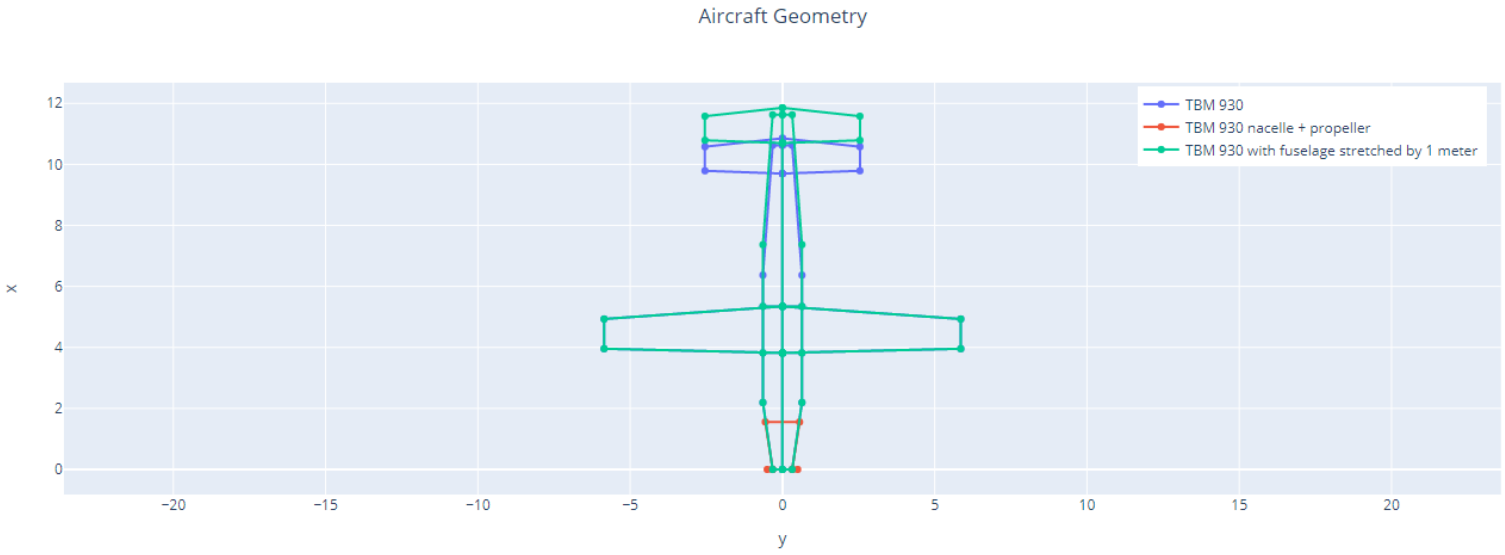

Figure 23 – Geometry of the reference and the modified aircraft

Increasing the cabin length this way is expected to have a lot of negative effects on the aircraft, since the rest of the geometry is not updated to counterbalance these effects. First

of all the variation of the CG position is analyzed. The figure below superposes the lateral views of both aircrafts and displays the aircraft empty CG and the extreme aft and fwd CG values which represent the worst flight and ground load cases scenarios. A zoom has been performed on the right side of the plot to distinguish more clearly the evolution of the barycenters.

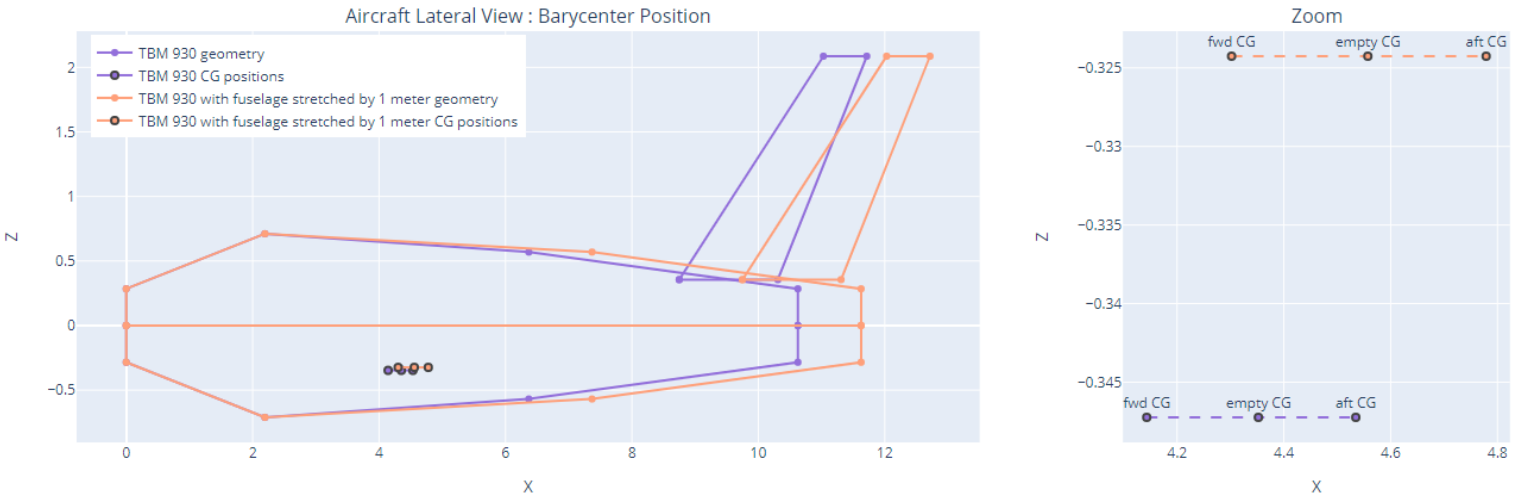

Figure 24 – Lateral geometry of the reference and modified aircraft and display of their barycenter range

As expected the CGs of the modified aircraft are at the back of the reference barycenter positions. Thus the CG of the empty aircraft has moved back by 20 cm. The z-coordinates have also varied but by an order of magnitude less (a bit more than 2 cm). These results are non negligible, as such CG variations have great influence on the stability of the aircraft. This can be seen by displaying the static margins of the two aircrafts :

```
TBM 930                                  static margin stick fixed: 0.1499     static margin stick free: -0.0267
TBM 930 with fuselage stretched by 1 meter  static margin stick fixed: 0.0322     static margin stick free: -0.1694
```

It can be clearly seen that both static margins (stick fixed and stick free) have deteriorated. The new aircraft is close to be unstable for the stick fixed case. For the stick free case the code has determined that the reference aircraft was already unstable but very close to be stable, and the fuselage stretched aircraft has obviousy become much more unstable.

For this particular use case it is also useful to exploit the load analysis of the fuselage. The following diagram shows very well that the fuselage will have to support much more efforts. In this study the maximum amplitude of the bending moments on the fuselage has increased by approximately 30% while the point of application of the inertia relief (the wing aerodynamic center) is the same for both aircrafts.

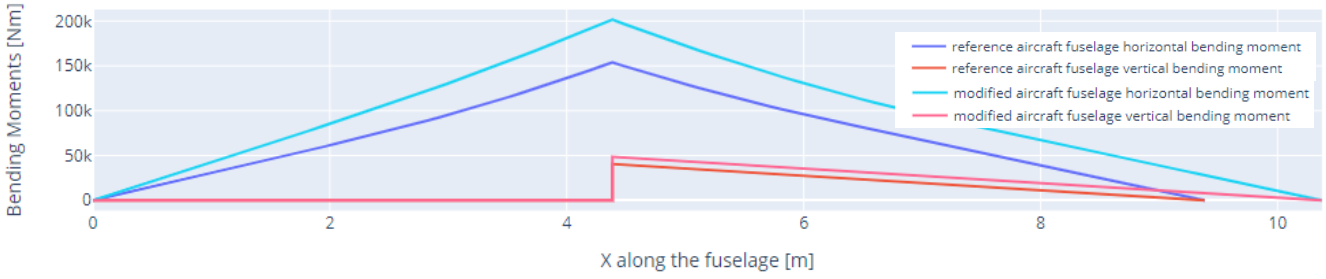

Figure 25 – Bending moment distributions of the reference and modified aircrafts

The designer has to take notice of such changes because the increase of the fuselage length means that the fuselage will be more sollicitated and the efforts can exceed the capability of the shell. In that case the manufacturer would need to reinforce the fuselage by increasing the skin thickness or by adding more local reinforcements where needed. Either way this would result in an additional mass increase that would have additional negative effects on the aircraft performance.

5.4 Limitations and further improvements of the use case

The main limitation of the use case definition is strongly related to the way the code arranges the different zones in the cabin and defines the length of the cabin. At the moment the thesis is written, the cabin layout generation is very simple. The code considers that the beginning of the cabin is the point where the nose ends. Then it puts side by side the different elements that constitute the inside of the cabin by extracting their length in the aircraft .xml files. Thus the flight instruments deck is directly interfaced with the pilot seats which are bound on their other end to the first row of passenger seats which is itself bound to the following passenger seats rows. After the last row of seats the length dedicated to the freight is then placed and the total resulting length is the cabin length. This approach requires to take in account the free spaces between components directly in their length definition. For instance the space for the passenger legs is already considered in the length of the seats. The consequences of this method are that the user cannot choose where in the cabin layout he wants to place the new cabin section, and that he cannot control the component that he want to add. Indeed, instead of adding a row of passenger seats a designer might want to add a table or another equipment between two already existing row of seats. This would result in a way different mass distribution in the cabin and the consequences could be interesting to observe. However as for other features that require to redefine a core method of the code this improvement would need to restructure all the FAST-OAD-GA sections that are related to the cabin and the fuselage (especially the mass breakdown and the CG estimations). It has been decided that reaching such a level of detail in the cabin was not the priority for the thesis. Please note that in such a project it has to be kept in mind that there are several people working in parallel each on their own version of the code and that these versions have to be kept as close to to original version as possible in order to conveniently merge the parallel branches into the main one. Therefore the restructuration of the way the cabin is computed would result in unnecessary time consuming efforts.

Chapter 6

Use Cases : Wing Span Increase

6.1 Use Case Description

The Use Case that is going to be described in detail in this section is the stretch of the wing span of the aircraft. The need that motivated the design of this use case is the will to add a new section at the tip of the reference aircraft wings, that is to say increase the wing span without interfering with the reference wing. The aim is to observe the impacts of such architecture modification on the structural loads along the span of the wing and on the static margins of the aircraft. The consequences on the flight performance will also be discussed. The use case presented in this section is the answer to this need that has been achieved in this thesis. As for the cabin stretch use case the user disposes in the related jupyter notebook of several parameters which allow him to affect the way the use case modifications will be determined by the code. Below is a screenshot of all the settings the user can play with for the stretch of the wings. In order to fully understand the presentation of this use case, it is useful to know how the code defines the majority of the wing related quantities, and especially the spanwise positions of the wing components. The position of the beginning and the end of the fuel tanks, the position of the flaps and of the ailerons, the position of eventual under the wings engines are all expressed in the .xml files as span ratios (called y-ratios in the code, the y coordinate being related to the span length). In the geometry module the code then computes the y-positions of these components by multiplying their y-ratios with the value of the aircraft span.

```
# Span modification parameters
span_length_multiplier = 1.1
fixed_engine_position = True
fuel_added_part = False

span_mod_parameters = [span_length_multiplier, fixed_engine_position, fuel_added_part]
```

Figure 26 – Parameters of the wing span increase use case

span_length_multiplier : This is the main parameter of the use case. This float determines the value by which the wing span is multiplied. In the screenshot this parameter is set to 1.1. It means that in that case the span will be extended by 10% of its original value. This parameter is used by the code to modify the geometrical quantities that define the whole wing geometry, that is to say the quantities that are taken as input of the geometry module. Said quantities are the wing aspect ratio, the taper ratio, and the wing area.

fixed_engine_position : Boolean stating if the engines stay at the same position along the span or if this position varies. This parameter only makes sense for engines under the wings. If the reference aircraft has a nose or rear mounted engine then the code will ignore this parameter regardless of its value. The quantity associated to this parameter is the y-ratio of the engine of the reference aircraft. If the reference aircraft engine span ratio is 0.4 then the engine position on the wing is at 40% of the semi-span. If the wing span is

increased with this use case and fixed_engine_position is False then the ratio will be conserved and the engine position will be updated (in absolute value it will be further from the wing root than its original value). On another note if this parameter is set to True then the span ratio of the engine will be updated by the code in order to obtain an invariant engine position. This parameter has also been studied to account for distributed propulsion.

fuel_added_part : Boolean stating if fuel tanks are added in the extended section of the wing. For structural reasons this feature is only activated if the fuel tanks of the reference aircraft go to the tip of the wing. If that is not the case, this parameter will be ignored regardless of its value.

When the module Modify Config is executed with generate_block_analysis and these parameters are given as input, the code automatically defines the new wing geometry. It is important to note that the added section of the wing is in the continuity of the reference aircraft wing. The chord / thickness / sweep angle / twist angle distributions are conserved. The control surfaces of the reference wing are also conserved. The flaps y-ratio is updated so that the modified wing will have the same flap length than the reference one. It has been decided to consider that ailerons would be added to the extra section of the wing. The code also automatically updates the y-ratio defining the position of the beginning of the fuel tanks. In that way the modified wing corresponds exactly to the reference wing with an extra section (except for the engine positions that can be modified with fixed_engine_position).

There is no limit in the code for the increase in span, but some modules will most probably have problems running with an extreme value (such as 1.5 times the original span). A wing span reduction can also be theorically computed by using a multiplier inferior to 1, but the consequences on the code have not been studied.

During the development of this use case the model computing the Maximum Fuel Weight (MFW) of the aircraft has been called into question. Its results have been used to analyze the impact of a new wing section able to store fuel. Its definition has been judged too simplistic and it has been decided that the code would benefit from a more detailed model. The following section presents this model.

6.2 Models to add to the code

6.2.1 Computation of the Maximum Fuel Weight

The maximum fuel weight has many purposes in FAST-OAD-GA. At the end of a Multidisciplinary Design Analysis iteration this quantity is used in the sizing of the wing area to ensure that the wing can stock enough fuel to compute the design mission. What's more this quantity is part of the equations of some statistical models, such as the estimation of the fuel systems in the mass breakdown. The computation of the maximum fuel weight is done at the end of the geometry module after the wing, the fuselage and the nacelle geometries have all been modelled. The main challenge for the estimation of this quantity is to be able to define how much volume inside the wing can be used to store fuel, that is to say the volume occupied by the wing fuel tanks. The MFW is then obtained by multiplying the density of the fuel employed by the aircraft.

The current prediction of the maximum fuel weight in the code is a basic analytical model. It makes the hypothesis that the wing tanks are between 30% and 60% of the Mean Aerodynamic Chord of the wing and that they extend from the center of the fuselage to the tip of the wing. As a consequence, the wing tanks occupy 30% of the wing. This area is then multiplied by the average thickness of the wing. The depth of the tanks is chosen to be 70% of the average thickness so the volume found is multiplied by a factor of 0.7. The resulting fuel tanks area is depicted in red in the following figure for one wing. The black solid line represents the fuselage width, while the dotted one represents the center of the fuselage.

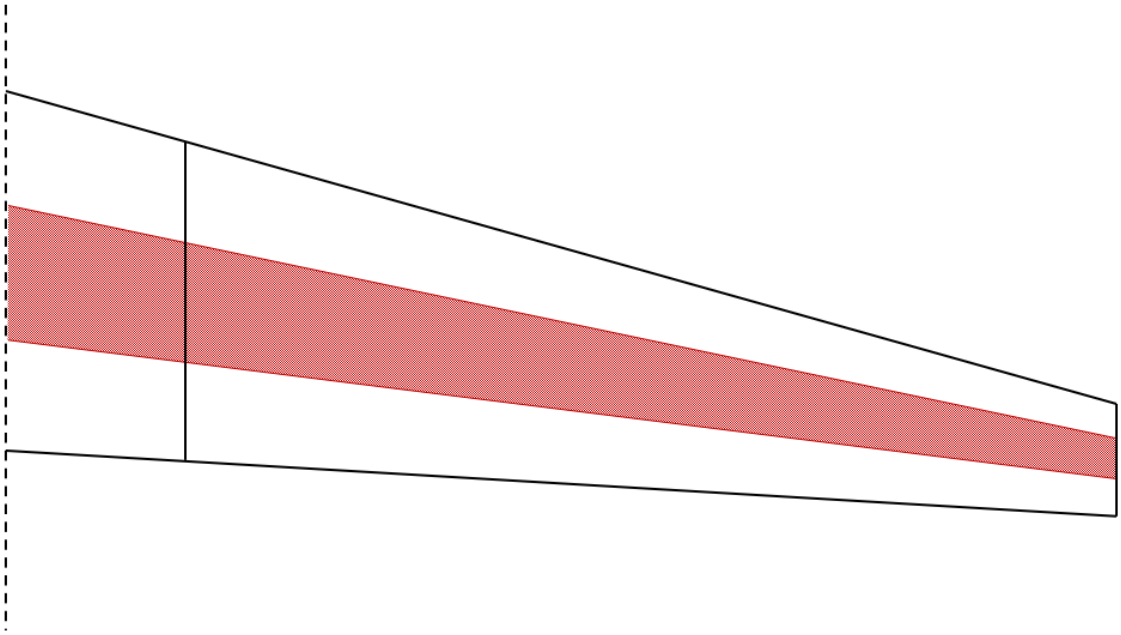

Figure 27 – Wing area occupied by the fuel tanks in the current FAST-OAD-GA model

This method approximates the maximum fuel weight in a convenient but in a little too simple way. First of all it does not take into account the different components of the wing that locally prevent to install a fuel tank, such as the control surfaces, the nacelle and the landing gears. Secondly this approach does not consider the place occupied by the tank structure which will reduce the fuel available for a given volume. Lastly the assumption that the fuel tanks cover the whole wing along the span has proved to be incorrect. All reference aircrafts of FAST-OAD-GA have fuel tanks in partial portions of the wings.

The model developed in this thesis is based on the Jenkinson model developed in [13] and is presented in figure 28. It is an analytical model developed for CS-25 aviation that discretizes the wing in three sections along its span. The first section (A1 on the figure) is at the span position of the beginning of the fuel tanks. In this model the assumption is made that this point corresponds to the center of the fuselage, that is to say that the wing structure goes through the fuselage. This hypothesis is plausible for transport aircrafts who commonly have this configuration. The third section (A3 on the figure) is located at the point where the fuel tank ends and the second one (A2 on the figure) is halfway between the first and the third sections. As can be seen on the figure, each section is defined to avoid wing components that prevent the implementation of fuel tanks. Each section is then assumed rectangular and the corresponding cross-sectional area is

computed. The volume of the fuel tanks is obtained by integrating the cross-sectional areas along the length of the fuel tanks. A factor to take in account structural restrictions and integral tankage is considered. Thus this model requires to know several quantities beforehand : the chord and thickness distribution of the wings, the length of the fuel tanks, the position along the span and the chord of the wing extra components, and the chord and span ratios of the control surfaces (flaps and ailerons). Jenkinson also allows the implementation of external tanks. It is worthy to note that FAST-OAD-GA does not at the moment this thesis has been written uses such external tanks, and that all the fuel necessary to compute the mission is stored in the wings, regardless of the aircraft computed.

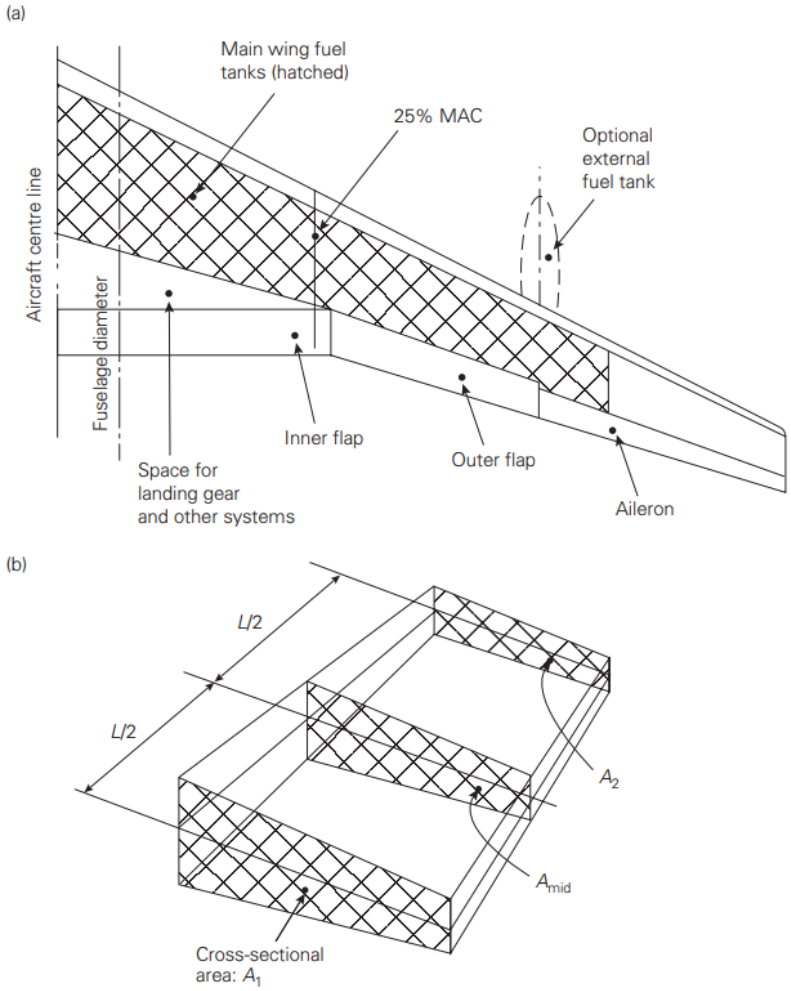

Figure 28 – Distribution of fuel in the wings in Jenkinson model [13]

Jenkinson's model has been adapted to FAST-OAD-GA and to general aviation aircrafts. The wing is discretized along the tank length by several sections which are uniformly separated. For each section of the wing the following terms are used : the width of the section denotes the length of the section along the chord of the wing. The depth of the section denotes the length of the section along the thickness of the wing.

The depth of the sections is calculated by computing and multiplying the chord and the thickness/chord ratio distributions along the span of the wing. A reducing factor is then applied to the thickness array obtained to account for the fact that the fuel tanks occupy approximately between 50 and 80% of the thickness of the wing. This range is defined by

Jenkinson and is assumed to be the same for general aviation aircrafts. This « depth factor » is defined in the .xml file instead of being directly defined in the class. It allows to modify its value for each aircraft.

The width of the sections of the tanks are defined by stating that the tanks are separated from the leading edge and from the control surfaces by certain lengths. These lengths are expressed as a percentage of the chord and so they can vary along the span. For example for the Cirrus SR22 the fuel tanks start at 10% of the chordfrom the leading edge and end at 10% of the chord before the control surfaces. As a consequence, the fuel tanks chord ratios will vary accordingly along the span with the control surfaces chord ratios, for example if different types of flaps are used or at the intersection between a flap and an aileron.

The width array and the depth array are then multiplied to obtain the array of the cross-sectional areas of the fuel tanks along the span. The landing gears and potential nacelles have to be considered as they inevitably restrict the quantity of storable fuel. Instead of simply avoiding the wing areas where extra components are present like in Jenkinson, the variation of the fuel distribution along the span is expressed as a variation of the available area of each section of the wing tank. The area of the tank is reduced by a percentage which depends on the component :

- For retractable landing gears the fuel distribution is reduced by 80% (note that for non-retractable landing gears no restriction is applied).

- For the powerplant of engines under the wing the fuel distribution is reduced by 50%.

The updated cross-sectional area array is then integrated along the tank length giving the volume of the wing tanks. Finally this value is multiplied by a reducing factor of 0.85 which takes into account the internal obstructions caused by the structural and system components within the tank, typical of integral tankage. The maximum fuel weight is logically the result of the product of the computed volume and of the mass density of the fuel.

The nacelle and landing gear spanwise positions are directly computed by the code in the geometry module. On the other hand, the control surfaces definition and the model settings such as the chord percentage defining the distance between the fuel tanks and the leading edge / control surfaces, the position of the fuel tanks along the span and the depth factor of the tanks are all extracted from the initial .xml file and as such are provided by the user when he defines his aircraft. To calibrate this model, it is recommended to use as much available data as possible. The chord and span ratios of the control surfaces can be estimated graphically with diagrams or pictures of the wings. What's more the fuel tanks are often roughly depicted in sketches of the wing structure in the aircraft information manuals and this allows the user to be able to define the fuel tanks position along the span and along the chord of the wing. However, the depth of the tanks is a very delicate data to find in the aircraft resources. As a result this parameter has been used in practice to calibrate the developed model. And for all the reference aircrafts, highly satisfying results were achieved by setting the model parameters to values identical to the real ones, and by setting a depth factor between 0.5 and 0.8 thus respecting its definition.

It may be interesting to note that the model has also been designed to account for distributed propulsion meaning that there would be several nacelles and wing mounted engines restricting the fuel distribution.

In the following section the results obtained by the developed model will be compared with the current model of the code and with data from the reference aircrafts. For the Beechcraft Duchess 76 and the Cirrus SR22 the shape of the fuel tanks is fully visible on their pilot information manuals. As a consequence the developed model geometrical parameters can be precisely defined and the model validated or not.

	Beechcraft Duchess 76	Cirrus SR22	TBM 930
Basic Model of the code	457 kg	303 kg	574 kg
Model Proposed	274 kg (depth = 0.6)	228 kg (depth = 0.5)	910 kg (depth = 0.74)
Aircraft Data	273 kg	220 kg	916 kg

Table 2 – Comparison of the models results

The possibility to calibrate the model by playing on the depth factor of each aircraft of the code renders the model very accurate comparing to the current analytical model. It can be seen that this parameter's value is included in the prescribed range [0.5 – 0.8] which means that the model is able to give satisfying results without getting physically irrelevant. This feature nevertheless imposes the user to be able to define the geometry of the wing tanks. These quantities are defined in the initial .xml files of the aircrafts and as such the model adds 4 quantities to the aircraft basic definition. Since FAST-OAD-GA aims to easily create or implement new aircrafts the engineers have to be careful not to ask for too many quantities in the .xml definition file. Such data could prove hard to come across for some aircrafts which are not heavily dcumented. An improvement of the model would be to take into account potential external fuel tanks that could be mounted on the wing or in the fuselage.

6.3 Application of the use case to a reference aircraft : TBM 930

As for the fuselage strecth use case the TBM 930 is used as the reference aircraft to test the use case and discuss about the impact of the wing span increase on the performances of the aircraft. More specifically exactly the same reference .xml file (which is the output of a Multidisciplinary Design Analysis for a certain set of TLARs) has been used for both use cases. For this analysis the wing span of the TBM 930 reference file will be increased by 15%. The nose mounted engine renders the fixed_engine_position unnecessary and no fuel is stocked in the extra section of the wing.

```
# Span modification parameters
span_length_multiplier = 1.15
fixed_engine_position = True
fuel_added_part = False

span_mod_parameters = [span_length_multiplier, fixed_engine_position, fuel_added_part]
```

Figure 29 – Parameters of the use case selected for the analysis

The analysis mode is then performed by following the method described in the related section, and the modified aircraft is effectively computed. The user now can compare this new aircraft with the reference one by using the postprocessing tools. First of all their geometry is displayed with aircraft_geometry_plot :

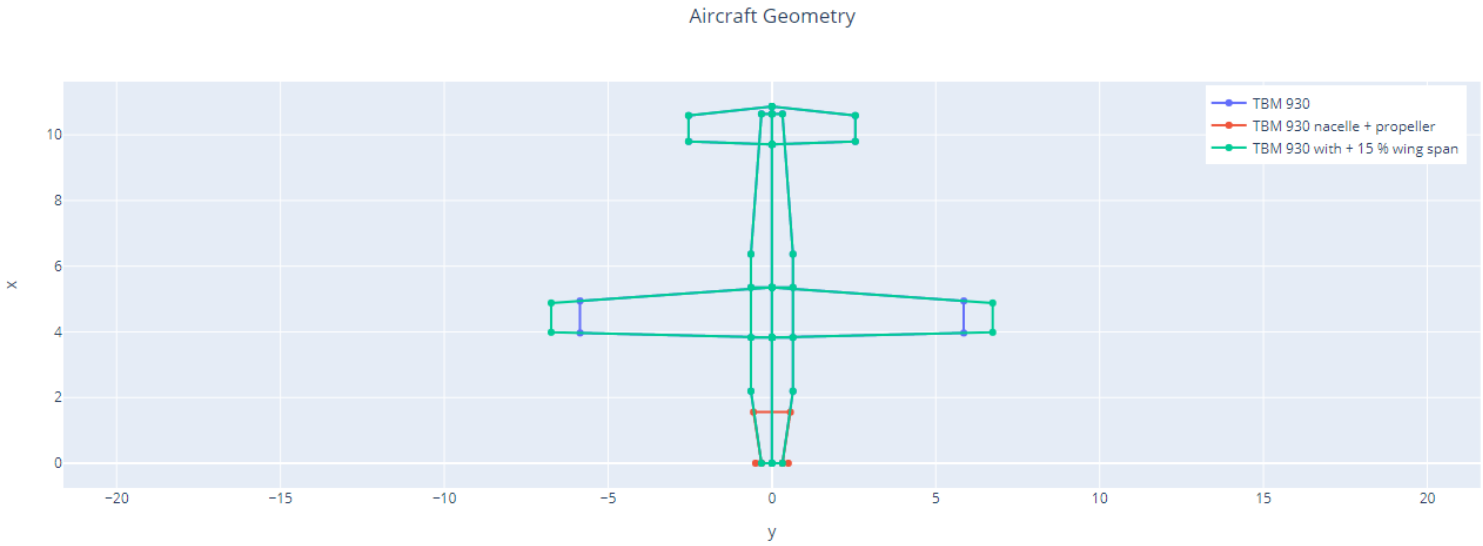

Figure 30 – Geometry of the reference and the modified aircrafts

As expected the modified aircraft geometry is exactly the same as the reference one except for the wings that have been stretched. The figure allows to clearly see that the original chord distribution is maintained through the stretch. The same goes for the thickness and the sweep angles distribution.

The postprocessing tools allow to visualize the impact of the wing stretch on several aspects of the aeroplane performance and characteristics. First of all the wing mass increase is analyzed through the increase of the airframe mass in the mass breakdown of the Overall Weight Empty of the aircraft which is displayed on the right side of the figure below.

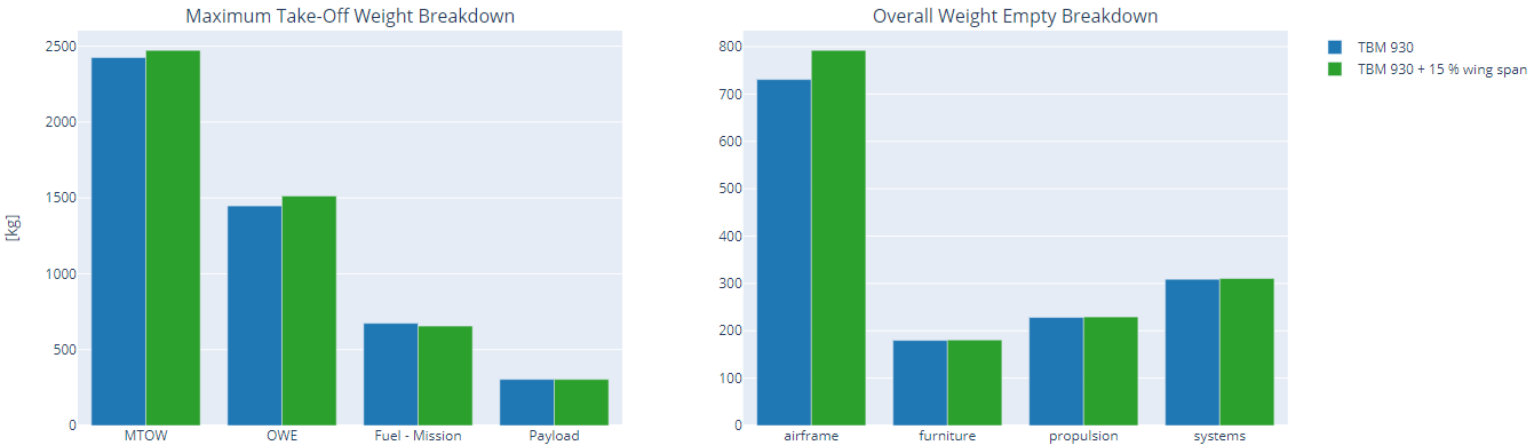

Figure 31 – Mass breakdown of the reference and the modified aircrafts

The weight of the total airframe has indeed increased due to the wing modification, but the increase in MTOW is not exactly equal to the increase in wing weight. One of the reasons for that point is that since a special analysis is performed for the weight and performance modules with a dedicated yml file, the code has indeed rerun the mass breakdown after having updated the MTOW value that increased with the wing stretch. It has been described briefly that a lot of the models present in the mass breakdown are statistical models that use the MTOW as an input, as the current fuselage airframe mass model. The MTOW variation results in a slightly difference in the weight of components such as the passenger seats, the power systems, or the navigation systems. What's more in this use case the aircraft modification definitely has a non negligible impact on the aerodynamic characteristics of the aircraft. Stretching the wings means more wing area to generate lift, at the cost of an increase in mass and a higher drag. There is a compromise to make between those related quantities and sometimes it would probably appear that it is for the best not to stretch the wings. In this example there is a 2.6 % reduction in the fuel burnt through the entire mission. As a result the MTOW also decreases by that amount and this compensates a bit the mass increase caused by the airframe modification.

A change in the aerodynamics and in the fuel consumption is also interesting for the overall performance of the aircraft. The superposed payload range diagrams of both aircrafts is analyzed with the figure below.

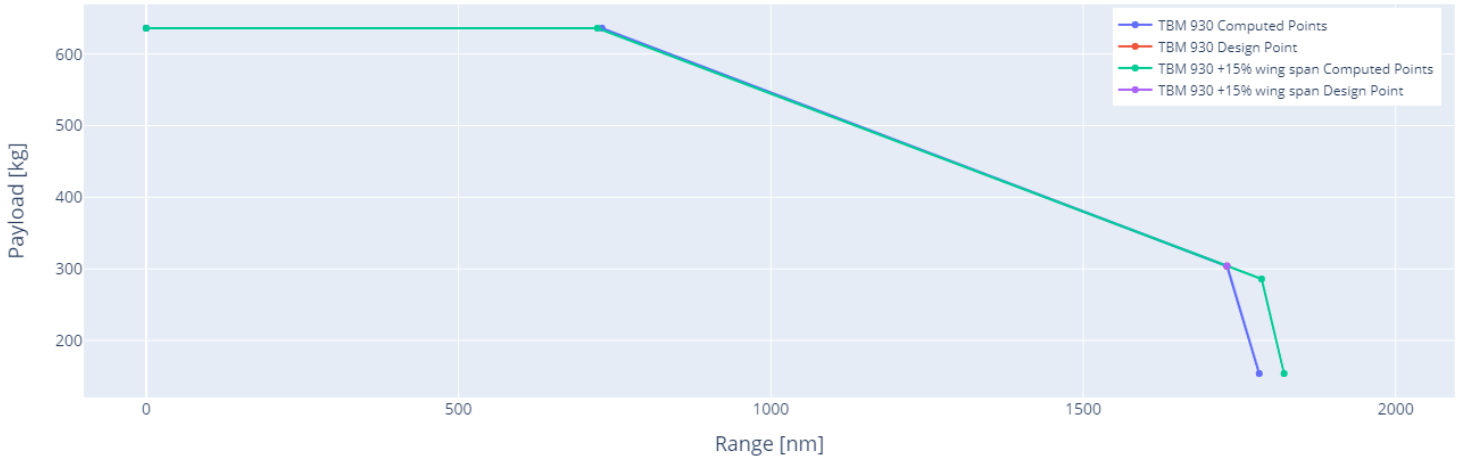

Figure 32 – Payload Range diagrams of the reference and modified aircrafts

With this diagram the changes in performance of the stretched wing aircraft can be clearly seen. As a reminder both aircrafts possess the same maximum fuel weight so the only changes that affect the payload range are the mass and the fuel consumption variations. The lower fuel consumption allows for the aircraft to reach a higher range than the reference aircraft. The range of the point E (the rightmost point on the diagram) has indeed stepped up by 2.2% with respect to the baseline configuration. The design point corresponds to the mission defined in the TLARs and written in the initial .xml file of the TBM 930 : a range of 1730 NM for a payload of 304 kg. This point is the same for both aircrafts.

6.4 Limitations and improvements of the use case

The use case developed in this thesis answers the original need which was to implement the feature to stretch the wing without interfering with the reference section of the wing or with the rest of the aircraft components. The use case even goes further by proposing the user the possibility to choose to maintain the engine position or not in the case of wing mounted engines. In that sense the objective that motivated the creation of this use case has been validated and achieved. The main limitation of the use case is the impossibility to model a new section that would have at least one geometrical discontinuity from the reference wing. The wing extension obeys to the same geometrical laws and distributions as the reference wing. For example the code cannot add a section that would have a constant chord, the value of which would be the chord at the tip of the reference wing. This line of argument applies for the chord distribution, the thickness distribution and the sweep angle distribution of the wing. This limitation is completely due to the structure of the code, and especially the geometry module. The whole section of this module that models the wing geometry would need to be redefined in order to implement such changes. As for the fuselage stretch use case improvements, it has been decided that this feature would be kept in mind but was not the priority at the moment this thesis has been written.

Conclusions

The analysis mode that has been described in this thesis fulfills the need that has been expressed. FAST-OAD-GA now can provide the user an efficient and user-friendly way to modify the baseline configuration of a reference aircraft and to study the impact of the chosen architecture modification. Two single use cases have been introduced along with a small analysis of their results. The presented use cases and potential future other use cases can also be coupled to perform a more global analysis of an aircraft. For instance a designer can decide to add an extra row of seats to a reference aircraft and then to balance the negative impacts of the fuselage length increase by modifying the geometry of the wing or of another component whose modification has been implemented in the code. The structure of the analysis mode and especially of the module Modify Config have been thought to be easily updated and to support new features. This thesis opens the way for additional use cases which creation will be motivated by the overall evolution of FAST-OAD-GA. The available postprocessing tools assist the user performing such complex analysis by helping to do decision making and to visualize all the impacts of an aircraft modification. By developing the analysis mode and the use cases some models of the code have been refined and new models have been adapted for general aviation and implemented.

Bibliography

[1] Amadori, K., Jouannet, C., & Krus, P. (2008). Aircraft conceptual design automation. *26th International Congress of the Aeronautical Sciences.*

[2] Lukaczyk, T., Wendorff, A., Colonno, M., Economon, T., & Alonso, J. (2015). Suave: an open-source environment for multi-fidelity conceptual vehicle design. *16th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, (p. 3087).

[3] Gray, J., Hwang J, Martins, J., Moore, K., & Naylor, B. (2019). Openmdao: An open-source framework for multidisciplinary design, analysis, and optimization. *Structural and Multidisciplinary Optimization*, (pp. 1075-1104).

[4] Jasa, J., Hwang, J., & Martins, J. (2017). *Openaerostruct: An open-source tool to perform aerostructural optimization.* Technical report, MDOlab Tech. Rept., Ann Arbor, MI.

[5] David, C., Delbecq, S., Defoort, S., Schmollgruber, P., Benard, E., & Pommier-Budinger, V. (2020). From FAST to FAST-OAD: An open source framework for rapid Overall Aircraft Design. *10th EASN International Conference.* Salerme, Italy.

[6] Torenbeek, E. (1982). *Synthesis of Subsonic Airplane Design.* Delft: Delft University Press, Martinus Nijhoff Publishers.

[7] Gudmundsson, S. (2014). *General Aviation Aircraft Design: Applied Methods and Procedures.* Butterworth-Heinemann.

[8] Roskam, J. (1985). *Airplane Design. Preliminary Sizing of Airplanes.* Kansas: Roskam Aviation and Engineering Corporation.

[9] Raymer, D. (2018). *Aircraft Design : A Conceptual Approach, Sixth Edition.* Blacksburg, Virginia: American Institute of Aeronautics and Astronautics.

[10] Wells, D., & Horvath, B. (June 2017). *The Flight Optimization System Weights Estimation Method.* Hampton, Virginia: NASA.

[11] European Aviation Safety Agency. (15 July 2015). *Certification Specifications and Acceptable Means of Compliance for Normal, Utility, Aerobatic, and Commuter Category Aeroplanes CS-23 Amendment 4.*

[12] Drela, M. (20 Mar 2010). *TASOPT 2.00 Transport AircraftSystem OPTimization.*

[13] Jenkinson, L., & Marchman, J. (2003). *Aircraft Design Projects for Engineering Students.* Oxford: Butterworth-Heinemann.