Weight Estimation for Pylons supporting large Aero-Engines

A KBE Approach

J.W.T. Snelders

Technische Universiteit Delft

TU Delft Delft University of Technology

Challenge the future

Cover image: Photo of a man inspecting an Airbus A320neo aircraft, clearly showing the nacelle and pylon. From: FAST 65 (Airbus technical magazine - April 2020)[1].

Weight Estimation for Pylons supporting large Aero-Engines

A KBE Approach

by

J.W.T. Snelders

in partial fulfilment of the requirements for the degree of

Master of Science
in Aerospace Engineering

at the Delft University of Technology,
to be defended publicly on Thursday August 31st, 2023 at 13:00 PM.

Student number:	4109562	
Supervisors:	dr. ir. R. Vos & dr. ir. M. F. M. Hoogreef	
Thesis committee:	dr. ir. R. Vos,	TU Delft (chairman)
	dr. ir. M. F. M. Hoogreef,	TU Delft
	dr. X. Wang M.Sc.,	TU Delft

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft Delft University of Technology

"I’m safe on Mars. Perseverance will get you anywhere."

‘NASA’s *Perseverance* Mars Rover’ twitter account upon successfully touching down on the surface of planet Mars

PREFACE

This thesis report marks not only the end of my master thesis project, but also of my time as a student at TU Delft. A period that was undeniably (considerably) longer than I could have ever imagined at the start of my university journey. Over the coarse of the last 13 years, I have experienced both the most amazing, but unfortunately also some of the most difficult periods of my life. I have had the opportunity to meet a lot of great people, many of which I have become lifelong friends with and who's support I have deeply felt during these years.

I was lucky enough to experience and see some fantastic things that I would never have been able to do without studying here. Highlights I especially hold dear are the year I got the opportunity to work on the design of a fully new electric car at the automotive start-up Amber in Eindhoven, between finishing my bachelor and starting the master. And of course my 9 month internship with Tesla in the US, which enabled me to fulfill one of my childhood dreams - living in the USA for a period of my life. That being said, it must be said that even with these amazing experiences, I have experienced this period as one of the toughest of my life. Especially difficult moments that come to my mind are not making the BSA mark in my first year, and having to extend my bachelor by a full year after failing my final course by 0.2 points. Truth be told, there have been many moments when I cursed the TU and her rigid, bureaucratic stance and impersonal approach towards students. Likewise, there have been multiple moments where I thought I was not going to make it at all. But in the end I kept going, maybe against better judgment, even if the situation looked dire. *"Perseverance will get you anywhere!"*, NASA tweeted in February 2021 when their *Perseverance* Mars rover successfully landed on the surface of Mars. From the moment I read this message, I knew this was going to be the motto of this thesis. Because I truly believe that if you keep your ground, work hard and keep your final goal in mind, anyone can fulfill their dreams. This report is the physical proof of that idea.

That being said, finishing this thesis project, and with it this degree, would not have been possible without the help and support of many people. First of all, I need to thank my supervisors during this project, dr. Roelof Vos and dr. Maurice Hoogreef, for their support and feedback. Many thanks also to professor Xuerui Wang, who was willing to accept a position in my graduation committee on a very short notice. And Thomas Pigeaud and Florian Garnier at Safran, who came up with the idea for this project and provided the necessary data for the validation of the code. Jelle Weigand at ParaPy, who wrote the biggest part of the ParaPy/Abaqus library on which I based the majority of my code, and who I have at times bombarded with questions which he all answered rapidly and without restraint. And finally Srikanth Vasudevan who has served as a sounding board and helped me with setting up my thesis plans, especially in the initial phase of the project.

On a more personal level, I would like to thank all the fantastic people I have met and/or was lucky enough to work with over the past years. My lovely office mates who struggled with me during the thesis period and have made it all so much more bearable (*grazie!*). A special mention goes to Thimo van den Berg, who entered the room two days before me and has been there with me virtually until the end of the thesis, saving the day whenever I had made some stupid coding mistake. Of course all my friends, who supported me this entire period and were never judgmental, even though I'm sure they must have wondered at times if I was ever going to graduate at all. And finally my family, who supported me so much throughout my life and during my time at this university. My mom and dad, and my brother. And my girlfriend, who was there by my side the entire time during the thesis project. I could not have done this without you all.

Jeffrey Snelders
Delft, August 2023

EXECUTIVE SUMMARY

The aviation industry faces challenges due to soaring air traffic and environmental concerns. Despite improved aircraft efficiency, greenhouse gas emissions from air travel persist, necessitating improved engine efficiency. Applying higher by-pass ratio's (BPR) can offer a solution to concerns regarding emissions by increasing the propulsive efficiency of the engine. However, integration of these new modern large ultra-high bypass ratio (UHBR) engines poses challenges affecting structural weight and aerodynamics. To evaluate new engine architectures and -integration types, early-stage design and weight evaluation methods that can properly incorporate the effects of these innovative designs on pylon structural weight are essential. Existing conceptual weight estimation methods for pylons lack the ability and precision to do so, as they mainly rely on statistical methods applied to previous generation engines which do not represent current trends. This thesis proposes a physics-based design approach for pylon structures using Knowledge-Based Engineering (KBE) principles, enhancing the evaluation of diverse engine-aircraft integration designs.

First, a new parameterization is proposed that involves a mathematical description of the so-called 'zero-thickness geometry' and of the structural elements that form the pylon structure. The zero-thickness geometry covers the description of the hardpoint locations, the section geometry of the pylon at the hardpoint planes, and the orientation angle. The structural parameterization covers the thicknesses and geometry of the structural members. In the end, a total of 80 parameters are defined to describe the pylon structure.

Second, the parameterization is captured in a KBE software application using the ParaPy Python platform. The core of the application is a ParaPy-class called 'PylonDesigner', in which the geometry is created, the weight is evaluated, a structural analysis process can be started and finally a sizing optimization is performed. A fundamental part of this is the so-called 'GeometryGenerator', which generates a CAD model of the pylon.

The generated pylon structural geometry is analyzed using the commercially available finite element code Abaqus. To enable a proper coupling between the ParaPy and Abaqus, a dedicated application programming interface (API) has been implemented. Using this API, the meshed pylon geometry is processed part-by-part, after which the full structure is assembled. Boundary conditions are then applied, and the analysis is defined including the loads. During the analysis, the pylon is subject to a total of 20 limit loads cases covering different maneuvers, thrust settings and gusts, and 4 ultimate load cases representing the critical fan-blade off event.

The results from the structural analysis in Abaqus and a weight evaluation procedure using the geometry in ParaPy are used as inputs for a sizing optimization procedure making use of the Scipy Optimize Sequential Least Squares Programming (SLSQP) algorithm. The objective of this optimization is to minimize the structural weight of the pylon, while subject to constraints on the maximum allowable stress in each component.

The application is validated using two different engine-aircraft integration reference cases. First, the structural weight of the pylon supporting the LEAP-1B engine as employed on the Boeing B737-MAX is evaluated. Secondly, the same is done for the pylon of the LEAP-1A engine as employed on the Airbus A320 neo. While the first analysis results in a structural weight of 236 kg, the second analysis unfortunately had trouble converging. When employing a '*FEM weight-to-realistic weight*' conversion factor between 1.8 to 2.0, it is found that the application is able to find the weight of the pylon structure with acceptable precision.

In conclusion, it is found that the proposed methodology shows promise in analyzing the structural design and weight penalty of a large UHBR turbofan engine, at least when considering wing-mounted engines with box-beam type structures. Nonetheless, future work is necessary especially in order to solve the numerical and structural problems experienced during the tools validation. Furthermore, a proper implementation of other pylon architectures such as fuselage mounted struts remains, as well as the implementation of a more extensive FBO model employing a rotor dynamic simulation of the engine. Finally, when coupled with

a proper engine model, the method shows promise in evaluating the effects of varying engine design parameters as well as integration parameters in the future.

Contents

List of Figures

List of Tables

NOMENCLATURE

LIST OF SYMBOLS

Latin letters and symbols

A	Area	[m^2]
c	Chord length	[m]
C_D	Total drag coefficient	[-]
C_L	Total lift coefficient	[-]
d	(1) Depth	[m]
	(2) (Equivalent) Diameter	[m]
D	Drag	[$kg\,m\,s^{-2}$] / [N]
E	Young's Modulus	[$kg\,m^{-1}\,s^2$] / [Pa]
F / P	Force	[$kg\,m\,s^{-2}$] / [N]
G	Shear Modulus	[$kg\,m^{-1}\,s^2$] / [Pa]
g	Gravitational acceleration	[$m\,s^{-2}$]
I	Moment of Inertia	[$kg\,m^2$]
K	(1) Constant	[-]
	(2) Stiffness Matrix	[-]
l	Length	[m]
L	Lift	[$kg\,m\,s^{-2}$] / [N]
m	Mass	[kg]
M	(1) Moment	[$kg\,m^2\,s^{-2}$] / [Nm]
	(2) Mach number	[-]
n	Load factor	[-]
n_{max}	Maximum load factor	[-]
n_{ult}	Ultimate load factor ($1.5 * n_{max}$)	[-]
N_x	Number of 'x'	[-]
r	Radius	[m]
R_d	Factor of non-circularity (height/width)	[-]
Re	Reynolds number	[-]
S	Surface/planform area	[m^2]
t	(1) Thickness	[m]
	(2) Time	[s]
T	(1) Torque	[$kg\,m^2\,s^{-2}$] / [Nm]
	(2) Temperature	[K]
u	Gust velocity	[$m\,s^{-1}$]
V/Q	Velocity	[$m\,s^{-1}$]
w	Width	[m]
W	Weight	[kg]
W_E	Aircraft Operational Empty Weight	[kg]
W_{feq}	Aircraft Fixed Equipment Weight	[kg]
W_F	Fuel Weight	[kg]
W_{pwr}	Aircraft Propulsion Group Weight	[kg]
W_{struct}	Aircraft Structural Weight	[kg]
$W_{T/O}$	Maximum Take-off Weight	[kg]

Greek letters and symbols

α	Geometric angle of attack	[rad]
β	Newmark-method β coefficient	[-]
γ	Newmark-method γ coefficient	[-]
ε	Strain	[-]
η	Efficiency	[-]
θ	(1) Pitch Angle	[rad] / [°]
	(2) Blade (release) angle	[rad] / [°]
ν	Poisson's ratio	[-]
ρ	Density	[$\mathrm{kg\,m^{-3}}$]
σ	Stress	[$\mathrm{kg\,m^{-1}\,s^{2}}$] / [Pa]
ϕ	Roll angle	[rad]
ψ	Yaw angle	[rad]
ω	Angular velocity	[$\mathrm{rad\,s^{-1}}$]

General subscripts and superscript

$\overline{x}$	Average value
x'	X per unit span
x^{ref}	Reference value
$x^{residual}$	Residual value
x_0	(1) Initial value
	(2) Sea-level value
x_{aero}	Aerodynamic value
x_{ac}	Aircraft
$x_{c/4}$	At quarter chord point
$x_{centrip}$	Centripetal
x_{cr} / x_{cruise}	Cruise conditions
x_e / x_{eng}	Engine
x_{EAS}	Equivalent Air Speed
x_{inl}	Inlet
x_{le}/x_{LE}	At leading edge
x_{max}	Maximum value
x_{min}	Minimum value
x_n	Nacelle
x_{ng}	Nacelle group
x_{oop}	Out of plane
x_p	Polar
x_{prop}	Propulsive
x_{py}	Pylon
x_{rev}	Reverse
x_{struct}	Structural value
x_t	Value at time t
$x_{t+\Delta t}$	Value at time $t+\Delta t$
x_{te}/x_{TE}	At trailing edge
$x_{T/O}$	Take-off conditions
x_{ult}	Ultimate/tensile
x_{wing}	Wing
x_y	Yield
x_∞	Free stream value
$\dot{x}$	Derivative of x
$\ddot{x}$	Second derivative of x

Vector and matrix quantities

$\mathbf{p}$	Vector of external forces
$\mathbf{u}$	Vector of DoFs
$\mathbf{V}$	Velocity vector

$\mathbf{x}$	Vector of design parameters
M	Structural mass matrix
C	Structural damping matrix
K	Structural stiffness matrix

LIST OF ABBREVIATIONS

AECMA	**A**ssociation **E**uropeene des **C**onstructeurs de **M**ateriel **A**erospatial (European Association of Aerospace Industries)
AMC	**A**cceptable **M**eans of **C**ompliance
BPR	**B**y-**P**ass **R**atio
CAD	**C**omputer **A**ided **D**esign
CAE	**C**omputer **A**ided **E**ngineering
CFD	**C**omputational **F**luid **D**ynamics
CG	**C**enter of **G**ravity
COC	**C**ash **O**perating **C**ost
CROR	**C**ontra- **R**otating **O**pen **R**otor
CS	**C**ertification **S**pecification
DEE	**D**esign and **E**ngineering **E**ngine
DoE	**D**esign **o**f **E**xperiments
DoF	**D**egree **o**f **F**reedom
DUUC	**D**elft **U**niversity **U**nconventional **C**oncept
EASA	**E**uropean **A**ir **S**afety **A**gency
ERF	**E**ffective **R**adiative **F**orcing
EU	**E**uropean **U**nion
FAA	US **F**ederal **A**viation **A**dministration
FAR	**F**ederal **A**viation **R**egulations
FBO	**F**an **B**lade-**O**ut / **F**an **B**lade-**O**ff
FEM	**F**inite **E**lement **M**ethod
FOD	**F**oreign **O**bject **D**amage
KBE	**K**nowledge-**B**ased **E**ngineering
KBS	**K**nowledge-**B**ased **S**ystems
MD(A)O	**M**ultidisciplinary **D**esign (**A**nalysis and) **O**ptimization
MoI	**M**oment **o**f **I**nertia
MTOW	**M**aximum **T**ake- **O**ff **W**eight
OAD	**O**verall **A**ircraft **D**esign
OEW	**O**perational **E**mpty **W**eight
OWN	**O**ver the **W**ing **N**acelle
RF	**R**adiative **F**orcing
rpk	**R**evenue per **P**assenger **K**ilometer
SMR	**S**mall to **M**edium **R**ange
TLR	**T**op **L**evel **R**equirement
UHBR	**U**ltra **H**igh **B**ypass **R**atio
UHPE	**U**ltra **H**igh **P**ropulsive **E**fficiency
UWN	**U**nder the **W**ing **N**acelle
VIVACE	**V**alue **I**mprovement through a **V**irtual **A**eronautical **C**ollaborative **E**nterprise
XDSM	e**X**tended **D**esign **S**tructure **M**atrix

1

INTRODUCTION

1.1. PROJECT BACKGROUND

Since the start of aviation in the early twentieth century, the air transport industry has matured into an important link in the global economic and cultural network, allowing people from all over the world to travel to distant locations fast and comfortably. And while the growth of the aviation industry saw a temporary but significant dip during the Covid-pandemic, predictions are that the volume of air traffic will quickly pick up its old pace and continue to grow massively over the coming decades [34]. At its latest (pre-Covid) peak in 2019, worldwide passenger activity exceeded 8.6 trillion *revenue per passenger- kilometer* (rpk)[1], 75% higher than in 2010 and more than 350% higher than it was in 2000. Cumulatively, total global rpk has grown almost 300-fold since 1950 and 75-fold since 1960. And with another 350% growth in air traffic expected until 2070, the end to this trend does not seem in sight [35]. With this continued growth however, the contribution of global aviation activities to *anthropogenic climate change*[2] and global warming rapidly increases as well, as the aviation industry significantly contributes to effective radiative forcing (ERF)[3] through both CO_2- and non-CO_2 (related) emissions. Lee *et al.* [2] estimates that in 2011 aviation alone was responsible for about 3.5% of net anthropogenic ERF. Between 1960 and 2018, aviation related CO_2 emissions increased by a factor of 6.8 to 1034 Tg CO_2 per year. But despite the industries efforts to deliver ever more efficient aircraft and aircraft actually becoming significantly more fuel efficient in that period (mainly as a result of continuous improvements in engine technology), the continuing explosive growth of air traffic has caused the share of aviation in total CO_2 emissions to grow steadily over the years. In Figure 1.1, the growth of global CO_2 emissions is shown, together with the fraction emitted by aviation with respect to total CO_2 emission. From this figure, the enormous increase in air traffic and aircraft efficiency are immediately clear. While Zheng and Rutherford [37] estimate a compound annual fuel reduction rate of 1.3% over the period between 1960 and 2019, the total emissions from commercial aviation have actually increased by 3.7% per year. This highlights even more the immense challenge of decreasing the industry's emission footprint.

An important factor in the emission of greenhouse gasses is of course the propulsion unit. But while radically new technologies like electric- or hydrogen-based propulsion are currently being investigated, these solutions are thought to be mainly suitable for short to regional flights and small- to mid-sized aircraft. As the 20% longest flights by distance worldwide are associated with almost 40% of global aviation CO_2 emissions output [35][36], further increasing the efficiency of the current engines is paramount. An important part of this efficiency increase is the trend to higher and higher *by-pass ratio's* (BPR). The by-pass ratio is the ratio of air flowing through the core of the engine (where it is mixed with fuel and burned to deliver power) to the

[1] Revenue passenger-kilometer (rpk) refers to the number of revenue-generating passengers (i.e. excluding crew) carried, multiplied by the distance flown. It is the common metric of passenger activity for commercial passenger aviation.

[2] anthropogenic climate change refers to the human induced component of climate change, opposed to natural climate change which is a constant historical phenomenon

[3] From [36]: "Radiative forcing (RF) is the change in the net, downward minus upward, radiative flux (expressed in Wm^{-2}) at the tropopause or top of atmosphere due to a change in a driver of climate change, such as a change in the concentration of carbon dioxide (CO2) or the output of the Sun. The traditional radiative forcing is computed with all tropospheric properties held fixed at their unperturbed values, and after allowing for stratospheric temperatures, if perturbed, to readjust to radiative-dynamical equilibrium. Radiative forcing is called instantaneous if no change in stratospheric temperature is accounted for. The radiative forcing once rapid adjustments are accounted for is termed the effective radiative forcing (ERF)."

amount of air pushed past the engine's core by the engine fan at the entrance. This part of the air 'by-passes' the core and never comes in contact with any fuel whatsoever. By increasing the size of the fan and spinning it slower, more air is moved with less velocity increase, leading to an increased *propulsive efficiency* (see definition in Equation 1.1). This in turn leads to a cleaner and more efficient engine.

$$\eta_{prop} = \frac{2}{\frac{v_e}{v_o} + 1} \tag{1.1}$$

With engine manufacturers seeking to increase the by-pass ratio's of their turbofan engines to higher values, the size and weight of the engines has increased significantly over the years. Whereas in the fifties a typical engine was a sleek, relatively small pod fitted to the wing or tail of the aircraft, today's engines have grown so large that they can hardly fit under the wings anymore, forcing some aircraft manufacturers to implement interesting 'design tricks' (like for example the 'hamster mouth' engine cowling under the wing of the Boeing 737NG and 737MAX, see Figure 1.2) to enable proper engine integration. With new *Ultra-High By-pass Ratio* engines with even higher diameter fans currently being designed, engine integration will likely pose even bigger challenges in the future. As the fan diameters go up, the engines will need to be placed more in front and higher with respect to the wing, or perhaps even require relocation to other locations on the aircraft. The pylons supporting these engines need to carry heavier engines, placed at further distance from the wing or aircraft main line. On top of this, the internal forces and inertia's in the engine will be larger, leading to larger dynamic and static loads on the pylons and mountings. All this leads to heavier structures, and more aerodynamic interference with lifting surfaces. As such the question arises to what extend the increased efficiency of the high by-pass ratio engines is counteracted by the increased required structural weight of the supporting structure and deteriorated aerodynamics of lifting surfaces.

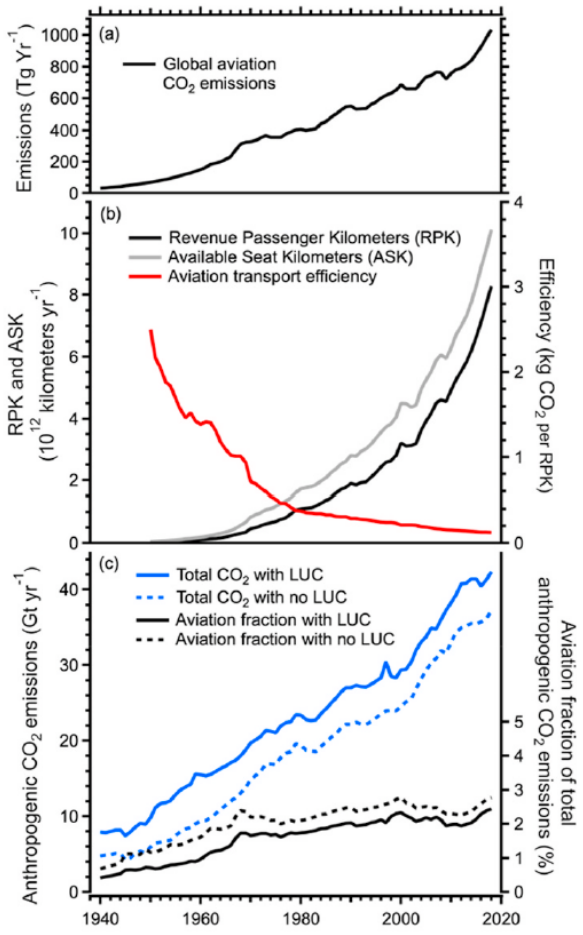
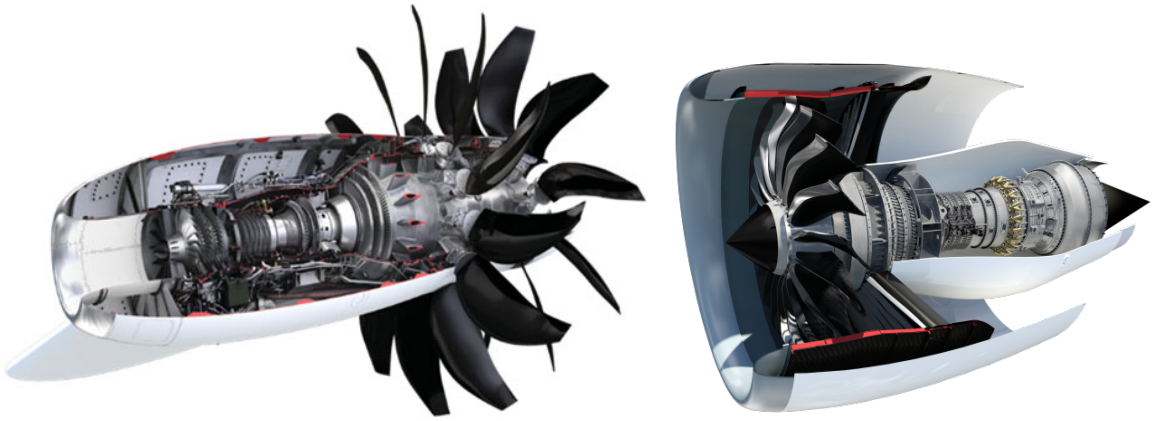


Figure 1.1: Global CO_2 emissions and the growth of aviation between 1940-2018 (LUC= land use change. Defined by UNFCCC as: "A greenhouse gas inventory sector that covers emissions and removals of greenhouse gases resulting from direct human-induced land use, land-use change and forestry activities.") [2].

Apart from further improving current types of engines, the desire for step improvements in engine efficiency also pushes engine manufacturers to think across the boundaries of the 'standard' UHBR turbofan engine design and (re-)consider radically different types of engine technologies. One of the most notable examples of this the recent renewed interest in concepts like the *Open Rotor* or *Unducted Fan* engine architecture, which can reach BPRs of up to 100:1. A recent example of this type of engine is the so-called 'Contra-Rotating Open Rotor' (CROR) engine concept developed by Safran Aircraft Engines in the EU funded 'CleanSky 2' project, shown in Figure 1.3a. This engine features two counter-rotating rows of unshrouded rotor blades, and promises an uninstalled efficiency gain of up to 30% with respect to current day engines, while reaching a BPR of 30:1 [38]. The CROR engine is expected to enter into service in the 2030-2035 time frame.

Concurrent with the CROR concept, Safran also developed a so-called 'Ultra High Propulsive Efficiency' (UHPE) demonstrator employing UHBR technology for introduction on a shorter term (2025-2030) (see Fig-

Figure 1.2: The Boeing 737NG features a nacelle with a flattened bottom, which was designed to provide more ground clearance while employing a high BPR engine without the need to redesign the air plane's landing gear and wings.



(a) Contra-Rotating Open Rotor demonstrator

(b) Ultra High Propulsive Efficiency demonstrator

Figure 1.3: Safran CROR en UHPE concepts as developed in the CleanSky 2 project [3].

ure 1.3b). This concept is mainly focused at small-to-medium range (SMR) aircraft. The main objective here is to validate low-pressure modules and nacelle technology bricks necessary to enable an UHBR engine; this includes a.o. advanced low-pressure fans and short air inlets adapted for UHBR, low-weight structures, low-drag and noise ducts and nacelle, and other technologies [3]. The engine has a BPR of 15:1. This concept is expected to deliver around 5-10% fuel efficiency compared to current market models like the LEAP engine.

The examples shown above represent the ingenuity of engine design companies and they drive to deliver step improvements in performance and efficiency. But in order to evaluate the full potential of these new engine architectures or draw conclusions on which type of engine could deliver the best performance for a specific aircraft or engine-aircraft integration type, airplane and engine designers need to be able to evaluate these engine-aircraft integration designs properly, and preferably as early in the design stage as possible as not to waste precious time and financial resources on unviable solutions. Therefore, a conceptual-design level method able to evaluate the performance of a given engine-aircraft integration design is needed. Unfortunately though, as will be further explained in chapter 2, commonly used conceptual-design level methods for engine-aircraft integration design are mostly empirical or semi-empirical by nature, meaning they are based on statistical data and curve fits of (outdated) engine data. Furthermore, these methodologies lack the ability to properly evaluate the effects of incorporating the radically new engine technologies and different engine architectures like the ones mentioned. A more flexible, physics-based design methodology that is less dependent on old engine data and is able to incorporate innovative engine architectures is therefore needed.

1.2. RESEARCH OBJECTIVE, -QUESTIONS AND -SCOPE

In this thesis, focus is put on the *structural weight penalty* part of the discussion. More specific, we would like to obtain a preliminary weight estimation tool that is able to provide us with a first approximation of the weight of the structure that supports the engine for a given engine and placement on the wing, based on the actual physical properties of the system. Or when using a completely different engine integration strategy, like over-the-wing or fuselage-mounting. In order to obtain this tool, a weight estimation methodology should be developed that is based on first-principle analysis, and is suitable for application during the preliminary and possibly even conceptual phases of the aircraft design cycle. This way, the obtained methodology is design- and geometry sensitive and can be applied to any type of engine and engine-integration design possible. Furthermore, it can be implemented both in a broader early design processes that look at the design of the full aircraft as well on a smaller sub-assembly scale of design. After validating the workings of the methodology, the hope is that the method is suitable for application to both fundamental research on the relation between engine installation and installed weight penalty as well as to analyze and assess the performance of a specific engine-aircraft integration as required.

Summarizing the above, the ***Research Objective*** of this thesis can then be defined as following:

> *""The objective of this thesis project is to create a new, design-sensitive and physics-based Class 2.5 pylon structural design and weight estimation methodology for use in early aircraft design processes, by developing a software tool making use of design principles from knowledge-based engineering (KBE) and validating this methodology against known weight data."*

To achieve the goal as laid out in the research objective, research questions should be answered that combined lead to a well-established conclusion. These research questions should guide the student in first of all getting a feeling for the train of thought and best practices of modern engine-airframe integration, then coming up with a suitable methodology, implement this methodology and validate the method against available engine and pylon data. As such, the following set of ***Research Questions*** were defined:

1. **What considerations govern the airframe-engine integration of modern (ultra) high bypass ratio engines?**
   (a) How is the engine-airframe integration process implemented in industry?
   (b) What structural architectures are generally used in pylon design, and what certification requirements, structural considerations and loads should be considered when sizing the pylon structure?
   (c) What methods are currently used to estimate the structural weight of the pylon, and thus the structural weight penalty of the installed engine, early in the design process of an aircraft?
2. **How can we define an efficient, accurate (with 10% of selected reference pylons), design-sensitive, physics-based Class 2.5 weight estimation method for pylons supporting large UHBR turbofan engines?**
   (a) What is the minimum set of design parameters required to obtain an efficient and reliable pylon parameterization to be used in a structural weight estimation method?
   (b) What application structure is required to produce an effective pylon design implementing structural optimization using a dedicated structural analysis tool leading to a feasible structure that accurately represents a structural design for a pylon with a structural weight prediction within 10% of reference pylon weights for that structural arrangement?
   (c) What methods can be used to model and apply the relevant forces and moments onto a structural pylon model such that a representative set of load cases is generated that is able to effectively size the pylon?

The project described in this thesis report is set up with the goal to answer the research questions as set forth above. As described, the ***Research Scope*** only covers questions regarding the *structural design* of the engine-airframe integration. Aerodynamic considerations are not taken into account in this project. That being said, the method developed in this project could be integrated into a more elaborate research project combining both structural and aerodynamic design into a multidisciplinary design project. The method will be designed with this application in the back of mind. Furthermore, only *podded engines* are considered in

this research, engine support structures of buried engines are not considered. The engine and nacelle will not be considered part of the design domain. Instead, the domain considered in this thesis will be bound by the *hard points* where the engine is attached to the supporting structure on the engine side, and the hard points where the supporting structure is attached to the rest of the airplane on the airplane side. Elasticity of the mounts and bushings will not be taken into account. Secondary effects like aeroelastic wing bending will therefore not be considered in this thesis. The supporting structure will be regarded as a *stand-alone structure*. And finally, we will only consider box-beam type pylon structures.

1.3. STRUCTURE OF THE REPORT

To answer the research questions as posed above, a method is developed that estimates the structural weight of the pylon for a given engine and engine placement. First, an extensive literature study is performed in chapter 2, exploring engine-airframe integration methods, pylon structural design specifics, weight estimation techniques, and KBE fundamentals. Based on the conclusions from the literature study, a methodology is proposed to solve the research questions in chapter 3. This methodology is comprised of four components,of which the specifics are described in the following chapters. In chapter 4, the mathematical parameterization of the pylon is described, which serves as a basis for the weight estimation tool. The implementation of the tool itself using the Python-based ParaPy KBE library is described in chapter 5, together with the geometry generation in ParaPy. To enable structural analysis of a proposed geometry, the tool is coupled with Abaqus using is dedicated API, that is described in detail in chapter 6. The forth component is the structural sizing using the Scipy Optimization Python library, that is described in chapter 7. And finally, the code is validated using two reference cases, the installation of the LEAP-1B engine onto the Boeing B737 MAX and that of the LEAP-1A engine onto the Airbus A320 neo. The results of this validation are described in chapter 8. In the final chapter, the conclusions of the research are presented and recommendations for future research are given.

2

LITERATURE REVIEW

In this chapter, the theory behind the (structural) design of the engine-airframe integration will be explored. In section 2.1 a broad overview of the design considerations that play a role in the choice for a specific engine integration strategy is given, and the generally applied methodology for conceptual engine-airframe integration design is explained, including some examples from literature. Then in section 2.2 we will focus our attention onto the structural design of the support structure for the engine, which is generally referred to as the *pylon* or *strut*. We will look at what basic structural concepts are often applied, and what loads should be considered during the sizing of these structures. Next up in section 2.4 structural weight estimation will be further explored, and we will specifically mention some recent examples of physics-based 'class 2.5' weight estimation methods used in conceptual and preliminary aircraft design. Finally in section 2.5 we will elaborate shortly on the concept of 'knowledge-based engineering' (KBE), a design strategy that has recently gained interest in the aircraft design world, and that will form the basis for the structural weight estimation method developed in this thesis. The chapter will conclude with a quick summary of the most important findings from this literature review, which enables us to answer the first Research Question presented in section 1.2 fully.

2.1. ENGINE-AIRFRAME INTEGRATION: AN OVERVIEW OF PROBLEMS AND DEVELOPMENTS

In this section we will look a the integration of the engine into the full airframe on a system level. We will look at the considerations that play a role in deciding where the engine should be placed. And we will look at how modern integrated engine-airframe integration design is done, using examples employing new design techniques like Multidisciplinary Design Optimization and Knowledge-based Engineering.

2.1.1. TRADITIONAL ENGINE-AIRFRAME INTEGRATION AND THE PROBLEM OF BIGGER ENGINES

One of the most fundamental choices to be made in the process of designing and aircraft is the type, number and integration of the propulsion system into the overall layout. Preliminary sizing provides the most important overall aircraft performance characteristics, which leads to a design choice for the type and number of engines required to deliver the required performance. Following this step, design choices need to be made like;

- location of the engines with respect to the airplane: in the nose, under the wing, at the rear fuselage, etc.
- integration of the engines into the airframe: buried into the fuselage or podded, in a nacelle.
- interference with lifting surfaces: under or above the wing, in combination with T-tail or canard, etc.

Due to the large impact this choice has on the whole aircraft performance (influencing among others aircraft stability, drag, maintenance and more), this choice is usually made in the earliest phases of conceptual design. But unfortunately, due to the relative broad and high-level character of this design phase, these choices

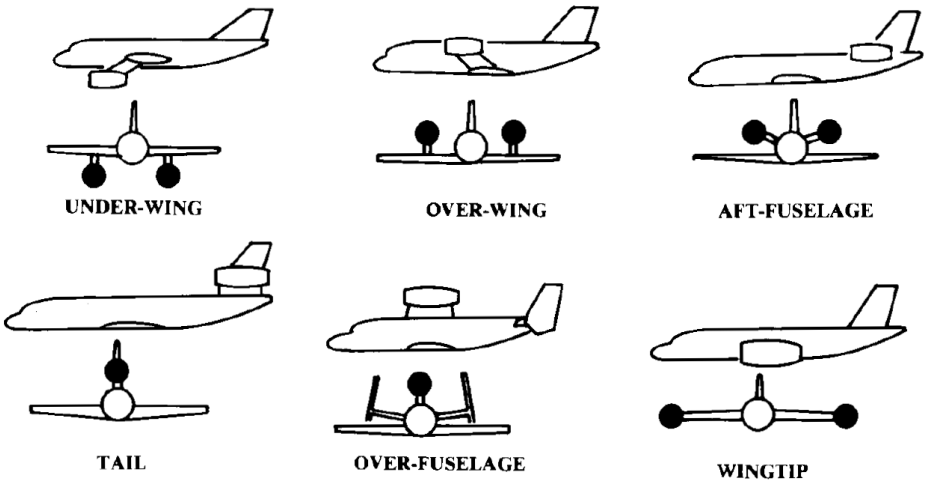


Figure 2.1: Possible locations of podded engines [4].

generally have to be made with little detailed knowledge on the differences between the design options (the so-called 'Knowledge Paradox'). Modern design approaches like MDO and KBE increase the designer's possibilities to weigh the pros and cons of different topologies to greater detail than ever before, and earlier in the process.

In this thesis project, focus is lays on podded engines mounted on the wing (underneath- or over the wing) or at the rear fuselage through pylons (also called struts or frames). Furthermore we focus our efforts on the integration of UHBR turbofan engines; large, high power engines suitable for mid- to long range commercial aircraft. Traditionally, these aircraft employ the so called 'tube and wing' layout (see upper left in Figure 2.1), with engines mounted in nacelles placed slightly under and forward of the wing. As explained in the introduction, new high efficiency UHBR or even Open Rotor engines lead to larger fan diameters, making integration of these engines using the traditional under-wing layout harder, as ground clearances start posing a problem and the additional structural weight and aerodynamic interference effects start playing a larger and larger role. For this reason, alternative locations like on the rear fuselage are also being considered as possible locations for future engine placement. But other locations could be considered, as shown in Figure 2.1.

2.1.2. ENGINE INTEGRATION IN INDUSTRY AND THE INTRODUCTION MODERN DESIGN TECHNIQUES INTO THE DESIGN PROCESS

When the location and type of the engine onto the airframe is set, the engine-airframe integration needs to be properly engineered. First, we will look at the 'classical' engine-airframe integration process as was employed roughly throughout the second half of the 20th century. Then, selected examples of more recent airframe-engine integration design and research projects using modern design techniques and numerical models are further examined to get a grasp of how the engine-airframe integration design process is tackled in modern day design projects.

EARLY TURBOFAN ENGINE INTEGRATION

With the emergence of the first turbojet engines and the push for aircraft flying at higher speeds and altitudes in the late '40s and '50s, the integration of the engines with the aircraft or wing more and more became a topic of interest. Pioneering research done in the late '40s by Boeing on the B47 and Sud Aviation on the Sud Caravelle had already shown the advantages of using podded- over integrated engines in terms of handling, maintenance and vibration, which from that point on led to podded engines to become the configuration of choice for most aircraft designs. A great paper showing the primary concerns for aircraft and engine designers at the time was written in the late '60s by Ward *et al.* [9] from Rolls Royce Ltd. In this paper, the most important factors influencing engine-airframe integration design are listed to be aerodynamics, weight distribution, vibrations, passenger comfort, accessibility, handling and maintainability. Extra attention is given

by the writers of the paper to accessibility and maintainability, and furthermore different choices of packaging of the engine with auxiliary systems like the starting unit, gearbox and (fuel) control systems on engine performance and integration are stressed. Several classical aircraft design textbooks furthermore explain the traditional integration of the engine onto the aircraft in great detail, like the standard works of Roskam [39][40] and Torenbeek [17]. When in the late 80s and early 90s engine power and BPRs started to increase, the aerodynamic and structural integration of the engines became more and more of a problem. A much referenced paper on the challenges and problems involved in the integration of these under-the-wing mounted high bypass ratio turbofan engines was written by Berry [41]. In this paper, Berry describes the design choices made while designing the engine-airframe installation of the engines of the first Boeing 777 airplane. The engines installed in the 777 were the largest both in size and thrust ever installed on an aircraft at that time, with a nacelle diameter of up to 160" (more than 4 meters), a BPR of 9 and over 100,000 lbf (more than 45,000 kg) thrust generated. Though the integration design process is described as 'multidisciplinary', the paper shows a clear aerodynamics-driven design strategy, indicating that other factors like the weight of the supporting structure were considered less problematic at the time. Indeed, the structural weight and stiffness of the struts was mainly limited by wing flutter design requirements, with the weight of the structure found only *after* flutter testing was already completed on the final design. The location and orientation of the engines was completely decided by the aerodynamic interference between engine and wing, and structural considerations played hardly any role at all in this decision. This is a common theme in many sources describing engine-airframe integration from this era, like for example Henderson [42], Carlson and Lamb [43], Naik [44], Henderson [45] and AGARD [46].

MULTIDISCIPLINARY DESIGN OF AIRCRAFT PYLONS

In section 1.1 we already mentioned that with increasing BPR and fan diameter, the risk exists that gains in uninstalled engine performance are undone by the increased aerodynamic and structural penalties in the installed case. This is recognized by many scientists, for example Magrini *et al.* [47]. He furthermore notes that the complex coupling between aerodynamics, structural design, operations and other stakeholders requires a different design approach than the more traditional procedures based on sequential and decoupled design, as an optimal overall design point is hard to find. One design strategy that has gained significant traction over the past 15 years and is specifically developed with this complexity in mind is the so-called **Multidisciplinary Design Analysis and Optimization** methodology. MD(A)O is a design method that aims at using the increased computational power of modern computers to approach conceptual aircraft design from a more holistic, multidisciplinary angle where multiple medium-to-high fidelity disciplinary analysis tools are coupled together such that dependencies between the different aircraft design disciplines can be taken into account early in the design process in an attempt to obtain a design that is optimized for several use cases and objectives. Sobieszczanski-Sobieski *et al.* [48] defines MDO as:

> *"... An assemblage of methods, procedures and algorithms for finding best designs, measured against a set of specified criteria, for complex engineering systems with interacting parts, whose behavior is governed by a number of coupled physical phenomena aligned with engineering disciplines. ..."*

The use of MDO for the design of engine support structures has seen an interesting growth over the past 15 years. The first European research programs where MDO was implemented successfully where executed at Airbus and ONERA between 2005-2010 under the VIVACE project, an EU funded research project executed under AECMA supervision addressing the 'European Aeronautics: a Vision for 2020'-objectives. Results of these efforts were described by Mouton *et al.* [5] and Grihon *et al.* [6]. Figure 2.2 shows the model used in these studies. The result of these studies showed that the pylon structure gets heavier when the horizontal and vertical distance between wing and engine is increased (mainly due to an increased arm leading to higher moments caused by blade-off loads) or when the pylon width is decreased (leading to thicker spars to sustain the same moments of inertia). On the aerodynamic side, the researchers found that drag was increased when *(1) placing the engine horizontally father away from the wing, (2) widening the pylon* and to a smaller extent *(3) placing the engine vertically father from the wing.* While point (1) and (3) are in agreement with theory, the credibility of point (2) is challenged by the researchers.

Though the research conducted in the VIVACE project showed very promising results, implementation into the actual industrial design process showed to be challenging. Therefore, a second attempt was performed by Gazaix *et al.* [7]. The design method proposed here is basically a more advanced implementation of the work done by Mouton *et al.* and Grihon *et al.*. The impact of engine position variation on global aircraft performance was measured in this research in terms of 'Cash Operating Cost' (COC), defined as the

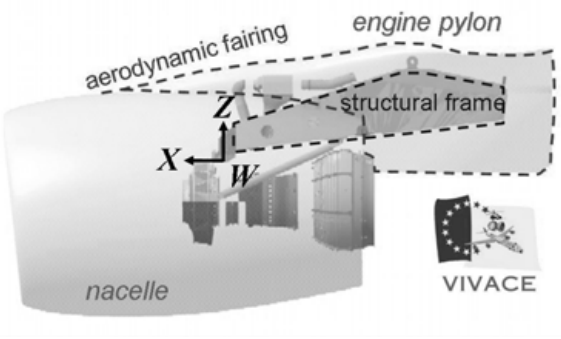
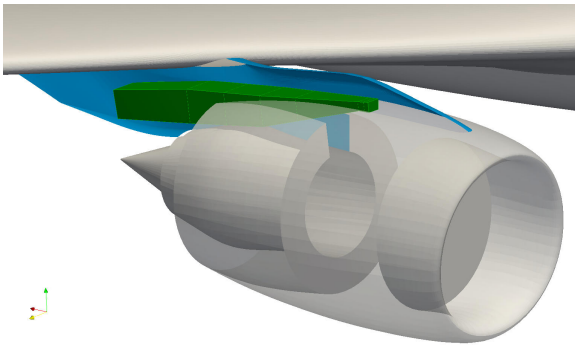
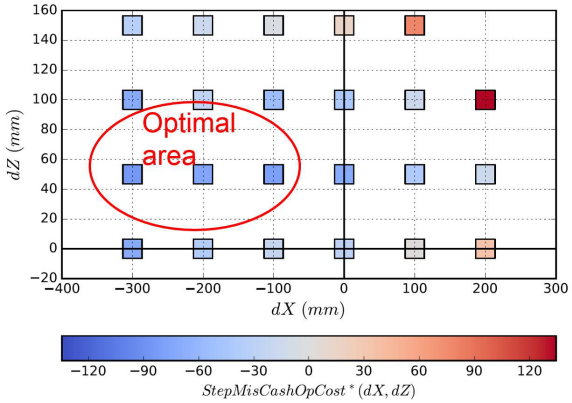


Figure 2.2: MDO model used in Mouton *et al.* [5] and Grihon *et al.* [6].



(a) Optimized structures obtained from the structural and aerodynamic mono-disciplinary optimization loops. Blue depicts the optimal aerodynamic fairing shape, while green depicts the optimal structural shape.



(b) |Graph showing the optimal 'trade-off' area for pylon design according to the DoE analysis.

Figure 2.3: Optimization results from Gazaix *et al.* [7] from the DoE analysis.

cost the airline plays to fly the aircraft; this is influenced by fuel costs, maintenance costs for aircraft and engine, crew costs, landing fees and navigation charges. This is clearly a more hands-on design approach than the one employed in the VIVACE projects. The process follows a two-step protocol; firstly, full high-fidelity mono-disciplinary aero- and structural optimization cycles were performed leading to two optimal mono-disciplinary designs. These were then coupled in a lower-fidelity 'Overall Aircraft Design' (OAD) level. Results from the structural optimization indicated that critical loads appear in the rear engine-pylon attachment. Flutter was never critical. The best design was found to be one where the engine is moved rearwards horizontally, bringing it closer wing, while vertically the engine should be placed down with respect to the wing. The aerodynamic optimization delivered an optimum engine placement further away and moved upwards with respect to the wing, so very different than the structural optimum. An optimum result was found to be clearly a compromise between the structural and aerodynamic solutions, where the aerodynamic solution showed a higher gain in cost reduction.

2.2. STRUCTURAL DESIGN OF ENGINE SUPPORT STRUCTURES

In the previous section, engine-airframe integration was discussed from a 'system-level' point of view; we looked at different engine integration architectures, their pros and cons, and what methods and considerations are employed in the design of the engine-airframe integration. Finally, we saw how recently modern design methodologies like MDO are being employed more and more in order to obtain a pylon-nacelle assembly that is able to achieve the best possible performance for multiple engineering objectives. However, as stated in section 1.2, the scope of this thesis is limited to the structural design of the engine support structure, which in most cases is either a pylon, a strut or sometimes a supporting frame. In this chapter, the design of this engine support structure will therefore be discussed in more detail. In subsection 2.2.1, we look at several much used engine support structure architectures for wing- and fuselage mounted engines. In sub-

section 2.2.2, we focus our attention on some of the legal requirements and certification regulations set by national and international government bodies regarding engine installation. Finally in section 2.3, the sizing loads and other structural considerations playing a role in engine support structure structural design are described in more detail.

2.2.1. TYPICAL ENGINE SUPPORT STRUCTURE ARCHITECTURES

In this section, the structural design of engine support structures is discussed. We will look at the most common structural arrangements for UWN type engine mounts, OWN type engine mounts and rear-fuselage mounted engines, respectively. In section subsection 2.1.2, we already saw some examples of wing-mounted engines and their support structures, which generally take the form of so called 'pylons' or 'struts'. The examples often showed a sort of 'box-beam' like structural arrangement; this is actually a common design for all types of engine support structures. Nevertheless, finding detailed information on the detailed structural designs of the pylons is quite difficult. Few sources treat this topic in detail, and most sources that do don't mention detailed specifications, like dimensions and weight. Also, descriptive studies on different types of pylons and their characteristics are very sparse. The only source of detailed structural information and drawings lay in a few technical reports of crashes (for example those prepared by the Australian Transport Safety Bureau [49], NTSB Bureau of Accident Investigation [24] or Wanhill and Oldersma [31]), in patent databases of in sparsely available technical documentation from (former) airplane design firms. One of the most well known sources describing detailed structural arrangements of pylons can be found in the book *'Airframe Structural Design'* by Michael C.Y. Niu, a former structural engineer at Lockheed [8]. In this book, mr. Niu describes the most common structural designs for both under-the-wing and rear-fuselage mounting systems and pylons, including important structural and aerodynamic configurations.

WING-MOUNTED ENGINE PYLON STRUCTURES

Niu described three different 'standard' arrangements for supports structures employed in wing-mounted engine integration, as shown in Figure 2.4. In all cases, the main structure of the pylon is made of a *box-beam*, with ribs and flanges, which may or may not be supported by secondary struts. Materials that are used are aluminum, steel or titanium. Titanium and steel alloys generally applied on the front engine mount bulkhead, the lower spars and aft engine mount bulkhead, all of which act as firewalls between the engine and the supporting structure, but in some cases are also used on other components of the pylon. In all cases, the pylon is attached to the *front spar* and *lower skin panel* of the wingbox structure. Pylon loads are distributed into the wing in such a matter that wing box secondary deformations are minimized.

- **The Drag Strut Installation.** Shown in Figure 2.4.a, this type of structure employs a cantilevered box-beam structure consisting of two upper and two lower longerons (longitudinal spars or stringers), supported by a secondary strut (the 'drag strut'). The two side panels of the pylon box transmit vertical shear forces, while the lower panel carries the lateral shear forces. The engine loads are transferred to the pylon via the front and aft mount bulkheads. The upper rear of the pylon is attached to the wing via a 'lug attachments'. The lower rear part of the pylon box is attached to the wing via the drag strut, which applies the lower longeron loads to a point between front- and rear wingbox spar. Examples of this arrangement are the pylons on the Lockheed L-1011, shown in Figure A.1.

- **The Box-Beam Installation.** In this type of pylon structure (shown in Figure 2.4.b), the box structure is extended underneath the wingbox up until the aft wing attachment point. Advantages of this design are a decrease in required wingbox structural strength (and thus weight), at the expense of a heavier pylon structure. Furthermore, this arrangement mitigates some of the fatigue problems at the lower wingbox skin surface. The pylon is attached to the wing using three attachment points. One *on the front spar* which transmits the vertical (z-) and lateral (y-) loads. One *underneath the front spar* to transmit thrust loads. And finally a fitting *on the lower wing box surface* for vertical (z-) and side bending (M_z) loads. Examples of this structure are numerous, for example on the Airbus A-300 (Figure A.2), the McDonnell Douglas DC-10 (Figure A.4), the Lockheed C-141 Starlifter (Figure A.6) and the Lockheed S03A Viking aircraft (Figure A.5). But this arrangement was also used in all research on pylon MDO design discussed in subsection 2.1.2.

- **The Upper Support Arm Installation.** The most complex but also most fail-safe structure, shown in Figure 2.4.c, interestingly leading to a relatively low weight. In this configuration, the pylon box-beam

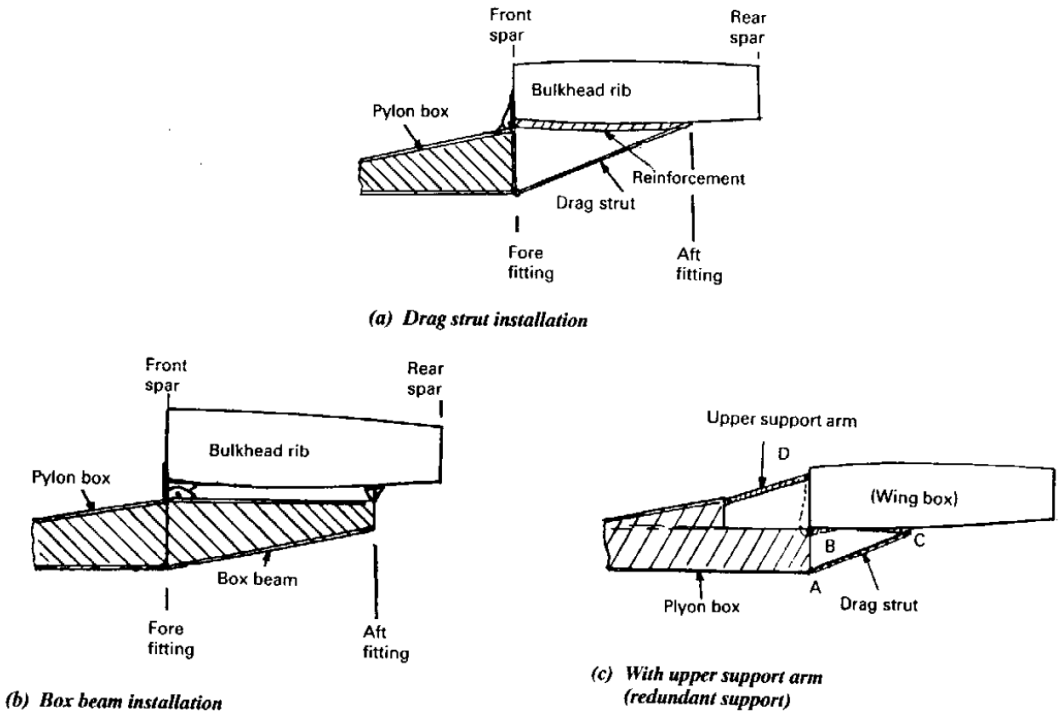


Figure 2.4: Three typical under-the-wing pylon structural topologies as identified by Niu [8].

structure is made up of not 2 but 3 longitudinal spars; the upper, mid- and lower longerons. The structure is further supported by an upper link and diagonal brace (drag strut). Due to this arrangement, this structure provides the most efficient load distribution onto the wingbox, leading to both a light pylon and a light wingbox structure. The structure is attached to the wing box at four points using so called 'fuse pins'; hollow carbon steed bolts that are heat-treated to shear-fail at a predefined load. Due to the inherent redundancy, the upper link can fail while maintaining an alternative load path through the lower diagonal brace. In case of an emergency engine landing (discussed in section 2.3), the structure is designed such that the upper and lower links fail first, enabling the engine and pylon to rotate upwards around the mid-spar attachment point. As a final advantage, the engine can be positioned further forward without severe structural weight penalties. All this does come at a cost - the statically indeterminate structure leads to a complex structural analysis process, making quick design and analysis very hard. Examples of this arrangement can be found on the Boeing 747 (shown in Figure A.7 and Figure A.8), the Boeing 767.

REAR FUSELAGE-MOUNTED ENGINE SUPPORT STRUCTURES

In case of rear-fuselage attachment, Niu again describes three different support structures, shown in Figure 2.5. The structures presented here are very different from the ones employed for wing-mounted engines. The structure does not consist of a box-beam, but instead consists of a set of struts with 'clamps' to keep the engine in place. In some cases, this strut may be supported by a diagonal link, called the thrust rod. These arrangements don't necessarily assume high BPR turbofan engines (the book was written in the late 80s). When the engines are placed further from the fuselage and the pylons get longer, a box-beam structure designed to take on the structural loads presented here makes sense, and in more modern designs, we do see this a lot.

- **Cowling Mount.** Shown in Figure 2.5.a, this structure features a cowling that fully encapsulates the engine. The structure consists of an integrated structure, with a light engine mount, but the heaviest cowling of the three structures. The cowling is used as the engine support structure. This design was very popular for fuselage-mounted dual engine or 'Siamese' engine concepts where turbojet engines are applied. An example of an aircraft using the structure is the Lockheed Jetstar, which features four rear mounted jet engines, embedded in a two-by-two pairing in a rear cowling. But also the Vickers VC10 and the Ilyushin Il-62 use this setup.

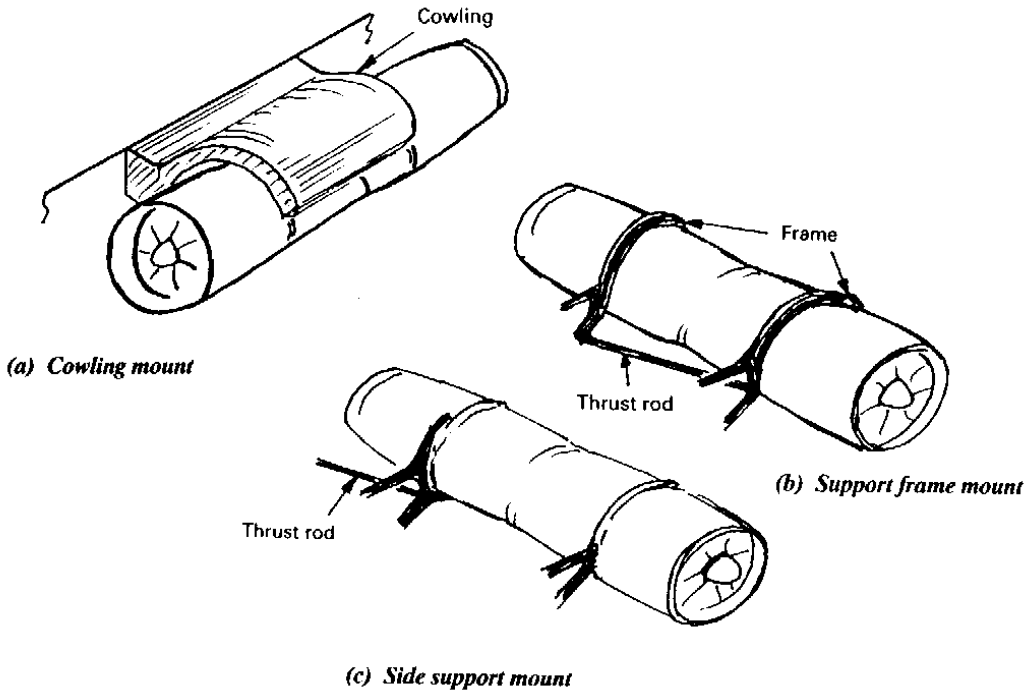


Figure 2.5: Three typical rear fuselage mounting structural topologies as identified by Ward *et al.* [9] and Niu [8].

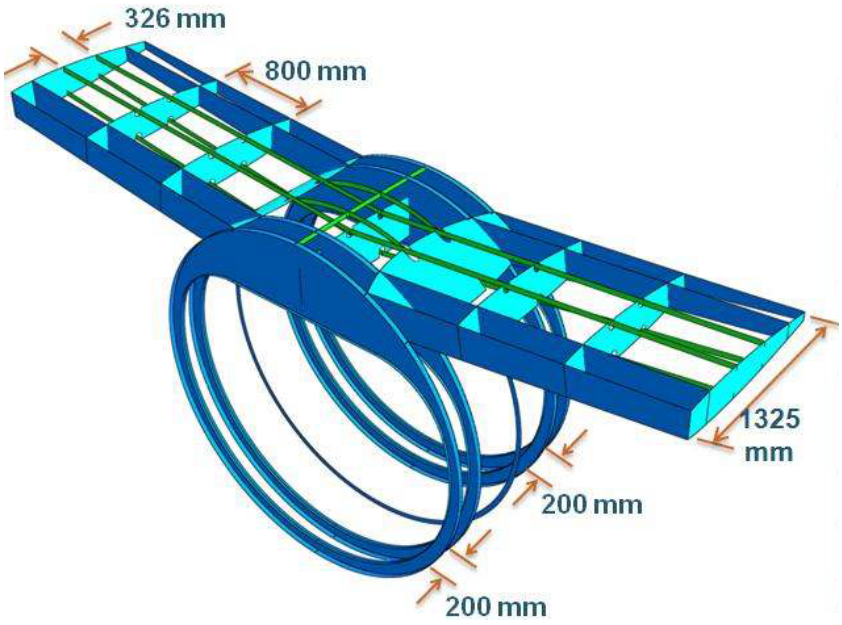


Figure 2.6: Fuselage installation employing box-beam type structures as designed by Verri *et al.* [10].

- **Support Frame Mount.** In this setup, shown in Figure 2.5.b, the engine is suspended in a frame of two arched cantilever beams bolted to the fuselage. These beams are mostly made from forged steel, using the 'safe-life' design philosophy. The engine is suspended underneath the arched beams at three points; two on the front beam, and one at the back. On the front beam, the engine is suspended at the upper side using a set of links, and with a trunnion (a pin protruding from the side of the engine suspended in a fork-type bracket) on the inboard part of the engine casing. The trunnion links the engine to the thrust rod, a tabular steel strut that is diagonally installed between the front side attachment point and a fitting at foot of the rear cantilever mounting beam. The side mounting itself takes axial, lateral and vertical loads, while the top link can only take vertical and moment (M_x) loads. The rear suspension point is at the top end of the arched beam and takes vertical and lateral loads. An example of such a structure can be found on the Fokker F-28 Fellowship.

- **Side Support Mount.** Finally a simpler, slightly modified version of the support frame uses side mounts to support the engine, as shown in Figure 2.5.c. Here, the attachment points are placed on the upper and lower side of the front suspension frame, taking up load in all directions including the torque M_x, while one attachment point at the rear only takes vertical and lateral loads. In this case, the thrust rod in placed at the root of the rear suspension arm and leads torque directly into the airframe.

With the advance of high bypass ratio's also being applied more and more in aircraft with side-mounted engines, leading to heavier engines that are harder to mount using lightweight support frames, a clear evolution towards the use of box-beam type structures for more structural stiffness can be observed in recent years. This leads to the development of a new, fourth type of side-mounted support structures:

- **Box-Beam Side Strut.** Shown in Figure 2.6, the box-beam side support strut employs the same box-beam type structural elements as seen in box-beam type wing-mounted pylons and wingbox structures. The strut is basically a further development of the side-support mount frame, where the frame is strengthened using extra webs and stringers for more stiffness and rigidity. The strut can either be designed as a stand alone structure that runs through the fuselage similarly as is the case for many wing box structures, or as one integral structural part with the fuselage, as is done in Figure 2.6. The design shown in Figure 2.6 was designed by Verri *et al.* [10] at Embraer in 2008 for a new medium-range, 50 passenger aircraft employing turboprop engines, based of the EMB-145 turbofan propelled commercial jet.

2.2.2. CERTIFICATION REQUIREMENTS

Certification requirements play an important role in the design of a pylon. A large part of the structural requirements of aircraft is directly related to it. The standards for certification can be different per country or group of countries, although much effort is put into unifying the rules and regulations as much a possible. In Europe, airworthiness and environmental certification requirements are managed and enforced by the European Union Aviation Safety Agency (EASA), representing in total 31 member states (EU + Switzerland, Norway, Iceland and Liechtenstein). Certification rules (*what* does the aircraft need to satisfy) are explained in 'Certification Specifications', one for each class of aircraft. The standard procedures for testing and data collection (*how* does the aircraft satisfy the rules) are described in the 'Acceptable Means of Compliance' (AMC). For Large Transport aircraft (with a MTOW of over 5700 kg), certification rules and AMCs are codified under CS-25. A summary of the rules applicable to engine mounts and pylons can be found in Appendix A.2, as taken from EASA [50].

In the US, certification and monitoring is governed by the Federal Aviation Administration (FAA) and certification rules are written in the Federal Aviation Regulations (FARs). These are codified into US law under 'CFR 14, Chapter 1: Federal Aviation Administration, Department Of Transportation'. In this text, the certification process and airworthiness standards are discussed in 'Subchapter C', in Parts 21 to 49 of the chapter. But for our purposes, 'Part 25 - Airworthiness Standards: Transport Category Airplanes' is the only relevant chapter. The FARs follow the same numbering as the CS's and their content is also similar, to simplify the certification process even further.

2.3. STRUCTURAL LOADS AND SIZING

The engine support structures discussed so far are all structural members, and their purpose is to transfer loads from the engine into the airframe. The character and sources of these loads can vary significantly,

spanning everything from continuous static thrust loads, to highly dynamic engine failure loads. One load case of specific interest is the 'fan blade-off (FBO) event', which is often identified in literature as one of the more important sizing load cases for pylon, but is also notoriously difficult to model and predict. The main goal of this section is identify all sources of loads that the engine support structure should be designed for to carry, and how these loads can be modeled effectively.

2.3.1. LOAD CASES AND LOAD FACTORS

External loads are defined as *all forces and moments applied to the aircraft structural components to establish the strength level of the complete aircraft* [11]. The goal of the substructure is to carry these loads, and redistribute them into the entire structure. Loads can be the weights of components and systems, inertia forces (maneuver loads), aerodynamic loads, vibrations and oscillatory loads, or impact loads caused by for example (crash) landing. But also fluctuating loads varying over longer periods of time can lead to structural failure. The loads are defined using load cases; sets of loads that together specify one specific condition of the aircraft or structure. For structural analysis, load cases can generally be subdivided using one of three types:

- Static load cases. These can contain weights, aerodynamic loads, impact loads, maneuver loads, gusts, etc. Static loads are constant and relatively simple to predict and model.
- Dynamic load cases. These can stem from oscillatory loading, vibrations or dynamic gusts (turbulence). Also foreign object damage leading to asymmetric blade damage can lead to dynamic load profiles.
- Fatigue load cases. Fatigue loads are important loads for structural design, as they do not lead to immediate failure when applied, but can lead to failure on the long run when applied a certain number of times over a period of time. These loads are especially dangerous as their limiting loads are usually much lower than the limits imposed by instantaneous gust or maneuver loads.

The loads and load cases for which an airplane or the airplane components need to be designed are largely determined by government imposed regulations and certification requirements, which dictate *minimum* loads the airplane is required to withstand; this was discussed in subsection 2.2.2 on certification requirements. But of course airplane manufacturers can always decide to design for higher load factors, for example to build in some safety margin for testing purposes or as a marketing tool.

An important concept when communicating about aircraft loads is the **load factor**, designated by the symbol n. The load factor is a multiplication factor which defines a load in terms of weight or g-forces [8]. Under 'standard' conditions (straight, level flight), the load factor would be 1; meaning the load (i.e. lift) the aircraft experiences is equal to its weight. But when the aircraft transitions into a different flight condition, for example a maneuver (a turn or a dive), the accelerations the structure experience lead to an increased or decreased (in case the acceleration is opposite to the weight vector) perceived loading. The maximum and minimum load factors an aircraft is expected to bear during its service life are graphically depicted in *V-n diagrams* (velocity-load factor diagram) and a *gust load diagrams*, which together depict the flight envelope of the aircraft. Typical V-n diagrams for transport aircraft are shown in Figure 2.7.

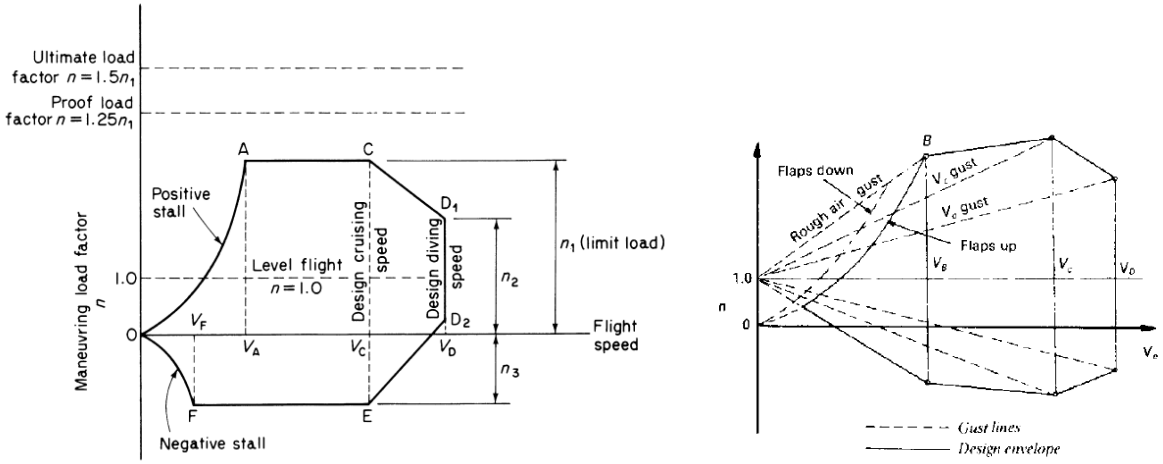
The maximum load factors (both positive and negative loads according to the standard sign convention) the aircraft is expected to experience during its 'normal' service life are designated as the 'limit loads', given by n_{max} and n_{min}. The ultimate load factor or design load factor n_{ult} represents the maximum load factor multiplied by a safety factor 1.5, as given in Equation 2.1:

$$n_{ult} = 1.5\, n_{max} \tag{2.1}$$

While limit loads are loads the aircraft should be able to support without permanent structural deformation, ultimate loads are those loads for which *structural damage* is deemed acceptable, but that the structure is required to bear without *structural failure*. Both limit and ultimate load factors are used in design, and can vary per component and per direction or axis (axial, lateral and vertical). For that reason, using the right load factors in design and analysis of a structure is detrimental in successfully deciding the required stiffness and topology of the structure.

2.3.2. STATIC LOADS

Knowing the importance of applying the correct loads and load factors for structural design, we would like to know what loads play a role in the structural design of engine support structures. Furthermore, it would be



(a) V-n diagram for maneuver loads [13].

(b) Gust envelope or V-g diagram [8].

Figure 2.7: Typical V-n/V-g diagrams for a transport aircraft.

nice to acquire a feel for the sizing loads in pylon and nacelle design. To investigate what loads are relevant, which of these are actually sizing and how they can be modeled effectively, we combine information from design textbooks (specifically Niu [8] and Niu [11]), certification regulations as discussed in subsection 2.2.2, discussions with engine designers and results reported in scientific literature.

The static structural loads that play a role in engine support structure sizing can mostly be characterized as either:

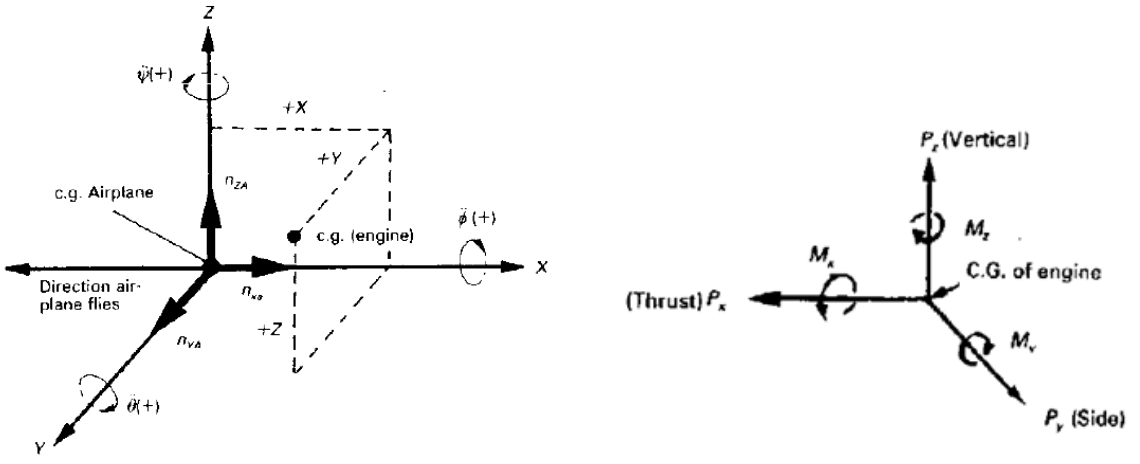
- *Engine loads.* For example thrust generated by the engine fan and/or core. Or; vibrations stemming from the rotations of the internal components of the engines. But also loads resulting from an FBO-event can be classified as engine loads.
- *Inertia loads.* These loads are generated whenever the engine experiences accelerations due to the inertia of the engine. For example when the airplane makes an in-flight maneuver, while landing or taxiing. Due to the inertia of the engine and its position away from the rotational center (which is generally the aircraft CG, the engine and supporting structure experiences heavy forces and moments. Furthermore, as the engine contains components that rotate, movement tends to generate gyroscopic moments which the engine support structure should bear.
- *Aerodynamic loads.* Finally some loads stem from interaction with the surrounding air; aerodynamic loads. These can be lift and drag, but also gusts or other turbulent air loads. Note that gusts can also lead to inertia and gyroscopic loads and should therefore be considered whenever an aerodynamic load case is considered.

Niu [8][11] describes the most important loads for propulsion integration sizing in his books on airframe structural design as thrust loads, air loads, inertia loads and gyroscopic loads. Moreover, he mentions that the engine supporting structure experiences very large inertia load factors in dynamic landing-, taxi- and gust-conditions; in the order of 5-8g. For preliminary sizing of the nacelle, nacelle strut and engine mounts, several loading conditions are suggested in his books, that can give some handholds for design. An overview of these suggested loading conditions is given in Table 2.1. The definitions of the positive forces and moments applied in this table are shown in Figure 2.8b.

On top of this, a designer should consider the possibility of a so-called 'engine break-away event'. This can occur either in the event of a wheels-up emergency landing, where the load factor can become as high as 9g [11], or as a result of the resonance occurring after a fatal engine blow-out due to the loss of fan blade material (caused by for example and FBO- or FOD-event). In these cases, certification requirements require that the airframe manufacturer ensures measures are installed to prevent fuel tank rupture in the area of separation (see Appendix A.2 - CS 25.721(c)). Logically, this is mainly interesting for wing-mounted engine installations.

Table 2.1: Inertia load conditions for preliminary design of nacelle, nacelle struts and engine mounts of commercial transport aircraft proposed by Niu.

Condition	**Ultimate Load Factor (n_{ult})** [8]	**Loading** [11]
P_z (Vertical)	6.5 $6.5 + 1.5\ T_{cruise}$ -3.5 $-3.5 + 1.5\ T_{cruise}$	Landing impact (-4.0 to +8.0 load factors on engine weight)
P_x (Thrust)	$3.0T_{max} + 3.0$ vertical $3.0T_{max} + 1.5$ vertical $3.0T_{reverse}$ $3.0T_{reverse} + 3.0$ vertical	3.0 load factors on engine thrust (alone or combined with vertical load) 3.0 load factors on reverse thrust (alone or combined with vertical load)
P_y (Side)	+3.0 −3.0	Roll side loads or 2.5 load factors side on engine weight
Gyroscopic	± 2.25 rad s^{-1} yaw $+ 1.5\ T_{cruise} + 1.5$ vertical ± 2.25 rad s^{-1} pitch $+ 1.5\ T_{cruise} + 3.75$ vertical	Yawing gyroscopic (alone or combined with vertical and thrust) Pitching gyroscopic (alone or combined with vertical and thrust)
Engine seizure	Torque = M_x ~ required to stop total blade mass in 0.6 sec	Dynamic unbalance and shock mounting



(a) Aircraft level

(b) Engine level

Figure 2.8: Notations and sign conventions used by Niu [8].

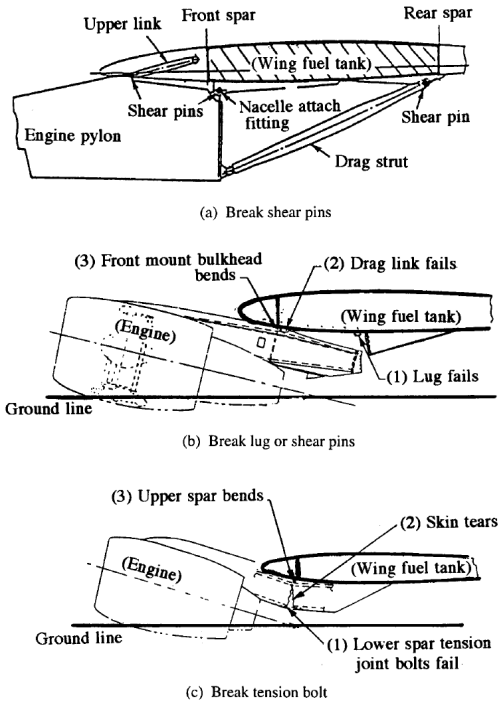


Figure 2.9: Engine breakaway case in emergency landing [11]

To prevent this situation from occurring, usually predefined 'separation points' are designed, usually in the form of fuse bolts or shear bolts, that fail under specific loads such that the entire engine-pylon assembly is shed off from the rest of the aircraft. Examples of often used measures to prevent fuel tank rupture in modern aircraft are shown in Figure 2.9, and was also shortly discussed in subsection 2.2.1 while discussing the *Upper Support Arm installation.*

THRUST AND WEIGHT LOADS

The primary objective of the engine-airframe installation is to redistribute thrust loads from the engine into the airframe. Several thrust settings should be considered, leading to different load vectors. Most importantly, maximum take-off thrust $F_{T/O_{max}}$, maximum continuous thrust $F_{cont_{max}}$, maximum continuous flight idle $F_{idle_{max}}$ and maximum reverse thrust $F_{rev_{max}}$.

AIR LOADS

Aerodynamic flows on the nacelle and pylon cause pressure loads in both axial, lateral and vertical direction. These contributions can increase under high angles of attack or yaw. A specific point of high aerodynamic loading is the inlet cowling, Figure 2.11. This part of the nacelle straightens and slows down the airflow before it enters the fan. In doing so it changes the momentum of the flow, leading to high aerodynamic loading. Note that this part is generally relatively far away from the pylon attachment point, meaning large moments can be generated. A second aspect is fan loads generated when the incoming air is set under an angle of attack. This can lead to blade loading varying when the fan blades rotates, leading to a non-uniform disk loading, which can lead to moments on the rotational axis that are transferred by the engine casing to the engine attachment points.

INERTIA LOADS

Inertia loads, as the name implies, are loads that are caused by the inertia of the engine under acceleration. The loads generated can be especially high, as noted by not only Niu and in discussions with engine integration design engineers at Safran Aircraft Engines. Inertia loads are typically generated whenever the aircraft

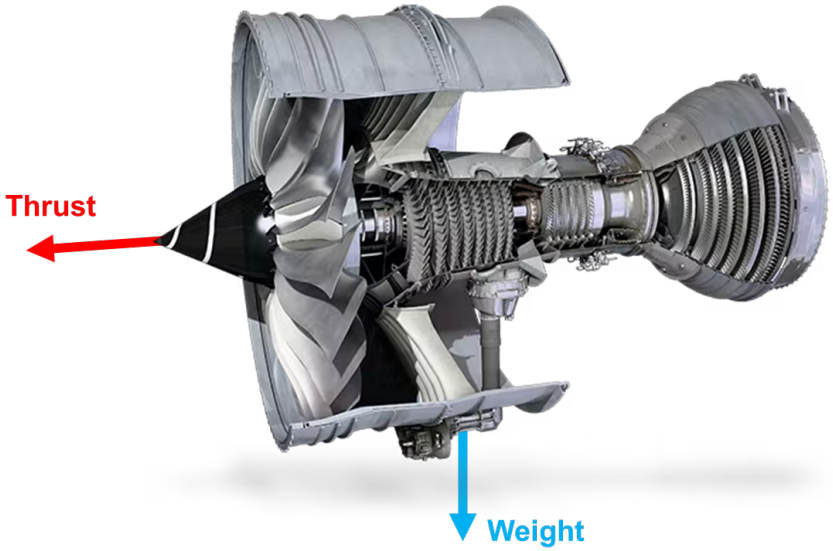


Figure 2.10: Thrust and weight loads on a Trent 1000 engine (adapted from [12]).

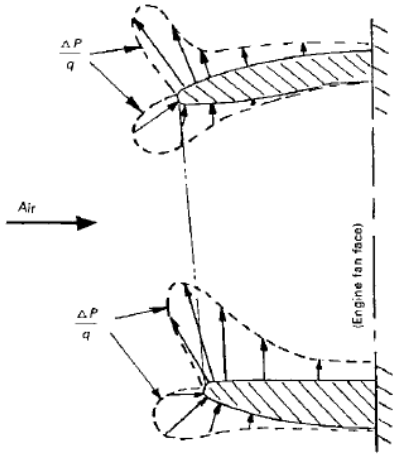


Figure 2.11: Pressure distributions on the inlet cowling can lead to heavy loads [8].

performs a maneuver, for example a yawing or pitching maneuver, or when the aircraft is subject to air loads like gusts or turbulence. Inertia loads are closely coupled to gyroscopic loads in engines, since these are also caused by the same accelerations that cause inertia loads.

To calculate the load factors the engine experiences during maneuvers, Niu [8] proposes the following relations, to be applied at the engine cg (see Figure 2.8a for definitions of the positive accelerations. Note that the relations proposed by Niu are defined using a left-handed coordinate system such that the positive roll-, pitch- and yawing angles equal those commonly used in flight dynamics analyses):

$$n_{x_N} = +n_{x_A} - \frac{z\ddot{\theta}}{g} + \frac{y\ddot{\psi}}{g} + \frac{x}{g}\left(\dot{\theta}^2 + \dot{\psi}^2\right) \tag{2.2}$$

$$n_{y_N} = +n_{y_A} - \frac{x\ddot{\psi}}{g} + \frac{z\ddot{\phi}}{g} + \frac{y}{g}\left(\dot{\psi}^2 + \dot{\phi}^2\right) \tag{2.3}$$

$$n_{x_N} = +n_{Z_A} - \frac{y\ddot{\phi}}{g} + \frac{x\ddot{\theta}}{g} + \frac{z}{g}\left(\dot{\phi}^2 + \dot{\theta}^2\right) \tag{2.4}$$

GYROSCOPIC LOADS

These loads occur when the rotating parts of the engine are rotated about an axis not parallel to the rotational axis of the spinning parts in the engine. With the engine placed longitudinally on the aircraft as usual, these axes are generally the pitch- and yaw axis of the aircraft (y- and z-axis). For the gyroscopic moments generated, Niu suggests the following equations *for a counterclockwise rotating engine*:

$$M_Z = -I_{p,e}\,\omega\,\dot{\theta} \tag{2.5}$$

$$M_Y = I_{p,e}\,\omega\,\dot{\psi} \tag{2.6}$$

Again, the coordinate system defined in Figure 2.8a is used in these equations. $I_{p,e}$ is the mass moment of inertia of the engine's rotating components about the spinning axis in $\mathrm{kg\,m^2}$, ω is the angular velocity of the engine in $\mathrm{rad\,s^{-1}}$ [51].

GUST LOADS

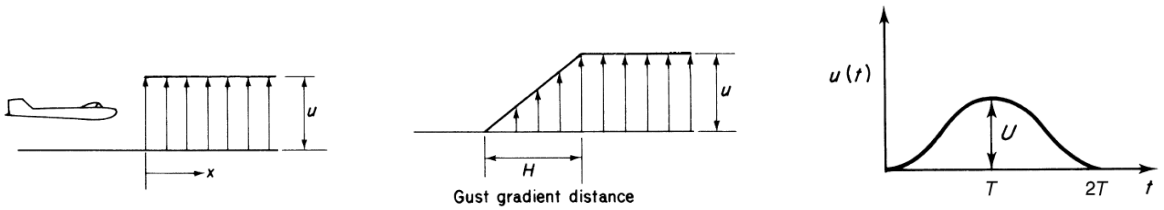
Gust loads are loads stemming from turbulence in the air surrounding the airplane. Sudden winds can increase the incidence angles of the wings and structure, leading to peak lift loads pulling on the structure. Modeling gusts can either be done statically, or by considering gusts as a dynamic load profile. Static gust modeling is usually done in one of three ways. As a *discrete* signal, where the airplane sees a sudden increase in vertical air speed increasing the angle of attack instantly (Figure 2.12a). This approach is required for the pylon per CS 25.341(c) (see Appendix A.2). The resulting load factor as seen by the aircraft wing can be calculated using equation Equation 2.7, with W_{ac}/S_{wing} the wing loading and u the gust velocity. Note that the obtained gust load factor should be multiplied using Equation 2.4 to find the load factor that the engine experiences.

$$n_{\text{gust, discrete}} = 1 \pm \frac{\frac{1}{2}\rho_0 V_{EAS}(\partial C_L/\partial\alpha)u_{EAS}}{W_{ac}/S_{wing}} \tag{2.7}$$

The discrete gust method is considered somewhat dated, as it does not take into account the build-up behavior of the circulation around the lifting structure in response to a changing angle of attack (Wagner effect) and therefore tends to over-estimate the upward acceleration of the aircraft and therefore the gust loads [13]. Another approach is to introduce a gradual build-up of gust speed, the *graded gust* (Figure 2.12b). To convert the load factor obtained by calculating using the discrete gust method to one resulting from the graded gust methods, a *gust alleviation factor F* is usually used. Equation 2.7 then transforms into:

$$n_{\text{gust, graded}} = 1 \pm \frac{\frac{1}{2}\rho_0 V_{EAS}(\partial C_L/\partial\alpha)Fu_{EAS}}{W_{ac}/S_{wing}} \tag{2.8}$$

Finally the *1-cosine gust* can be used, where the gust profile follows a cosine wave shape (Figure 2.12c). The most advanced method of modeling gust is the *power spectral analysis* approach, which enables the designer to take into account a host of gust shapes and sizes. For gust loads on pylons of wing-mounted engines, Niu suggest applying a dynamic gust with load factor of 5.0 or 6.0, as was mentioned in the introduction text of this section.



(a) Discrete (b) Graded (c) 1-cos

Figure 2.12: Gust types [13]

2.3.3. FAN BLADE-OFF (FBO) EVENT

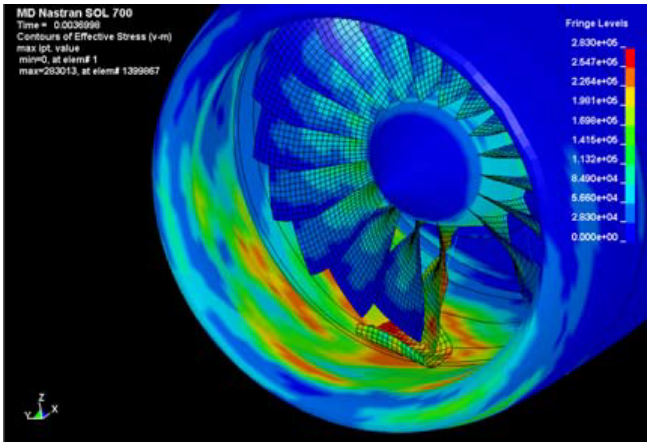
One of the most important load cases for which the engine and engine support structure must be designed is the so-called 'Fan Blade-Off Event' (FBO, also called 'Fan blade-out'). FBO is an event where one of the fan blades of the engine fan separates from the hub and is ejected into the casing of the engine, generally as a result of either 'Foreign Object Damage' (FOD) or fatigue. The FBO event is a very intense, sudden event that causes heavy non-linear impact loads on the engine casing and the pylon combined with high non-linear dynamic asymmetric thrust loads in the engine due to rotor imbalance. In fact, engine blade-off was identified as *the* primary sizing load case for both the engine casing and the engine support structure in multiple publications. For example, Grihon *et al.* [6] noted that *"FBO static load cases are the dimensioning loads in panel webs with respect to major principle stresses and in spar webs with respect to the shear stress allowables used in this study"*. Gazaix *et al.* [7] supports this claim, stating that *"fan blade out events generate very large loads on the pylon and are consequently critical sizing load cases of the structural elements"*. Mouton *et al.* [5] states that *"fan blade off cases drive the limit static loads to sustain"*. For this reason, demonstration of the ability to deal with this type of event is considered a major milestone in engine and aircraft certification, and the requirements on engines and engines support structures are laid out extensively in certification regulations and directives, such as expressed in Appendix A.2 - CS 25.362 and CS 25.367.

Considering the important place the FBO event takes in the design of engines and engine support structures, it is clear that the ability to effectively model and implement the loads generated from an FBO event in an early stage of the engine and airframe design cycles is critical. Unfortunately at the same time, due to its highly non-linear and dynamic character, this is quite difficult and requires detailed numerical models in order to accurately predict the generated loads. For that reason, it is mentioned by Bettebghor *et al.* [26] that for pylon sizing and in the initial stages of design the airframe manufacturer will generally only use *coarse models of the engine* or even just a *concentrated mass at the center of gravity of the engine*, and will assume that the nacelle-engine assembly is *rigid*. Furthermore, typically only FBO loads *transferred to the center of gravity of the engine* are used. Nevertheless, several authors have published detailed works on the detailed numerical modeling of the FBO event, especially in the past 15-20 years. Notable works describing FBO analysis or specific parts of the process are Stallone *et al.* [52], Lawrence *et al.* [53], Carney *et al.* [54], Husband [55] and Heidari *et al.* [14].

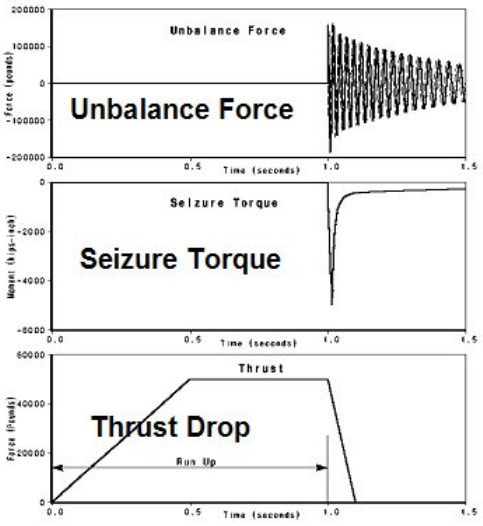
Heidari *et al.* [14] and Bettebghor *et al.* [26] describe the process of conducting a full numerical FBO analysis as would be done in industry as follows:

1. **Pre-stress step.** Executed by the engine manufacturer using a full engine FEM model using an implicit numerical integration scheme. Fan blade pre-stress loads originating from thrust, gravity and gyroscopic effects are calculated and applied onto a detailed finite-element model of the engine. The resulting state from this analysis is used as initial conditions for step 2 in this process.

2. **Fan blade-off simulation step.** Executed by the engine manufacturer using a full engine FEM model using explicit numerical integration schemes. In this step, the actual fan blade-off event is simulated numerically for a relatively short period of time (about 3-10 revolutions). Release of the blade, damage of the blade, impact of the blade onto the casing, rubbing of the released material and the remaining blade with the engine casing due to the unbalance force are all simulated. This simulation is the most heavy, using explicit schemes to take into account contact, plastic deformation, impact etc. The meshes used in this step are very fine, and the time steps are very small as usual in explicit analyses, which makes this part of the analysis very numerically heavy and time consuming.



(a) Close-up of stress contours during the release of a fan blade in full explicit FBO simulation.



(b) Typical loading profiles as generated during detailed explicit FBO simulation.

Figure 2.13: Results of an explicit FBO simulation (step 2) using MSC Nastran as presented by Heidari *et al.* [14].

3. **Post-impact rotordynamics analysis step.** Executed by the airframe designer using a coarse engine model using implicit numerical integration schemes. Loads calculated in step 2 are transferred to the airframe manufacturer to be used in this step, together with the fan rotor speed at impact. The total set of loads transferred from step 2 to be used in this step consist of:

   - Rotating unbalance load coming from mass of the missing fan blade, applied at the fan rotor location. This can be calculated using the equation with m the mass of the missing blade, r the height of the fan blade and ω the rpm in rad/s:

   $$F_{unblc}(t) = mr\omega^2(t) \tag{2.9}$$

   - Blade impact loads on the engine case, applied to the nodes at the location of impact.
   - Seizure torque generated due to rubbing of the remaining blades against the casing, generally applied as a tangential force at the end of the blades.
   - Thrust, applied to the thrust links.
   - Gyroscopic loads, applied at the fan rotor location.

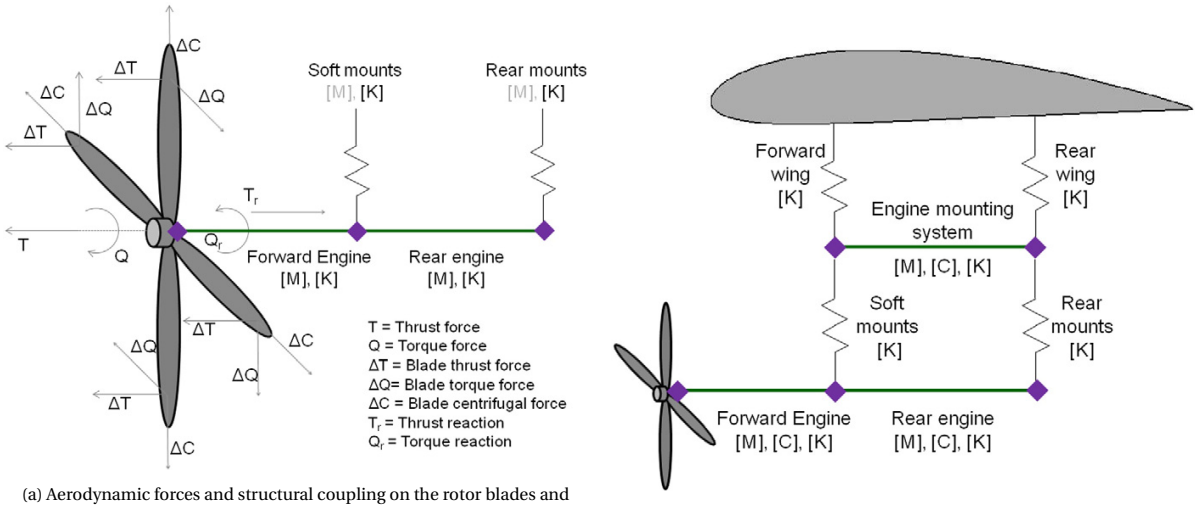
   The analysis generally will consist of a transient rotordynamics analysis that simulates the behavior of the rotor in the phase starting right after blade impact up until rotor windmilling. This process is governed by the following equation of motion:

   $$[M]\ddot{\mathbf{q}}(t) + \big([C]+[G]\big)\dot{\mathbf{q}}(t) + \big([K]+[N]\big)\mathbf{q}(t) = \mathbf{F}(t) \tag{2.10}$$

   with [G] the gyroscopic damping matrix and [N] the centrifugal forces. Several authors have published work specifically focused on the details of this rotordynamics analysis, noticeably Heidari *et al.* [56], Sinha and Dorbala [57], and Sinha [58].

As indicated, step 1 and 2 are executed by the engine manufacturer and is mainly used to secure the structural integrity of the engine itself in case of FBO. Step 3 however is generally done at the airframe manufacturer (which also designs the engine support structure) and is done to find the FBO loads to be used for pylon sizing.

The analysis and modeling of a supporting structure for a turboprop engine in the event of blade loss was investigated at the Spanish National Institute of Aerospace Techniques (INTA) in cooperation with Airbus Spain, and published in Armendáriz *et al.* [59][15][60]. In this research, the Engine Mounting System (EMS)



(a) Aerodynamic forces and structural coupling on the rotor blades and engine mounting

(b) Structural spring-element model of engine-wing attachment

Figure 2.14: Structural model and forces applied in the model of Armendáriz *et al.* [15].

was modeled using an FEM model in MSC.DYTRAN and MSC.PATRAN. The attachment point between engine and pylon and pylon and wing are modeled as springs, which represent the elastomeric coupling between the structures at these points. The engine is modeled using beam and mass elements. An schematic overview of the model including applied forces is shown in Figure 2.14. The centripetal forces on the blade hub due to the fan blade loss are calculated using the following equations, with θ the 'blade release angle':

$$F_y = -F_{centrip}\, cos(\omega t) + \theta \qquad (2.11)$$

$$F_z = -F_{centrip}\, sin(\omega t) + \theta \qquad (2.12)$$

$$M_y = M_{oop}\, \sin\omega t + \theta \qquad (2.13)$$

$$M_z = -M_{oop}\, \cos\omega t + \theta \qquad (2.14)$$

The model is subjected to a range of load cases representing FBO events for engines operating at different rpm, with different blade loss angles and blade heights, and the behavior is recorded. It was found that multiple factors influence the severity of the forces and the behavior of the structure. Among others, blade release angle, rpm, blade size, mounting stiffness and structural damping, and flight condition were found to significantly influence the severeness of the structural failure.

2.3.4. FATIGUE LOADS

Fatigue is defined as *"the progressive deterioration of the strength of a material or structural component during service, such that failure can occur at much lower stress levels than the ultimate stress level"* [13]. Fatigue can have several causes, for example cyclic loading, corrosion, thermal expansion and contraction or vibration. But no matter the cause, the degradation principle is the same; small cracks are introduced into the structure, that grow in time and under fluctuating load cycles, leading to large cracks that propagate through the structure and finally lead to structural failure. The engine mounts and pylons are designed with a *safe life* design philosophy in mind, meaning internal stresses and loads should be lower than the allowable fatigue stresses for a pre-specified number of load cycles (such that the service life of the pylon is covered) at all times. This can be reached by applying a material with an *endurance limit* (the stress the material can withstand an infinite amount of times, see Figure 2.15) higher than the required fatigue strength or by dimensioning the structure such that the stresses maintain below the specified fatigue limits, as was done in for example Grihon *et al.* [6]. Fatigue is an important factor in pylon design, as it forms a critical load case for several structural members of the pylon. Results obtained by Grihon *et al.* [6] indicate that fatigue flight load cases are the dimensioning load cases for the flanges with respect to longitudinal stresses in his structural pylon analysis. The same is true in Gazaix *et al.* [7] which mentions fatigue loads as critical load cases mainly driving the sizing properties of the corners of the pylons.

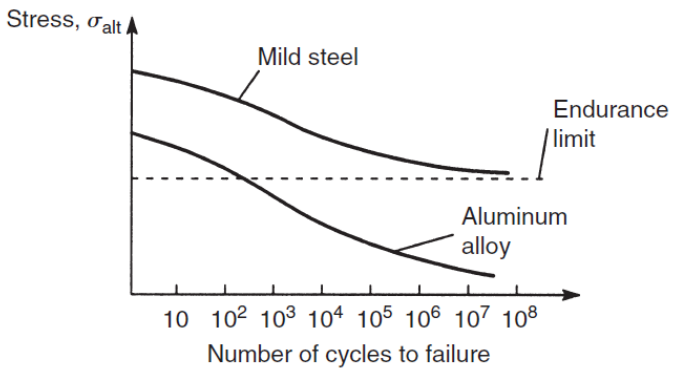


Figure 2.15: Typical S-n curve showing endurance limit for mild steel [13].

2.4. WEIGHT ESTIMATION METHODOLOGIES AND DEVELOPMENTS

In the previous sections, the geometrical design of the engine-airframe integration was investigated. In this chapter, attention is shifted to estimating the weight of the pylon-nacelle assembly. First we look at how weight estimation works in the aeronautical industry. Then we investigate what weight estimations currently exist for pylons and nacelles. And finally, we will look at some recent development in the weight estimation field.

2.4.1. AN INTRODUCTION TO WEIGHT ESTIMATION METHODS

Proper weight estimation is vital for the successful development of new aircraft. Due to its importance, weight estimation methods are described in many well-known airplane design textbooks, notably Roskam [61], Torenbeek [17], Raymer [4], and Jenkinson *et al.* [16]. Aircraft weight is influenced by many of the most important aircraft design characteristics (see Figure 2.16), and is often a key factor in achieving the aircraft's required performance as stated in the mission statement. Finally, the system of weight and balance of the different aircraft components is delicate and easily disturbed. A weight increase in one component often leads to a weight increases in another; which in turn might bounce back to affect the first. This way, a 'snowball effect' is created that is very hard to mitigate once it occurs. This dynamic makes good weight prediction extremely important, and many projects have failed or suffered severe performance penalties due to mistakes in weight prediction or unexpected weight growth, leading to high financial losses.

Speaking in general terms, the type and accuracy of the weight estimation methods employed by a the designer is closely correlated to the design stage of the aircraft [39][17]. Overall, four 'traditional' classes of weight estimation methods can be distinguished [62]:

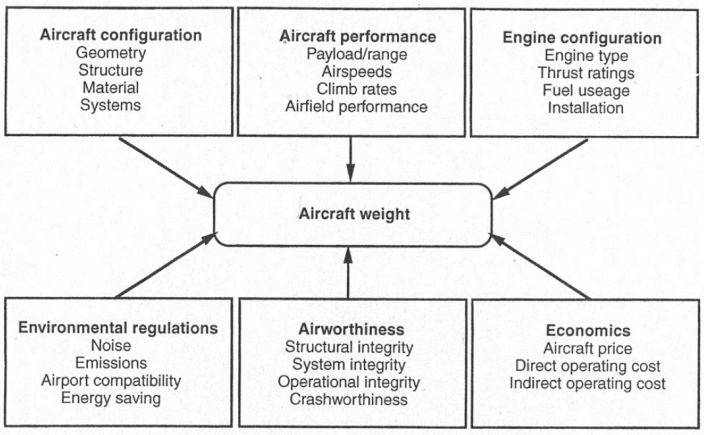


Figure 2.16: The relation between weight and other factors influencing aircraft design [16].

- **Class-I methods** (also known as *fraction methods*) are the simplest weight estimation methods available to the designer. They comprise first estimates for the top level weight characteristics of the aircraft like maximum take-off weight (MTOW), operational empty weight (OEW) and fuel weight, but also for the so-called 'major component fractions' like fuselage weight, propulsion system weight and avionic systems weight. These methods are mainly used in the early stages of design, during the *preconfiguration development* and *configuration development* stages. In this stage of design, the only source of information available to the designers is the Mission Statement or a list of 'Top-level Requirements' (TLRs) indicating key design characteristics (i.e. range, payload, climb requirements, etc.). By comparing the aircraft with other aircraft with similar mission statements, so called 'guesstimates' for MTOW, OEW and fuel weight are identified. Using this information, first estimates for typical weights for major component groups can be found by applying prescribed weight fractions [61].

- **Class-II methods** employ statistical information and semi-empirical equations to provide more detailed an more accurate estimations for aircraft on a component level. These methods are mainly used in *initial baseline- and preliminary design*. Though more detailed, the level of detail is still at a component-level, not at a parts level. So while the weight of an engine can be generally predicted, the weight of individual fasteners or spars can not. An important component of Class-II methods is their use of airplane specific values, like wing design parameters (sweep angle Λ, wing area A, etc.), design cruise or dive speed V_C or V_D and other factors. Furthermore, maximum load factor n_{max} or ultimate load factor n_{ult} often play an important role in the equations. Nonetheless, problem of 'old' weight data upon which the semi-empirical equations are based also plays a role in these methods, at least for the methods found in open literature. Finally, the equations are iterative by nature, as they often build on first estimates from Class-I methods. The results of the Class-II methods are then used to build a new aircraft weight definition, which is then iterated until the calculated GW and newly obtained GW are in a satisfactory range.

- **Class-III methods** are methods that employ physics-based analytical and numerical analysis methods like Finite Element Method codes (FEM) and Computational Fluid Dynamics (CFD) to make detailed analysis of aircraft on a component level. The methods are mainly employed during the (advanced) *preliminary design* phase. This means main components like wing boxes and fuselage structures are analyzed, but without including auxiliary parts like firewall material, fasteners, electric wiring, hydraulic systems etc. Full load set are required, as well as a component level CAD model. The topology of the aircraft is usually set in or before this stage of design, leading to decreased flexibility. Furthermore the cost in terms of money and computing time is usually several orders of magnitude higher than those of Class-I and -II methods, making these methods unsuitable for the highly flexible high-level design approach used in the early phases of conceptual design.

- Finally **Class-IV methods** are the most detailed weight calculation methods, employed in the *detailed and (pre-(production) design phase*. The airplane is analyzed on a detailed component and even part-based level, taking into account all sub-assemblies, auxiliary systems like hydraulics, lubrication systems, fasteners, seals and so on. Weights are calculated in expensive CAD or CAE models using exact geometries, material and employing manufacturing-grade finishes, or in the later design stages measured from actual manufacturing grade parts. These methods are clearly the most accurate, but also require a full understanding of the systems and can therefore only be employed when the final design is already decided on, leaving limited freedom in case changes need to be applied. On top of this, they usually require heavy (computing) resources, leading to high costs and long computational times. This means that these tools are usually only used in the final stages of design, when the level of accuracy and the high costs associated with them is worth the added qualitative value to the final production design. Or as a means to replace even more expensive certification testing campaigns, if this is allowed.

2.4.2. PRELIMINARY WEIGHT ESTIMATION METHODS FOR PYLONS AND NACELLES

The following text features a summary of known Class-I and -II weight estimation methods for pylons, nacelles and related secondary components. Unfortunately, few detailed weight estimation methods for pylons and nacelles are reported in literature. This mainly has to do with the relatively small engines used on aircraft in the '60s and '70s, when most of the reported methods where conceived. With nacelle weight only constituting about 1.2-2.2% of MTOW for most older aircraft, many authors considered a general 'propulsion group' weight sufficiently detailed for the early stages of design. But of course as the weight and size of the engine

grows, so do the weight and size of the nacelle and the pylon. So for the purpose of this thesis, the high-level group weights previously considered are no longer sufficient.

CLASS-I WEIGHT ESTIMATION METHODS FOR PYLON-NACELLE ASSEMBLIES

Though several aircraft design books discuss (initial) weight estimation, one of the most elaborate and well known texts on Class-I weight estimation methods is without a doubt found in the books of prof. J. Roskam [39][61][63]. In his book series, especially in [63], Roskam shows a number of Class-I weight estimations for different classes of aircraft ranging from twin-engine propeller aircraft to large commercial transport turbojet aircraft. All methods are based on an averaging of weight data from existing aircraft in that class, leading to weight fractions that are than multiplied with the obtained 'Flight Design Gross Weight' GW. This is defined as "the weight at which the airplane can sustain its design ultimate load factor n_{ult}, and is equal to the MTOW for most civil aircraft. The method divides the weight of the aircraft in three main groups;

1. Structure weight, W_{struct}, subdivided into wing weight, empennage weight, fuselage weight, nacelles weight and landing gear weight.
2. Powerplant weight, W_{pwr}, subdivided into engine weight, air induction weight, propeller weight (if applicable), fuel systems weight and propulsion systems weight.
3. Fixed equipment weight, W_{feq}, subdivided into flight control systems weight, hydraulic systems weight, electrical systems weight, instrumentation and avionics weight, oxygen system weight, and all other equipment weight items.

Airplane Empty Weight is than defined as a summation of these components:

$$W_E = W_{struct} + W_{pwr} + W_{feq} \tag{2.15}$$

Then, based on a weight data averaging for which tables of typical airplane weights are used, the weight fraction are calculated and multiplied by the earlier defined GW. This is clearly a very simple and quick method that gives indicative numbers. But the method is not very accurate and relies heavily on the data set used, which in case of Roskam is particularly old. Furthermore, all airplane data used in the book refers to largely aluminum aircraft. If the aircraft employs other materials, like composites, the weight factors have to be adjusted accordingly. Similar methods are found in other textbooks, for example Corke [64] and Jenkinson *et al.* [16].

CLASS-II WEIGHT ESTIMATION METHODS FOR PYLON-NACELLE ASSEMBLIES

Though still relatively coarse, the equations used in Class-II methods can already provide quite reliable weight estimates for large component groups. Most of the equations are found from statistical regression analyses and interpolation of existing aircraft data. When using Class-II methods, the weight of aircraft is distributed over at least the same amount of main systems as is used in the preceding Class-I analysis, but often the analysis is somewhat more specific. Nevertheless, the pylon-nacelle assembly is still mostly interpreted as a single component, and more detailed weight estimation equations just for pylon weight are hard to come by. An overview of Class-II weight estimation methods for pylon-nacelle assemblies can be found in Appendix A.3.

As for Class-I methods, Roskam [63] also gives a pretty elaborate overview of Class-II weight estimation methods from different sources, notably originating from sources like Schmitt *et al.* [65] ('the Cessna method'), Nicolai [66] ('the USAF method' or 'the General Dynamics method') and Torenbeek [17] ('the Torenbeek method'). In the book, equations are given for different types of aircraft, but for our purposes only equations which can be used for weight estimation of Commercial Transport Aircraft are considered relevant. The weight of the pylon and nacelle assembly is captured in one umbrella term, 'Nacelle Weight' W_n. This terms covers the structural weight of the engine external ducts, cowling and pylon(s). Roskam mentions two equations for Class-II weight estimation of the Nacelle group, one from Schmitt *et al.* [65] (Equation A.1) and another one from Torenbeek [17] (Equation A.2). Torenbeek in his book, from which the equation originally comes, further adds some more detailed estimates for subsystems, shown in Figure 2.17.

Jenkinson *et al.* [16] bases his weight estimation method on a quadratic curve fit of a selected set of transport aircraft, which leads to two equations; one for aircraft with a total installed sea level take-off thrust of

WEIGHT CONTRIBUTION	METHOD
ENGINE MOUNTS AND VIBRATION ABSORBERS	5% of engine plus propeller installation weight
NACELLE STRUCTURE, PYLONS AND STRUTS, ENGINE COWLINGS, FLAPS AND BAFFLES	$.03\sqrt{V_D}\,S_{wet}^{1.3}$ (lb); V_D~kts EAS; S_{wet}~sq.ft $.405\sqrt{V_D}\,S_{wet}^{1.3}$ (kg); V_D~m/s EAS; S_{wet}~m^2 S_{wet} ~ total area per nacelle wetted by the cold airflow, both internally and externally*
GAS GENERATOR COWLING AND PLUG	3 lb/sq.ft (14.6 kg/m^2) of wetted area
NOISE SUPPRESSION MATERIAL (EXTRA WEIGHT)	.35 lb/sq.ft (1.71 kg/m^2) ~ nacelle walls 1.75 lb/sq.ft (8.53 kg/m^2) ~ splitter plates
FIREWALLS AND SHROUDS FOR FIRE PROTECTION	1.13 lb/sq.ft (5.51 kg/m^2)

*for straight jet engines the external nacelle area plus the inlet duct area

Figure 2.17: Data for estimating nacelle group weight [17].

under 600 kN (Equation A.3), and one for for aircraft with a total installed sea level take-off thrust of *over 600 kN* (Equation A.4).

Another very famous aircraft design book is the book of Raymer *et al.* [67]. In his book, dr. Raymer lists an abundance of weight estimation methods, most of which are taken from Staton [68] and Jackson and Christian [69]. The nacelle group weight equation (Equation A.5) includes the nacelle, air induction and the pylon and is the 'recommended' formula as per the original source, though the report by Staton also lists three other equations which can also be used (Equation A.6, Equation A.7 and Equation A.8).

A more recent report containing an overview of the Weight Estimation methods employed in the NASA's aircraft design code 'FLOPS' (Flight Optimization System) is given by Wells *et al.* [70] (Equation A.9). FLOPS employs weight equations synthesized by curve fits on 17 transport aircraft and 25 fighter aircraft. To obtain the equations, curve-fitting optimization using non-linear programming techniques was used. This leads to statistics-based equations that allow reasonable predictions for new aircraft employing off-database designs. Furthermore, this equation is not only applicable to under the wing nacelles, but can also be used for rear mounted nacelles.

Finally, Liu [71] of DAR corporation mentions the only pylon and nacelle specific Class-II weight equations found. Equation A.10 and Equation A.11 describe the weight of an under the wing pylon and the weight of a high BPR turbofan nacelle, respectively. In their methods, they specify different weight equations for single engine and 'Siamese' engine setups, where two engines share a nacelle and pylon. The equations reported in Appendix A.3 cover the standard, single engine per nacelle configuration.

2.4.3. DEVELOPMENTS IN WEIGHT ESTIMATION: PHYSICS-BASED APPROACHES

In the previous text, the 'classical' Class-I, -II, -III and IV division of weight estimation methods was explained, and some important methods found in literature for conceptual aircraft design were summarized. But as explained, the equations listed are either of an empirical/statistical nature (meaning they depend on the availability of data) and quick, or based on physical (numerical) models but expensive. In this section, we will look at an intermediate solution based on elementary physics models, which provides a quick methods of analysis, but also an acceptable accuracy.

BETWEEN CLASS-II AND CLASS-III METHODS: CLASS-2.5 METHODS

The Class-I and -II weight estimation equations provided in subsection 2.4.2 provide a reasonable starting point for preliminary design weight prediction of the nacelle group of a 'standard' UWN tube-and-wing type aircraft employing 'current-day' engines. But they pose significant limits on further developments in the fu-

ture. As explained, the databases used to generate the equations is often relatively old (companies do not share detailed weight data anymore as they would do in the 70s and 80s). Furthermore, they are usually based on the 'standard' aircraft types as described earlier, like the tube-and-wing type aircraft with low BPR turbofan or turbojet engines which as we saw has been the standard topology for the past 60 years. When we would like to analyze alternative designs, this means the obtained results might deviate from what the actual, real world result would be. As the societal challenges ahead of us require radical technological developments as explained in section 1.1, these shortcomings undermine our capability to deal with these challenges in the near future. This is especially relevant with the introduction of new design methodologies in mind. As we saw in subsection 2.1.2, these methods are becoming increasingly important in the design and analysis of new aircraft. They allow the designer to quickly analyze the effects of for example varying aircraft topologies, employing different engines or implementing new technologies. An important condition for them to be successful is a fast and accurate analysis module, which is able to quickly analyze a specified design. Clearly, empirical relations such as the ones used in traditional Class-I and -II aircraft analysis tools do not provide the required flexibility to provide a reliable basis for these types of design methods.

A solution would be to employ higher-fidelity numerical analysis tools (Class-III methods) like full-scale CFD- or FEM models, but these come with problems like high costs and high computational times as explained. Furthermore, this would significantly limit the design freedom of the designer. After all, the high costs and running times of high-fidelity analysis tools constrain the number of different design options that can be analyzed without leading to exploding financial and time-related costs. Furthermore, high-fidelity methods are not well suited for optimization, which requires fast numerical analysis modules. As such they are not very suitable for conceptual design.

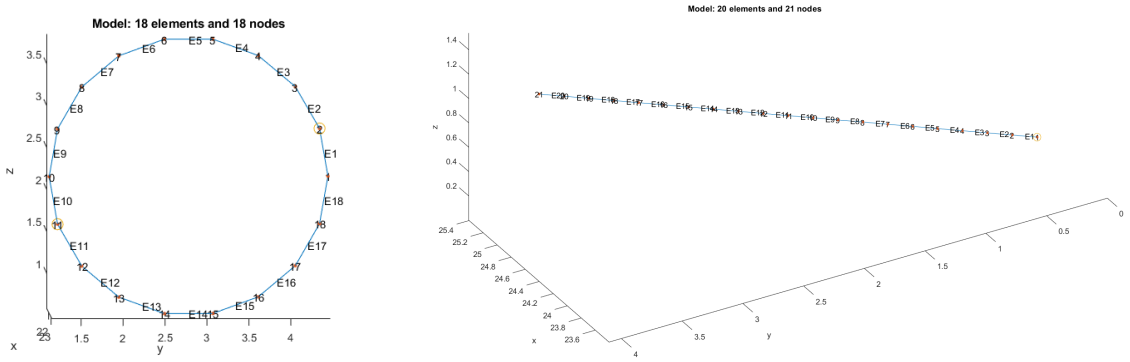
New methods that are more first-principle / physical-model based are required to bridge this gap; a fifth class of weight estimation methods called **'Class-2.5 methods'** has the potential to provide a solution. Class-2.5 methods employ elementary physics-based but low fidelity analytical models based on first-principles like elementary beam theory, augmented by the semi-empirical equations employed in Class-II methods. They form a sort of intermediate form between Class-II and Class-III methods; being physics-based, they are more accurate than 'standard' Class-II methods and thus less dependent on empirical analysis of weight data from existing aircraft. At the same time, they are not as computationally intensive as Class-III methods with their complex CAE models. For that reason, they are very useful in (multidisciplinary design) optimization problems, providing a good balance between speed and accuracy.

EMWET BY ELHAM *et al.* [72][73]

A great example of a successful Class 2.5 method developed in recent years is the wing weight estimation code EMWET, designed by Elham, La Rocca and Van Tooren [72][73]. This method employs elementary wing box sizing techniques to compute the structure required to resists the applied loads, complemented by empirical methods to estimate the weight of second-order structures and 'non-optimum weights', like those of bolts and joints, cutouts, etc. In the method, beam theory is used to size the wing box structure. The wing box is represented by a simplified structure of 'equivalent panels'. The wing box upper skin, stringers and spar caps are modeled by an 'upper equivalent panel', while the skin, stringers and spar caps of the lower skin are represented by a 'lower equivalent panel'. Spar webs are modeled using two 'vertical panels'. Loads are calculated using a Vortex Lattice Method (VLM) code, taking into account among others maneuver loads and gust loads. But also wing relief factors due to fuel weight and engine weight are taken into account. Based on the calculated loads, an optimal wingbox structure is calculated, onto which the weight is secondary structures like flaps, slats, etc. as well as the non-optimal weight is added. In total, as very accurate wing box weight is founds, which is shown to possess error margins in the order of 2%, extremely low for a quasi-analytical weight estimation method. Elham later expanded and transformed his EMWET tool into an even more precise tool, FEMWET (finite element based EMWET) [74][75]. In this method, the original EMWET script was used as a starting point for a more advanced analysis employing a finite beam element method, which can be used not only for weight estimation, but also for structural sizing.

STRUCTURAL SIZING AND WEIGHT ANALYSIS METHOD FOR THE DUCTED PROPELLER PROPULSIVE EMPENNAGE OF THE DUUC BY STAVREVA [18]

An example of a study where physics-based weight analysis methods were used specifically to analyze a pylon and nacelle is the M.Sc. thesis of Stavreva [18]. The main research goal of this thesis was to define and



(a) Nacelle approximation model

(b) Pylon approximation model

Figure 2.18: Simplified FEM models used in Stavreva [18].

implement a "physics-based design sensitive structural sizing method" capable of calculating the weight of the nacelles and pylons for the DUUC aircraft. To achieve this, Stavreva looked at the DUUC's ring wing nacelle and pylon separately, identified the sizing load cases, and then applied these onto a simplified FEM model consisting of beam elements. The ring wing is modeled using a circular ring of beams and nodes (see Figure 2.18a), each with their respective cross section and moment of inertia (found by approximating the airfoil by a trapezoid). For the pylon, a simple line model of beam elements is used (Figure 2.18b). Static loads are applied to the nodes of each model, after which the internal bending, axial- and shear stresses and displacements are calculated using the equilibrium equations $\mathbf{F} = [k]\,\mathbf{u}$ where $\mathbf{F}$ are the forces, $[k]$ the stiffness matrix and $\mathbf{u}$ the displacement vector. From this, the element thicknesses $\mathbf{t}$ required to handle the applied stresses can be found. Critical load cases considered for the nacelle are 1. *take-off with engines at full thrust*, 2. *cruise* and 3. *diving conditions* and for the pylon 1. *take-off with engine at full thrust*, 2. *diving conditions* and 3. *crash landing*. The aerodynamic loads are found from a VLM method using AVL software. Fan-blade off is not taken into account in the initial loads calculation as its determination was deemed too complex for an analysis in the conceptual design stage. Instead, a secondary analysis was performed in Abaqus making use of the obtained structure. The full analysis was performed for a fully aluminum structure, a fully composite fiber structure and one where the duct was made of composite fiber and the pylon of aluminum. In verifying the method and using is to analyze the DUUC's structure, the methods was found to provide a decrease in estimated nacelle weight of almost half that of the weight estimated using the empirical relations. For the pylon, the weight estimated by the method was higher than the weight estimated by the empirical relations, which was contributed to the higher thrust loading caused by the engines thrust. Unfortunately the methods was deemed not suitable for conceptual structural pylon design as it is not able to truly handle dynamic FBO loads, which play a major role in pylon design.

2.5. KNOWLEDGE-BASED ENGINEERING AS A METHOD FOR DESIGN AND ANALYSIS

In subsection 2.1.2, a brief introduction to the use of modern design methodologies like MDO into the design process of engine support structures was given. Furthermore in subsection 2.4.3, the rise of physics-based methods for weight estimation of aircraft components in the early phases of the aircraft design cycle were discussed. In this section, another method that can be applied in combination with both these concepts will be briefly discussed; 'knowledge-based engineering' (KBE). The use of KBE in conceptual design of aircraft and other mechanical systems has gained significant research interest in recent years, and will play a vital role in the development of the structural design and weight estimation method developed in this thesis.

2.5.1. KBE THEORETICAL BACKGROUND

Knowledge-based engineering (KBE) is a design strategy that uses dedicated software tools and systems to capture and formalize product- and process knowledge, and then re-apply this to systematically design and analyze new products, as described by La Rocca [21][22]. Ultimately, the goal of KBE is to decrease the number of repetitive tasks performed by engineers, and allow them to put more focus into the creative part of

design where new innovations are developed and capital is created.

More concrete, a KBE system is one that combines Knowledge Based- or Rule Based Systems (KBS) containing 'design rules' with the geometry manipulation and data handling capabilities of a Computer Aided Design (CAD) and/or Computer Aided Engineering (CAE) system, enabling the user to manipulate and analyze geometries using the rules in the Knowledge Base of the KBS. This is shown graphically in Figure 2.19.

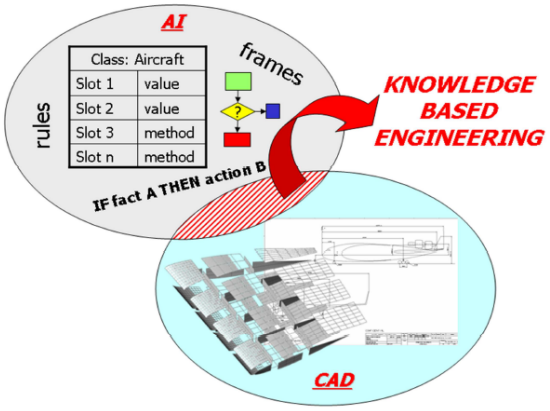


Figure 2.19: Knowledge-based Engineering systems form the combination of a Knowledge-based system (KBS) with a CAD system [21][22]

The use of KBE in the design of components in both the aerospace and automotive industry is not a recent phenomenon, and has been around for over 30 years. That being said, until recently is was mainly used in the detailed design phase to design relatively simple components. Yet, with its enormous power in doing quick and accurate, physics-based generative design, KBE has more and more cemented itself into earlier stages of the design process. This is because KBE and MDO have shown great potential to provide a solution to the well-known 'Knowledge Paradox' problem that is often experienced in design projects - the notion that most design decisions and cost commitment are set during the early stages of the design process, when knowledge on the design is low, leading to high risks. Techniques like MDO and KBE allow for advanced, relatively cheap analysis of several design options early in the design process, increasing the knowledge of the designer during the early design stages and in the process significantly increasing the freedom of the designer to investigate alternative, innovative solutions while lowering the allocated costs during that stage. The advantages of applying new design methodologies on the committed costs, knowledge and design freedom at every stage of the design cycle is clearly visible in Figure 2.20. For an extensive, in depth explanation of the history, benefits and characteristics of KBE (and MDO), the reader is referred to the PhD thesis of La Rocca [21].

2.5.2. USE OF KBE IN DESIGN AND ANALYSIS AT DELFT UNIVERSITY OF TECHNOLOGY

Although the body of research on KBE implementation in the conceptual design of aircraft or aircraft components is still relatively small, several recent examples of successful implementations of the technique do exist. Especially here at TU Delft in the Flight Performance and Propulsion group, the application of KBE in different fields of aeronautical design has been widespread. Many times in combination with MDO, but often also as a standalone design technique. This has produced some very interesting results, most noticeably the doctoral thesae of Van Der Laan [76] and La Rocca [21]. Both authors worked extensively on the application and development of KBE-based tools in the framework of the *Design and Engineering Engine* (DEE) [77], of which a schematic flow diagram is shown in Figure 2.21a. The DEE consists of a multidisciplinary collection of design and analysis tools, able to automatically interface and exchange data and information, with the purpose of supporting and accelerating MDO of complex products, through the automation of the non-creative and repetitive design activities [21]. Van Der Laan applied KBE to automate generative modeling, structural analysis and manufacturability analysis of aircraft components, and applied this to the design of aircraft moveables. La Rocca built on the work of Van der Laan to develop his *Multi-Model Generator*, a tool that uses KBE to generate 3D geometric models of the aircraft and prepares them for further analysis in the analysis tools connected to the DEE. To generate the geometric models, La Rocca uses so-called 'high-level primitives'. These are then fed into 'capability modules' in which the 3D geometry is translated into disciplinary models that are used in the specialized (numerical) analysis tools. A graphic representation of this process for the structural analysis workflow in the MMG is shown in Figure 2.21b, but silar models can be applied to aerodynamic analysis, cost analysis, manufacturability analysis and so on. The results of the separate disciplinary analyses are then fed back into the 'Converger & Evaluator' of the DEE, where an optimization algorithm is applied to analyze and weigh the results. This perfectly shows the power of new design

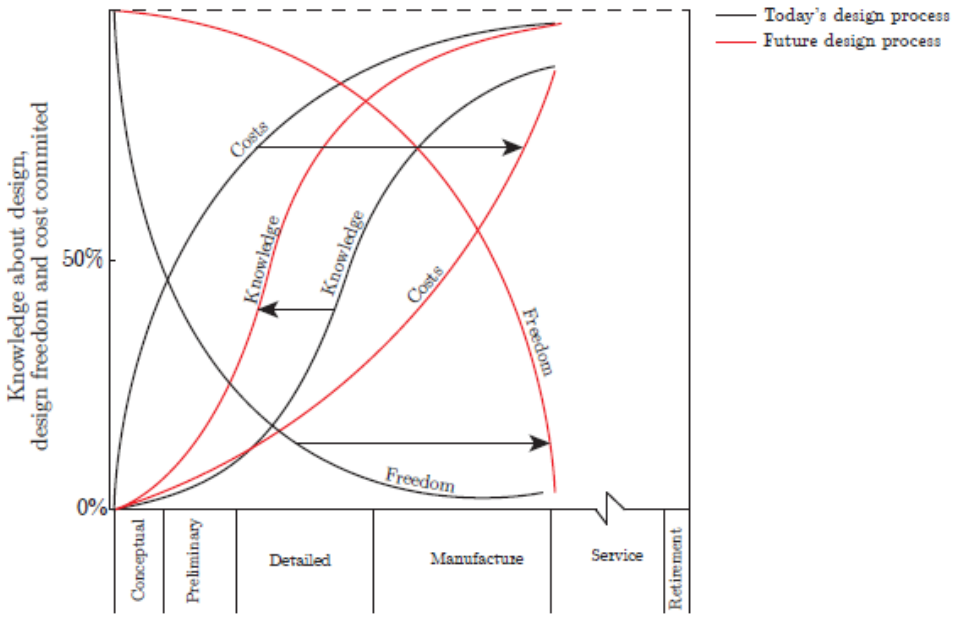
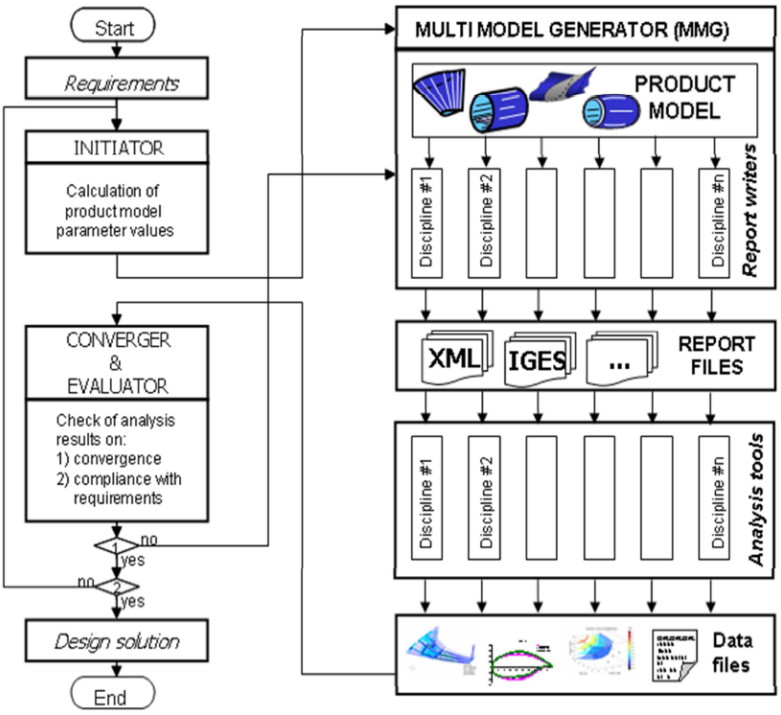


Figure 2.20: "Cost-knowledge-freedom" shift for future design methods (adapted from Mavrist *et al.* [19] by Zaccai [20]).

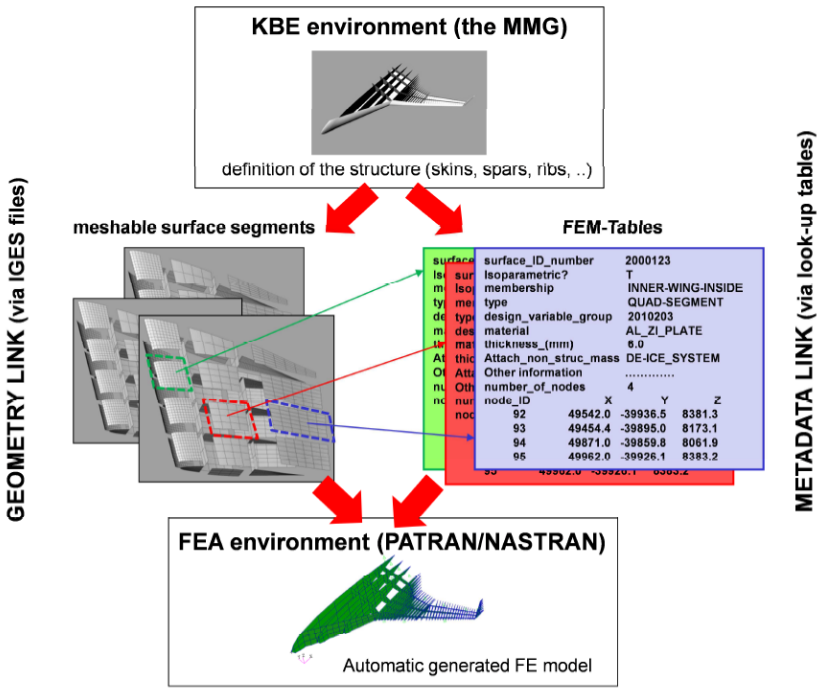
methodologies like MDO and KBE, and how they can be applied to create a powerful tools that can be used in the conceptual design of new aircraft.

Over the past 10 years, KBE was also applied successfully in student thesae. A great example of this is the Masters thesis of Zaccai [20]. In his research project, Zaccai developed a KBE-based weight- and power estimation tool for trailing edge high-lift devices, based on earlier work done by Bertels [78]. The combined work of these two students was later published in Zaccai *et al.* [79]. Using the tool, the designer is able to quickly size realistic 3D trailing edge high-lift devices and analyze the influence of design choices on the flap weight and power consumption. The tool Zaccai and Bertels developed is capable of performing advanced kinematic- and structural analysis on four of the most used flap mechanisms (dropped-hinge, four-bar linkage, straight track with drive link and hooked-track with screw jack) taking into account key design parameters like flap gap and -overlap, as well as several sizing load cases. It does this by implementing a full preliminary design sequence, including aerodynamic, structural, weight and cost analysis codes, into one KBE application.

Proesmans [23] in his Master thesis used KBE to develop a method to do conceptual design and analysis of gas turbine engines, which he named **GTpy**. For his work, Proesmans used the the Python-based ParaPy®KBE-platform as a basis. GTpy takes as input a CPACS-file [80], containing the characteristic design parameters of a given engine, which it then uses to perform a thermodynamic analysis using the gas turbine engine cycle analysis code GSP by Visser [81]. These outputs are sub-sequentially used to generate a realistic geometric 3D model of the engine as well as a possible nacelle, shown in Figure 2.22. GTpy can also create a rudinemtary pylon, as shown in Figure 2.23, but this only consists of an shell created from airfoil extrusions, without any real structure in it. The tool is coupled with the MMG-framework such that the combined geometry of airplane and engines can then be used in discipline-specific analysis codes. In the thesis of Proesmans, the model is coupled to both the aerodynamic analysis code VSAERO and the flight mechanics analysis code PHALANX in an MDAO analysis to find the optimal placement of the engine on a given aircraft, again showing how KBE and MDO can be coupled to create powerful aircraft design and analysis tools.

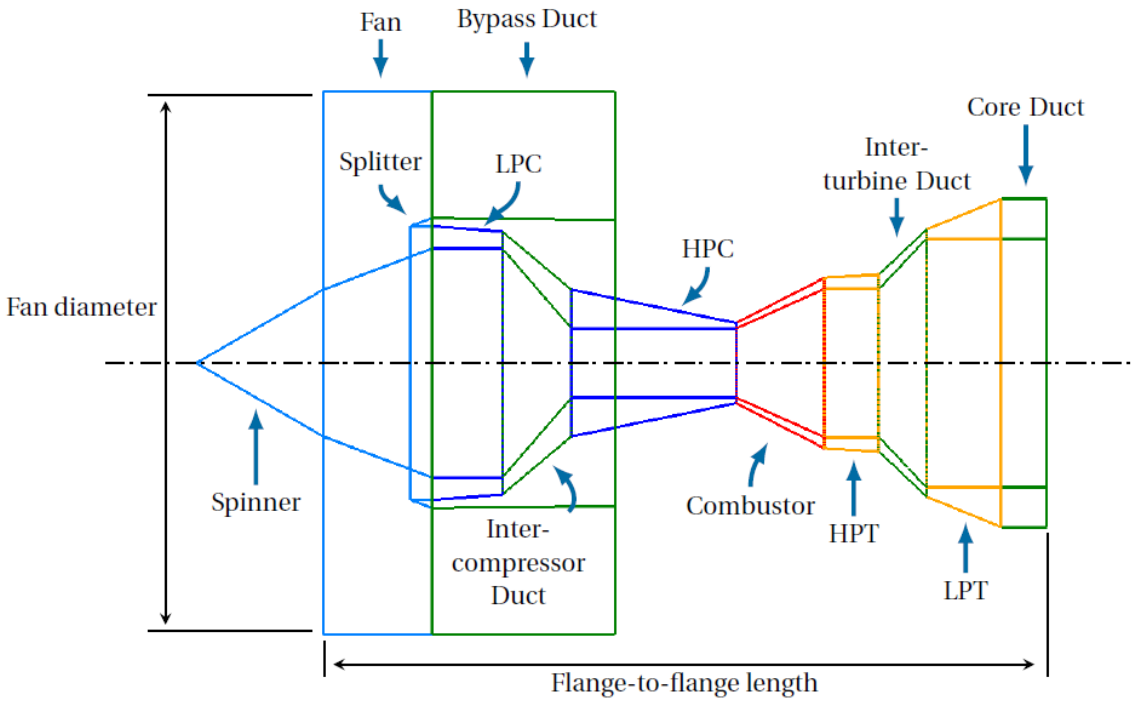


(a) Schematic process overview of the DEE.

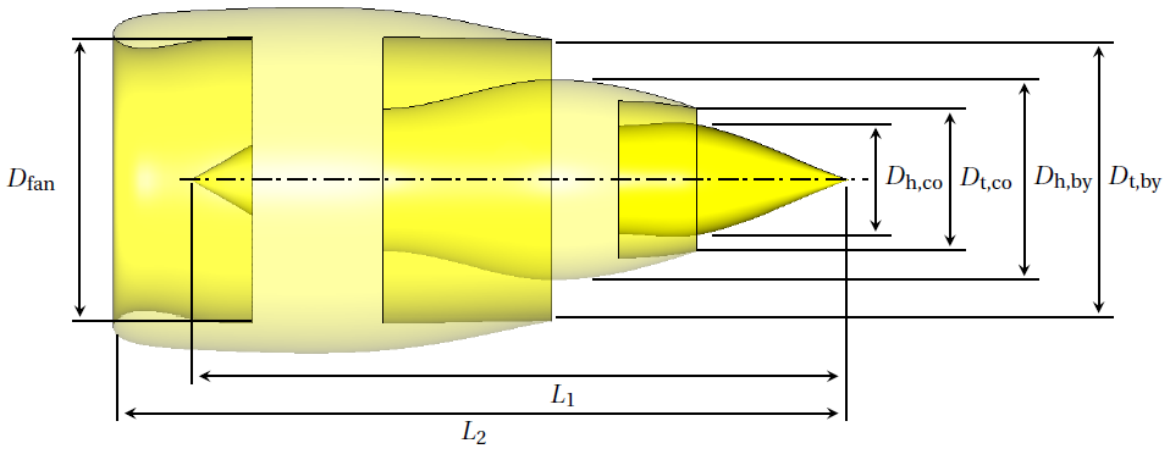


(b) Graphic depiction of geometry and metadata transfer from the MMG to the FEM environment (in this case PATRAN/NASTRAN).

Figure 2.21: Process diagrams describing the Design and Engineering Engine developed at TU Delft [21].



(a) Schematic cross-section of the engine geometry generated by GTpy



(b) Nacelle geometry (based on the GE90 engine).

Figure 2.22: Engine and nacelle geometry generated by GTpy [23].

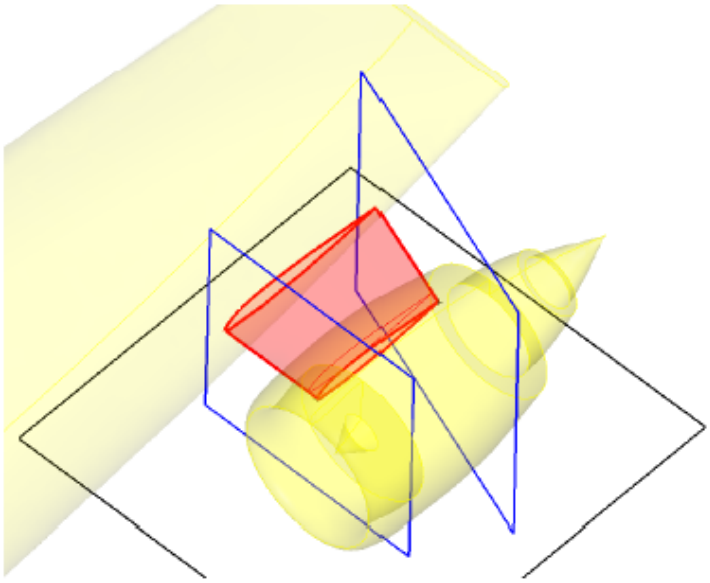


Figure 2.23: Pylon geometry creation in GTpy

2.6. LITERATURE REVIEW CONCLUSIONS

At the end of this chapter, lets review the material that was presented so far and try to draw some conclusions. In doing so, Research Question 1 will be answered fully. Considering the topics that were discussed and the information found in scientific literature, it is clear that:

- Engine-airframe integration is one of the most fundamental problems that needs to be considered in aircraft design, and requires careful investigation as early in the design process as possible due to the large impact it has on overall aircraft design parameters.
- Several types of engine-airframe integration exist, and the choice on which type of engine-airframe integration is applied is dependent on multiple factors, most important of which are the type and size of the aircraft and the engine, the aerodynamics of the engine-airplane integration including effects like noise, the manufacturability of the proposed structure and the handling and maintability qualities of both engine and airframe.
- Early engine-airframe integration was mainly governed by aerodynamic considerations, as the structures were relatively small and pylon weight was only a relatively small fraction of the total aircraft weight.
- With the increase in fan diameters and engine weight, the coupling between aerodynamics, structures, operations and other stakeholders has become more important than ever. This has led to a more multidisciplinary design process for the design of the engine-airframe integration. The rapid increase in computing power of PCs over the past 20 years has thereby led to the rise of innovative numerical design methodologies like Multidisciplinary Design Analysis and Optimization (MDAO) and Knowledge-based Engineering (KBE) in the early stages of the design process.
- Considering the structural design of pylons or struts, a clear distinction can be made between wing-mounted vs fuselage- or side-mounted engine integration. Nonetheless, similar structural concepts are employed in both architectures, most important of which is the box-beam structure.
- Loads and load cases that are relevant to engine-airframe structural design are either static, dynamic or fatigue related loads. The 'Fan-blade Off' load case plays and exceptionally important role in the design of engine-airframe integrations, as it is often found to be the sizing load case for these types of structures.
- Other important *static* loads that play a role in structural design of engine-airframe integrations are inertia loads, thrust loads, gyroscopic loads and aerodynamic loads.

- Weight estimation methods are traditionally divided in 4 typical 'categories', depending on the complexity of the method and the design stage at which they are employed.
- Conceptual and preliminary weight estimation methods are generally highly empirical in nature, and are often found by linear regression of (old) aircraft data. More advanced weight estimation methods usually make use of numerical models, but are often considered too detailed and to complex for the early stages of design.
- Preliminary weight estimation methods for pylons and nacelles are especially problematic, due to their empirical nature, which moreover relies heavily on ancient aircraft data. Moreover, detailed preliminary weight estimation methods are difficult to find, since these methods tend to consider the propulsion unit and supporting systems as one entity.
- A promising development in the world of weight estimation is the development of so-called 'Class 2.5 weight estimation methods, that combine the power of physics-based and design sensitive numerical analysis models with low-level empirical weight estimation methods, enabling the accurate analysis of a wide range of concepts earlier in the design stage than ever before.
- Unfortunately, Class 2.5 weight estimation methods for pylons or nacelles are not yet developed, and therefore not available.
- Knowledge-based Engineering (KBE) is a design strategy that has gained significant interest in the last 10 years as a method to create tools that can help to automate repetitive design tasks such that engineers can focus on creative design task and boost innovation while lowering financial risks.
- By applying KBE, possibly in combination with MDO, engineers are able to create tools that can help the engineer to quickly analyze multiple design options early in the design process based on physics-based models, enabling more design freedom, and lessening committed costs in the early stages of design.
- KBE has been implemented successfully multiple times over the past 10-15 years at Delft University of Technology to create powerful (multidisciplinary-)design and analysis tools that can help engineers make informed design decisions earlier in the design process and create better, more innovative technological solutions.
- Yet, to this date, a proper KBE tool for analysis and design of engine integration structures is not available.

3

METHODOLOGY OVERVIEW

In the previous chapter, the most important concepts surrounding engine-airframe integration, structural pylon design, weight estimation and knowledge-based engineering were explained. From this review, a set of conclusions was drawn, providing the answer to Research Question 1 in section 1.2. The information provided in the previous chapter forms a solid basis for the development of a KBE-based pylon design- and weight estimation method, that can be used to answer the remaining research questions. In this chapter, a road map is set out, explaining the steps taken and tool implemented during the main part of the project.

Looking at the concluding remarks from the literature review, it is clear that engine-airframe integration plays a vital role in the development of successful new aircraft, and that the ability to quickly review different types of engine integration architectures early in the design cycle is a major asset in this era of radical innovation. Yet the methods currently available to the designer do not allow proper physics-based conceptual analysis of the engine-airframe architecture, nor are they sophisticated enough to properly estimate the weight penalty of integrating radically new engines or significantly changing the engine integration architecture. As such, the need for a new design-sensitive and physics-based Class 2.5 structural design and weight estimation methodology for engine-airframe integration structures is clear. In this thesis, it is proposed to develop this methodology in the context of a KBE-based software application for maximum flexibility and re-usability. This way, the tool will be able to capture and apply available knowledge of engine-airframe integration design, and couple this with advanced numerical analysis tools to create feasible 3D structures. Then, the tool can be applied in the analysis of new engine types or integrations. And finally, in a later stadium, it can even be further integrated into a higher level design methodology in combination with MDO, to achieve a result that represents the best possible solution in the context of the full aircraft design problem.

3.1. DEVELOPMENT PRINCIPLES

The proposed software application should be built with a number of key development principles in mind, in order to fill the knowledge- and capability gap as identified and successfully play a role in future design projects. The most import development principles are summarized and categorized in Table 3.1. Four different categories are identified, covering the modeling of the structure and its characteristics, the geometry generated and manipulated in the tool, the structural design of the engine-airframe integration and the functionality of the application.

As is clear from the literature review, a newly conceived method should move away from the rigid, outdated empirical relations currently used in the conceptual design of engine-airframe integrations towards more flexible, physics-based models. Since we are dealing with a conceptual design method, the level of detail of the model should be such that it fits this design stage; meaning it should cover the most important characteristics of the structure in terms of geometry and (material) behavior, without being unnecessarily detailed. This way, an optimal balance is found between level of detail and effective cost of the method, both monetary as well as time-wise. Finally, the analyses performed in the method should be based on first-principles, to ensure maximum flexibility and design-sensitivity.

Table 3.1: Key development principles for a new design- and analysis methodology for engine support structures

Modeling	
1.	Proper level of detail (conceptual design phase)
2.	Physics- & first principles-based
3.	As little empirical relations as possible
4.	Design-sensitive
Geometry	
5.	Able to capture relevant geometry
6.	Flexibility to generate multiple pylon architectures
Structural design	
7.	Able to withstand all loads and load cases in accordance with relevant regulations
8.	Feasible structural concept
9.	Starting point for detailed structural design
Functionality of the tool	
10.	Easy-to-use
11.	Re-usable
12.	Flexible yet robust execution
13.	Ability to couple with other tools

One of the striking capabilities of KBE is the ability to generate and manipulate 3D-geometry, and this will definitely be applied in this project. Note though that the comment regarding the level of detail for conceptual design models applies especially to the geometry. Therefore the tool should be able to capture the essence of the geometric design of the pylon, but there is no need to include design details like rivets of secondary structures. Finally, it would be desirable to have the ability to consider multiple pylon architectures.

The generated geometry should be structurally sound and feasible. It should be able to carry all relevant structural loads and load cases. Ideally it should be such that it could be used as a starting point for further detailed structural design.

Finally the application should be functional, in the sense that it should be relatively easy-to-use but also re-usable in other applications or in conjunction with other design and analysis tools. And of course it should be accurate; preferably the solution should be within 10% of reference pylons.

3.2. METHODOLOGY INGREDIENTS

Based on the development principles as stated above and revisiting the questions posed in the research definition at the beginning of this thesis, four essential ingredients are recognized as imperative for the successful development of the tool. These ingredients are shown graphically in Figure 3.1. They will discussed here shortly. The full implementation of each of these parts will be explained in more detail in the subsequent chapters of this report.

3.2.1. PARAMETERIZATION

To be able to properly capture and manipulate the geometry of the engine support structure, a well-defined mathematical parameterization of the structure is paramount. This parameterization will be the basis of the application. Sobieszczanski-Sobieski *et al.* [48] defines a good parameterization as one that is both **effective** and robust while at the same time being **efficient**. Effectiveness in this case refers to the ability to capture all the required features of the system to be abstracted, while at the same time allowing the designer to generate solid and sound models; meaning it should not lead to errors or contradicting or infeasible geometry. Efficiency means that the number of parameters used to describe the structure should be chosen such that it has the right level of detail for the level of abstraction that is required for the design stage, but no more than that. This way computational cost are minimized, the possibility for errors is reduced and the overall efficiency of the application is increased.

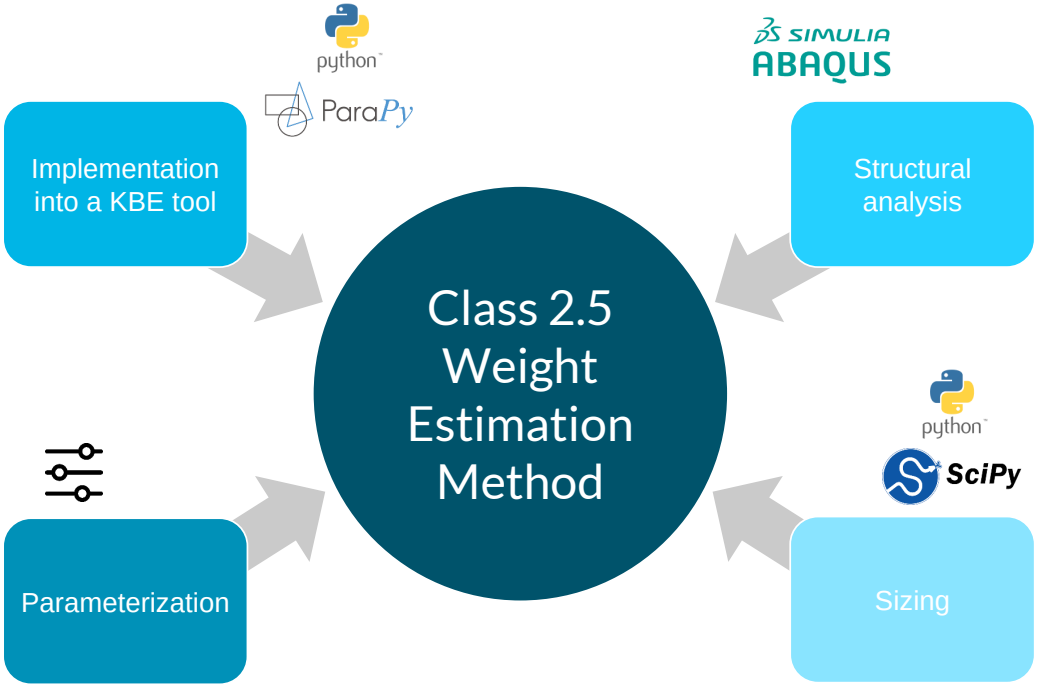


Figure 3.1: Ingredients of a KBE application for conceptual design and weight estimation of the pylon.

3.2.2. DEVELOPMENT OF A KBE-BASED SOFTWARE TOOL

Next, the defined parameterization should be implemented into a software application that can manipulate and analyze resulting geometries and communicate this with other software tools. As was explained in subsection 2.5.1, a knowledge-based engineering platform is capable of exactly that. In this project, the python-based KBE-platform ParaPy[1] is used. This platform was developed by former PhD-students of the Flight Performance and Propulsion group at TU Delft, and has seen a considerable growth in use both in academia and the industry over the past 5 years (like for example in GTpy by Proesmans [23], as was explained in subsection 2.5.2). The ParaPy platform uses the openly available Python[2] programming language as a basis, which is currently one of the most widely-used object-oriented programming languages available. Python is renowned for being easy-to-learn, extremely well-readable and widely applicable, which makes it an excellent choice to create tools that should be user-friendly and easy to integrate with other software. Moreover, Python is extremely well-equipped to deal with large quantities of data and has excellent data structures to store and manipulate data, needed for complex operations like geometry manipulation and optimization. Finally, the language is open-source and has a large and active developer community, which ensures the language's security and had led to a wealth of freely available dictionaries to work with ensuring its compatibility with many other software tools and applications.

3.2.3. STRUCTURAL ANALYSIS

While the model information is stored inside the KBE-application, dedicated disciplinary analysis tools are required to properly analyze the structural stability of the generated structure. Since the intention of this project is to provide an accurate first-estimate of the structural weight of the engine support structure, an overly simplified model as the ones used in most analytical design codes (like for example the stick model described in Stavreva [18]) is not sufficient for this project. Furthermore, from section 2.3 where the most commonly experienced loads by the engine support structures were listed, it is clear that the type of loads which the structure can experience varies. Loads can be static or dynamic and linear or non-linear. Moreover, several papers mentioned that they had found the FBO load case to be sizing for the structure, which is

[1] https://www.parapy.nl/
[2] https://www.python.org/

both dynamic and non-linear. A structural analysis method that can deal with these types of loads is therefore desired.

Considering the above, a Finite Element Method code that can analyze more complex geometry and deal with non-linearity and dynamic analysis seems to be the best option for the problem at hand. The commercially available FEM software suite Abaqus[3] by Dassault Systèmes will be employed in this project, as it is an extremely complete but at the same time user-friendly software, based on the Python programming language. Secondly, it is the software package of choice in the Structures and Materials group at the Aerospace faculty here at TU Delft.

3.2.4. STRUCTURAL SIZING

The final ingredient that is required for the method to be successful is a proper sizing and optimization algorithm. While the KBE application is able to generate a geometrical structure that is sound, and the structural analysis code can be used to evaluate the structural integrity of that structure, this doesn't necessarily mean that that proposed structure will fully meet the structural requirements which are set for the engine support structure. Or the proposed structure might be over-designed in order to make sure that it is indeed strong enough to carry the required loads for a specific engine, leading to a structure that is bulkier than the actual pylon would reasonably be. This then might lead to a weight over-estimation. An optimization algorithm that is able to size the structure will therefore be added to the code, in order to try to approach an actual structure as it might appear on an airplane as much as possible.

3.3. TOP-LEVEL PROCESS OVERVIEW

The ingredients that are described above come together into the pylon structural analysis and weight estimation application developed in this thesis project. Figure 3.2 visually describes the general top-level structure, process flow and dependencies that make up the application using an eXtended Design Structure Matrix or XDSM-style diagram.[4] The inputs to the system that will be used in this project are:

- The zero-thickness geometry parameterization of the pylon, defined using:
  - The locations of the hardpoints, as defined using the parameterization of subsection 4.2.1.
  - The outer mold shape of the pylon, as defined using the parameterization of subsection 4.2.2.
  - The orientation of the pylon, as defined using the parameterization of subsection 4.2.3.
- The thermodynamic and geometric characteristics and location of the engine.
- The material used in the pylon structure.

The inputs are provided by the designer and define the design space of the pylon structure. In a further research project, this application can be coupled with an aerodynamic solver into a full MDO design project, where the aerodynamic solver would define the pylon's aerodynamic fairing into which the structure is to be fitted and with that the constraints for the outer mold shape. This is not the focus of this thesis project though, and as such will not be implemented.

The process flow in Figure 3.2 visually represents the overall design philosophy of the application that will be applied in this thesis project. The idea behind this application is to capture a mathematical parameterization of the pylon structure into a KBE-based environment, and use structural sizing in the form of an optimization code to size the zero-thickness structure. Structural analysis is provided by coupling the KBE environment with Abaqus. In the end, the structure found by the application should be a good representation of the structure used in the final design, and offer a precise indication of the structural pylon weight that is to be expected for a specific combination of engine and engine integration choice.

[3] https://www.3ds.com/products-services/simulia/products/abaqus/

[4] XDSM was developed by Lambe and Martins [82] as a way to efficiently visualize the structure of MDO-architectures by extending the strengths of the standard Design Structure Matrix or N2-diagram with capabilities to enable clear visual representation of the type and execution order of procedures, as well as the flow of information between systems.

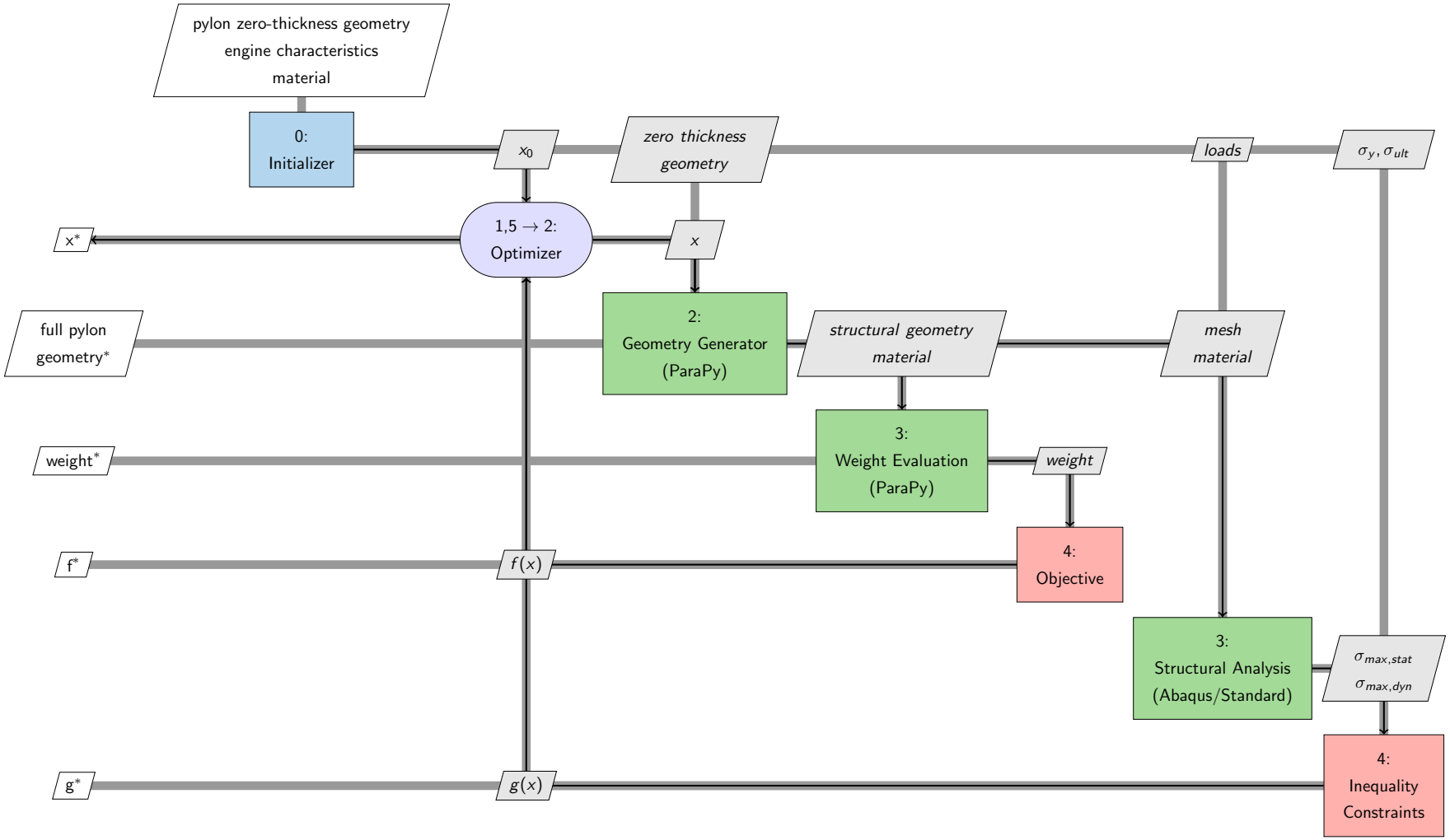


Figure 3.2: XDSM depicting the top-level structure and process flow of the pylon structural design and weight estimation application developed in this thesis project.

The general process that is executed by the application and is described in the XDSM can be explained as follows:

0. **Initiator:** The inputs are entered into an initialization code, that initializes the design process and starts the optimization. Based on the inputs, an initiator defines the starting design vector and bounds of the optimization. Furthermore, it provides the necessary loads for the structures model to define the load cases, and based on the chosen material defines the yield- and ultimate stresses that the pylon should be able to withstand.

1. **Optimizer:** The optimizer starts and controls the optimization loop that sizes the pylon structure. For each iteration, it provides a design vector that is used to define the full geometry of the structure to the geometry generator.

2. **Geometry generator:** The geometry generator combines the zero-thickness geometry and the structural parameter definition provided by the optimizer into a full 3D structural geometry for the pylon in ParaPy. From this 3D geometry definition, an instantaneous weight can be found. Furthermore, a mesh is generated and sent to the Structural Analysis sub-process for analysis.

3. **Weight- and objective function evaluation:** Using the 3D geometry and the material definition, both of which are created in the Geometry Generator, the weight of the resulting structure can be evaluated. The weight is fed into the objective function, and the result is sent back to the optimizer.

4. **Structural analysis and constraints evaluation:** In the structural analysis module, the information from the mesh, material definition and engine characteristics are combined into an Abaqus input file that defines the full Abaqus analysis including geometry, constraints, load cases and solving procedures. The maximum stresses in the material at each load case is evaluated in Abaqus and the obtained stresses are sent to the constraints evaluation module.

5. **Optimizer:** Finally the results from the objective function and constraints evaluations are sent back to the Optimizer. Based on these results, the Optimizer can decide to re-initiate the evaluation loop using a different design vector. Or, if the optimal solution is found, the process is terminated and the optimized result is presented to the user.

4

PARAMETRIC MODEL

The first step that needs to be taken in order for the structural design and weight analysis to be implemented is the definition of a solid and sound parametric model to represent the engine support structure. Therefore, a 'zero-thickness' parametric model is created that represents the rough 3D geometry of a pylon. The structural elements are represented by an additional set of parameters, adding material to the structure. The combination of the two will result in a 'full' parametric model at the end of this section.

4.1. CHARACTERISTIC FEATURES OF PYLON STRUCTURES

In order for any parameterization to be successful, it should form a good model description of the object it represents, of course tailored to the needs of the designer and the design stage in which the model is going to be used. Therefore it is important to properly assess and define the object that is to be modeled. For that reason, we will start by analyzing the key components of a pylon structure before proposing a parameterized model.

4.1.1. PYLON BOXES AND BOX-BEAM STRUCTURES

A comprehensive overview of some of the most important structural design concepts used in aircraft for podded engines was given in the literature review in subsection 2.2.1. As defined in the research scope in section 1.2, only podded engines are considered in this thesis. Engine support structures for these 'externally mounted engines' generally take the form of pylons or struts. This is especially true for wing-mounted engines, but as was explained in the text, also for fuselage-mounted engine support structures the strut type engine support has become the structure of choice since it was pioneered on the B727. A defining feature of these types of structures is their application of the so-called 'box-beam' structure. A box-beam is a structure that is commonly used throughout the aircraft, most famously as the supporting structure in the wing in the wingbox. It consists of a box-shaped beam of spars and webs stiffened at the corner points by stiffening elements and inside using stringers and ribs. Bulkheads are added to the structure at the locations where the forces are introduced into the structure for more structural strength.

In Appendix A, an overview of examples of historical pylon designs from existing aircraft is shown, showcasing the real world implementation of some of the concepts explained in subsection 2.2.1. Looking at these structures, the defining features described above are clearly visible. Especially for the wing-mounted pylons, a stiffened 'box-beam' structure is visible on all examples. The exact design of the structures varies somewhat - for example, the box-beam used on the Lockheed C-141 (Figure A.6) is built up as a real 'box', employing stiffened panels with stringers and ribs, while the 'box-beam' used on the Boeing 747 (Figure A.7-A.9) looks more like a cage-like structure with lots of holes in it. Nevertheless, the defining load-carrying elements are there. Take for example the pylon employed on the DC-10 as a reference, Figure A.4. Figure 4.1 shows the pylon in more detail, with the defining elements indicated. As shown, the pylon is made of longitudinal spars, which are stiffened at the corners using longerons. Ribs are added for more torsional stiffness, and bulkheads are placed in the beam at strategic places to introduce loads into the structure. The shear forces are taken up using webs that run between the longerons and form the outer skin of the structure.

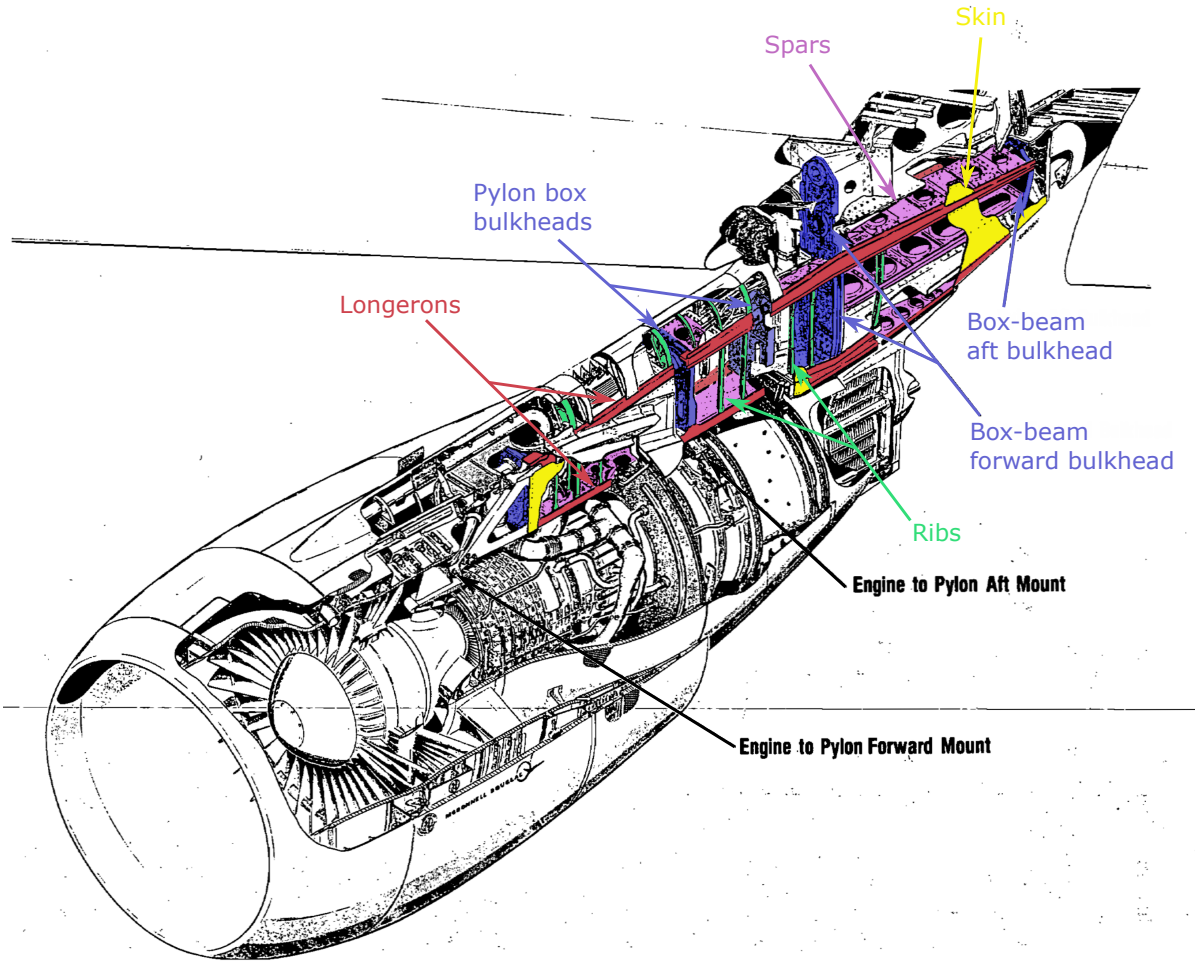


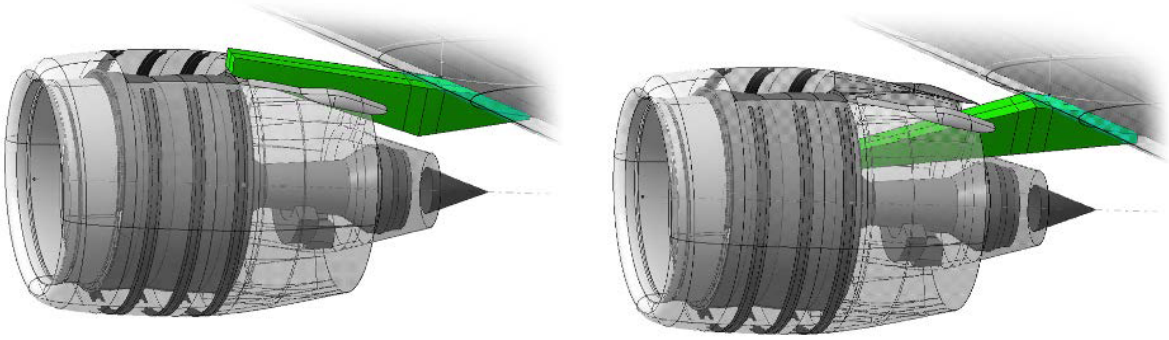
Figure 4.1: Schematic drawing of the DC-10 pylon, showing common structural components that make up an engine pylon [24].

In line with the concepts explained in subsection 2.2.1, the extent to which the box-beam structures are employed varies among the pylons shown in Appendix A.1; in structures applying a drag-strut type pylon or an upper support arm installation, a box-beam is only applied in the first section of the pylon between the engine front mount and the wing front mount - the so-called 'pylon box'. Further support is provided by struts or braces, running from the end of the box-beam towards the wing. In pylons employing the full box-beam concept, like the one on the DC-10, the box-beam is extended beneath the wing, replacing the struts by a more rigid structure.

4.1.2. ENGINE-PYLON CONNECTION

When we look at the engine-to-pylon connection, it is clear that all pylons are connected to the engine at two locations; using one front- and one rear engine mount. The locations where the mounts are placed on the engine is set for a given engine, although the location of the front mount can vary depending on the manufacturers wishes and the size of the engine. In general, two configurations are used, shown in Figure 4.2; In pylons employing the 'fan mounting' concept in Figure 4.2a, the front mount is placed on the fan casing and the near mount on the core at the turbine exhaust. The 'core mounting' concepts places the front engine mount on the engine core, roughly around the compressor entry, as shown in Figure 4.2b.

The exact layout of the connection varies widely per pylon, as is clear from Appendix section A.1, but in general a system of links and joints is used for both front and rear mount. Figure 4.3 shows the engine mounting configuration of the CFM56-7B engine as installed on the Boeing B737-600, -700, -800, -900, -BBJ, COMBI and C40A aircraft, representing a typical connection mounting strategy as can be seen on modern



(a) Fan pylon concept

(b) Core pylon concept

Figure 4.2: Two different strategies for engine-pylon attachment; the concept on the left shows an engine attached to the pylon via a clamp on the fan casing. The concept on the right has the engine attached purely via the core of the turbofan engine [1].

aircraft. As is clear from the figure, the CFM56-7B has its forward engine mount placed on the fan, and the rear engine mount placed on the turbine rear frame. Detailed schematic drawings of the mounts can be see in Figure 4.4. The forward mount, Figure 4.4a, carries the vertical and lateral loads, and is connected to the pylon using a combination of bolts and shear pins with two fittings, one on the engine and one on the pylon. The aft mount, Figure 4.4b, constraints the engine in all directions, and is connected to the engine using only a fitting on the pylon side, that is connected directly to the turbine frame using four links. Finally,

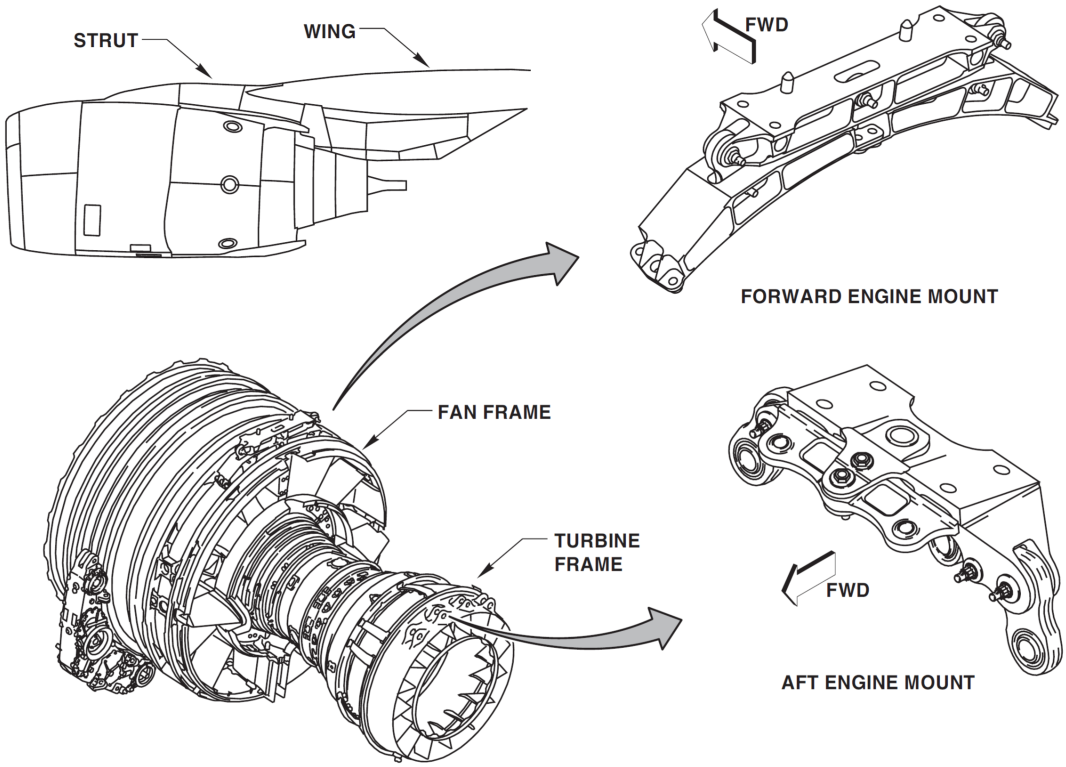
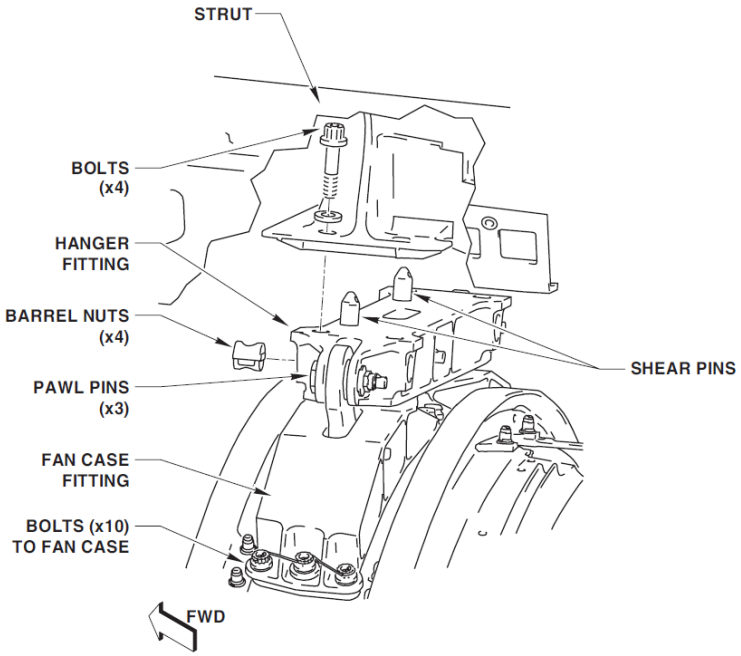
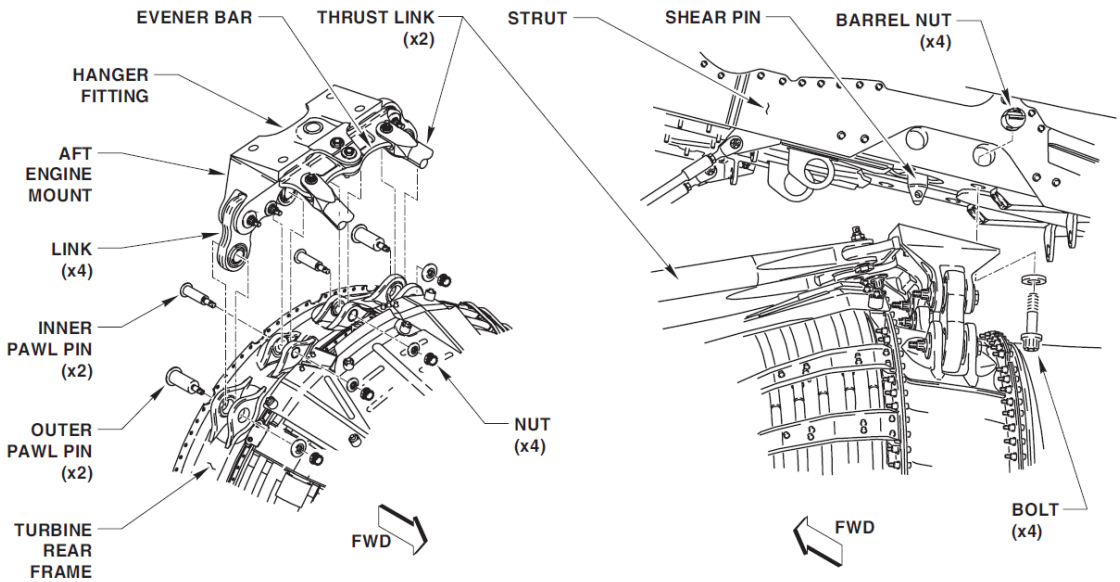


Figure 4.3: Engine to pylon attachments on a CFM56-7B engine as installed on the Boeing B737-600, -700, -800, -900, -BBJ, COMBI and C40A/ALL aircraft [25].

thrust bars are installed that connect the fan frame with the aft mount and take axial thrust loads (forward and rear thrust). These are shown in Figure 4.5, and may or may not be installed depending on the airframe manufacturers desires and wishes. Thrust bars generally connect to the engine at the core, on or around the compressor entrance around the 11- and 1 o'clock points. On the pylon side, they are generally connected to



(a) Forward mount



(b) Aft mount

Figure 4.4: Detailed schematic drawings showing the forward and aft engine mounts of the CFM56-7B engine as installed on the Boeing B737-600, -700, -800, -900, -BBJ, COMBI and C40A/ALL aircraft [25].

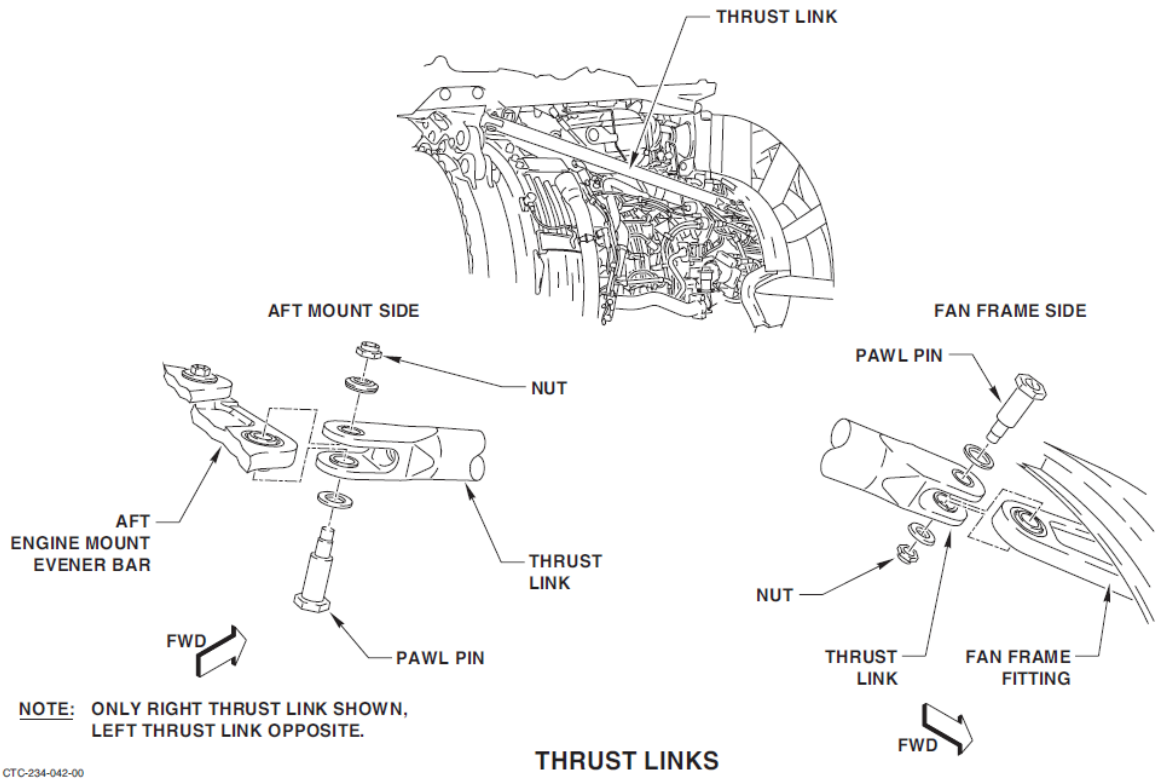


Figure 4.5: Schematic drawing of the thrust links of the CFM56-7B engine as installed on the Boeing B737-600, -700, -800, -900, -BBJ, COMBI and C40A/ALL aircraft [25].

the engine aft mount, as is the case for the CFM56-7B engine as shown in Figure 4.4b and Figure 4.5.

4.1.3. PYLON-AIRFRAME CONNECTION

On the airplane side, the pylon is attached to the airframe on at least two locations; the front and the rear airframe mounting points. For wing-mounted engines with the engine placed in front of the wing, the front wing connection is generally placed at the front wing spar, while the rear wing connection placed on the skin of the wing box, somewhere between the front and rear spar. This setup can be seen clearly on the examples shown in Appendix A.1. Both the L-1011-, A300-, S3A- and C-141 are mounted to the airframe using a mounting configuration that uses two mounting points at the front wing mounting and one at the rear wing mount. The DC-10 splits the front wing mount into two brackets; one placed on the surface of the front spar taking vertical and side loads and one directly underneath the front spar for thrust loads, often designated as the spigot. The C-141 splits up the front wing mount even further; here the pylon is connected to the wing using an elaborate framework links that distribute the forces and degrees of freedom amongst themselves.

4.2. ZERO-THICKNESS GEOMETRY PARAMETERIZATION

Having thoroughly investigated the characteristic features of several aircraft pylons, a proper assessment can be made on how to capture these characteristic features in a comprehensive parameterization model that is able to represent an as broad as possible collection of pylons while at the same time assuring a good level of efficiency and robustness.

4.2.1. HARDPOINTS DESCRIPTION

The prime insight that is required to be able to come up with a proper parameterization is that a pylon is first and foremost a structural connection between the engine and the aircraft; this realization stipulates the starting point of any parameterization. Regardless of the structural concept, every pylon investigated in the previous text was connected to the engine and airframe at at least four points, designated by the numbers 1,

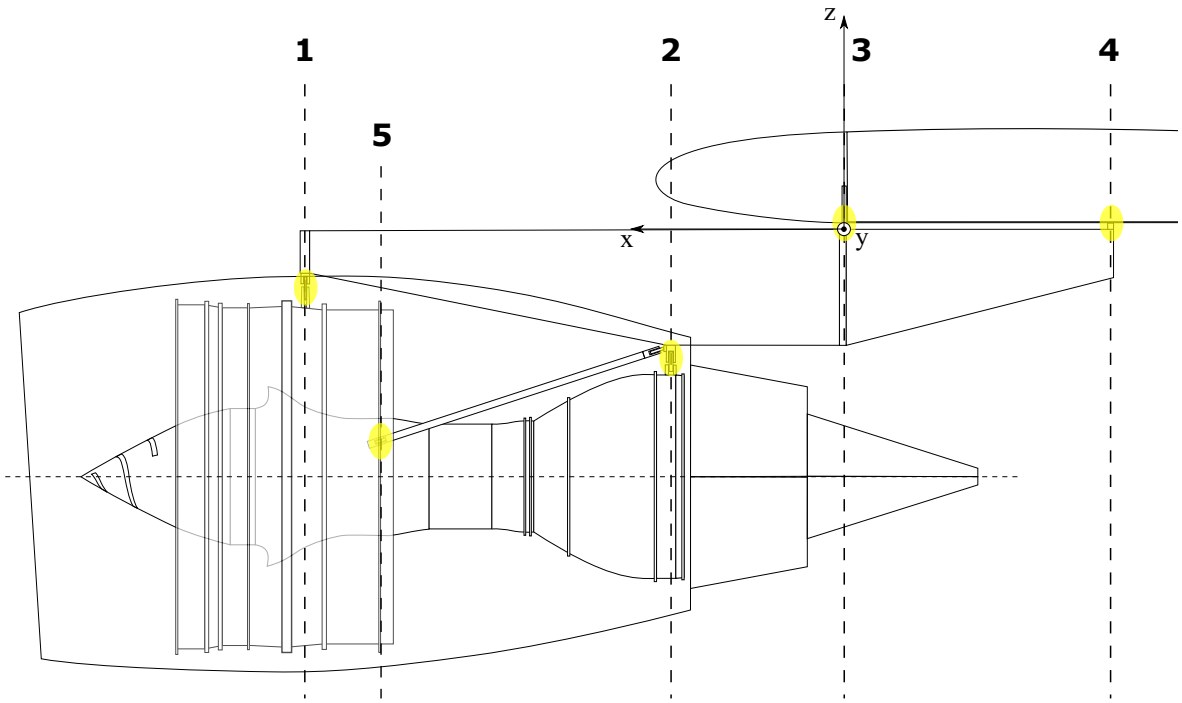


Figure 4.6: Schematic representation of a typical engine mounting installation with the mounting points or stations indicated in yellow.

2, 3 and 4;

- Two engine-to-pylon connection points:
  - Forward engine mounting point → **station 1**
  - Aft engine mounting point → **station 2**
- Two pylon-to-airframe connection points:
  - Forward airframe mounting point → **station 3**
  - Aft airframe mounting point → **station 4**

Moreover, in some cases thrust links were used to take the axial thrust loads. The thrust links are mounted to the engine using a fifth set of hardpoints, who's location will be indicated as **station 5**, and usually sit somewhere between the engine front- and rear mounting points. On the pylon side, the thrust links usually connect to the rear engine-to-pylon mount, indicated here as station 3, and therefore that connection is already taken into account.

- One **optional** engine-to-thrust link connection point:
  - Forward engine-thrust link mounting point → **station 5**

Figure 4.6 shows the situation schematically in side view for an under-the-wing mounted turbofan engine. Note that the origin of the reference axis system in this parameterization will be set at location of the front airframe mounting point, as this is generally the easiest point to place in the reference frame of the aircraft. This decision was taken with the further implementation of the pylon model in an aircraft level model in mind.

In Figure 4.7, the situation is translated into a 3D isometric model, where only the hardpoints are left visible. This is the starting point of the parameterization process. Each hardpoint is depicted using a yellow triangle, facing upwards for engine-to-pylon hardpoints and downwards for pylon-to-wing hardpoints. For

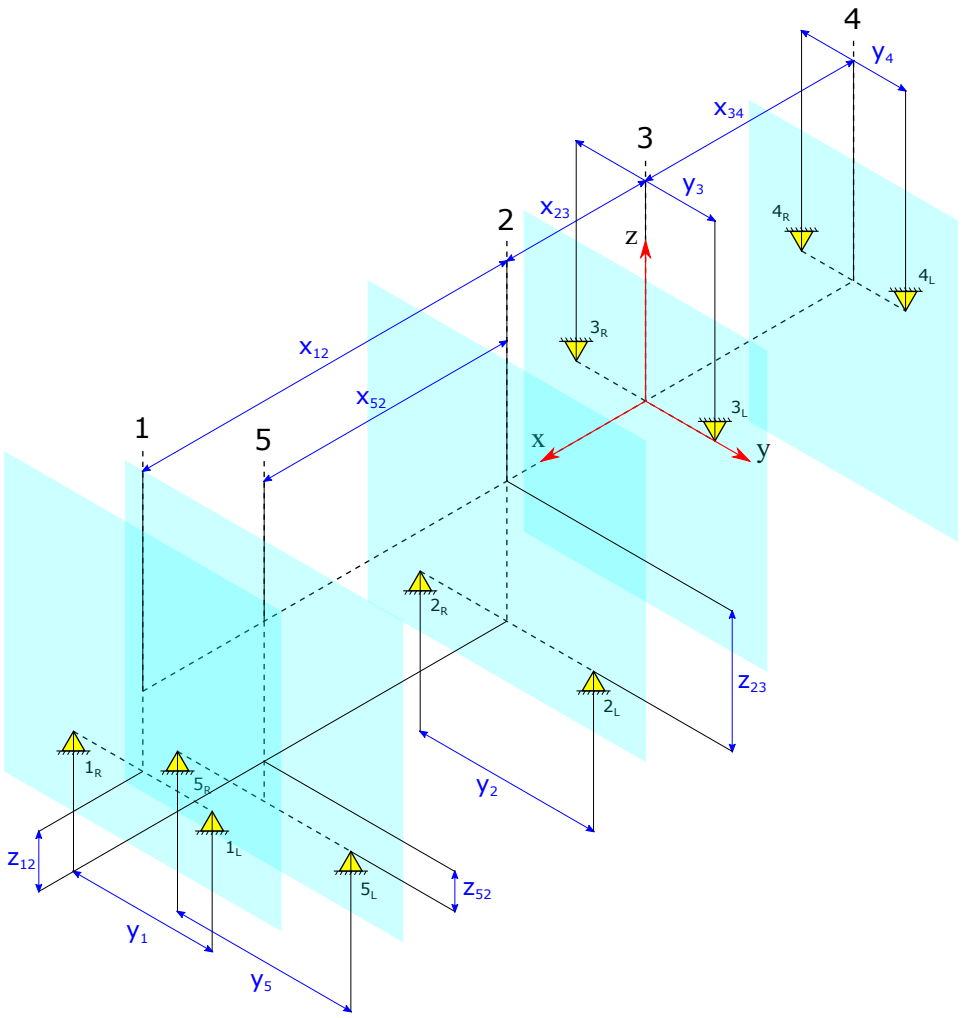


Figure 4.7: The situation shown in Figure 4.6 can be translated into a 3D model that will be the starting point of the parameterization process. In this figure, the 3D-model is shown in isometric view. The depiction is scaled down to only include the actual hardpoints of the pylon, indicated by their station numbers. Measurements in blue depicts the parametric representations of the hardpoint locations.

every interface, two hardpoints are assumed - one on the right side (indicated by the index 'R') and one on the left side (indicated by the index 'L'). Of course as we saw in the previous text, the actual number of links between engine- and pylon mount or pylon- and airframe mount can vary. In cases where only one link is required, the y-distance between the two hardpoints can be set to 0 such that the hardpoints coincide. Mounts with more than two links or hardpoints are modeled here by their most outward connections. In the figure, the locations of the stations are indicated by their respective station numbers. YZ-planes are shown in cyan at the location where they run through the station. The origin of the coordinate system, as explained above, is set midway of the hardpoints at station 3 in red, indicating the location where the pylon front mount to the airframe is situated. For a 'standard' front under-the-wing mounted engine, this point will be located at the front spar of the wingbox, making it a good reference point from the perspective of further integration of the structural model in higher level structural models.

The parameters describing the locations of the hardpoints with respect to the origin and each other are indicated in blue. As shown in the picture, the parameters follow the notation of the reference coordinate system (in red), with its x-axis in longitudinal direction, y-axis in lateral direction and z-axis in vertical direction. Longitudinal distances are indicated by the symbol 'x', complemented by a subscript indicating what distance the parameter describes. For example, the parameter x_{12} describes the longitudinal distance between the engine front and rear mounts (stations 1 and 2). In the same way, the parameter z_{23} indicates the verti-

cal distance between the engine rear mount and the pylon-to-wing front attachment point. For parameters describing the lateral- or y-distances, the subscripts simply indicate the station number, as they simply describe the lateral distance between two mounting points at the same station. A full overview of the hardpoint parameters and their definition is given in Table 4.1. In total, these parameters form a complete description of the locations and distances of the hardpoints of the engine, describing more or less the 'outer' dimensions of the structure. The parameters are chosen such that they describe meaningful distances that can be simply measured and communicated by both engine and airframe manufacturers; after all, distances like the front and rear engine mounts are generally set for a specific type of engine and can be found in an engine's technical description of manual. Other parameters like the distance between engine rear mount and wing front mount can easily be measured or decided on by a designer. The same is true for other measurements described in Table 4.1.

4.2.2. OUTER MOLD DESCRIPTION

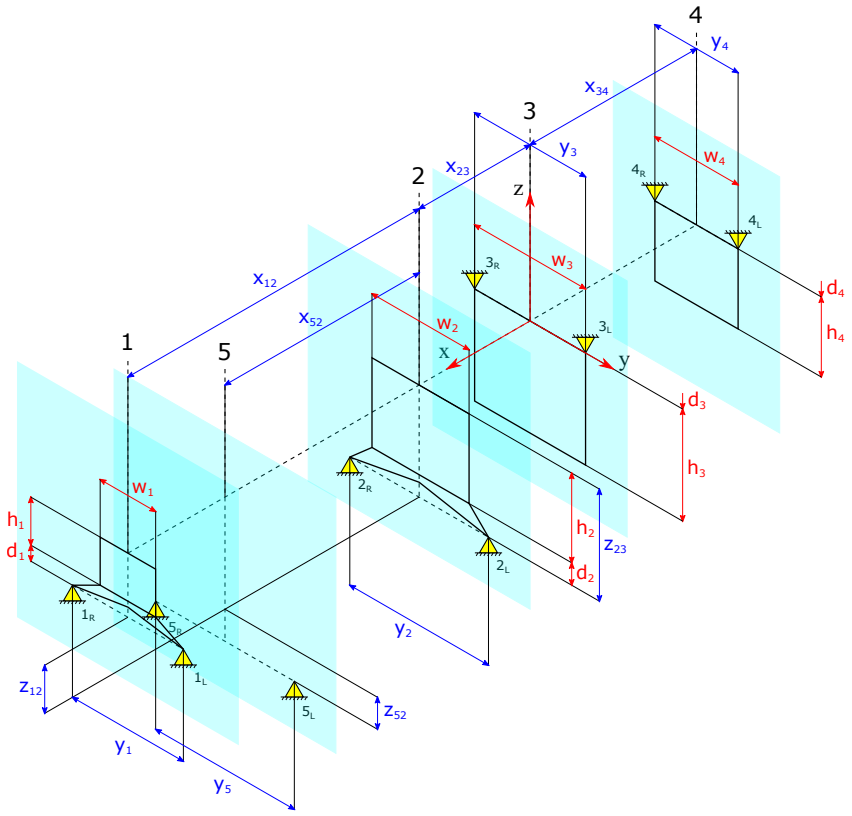
Now that the parameters describing the hardpoints are defined, the next step is to define a pylon structure to connect these hardpoints. This starts with the realization that the planes defining the stations in Figure 4.7 can be regarded as planes sectioning the pylon structure at the locations of the hardpoints. Furthermore, assuming that the fittings on the engine/airframe and the pylon side are placed in the same plane, these sections will also define bulkheads in the box-beam structure. As such, they define primary outer shape of the box-beam. In Figure 4.8a, the sections are drawn onto the station planes. Assuming a rectangular box, each section is defined using a width an height, and a vertical distance between the box and the hardpoints, essentially defining the length of the linkages between the engine/airframe- and pylon fittings as well as the angles at which the forces are introduced into the structure. Note that these distances are set to 0 for this example, signaling the assumption that the pylon is connected directly to the airframe using lug fittings, as is generally the case. With the bulkheads defined, the full pylon can be defined by connecting the corner points of the sections as is done in Figure 4.8b. Finally, the thrust links are drawn by connecting the engine-to- thrust link hardpoints to the engine rear attachment hardpoints.

Summarizing this part of parameterization, the full zero-thickness pylon parameterization as described to this point is shown in Appendix B in Figure B.1. In this figure, the station planes have been removed for reasons of clarity. The outer geometry of the zero-thickness pylon model is clearly visible.

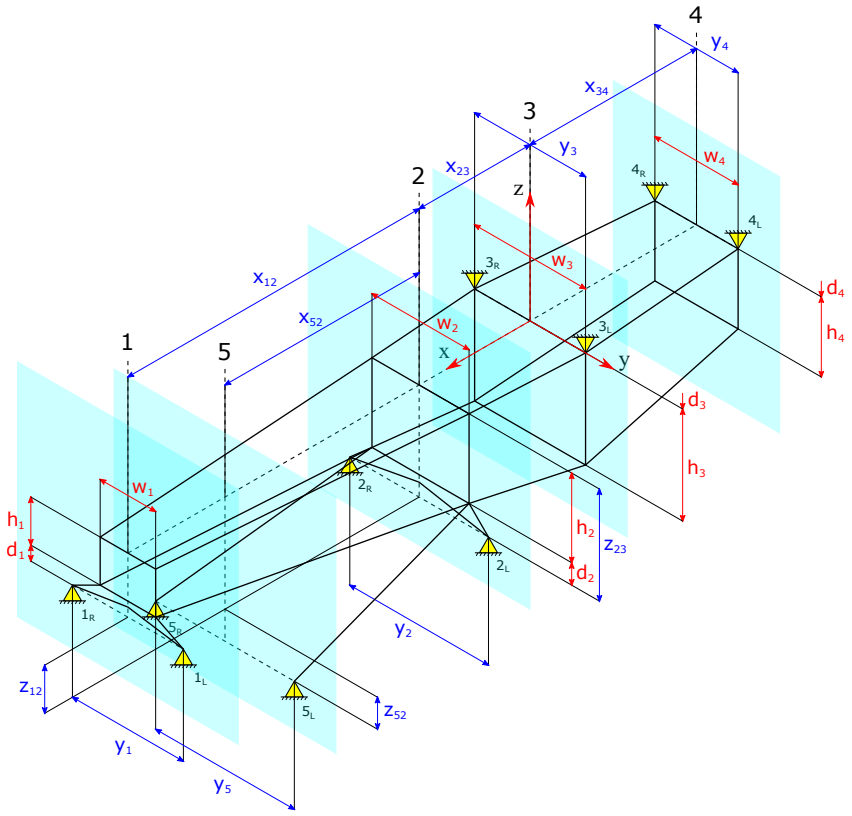
4.2.3. ROTATING THE STRUCTURE

So far the pylon parameterization was based on a standard under-the-wing mounted engine configuration with the engine installed in front of the wing. But many other configurations are possible, and will need to be captured by the model as well. Examples of this are the over-the-wing nacelle (OWN) type engine installation and the side-fuselage mounted engine installation type, which are considered frequently for new aircraft. The proposed pylon model should be equally useful for aircraft concepts employing those engine installation types.

To enable the model to capture these engine installation architecture, the **orientation angle** φ will be added to the parameterization as an additional variable. The definition and use of this angle is shown in Figure 4.9, which shows the pylon in front view. The orientation is defined by taking the angle between the vertical longitudinal plane running though the origin of the local axis system and the xz-plane of the global axis system. When the pylon is rotated in anti-clockwise direction, as shown in the top right image, the angle φ increases. For side-mounted engines, the pylon is placed at an installation angle around 90°, as shown in the bottom left. For over-the-wing (OWN) mounted engines, an orientation angle of 180° is applied (bottom right). By applying the orientation angle in this way, the full range of engine installation types can be captured using just 1 parameter, which makes this parameter extremely effective.



(a) Pylon sections at the various stations.



(b) The sections are connected at the corner points to obtain a full 3D model.

Figure 4.8: The drawing in Figure 4.7 is extended by drawing a pylon outer mold that connects the hardpoints to each other (isometric view). The dimensions of the pylon sections at each station are shown in red.

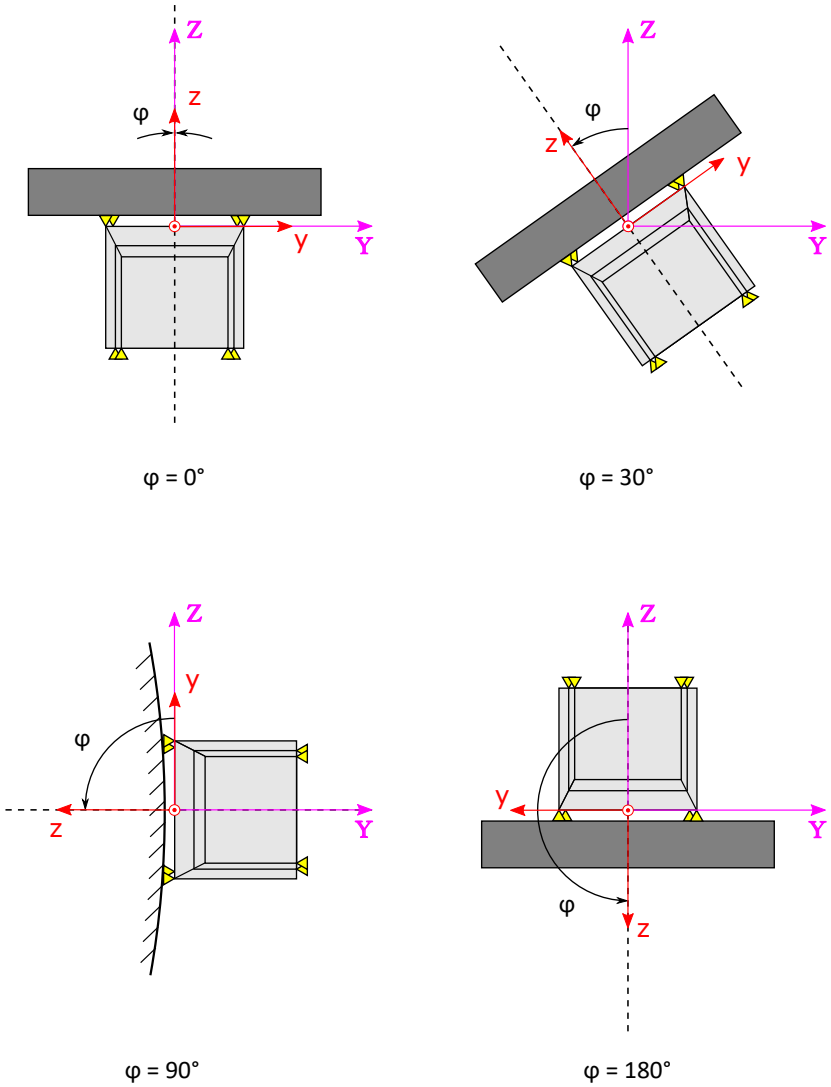


Figure 4.9: Front view of a schematic pylon showing the orientation of the pylon at different orientation angles φ.

Table 4.1: Parameters describing the locations of the pylon hardpoints.

Parameter	Definition	Unit
x_{12}	Longitudinal distance between the front- and rear engine mounting points	m
x_{23}	Longitudinal distance between rear engine mount and front airframe mount	m
x_{34}	Longitudinal distance between front airframe mount and rear airframe mount	m
x_{52}	Longitudinal distance between thrust link front engine mounting point and rear engine mount	m
y_1	Lateral distance between left and right hardpoints on the front engine mount	m
y_2	Lateral distance between left and right hardpoints on the rear engine mount	m
y_3	Lateral distance between left and right hardpoints on the front airframe mount	m
y_4	Lateral distance between left and right hardpoints on the rear airframe mount	m
y_5	Lateral distance between left and right hardpoints on the thrust link front engine mounting point	m
z_{12}	Vertical distance between the front- and rear engine mounts	m
z_{52}	Vertical distance between the thrust link front engine mounting and the rear engine mount	m
z_{23}	Vertical distance between the rear engine mount and the front airframe mount	m

Table 4.2: Parameters describing the pylon outer mold geometry.

Parameter	Definition	Unit
$w_{1...4}$	Width of the box-beam section at station 1...4	m
$h_{1...4}$	Height of the box-beam section at station 1...4	m
$d_{1...4}$	Vertical distance between the hardpoints and the nearest spar at station 1..4	m

Table 4.3: Parameters describing the orientation of the pylon.

Parameter	Definition	Unit
φ	Angle of orientation of the pylon	°

4.3. STRUCTURAL ELEMENTS PARAMETERIZATION

Now that the zero-thickness geometry of the pylon is defined, it is time to add material to the structure. This means defining first of all the plate thicknesses, and secondly the thickness and geometry of structural elements.

To simplify the definition of structural parameters in this section, it is recognized that the pylon model of Figure B.1 can be divided into 3 separate compartments, each representing box-beam structures when using a box-beam type structure, as indicated in Figure 4.10; for each box-beam, the set of parameters used to describe the model is the same. The compartments defined this way are:

- **Compartment A:** between station 1 and station 2
- **Compartment B:** between station 2 and station 3
- **Compartment C:** between station 3 and station 4

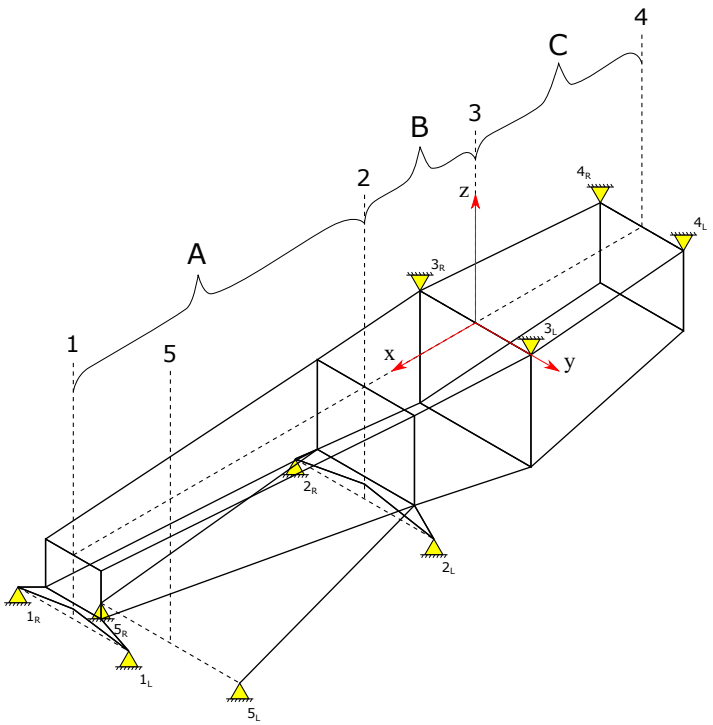


Figure 4.10: Pylon model with box-beam compartments A. B and C indicated.

4.3.1. BOX-BEAM STRUCTURAL ELEMENTS DESCRIPTION

The structural elements making up the box-beam are described here for one section. The same definitions apply for all three box-beam compartments of the pylon. A visual representation of some of the most important structural members for a pylon box-beam are shown in Figure 4.12. An overview of all structural parameters is given in Table 4.4.

LONGERONS

Longerons are strengthening profiles at the corners of the box-beam which form the outer structural frame of the pylon and run along the length of the pylon from front to rear, see Figure 4.12a. Similar to how they are used in the wings, longerons are used to transfer the bending loads that are exerted by the engine into

the pylons spars and ribs. Longerons are L-profiles that have a thickness t_{long} and flange height $h_{fl,long}$ as shown in Figure 4.11. The length of the longeron l_{long} is set to the length of the box-beam compartment. Their number is set to 4, as they run along the four corner points of the rectangular box-structure.

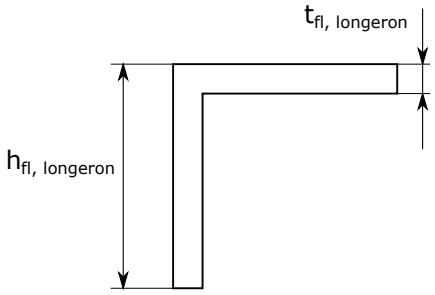


Figure 4.11: Definition of the longeron parameterization

RIBS

Secondly, ribs are added to the structure. Ribs add torsional stiffness to the pylon box, and are placed using a continuous spacing in the length of the box structure, as shown in Figure 4.12b. This spacing is controlled by the number of ribs n_{ribs} which are added to the section. In a practical application, cutouts will be made in the ribs to accommodate for thing like electrical wiring, fuel- and other fluid piping and other components that need to be placed in or around the pylon structure, but this is not taken into account in this model. Here the ribs are simplified to consist of thin plates with thickness t_{ribs}. The height and width of the ribs is equal to that of the box-beam at the location of the rib under consideration.

BULKHEADS

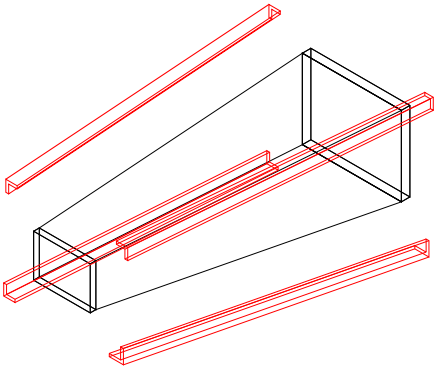
The bulkheads form the ends of a compartment, and are the locations where the forces are fed into the structure, Figure 4.12c. Like ribs, they add to bending and torsion stiffness of the beam, but due to their special function they will be much thicker and stronger. Furthermore, they are part of the structural frame of the box-beam. For each box-beam, two bulkheads can be defined, numbered 1 and 2. Bulkheads are defined by the height and width of the cross section at that station. Furthermore, each bulkhead has a flange by which the bulkhead is fastened to the spars and panels. The flange width is defined using the parameter $w_{blkh,flanges}$. The thickness of the bulkheads is defined using a separate parameter for the 'web' part $t_{blkh,web}$ and one for the flanges $t_{blkh,flanges}$. These parameters are defined for each of the two bulkheads of the box-beam structure.

STRINGERS

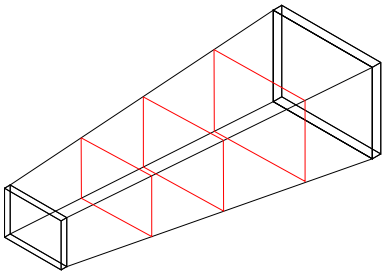
Stringers are longitudinal members that add buckling stiffness to the plates onto which they are mounted. In this model, stiffeners can only be added to the lower spar of the box-beam, as is done in Figure 4.12d. The number of stringers is defined using the parameter $n_{stringers}$ and they are spread equally across the spar width depending on this number, similar to the ribs. Stringers come in different forms, like Z-stringers, I-stringers, C-stringers and T-stringers. No matter the exact shape, each stringer will have a height $h_{stringer}$ and a thickness $t_{stringer}$. The height-to-weight ratio of the stringers is set to 1 here, so the stringer width will equal the stringer height. The thickness is distributed equally across the stringer cross-section.

SPARS AND WEBS

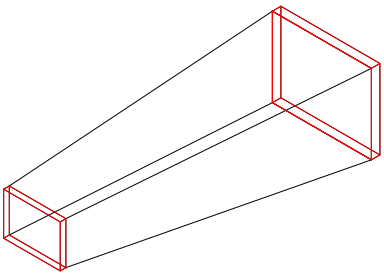
The final components considered are the spars and webs of the box-beam; spars are the upper and lower 'plates' making up the box-beam, while the webs are the side panels. Their geometry is defined by the parameters describing the outer mold of the pylon box. Each panel has a thickness that can be varied, designated by the parameters $t_{spar,upp}$, $t_{spar,low}$, $t_{web,left}$ and $t_{web,right}$.



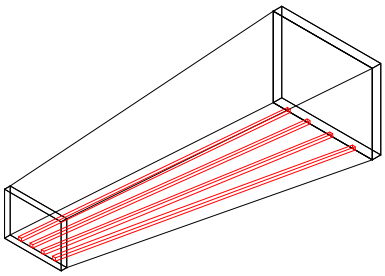
(a) longerons (exploded view)



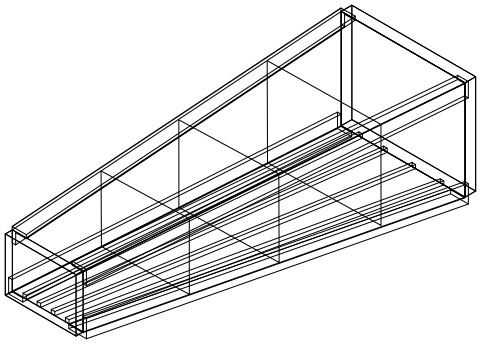
(b) ribs



(c) bulkheads



(d) stringers (general shape)



(e) total pylon box-beam element

Figure 4.12: Structural elements added to the box-beam geometry

4.3.2. THRUST LINKS

Finally, the thrust links (if present) are also considered structural members of the pylon. In this thesis project, the thrust links will be modeled as beams with a circular section that are attached to the rear engine-to-pylon mounting structure using a pin-type connection. While the length of the thrust links is set by the distance between the hardpoints at stations 5 and 2, the diameter of the rod, defined using the parameter $d_{thrustlink}$, decides the amount of material used in the thrust link, providing the structural rigidity of the link. As such, this last parameter is added to the other structural parameters defining the structural elements parameterization in Table 4.4.

Table 4.4: Summary of parameters describing structural elements in the pylon model.

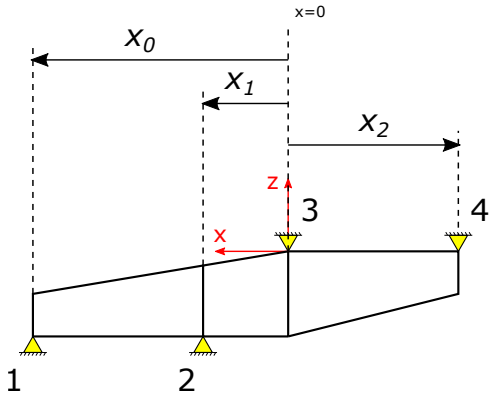
Parameter	Definition	Unit
$t_{longeron,flanges}$	Longeron flange thickness	[m]
$h_{longeron,flanges}$	Longeron flange height	[m]
$l_{longeron}$	Longeron length	[m]
n_{ribs}	Number of ribs	[-]
t_{ribs}	Ribs thickness	[m]
$t_{blkh1,web}$	Bulkhead 1 web thickness	[m]
$t_{blkh1,flanges}$	Bulkhead 1 flange thickness	[m]
$w_{blkh1,flanges}$	Bulkhead 1 flange width	[m]
$t_{blkh2,web}$	Bulkhead 2 web thickness	[m]
$t_{blkh2,flanges}$	Bulkhead 2 flange thickness	[m]
$w_{blkh2,flanges}$	Bulkhead 2 flange width	[m]
$n_{stringers}$	Number of stringers	[-]
$h_{stringers}$	Stringers height	[m]
$t_{stringers}$	Stringers thickness	[m]
$t_{spar,upp}$	Upper spar thickness	[m]
$t_{spar,low}$	Lower spar thickness	[m]
$t_{web,left}$	Left web thickness	[m]
$t_{web,right}$	Right web thickness	[m]
$d_{thrustlink}$	Thrust link diameter	[m]

4.3.3. LONGITUDINAL SHIFTS

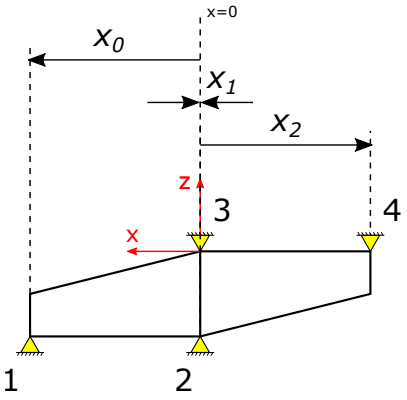
With the parameters defining the zero-thickness geometry and the structural geometry defined, we now have a full mathematical description of the pylon geometry in the 3D space. Yet to properly define the pylon mathematically, we need one additional condition to define the longitudinal relationships between the pylon sections at stations 1 to 4, which ensures that the right sections are coupled with each other when defining the final pylon geometry. To explain this condition, we will look at what happens to the pylon geometry when the engine is shifted from forward of the wing to aft of the wing.

Lets consider an under-the wing mounted engine configuration, as is shown in Figure 4.13. The stations are indicated in the same way as is done throughout this chapter. The image shows what happens when the engine moves along a trajectory in the axial direction from an initial position where it is fully in front of the wing, to a position underneath the wing and eventually a position fully behind the wing. In this figure, the axial distances between the origin of the reference axis system and the locations of the stations 1, 2 and 4 are defined as x_0, x_1 and x_2 respectively. As before, the origin of the reference coordinate system is located at station 3 (the front wing mounts) which is assumed fixed, and the positive x-axis is pointing in the direction of the aircraft nose. The distances are defined positive if the station is in front of the $x = 0$-plane, and negative when the station is behind that plane.

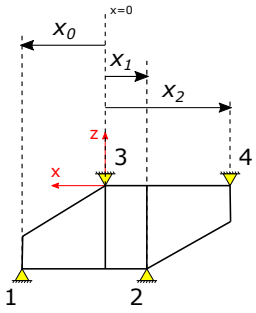
Initially, the engine is placed at a location far ahead of the wing as shown in Figure 4.13a, and we are basically dealing with a cantilever beam with its primary axis in the direction of the x-axis. Both the engine front mount and the engine rear mount are positioned ahead of both wing mounting points. Four bulkheads and three beam sections can be defined, where the order of the stations is simply 1-2-3-4. In this situation, x_0 and x_1 are positive non-zero numbers, while x_2 is negative. And as such $x_0 > x_2$.



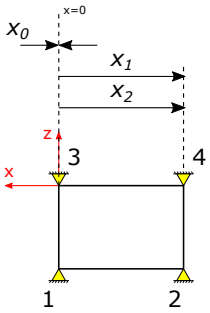
(a)
$x_0 > 0$
$x_1 > 0$
$x_0 > x_2$



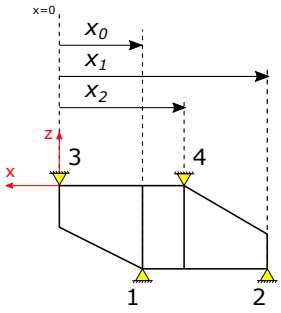
(b)
$x_0 > 0$
$x_1 = 0$
$x_0 > x_2$



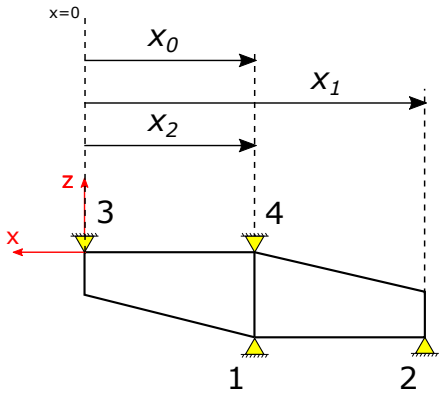
(c)
$x_0 > 0$
$x_1 < 0$
$x_0 > x_2$



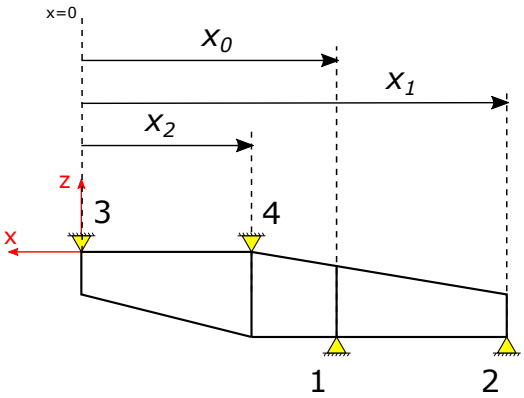
(d)
$x_0 = 0$
$x_1 < 0$
$x_0 > x_2$



(e)
$x_0 < 0$
$x_1 < 0$
$x_0 > x_2$



(f)
$x_0 < 0$
$x_1 < 0$
$x_0 = x_2$



(g)
$x_0 < 0$
$x_1 < 0$
$x_0 < x_2$

Figure 4.13: Visual representations of the pylon geometry changes while shifting the engine from front to aft

When the engine moves rearward, at some point the rear engine mount will end up in the same plane as the front wing mount, meaning distance x_1 becomes zero. This situation is shown in Figure 4.13b. Now, the bulkhead positioned at the rear engine mount and that at the front wing mount coincide. Only three bulkheads remain and two beam sections can be identified. At the same time, the front engine mounting point is still ahead of the front wing mounting point, so $x_0 > 0$ and $x_0 > x_2$.

When the engine moves even further aft, the longitudinal position of the rear engine mount will move with it and the position of the rear engine mount will suddenly be situated *behind* the front wing mount. This is the situation shown in Figure 4.13c. At this point, we can no longer speak of a cantilever beam that is extended outwards in the (positive) x-direction. The structure is again comprised of 3 *compartments* and 4 bulkheads, but the order of the stations is now 1-3-2-4. And subsequently, the pylon section at station 1 now needs to be connected with that of station 3, and not of station 2. Station 2 need to be connected to station 3 on the left and station 4 on the right, and station 4 is only connected to station 2. In this situation, $x_0 > 0$, but x_1 has turned negative, yet remains smaller than $x < 2$. Or in mathematical terms: $0 > x_0 > x_2$.

This situation is maintained as the engine moves further aft, moving though the positions shown in Figure 4.13d, when x_0 also turns negative. If the engine moves even further aft, at some point the rear engine mount will surpass the position of the rear wing mount as in 4.13e, and $x_1 < x_2$. Finally, at some point the front engine mount lines up with the rear airframe mount, and the situation basically mirrors the starting situation, but with the mounts shifted. Figures 4.13f and 4.13g show the pylons when the engine in placed aft of the wing mounts. In the final position, both engine mounts sit behind both wing mounts, and the condition $x_2 > x_0 > x_1$ becomes valid, with all three parameters $x_0, x_1, x_2 < 0$.

4.4. FULL PARAMETRIC MODEL

Having defined both the hardpoint-, outer mold and structural parameters, a full description of the pylon model is now available. Table 4.5 summarizes all parameters defined in this chapter. A total of 80 parameters is defined to describe the model. This will form the basis for the KBE application used to define and size the pylon structure, as described in section 3.2.

Table 4.5: Summary of the pylon structural parameterization.

Parameter	Definition	Unit
x_{12}	Longitudinal distance between the front- and rear engine mounting points	m
x_{23}	Longitudinal distance between rear engine mount and front airframe mount	m
x_{34}	Longitudinal distance between front airframe mount and rear airframe mount	m
x_{52}	Longitudinal distance between thrust link front engine mounting point and rear engine mount	m
y_1	Lateral distance between left and right hardpoints on the front engine mount	m
y_2	Lateral distance between left and right hardpoints on the rear engine mount	m
y_3	Lateral distance between left and right hardpoints on the front airframe mount	m
y_4	Lateral distance between left and right hardpoints on the rear airframe mount	m
y_5	Lateral distance between left and right hardpoints on the thrust link front engine mounting point	m
z_{12}	Vertical distance between the front- and rear engine mounts	m
z_{52}	Vertical distance between the thrust link front engine mounting and the rear engine mount	m
z_{23}	Vertical distance between the rear engine mount and the front airframe mount	m
$w_{1...4}$	Width of the box-beam section at stations 1 to 4	m
$h_{1...4}$	Height of the box-beam section at station 1 to 4	m
d_1	Vertical distance between the front engine mounting points and the lower spar at station 1	m
d_2	Vertical distance between the front engine mounting points and the lower spar at station 2	m
d_3	Vertical distance between the front engine mounting points and the lower spar at station 3	m
d_4	Vertical distance between the front engine mounting points and the lower spar at station 4	m
φ	Angle of orientation of the pylon	°
$[t_{longeron,flanges}]_{A...C}$	Longeron flange thickness for compartment A ... C	[m]
$[h_{longeron,flanges}]_{A...C}$	Longeron flange height for compartment A ... C	[m]
$[l_{longeron}]_{A...C}$	Longeron length for compartment A ... C	[m]
$[n_{ribs}]_{A...C}$	Number of ribs for compartment A ... C	[-]
$[t_{ribs}]_{A...C}$	Ribs thickness for compartment A ... C	[m]
$[t_{blkh1,web}]_{A...C}$	Bulkhead 1 web thickness for compartment A ... C	[m]
$[t_{blkh1,flanges}]_{A...C}$	Bulkhead 1 flange thickness for compartment A ... C	[m]
$[w_{blkh1,flanges}]_{A...C}$	Bulkhead 1 flange width for compartment A ... C	[m]
$[t_{blkh2,web}]_{A...C}$	Bulkhead 2 web thickness for compartment A ... C	[m]
$[t_{blkh2,flanges}]_{A...C}$	Bulkhead 2 flange thickness for compartment A ... C	[m]
$[w_{blkh2,flanges}]_{A...C}$	Bulkhead 2 flange width for compartment A ... C	[m]
$[n_{stringers}]_{A...C}$	Number of stringers for compartment A ... C	[-]
$[h_{stringers}]_{A...C}$	Stringers height for compartment A ... C	[m]
$[t_{stringers}]_{A...C}$	Stringers thickness for compartment A ... C	[m]
$[t_{spar,upp}]_{A...C}$	Upper spar thickness for compartment A ... C	[m]
$[t_{spar,low}]_{A...C}$	Lower spar thickness for compartment A ... C	[m]
$[t_{web,left}]_{A...C}$	Left web thickness for compartment A ... C	[m]
$[t_{web,right}]_{A...C}$	Right web thickness for compartment A ... C	[m]
$d_{thrustlink}$	Thrust link diameter	[m]

5

APPLICATION OVERVIEW AND GEOMETRY GENERATION

In the previous chapter, the first of the four ingredients as specified in section 3.2 was established - a mathematical model description of the pylon, in the form of a parameterization. But as was shown in Figure 3.1, the parameterization is only one piece of the puzzle. The next step is to implement this parameterization into a software application that can use the parametric definition of the pylon to generate a 3D geometry for any given engine type and placement, and subsequently size this geometry such that it is able to withstand relevant loads. In this chapter, the general structure of that application is described. Furthermore, the implementation of the parameterization onto a KBE-system is described. The in-depth description of the coupling with the structural analysis tool is further examined in chapter 6.

5.1. OVERVIEW OF THE KBE-APPLICATION ARCHITECTURE

To perform the structural analysis and weight estimation procedures required in this thesis project, a software application was built in object-oriented Python 3.7 using the ParaPy V1.9.0 KBE-platform as a basis for most of the 'general' functionality, geometry generation, weight extraction and meshing. To perform the structural analysis, this code was coupled with the commercially available FEM-solver Abaqus using an extended version of the ParaPy/Abaqus API. A basis version of this ParaPy/Abaqus API was provided by ParaPy, but at the time of delivery the functionality of the API was not extensive enough to support all capabilities required to perform the analysis required in this thesis project, which meant that this functionality also needed to be added during the thesis project. More on this part will be explained in chapter 6.

Figure Figure 5.1 shows an activity diagram representing the general operational steps executed by different components of the code when trying to obtain a feasible pylon structural geometry and weight for a given set of inputs as defined in section 3.3. As a reminder, the inputs to the system were defined as:

- The zero-thickness geometry parameterization of the pylon, defined using:
  - The locations of the hardpoints, as defined using the parameterization of subsection 4.2.1.
  - The outer mold shape of the pylon, as defined using the parameterization of subsection 4.2.2.
  - The orientation of the pylon, as defined using the parameterization of subsection 4.2.3.
- The characteristics and location of the engine.
- The material used in the pylon structure.

The process starts with the user inputting the required input data for the analysis case, as mentioned above. With this information, the pylon geometry is created. The designer is asked to provide a starting point for the structural elements geometry, and the optimization process is started. The structure is meshed and the resulting mesh, structural data and loads data is transferred into a process that uses the data to construct an Abaqus input-file and job definition. With this, the Abaqus FEM analysis can be executed, which performs

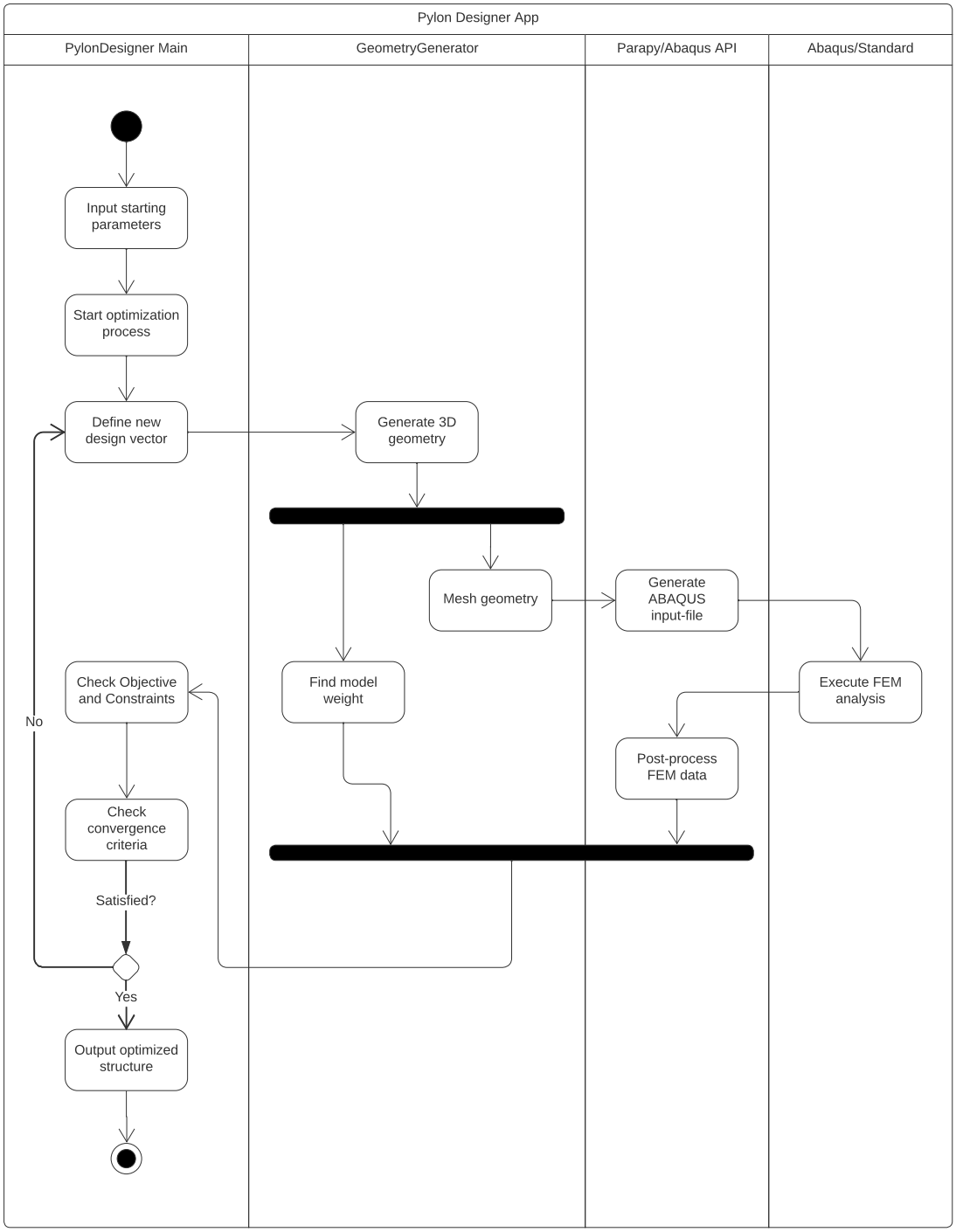


Figure 5.1: Activity diagram showing the general process executed in the pylon designer app.

a structural analysis of the design under consideration. Meanwhile, the structural weight of the pylon is retrieved from the model in ParaPy. The results of the two processes are fed back to the optimizer which uses them to evaluate the objective function and constraint functions of the optimization run. When the optimality conditions for the optimization case are met, the process returns the final results and structural pylon geometry. If not, a new design vector is generated and the evaluation process starts over from the beginning.

Figure 5.2 shows the simplified structure of the code that performs these activities. This UML diagram is created in the class-object style tailored specifically for KBE applications, and shows both the classes used and the objects created when using the classes. A more elaborate version of this UML showing all the classes and object used in the project including their inputs, attributes and parts can be found in Appendix B.

As shown in the UML, the code is initiated by creating an instance of the superclass `PylonDesigner`. In this class, the pylon is created based on the required type of pylon as defined by the user. The structural geometry of the pylon is created inside the `pylon_structure`-part using either one of the classes `DragStrutType`, `BoxBeamType` or `RedundantLinkType`. Each of these is a generalization of the superclass `PylonStructure`, in which the pylon structure geometry is created and the meshes are defined. The inner working of this class are explained in detail further in this chapter in subsection 5.2.2. In this thesis project, only the `BoxBeamType`-class is fully implemented, as the implementation of the other two types was not possible in the given time- and scope restrictions of the project. Full implementation of these classes is left as a recommendation for further research.

The geometry created in the ParaPy `Part`-object '`pylon_structure`' is used in two `Attributes` that obtain the mass of the structure under consideration and perform structural analysis on it, respectively. The structural weight of the full resulting assembly is gathered inside the `structural_weight`-attribute. The `abaqus_input_file`-attribute uses the meshed geometry to generate a '.inp' input file containing the full definition of the geometry, loads and load cases, constraints, solving procedures and required output. This file is then fed into the `stress_analysis`-attribute which use dedicated functional routines employing Abaqus/Standard to evaluate the structure. As stated before, a detailed evaluation of the components and procedures executed in this class will be given in chapter 6, which focus on the structural analysis part of the code.

Structural sizing is performed when activating the '`optimize_structure`' action of the `PylonDesigner` class. This requires definition of a starting design vector, upper- and lower bounds and an optimization case definition. A detailed description of the processes involved in this is given in chapter 7.

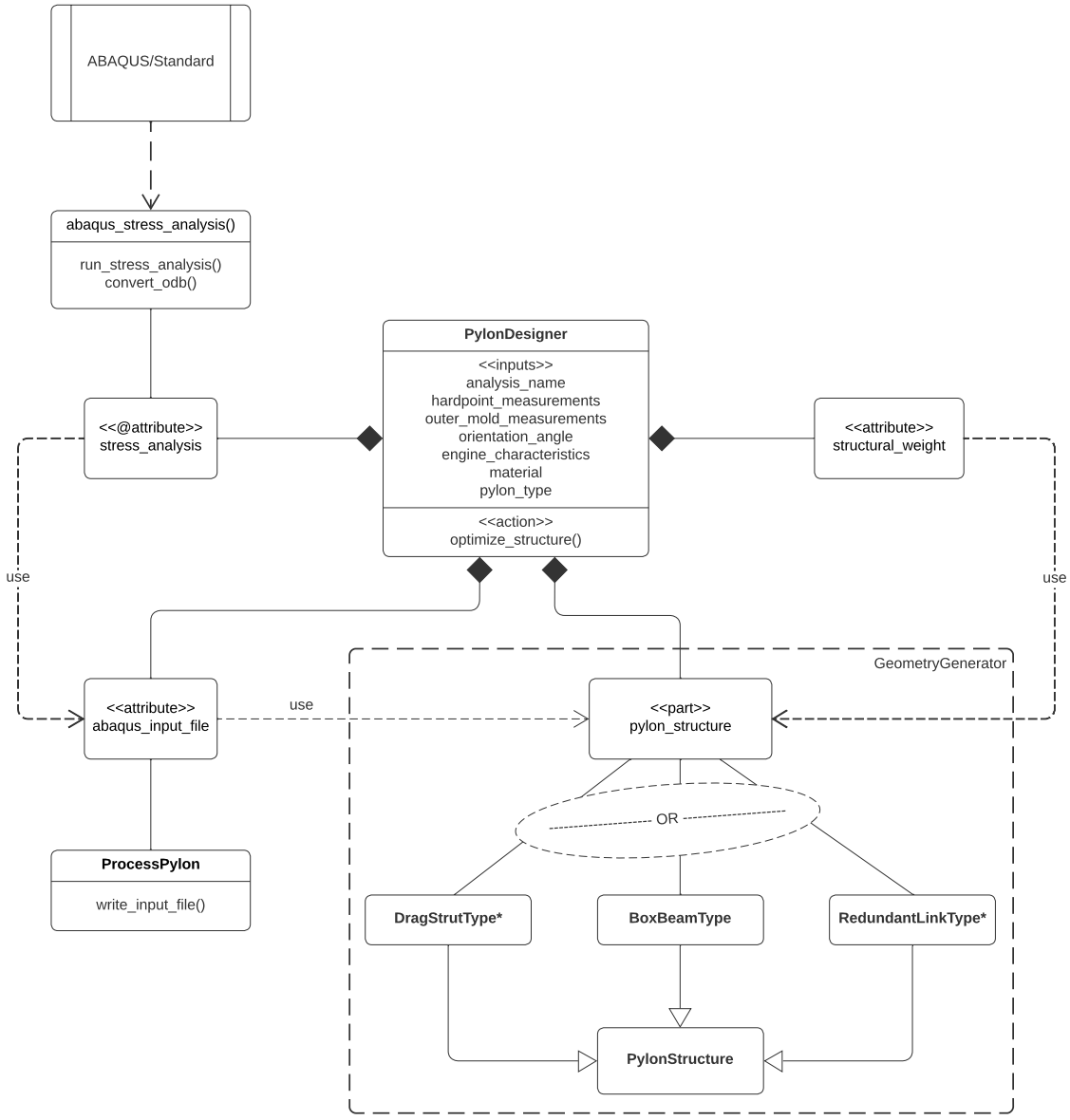


Figure 5.2: Simplified UML class-object diagram showing the most important classes and objects that make up the pylon designer app. Classes indicated with an asteriks (*) are not fully implemented in this thesis project.

5.2. INITIALIZATION AND GEOMETRY GENERATION USING PARAPY

Having explained the general structure of the `PylonDesigner`-code, we now quickly focus our attention into the initialization of the code and the geometry generation using the `BoxBeamType`-class.

5.2.1. INITIALIZING THE PYLONDESIGNER APP

To generate the pylon structure using the PylonDesigner-app, a user has to create an instance of the `PylonDesigner` ParaPy-class. As explained, the use of this app requires the input of a specific set of input variables, the details of which can be found in section C.1 in Table C.1. Note that the exact set of input variables required to run the code depends somewhat on the pylon type and location of the engine with respect to the aircraft. For example, the exact zero-thickness geometry of the pylon at station 2 (the rear engine mount) can be defined as an input, but if engine is mounted in the traditional forward-of-the-wing position (as represented in Figure 4.13a), it can also be left open, and the coordinates of the pylon section vertices will be calculated by interpolating between the dimensions of the pylon at stations 1 and 3 (assuming that a continuous change in structural dimensions). Note though that when the engine is placed aft of the wing (as in Figure 4.13g), station 2 will be at the end of the pylon structure, and now the dimensions of station 1 (the pylon at the front engine mount) can be calculated by interpolating between stations 3 and 2. The exact locations of the hardpoints are calculated inside the `PylonDesigner`-class using dedicated attributes. At this moment, no initial design procedure is implemented to provide initial values for the generation of the structural members inside the pylon geometry, but this information has to be provided by the user based on engineering design knowledge. The implementation of an initiator to provide a first guess for in an initial design vector is left as a recommendation for future work on the method.

5.2.2. GEOMETRY GENERATION

The next step in the process is the actual generation of the 3D-geometry in a CAD terminal. To be able to do this, the mathematical parametric model defined in chapter 4 was encoded into the applications knowledge base using the standard ParaPy KBE-language appended by the ParaPy *construction*-library for beam generation. This way, the knowledge of the relations between the different components of the structure is captured in the application, which uses this information to generate the actual geometry. In ParaPy, the Python-based ParaPy KBE-language is connected with the *Open Cascade*[1] CAD-terminal for geometry generation, while the meshing is done using the *Salome SMESH* meshing platform[2]. A UML class-diagram showing the structure of the code for a case where the 3D geometry for a box-beam type pylon is created is shown in Figure 5.3. As before, a detailed version of this UML can be found in Appendix B.

INITIALIZING THE GEOMETRYGENERATOR

As explained in section 5.1, the geometry generation process starts when the `pylon_structure` part inside the `PylonDesigner`-class object is initiated. The specific class used by this part to generate the structure is selected based on the type of pylon that is defined in the inputs of the tool by the user. As explained, only the `BoxBeamType`-class is employed in this thesis project, which is a subclass of the `PylonStructure` superclass. The inputs to the system are the hardpoint definitions of stations 1 to 5 prepared in `PylonDesigner` based on the hardpoint geometry input variables, the outer mold parameters and the structural parameters of each compartment A, B and C (see Figure 4.10) of the box-beam structure (set by the optimizer when using the tool in an optimization context). In the code, the compartments are designated by the names `pylon_box1` for the part between the front and rear engine mount (compartment A), `pylon_box2` for the part between the rear engine mount and front aircraft mount (compartment B) and `box_beam` for the part between the front and rear aircraft mounts (compartment C). This is in line with the definition used by Niu for a standard, forward mounted UWN engine installation employing a box-beam installation (see Figure 2.4). The thrust links are created inside a `thrust_links`-part that is only created when the user enters the hardpoint measurements related to station 5. The exact geometry of the mounts is not parameterized in this project but instead is only implemented here for visual reasons, as it is not relevant for the further analysis of the pylon main structure and their weights contribute only a small fraction to the total weight.

[1] https://www.opencascade.com/
[2] https://docs.salome-platform.org/latest/gui/SMESH/index.html

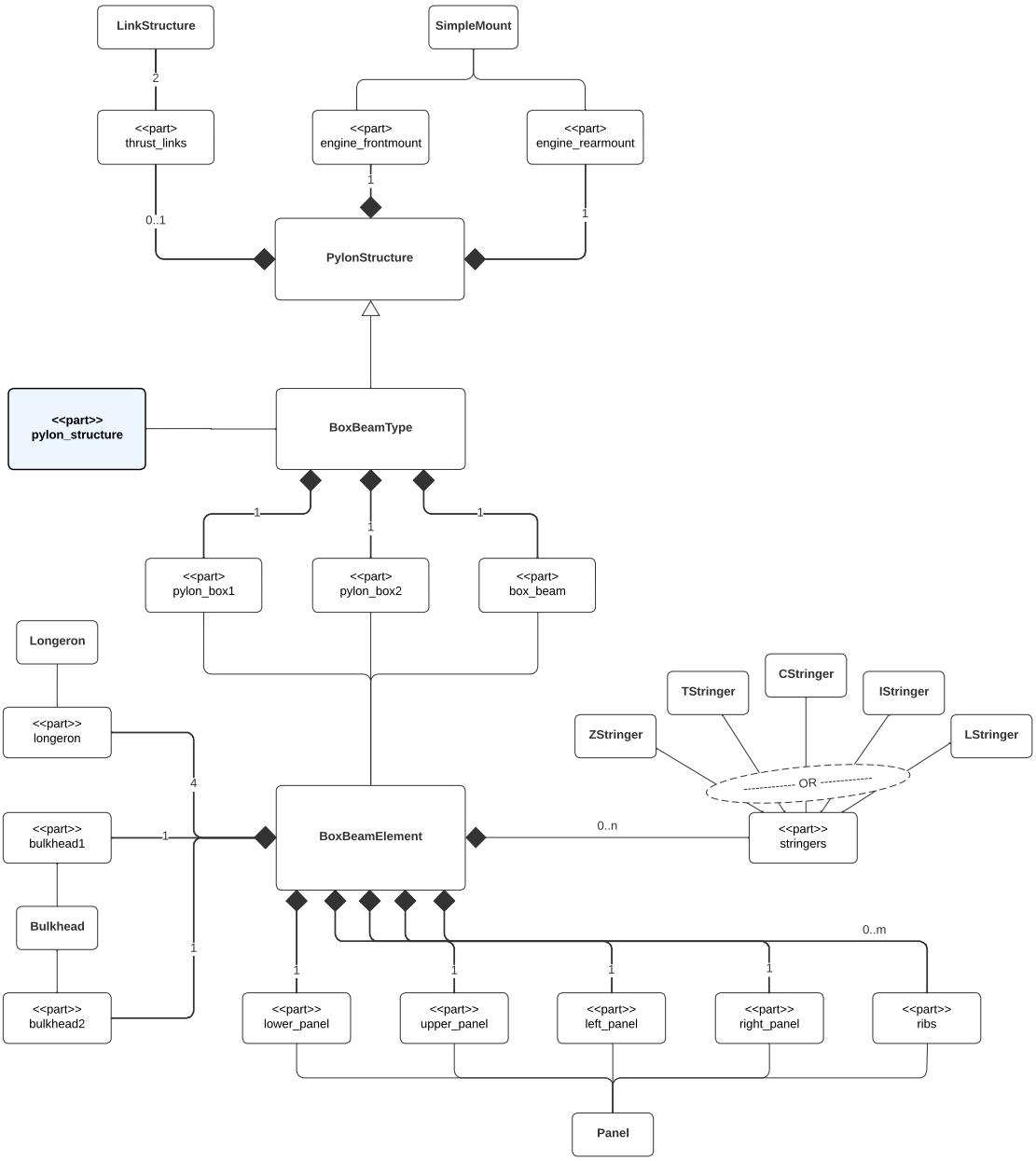
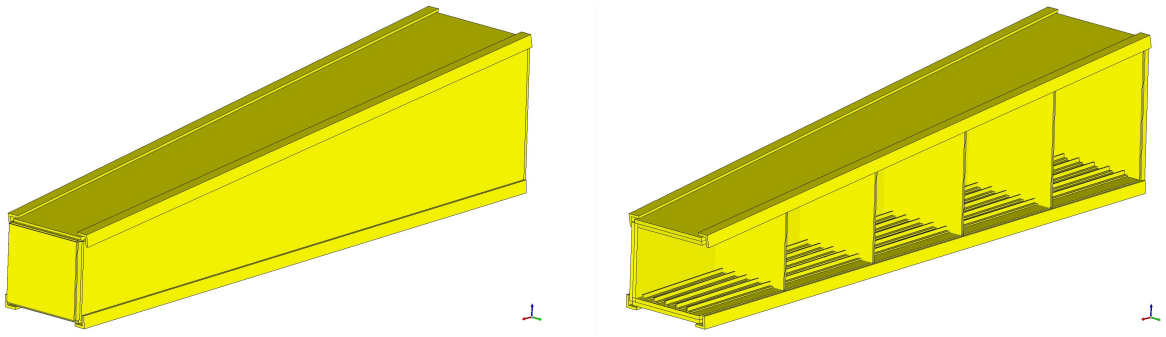


Figure 5.3: Simplified UML class-object diagram showing the most important classes and objects used when creating an instance `pylon_structure` using the `BoxBeamType`-class.



(a) Full

(b) Internal structures visible

Figure 5.4: Box-beam element geometry as generated by `BoxBeamElement`-class.

BOX-BEAM ELEMENTS

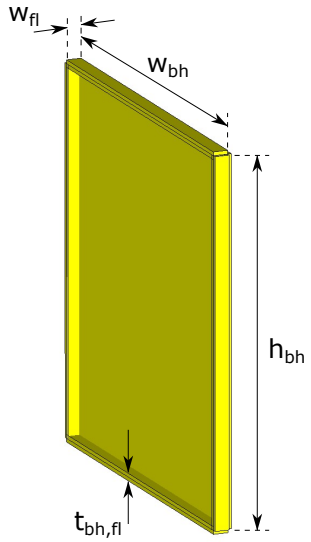
Each box-beam compartment generated by the `BoxBeamType`-class is an instance of the subclass `BoxBeamElement`. This class generates a box-beam structure including all the structural elements like spars or panels, bulkheads, longerons and stiffeners, as indicated in the class-object UML. The inputs to this class are locations of the corner points of the pylon sections at the section planes between which the compartment is bounded (see Figure 4.8a), and the structural design parameters for the respective element as defined in Table 4.4. The resulting geometry is shown in Figure 5.4a. It consists of the upper and lower spar of the element, the left and right side panels, four longerons for each longitudinal corner of the box, n stringers and m ribs, mimicking the parameterized model as defined in subsection 4.3.1 and Figure 4.12. Figure 5.4b shows the box-beam without the front bulkhead and the left panel. In this figure, the structural elements that make up the box-beam element are clearly visible. Figure 5.5 shows some of the elements in more detail.

The input vertices provided to the `BoxBeamElement`-class act as the starting point of the box-beam geometry generation. Using these vertices as inputs, the bulkheads are created with the help of a dedicated `Bulkhead`-class, the result of which is shown in Figure 5.5a. Each bulkhead consists of a rectangular shaped web, with flanges protruding to the sides perpendicular to the main web that connect the bulkhead to the rest of the structure. The thickness of the bulkhead is uniform over both the web and flanges.

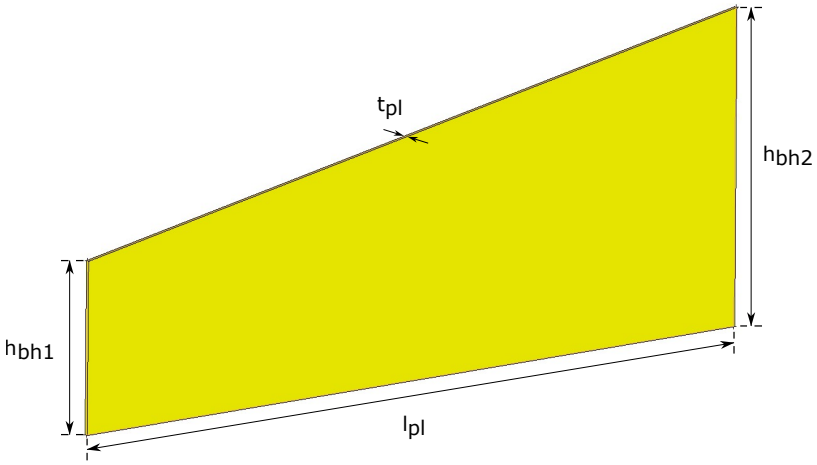
The upper-, lower- and side panels are generated using a `Panel`-class that has as inputs the corner points of the bulkhead flanges between which the respective panel is to be generated, and the thickness of the panel. The resulting structure is shown in Figure 5.5b. Ribs are generated using the same class and placed equispaced on the longitudinal axis between the front and rear bulkhead, where the spacing distance is dependent on the number of ribs required.

A predefined number of stringers is placed on the lower panel for additional buckling stiffness of the lower plate as per subsection 4.3.1. To create the stringers, the `UniformBeam`-class native to the ParaPy *construction*-library is used, which is required in the ParaPy/Abaqus API to generate 1D beam elements, as explained in the next chapter. The `UniformBeam`-class takes as inputs a `path`-variable indicating the 1D path the beam should follow (shown as a black line in the figure), a `section` definition, a `placement` variable and a `material` definition. For the `section` definition, a user can choose from any of the the native classes `CSection`, `LSection`, `TSection` or `ISection`. An additional `ZSection`-class is created that can be used here as well as this is one of the most commonly used sections used in box-beams in the aerospace industry. The definition of the control variables is as in subsection 4.3.1. An example of an instance of the lower plate with L-type stringers is shown in Figure 5.5c. In this image, only two of the five stringers for this instance are shown, while the rest is indicated by their paths using black lines. Like with the ribs, the stringers are equispaced along the plate width, with the internal spacing between stringers decided by the number of stringers used in an iteration.

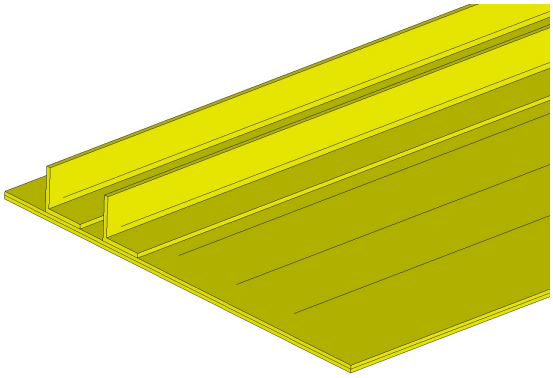
Finally the longerons are created. These are generated using a dedicated `Longeron`-class, which is essentially a wrapper for an `LStringer` definition with the box-beam edges as `path` inputs, a specific `placement` variable and an `angle` that ensures the correct orientation of the stringer for every corner of



(a) Bulkhead



(b) Side panel



(c) Lower panel with L-stringers

Figure 5.5: Detailed view of selected structural components created in the `BoxBeamElement`-class.

the box-beam.

THRUST LINKS

The thrust links are created as a separate element in the `PylonStructure`-superclass whenever a user defines the thrust link hardpoint locations parameters during the `PylonDesigner` initialization. To generate the thrust links, a special `LinkStructure`-class is created that can also be used by other link- or strut-like structures in the future. Its inputs are the points between the link should be created, the diameter of the link, the material used and the number of elements required for mesh generation. Like for the stringers and longerons in the `BoxBeamElement`-class, the `UniformBeam`-class from the ParaPy *construction*-library is used for the generation of the link, using a `Circle`-type section assignment. More on the modeling of the thrust links in the context of the structural analysis method can be found in the coming chapter on structural analysis.

FULL 3D PYLON MODEL

An example of the complete resulting pylon structure geometry generated for one specific instance of the `pylon_structure` object using the `BoxBeamType`-class for an UWN-type pylon with forward mounted engine employing thrust links is shown in Figure 5.6. In this figure, the three `BoxBeamElement`-instances are clearly visible, colored in yellow. The thrust links, generated using the `LinkStructure`-class, are shown in blue. Being an UWN- forward mounted engine configuration, the model mimics the pylon parameterization model shown in Figure B.1, with the stations ordered 1-5. The engine mounts are added here for clarifi-

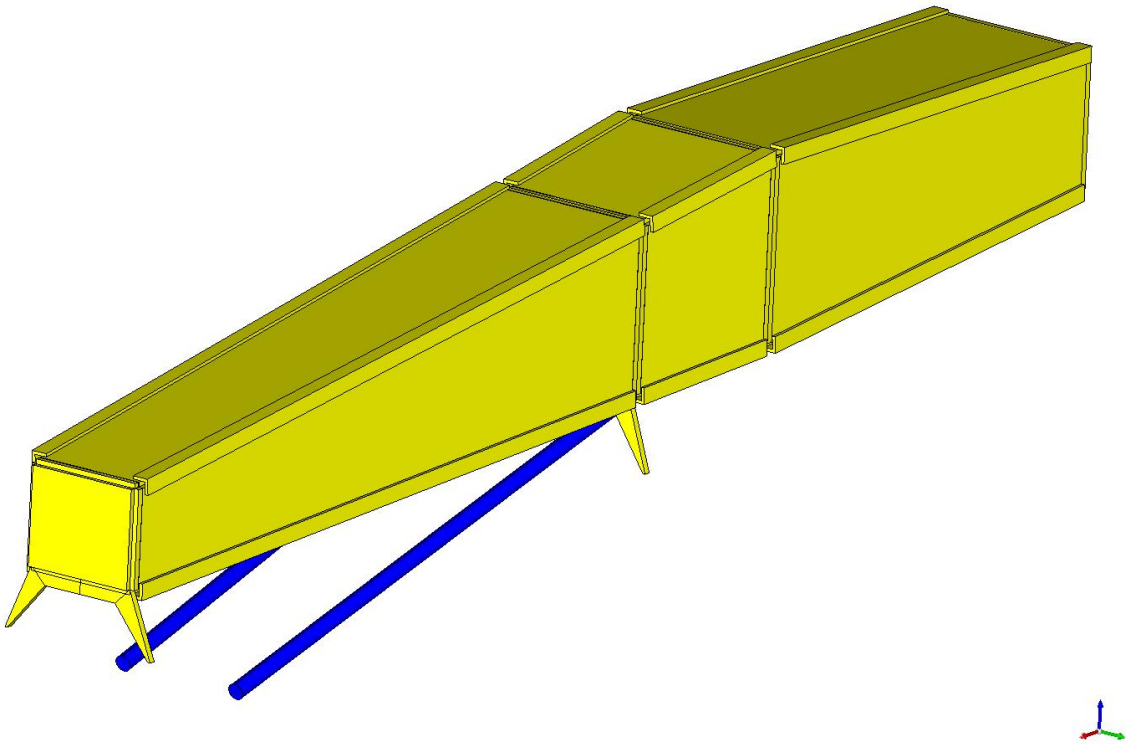


Figure 5.6: ParaPy model of the full pylon with thrust links for an UWN forward mounted engine (3D view).

cation, and will not be sized in the remainer of the text.

6

STRUCTURAL ANALYSIS USING ABAQUS

In the previous chapter, the generation of a 3D pylon structure geometry using the geometry generator code in ParaPy was discussed. The resulting geometry is a structure that is specific for one choice of values of the parameters. The next step in the process is structural analysis of the geometry to check if the structure of choice can actually hold all the relevant loads generated by the engine and airplane. To be able to perform this analysis, the geometry generation tool needs to be coupled with a structural analysis tool - in this thesis numerical analysis using the FEM-code Abaqus is employed.

6.1. PROCESS OVERVIEW

Figure 6.1 shows an overview of the most important steps that are executed by the code when performing a structural Finite Element Analysis of the pylon geometry with Abaqus. Note that the figure only shows a 'top-level' overview of the steps performed by the code. Furthermore, different parts of code are used to perform the steps, each of which will be explained into more detail in this chapter.

The process starts of with the geometry in ParaPy as generated in the previous step. Furthermore, several other input parameters are required, like the characteristics of the engine and the orientation angle, which are used to calculate the magnitudes and the directions of the forces. The geometry is then discretized, the result of which is a grid of nodes that form the basis of the Abaqus model. With the discretized geometry in ParaPy created, all the ingredients required to start the process of generating an analysis model for Abaqus are available.

To be able to run an analysis in Abaqus, two main elements are required;

- A so-called **'input file'** (extension '.inp'), a text file that contains all the necessary *model* and *history* information, including loads, boundary conditions and analysis information required to perform the analysis.

- A **job definition**, which defines the name of the job, tells Abaqus what model to use and where on the PC it can find the input file, where to put the output and log files, how many cores should be used for the analysis, if parallel computing should be used or not, etc.

First, a .inp-file needs to created in which not only the relevant data describing the ParaPy model is stored, but also the loads and load cases and the analysis steps that should be performed during the analysis, including output- and history requirements. This is where the Abaqus library of ParaPy is used, extended with a set of self-implemented functionalities to enable the use of a more extended range of Abaqus functionalities that were not implemented in the library to this moment, to create an API that translates all this information into data that is subsequently written to the .inp-file in the correct format as described by the SIMULIA Abaqus Online User Manual [27]. A short summary is given below, a detailed discussion of the process and choices made in this chain of steps can be found in section 6.3.

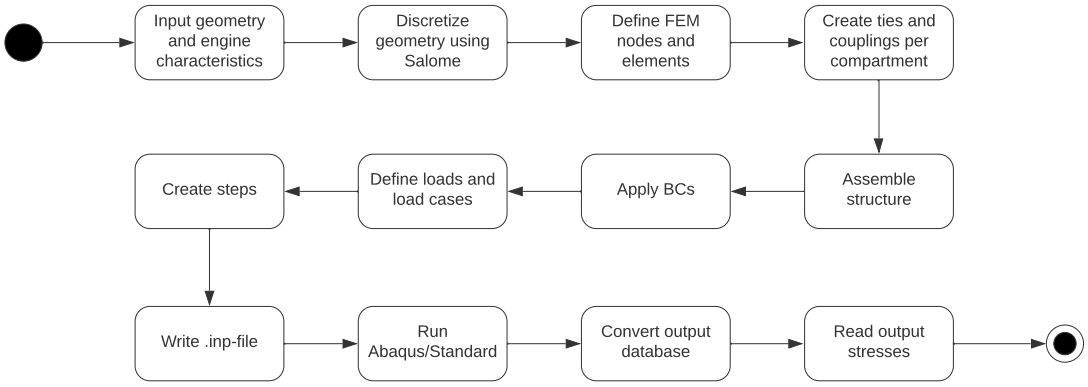


Figure 6.1: Activity diagram showing the most important steps in the structural analysis procedure in the PylonDesigner app.

The first step in this translation process is to process the discretized geometry in ParaPy into an actual mesh that can be understood by Abaqus. This processing step is done per part and per box-beam compartment. For each part, he node grid generated by ParaPy is processed and enhanced with information regarding the material and mesh element type used. Then, the relevant nodes and surfaces of different part meshes are tied together, adding information to the model on the relative relations between the different nodes and adding stiffness to the structure. Longerons are connected to the longitudinal edges of the panels, bulkheads are connected to the adjacent edges at their respective sides and the ribs are connected to the surfaces where they meet the surrounding panels. Finally, the different box-beams need to be tied together to generate the full structural geometry of the pylon.

In the next step, all couplings and boundary conditions, which define the interfaces of the structure with the engine and the aircraft, are applied. Subsequently, the loads need to be defined, which first have to be calculated from engine information like the engine weight and generated thrust (discussed in section 6.2) and applied to the correct nodes. Using these forces, load cases can be defined. Finally, the loads and load cases are combined with other step-specific properties information into a step-definition. The complete set of resulting data is written into an Abaqus input file, which defines the full problem definition for the analysis problem.

Using the input-file, an Abaqus structural analysis job can be executed with *Abaqus/Standard*, using the procedure explained in section 6.5. The resulting data is needs to be converted and post-processed to extract the required structural strength data. Finally, all acquired data is fed back to the ParaPy main terminal for further use in the structural sizing process.

6.2. LOADS AND LOAD CASES

Now that we have a broad overview of the steps and process flow in the ParaPy/Abaqus coupling and analysis procedures, let's take back a step and discuss the forces and load cases that will be applied during the analysis. After this section, the practical implementation of the load cases into the analysis steps is discussed in detail in section 6.4.

A short discussion on some of the most important load types reported in literature, which are relevant for the design of engine support structures, was given in section 2.3. Moreover, as mentioned in subsection 2.2.2, the minimum loads for which the the airplane design company is required to design the engine support structure, and for which it is required to show that the structure can hold these loads within an acceptable degree of confidence, are listed in the airworthiness regulations set by national- and transnational government bodies. For the purpose of this analysis, the structure will be subjected to a set of load cases that are derived from both the EASA CS-25 airworthiness standards [50] combined with design rules described in literature. A total of 24 load cases was derived this way.

As mentioned in the literature review, the loads that are applied to the structure can be categorized in two classes. 'Limit load cases' cover the most important important loads the airplane will see during its service life based on the flight envelope, and should be withstood without permanent deformation to the structure. And secondly 'ultimate load cases', which mainly cover the FBO load cases, and should not lead to immediate critical failure. Each are discussed in more detail in the following subsections.

6.2.1. LIMIT LOAD CASES

A total of 20 limit load cases were derived based on the regulations and design rules in literature, which are summarized in Table 6.1. Note that in the load factors applied in this table are *ultimate* load factors, meaning they incorperate a safety factor of 1.5 as prescribed in the certification standards. On top of this, one additional 'reference' load case is defined, which will act as the *base load case* for the Abaqus analysis (as will be explained in subsection 6.4.2), and which is meant to give some indication of the loads experienced by the pylon in its 'standard' operating condition. This load case is indicated using the ID tag 'CR-00'.

The limit load cases are further divided based on the operational regime into 7 *cruise* load cases, 2 *take-off* load cases, 4 *landing* load cases and 2 ground handling load cases, and completed with 4 additional *aerodynamic* load cases representing the gust loads. Note that per item **CS 25.371 - Gyroscopic loads**, the effects of gyroscopic loads are included in all load cases covering *CS 25.331 - Symmetric manoeuvring conditions*, *CS 25.341 - Gust and turbulence loads*, *CS 25.349 - Rolling conditions*, *CS 25.351 - Yaw manoeuvre conditions*, *CS 25.473 - Landing load conditions and assumptions*, *CS 25.479 - Level landing conditions*, and *CS 25.481 - Tail-down landing conditions*, with the engine at the maximum rpm appropriate to the condition. Further notice that the aerodynamic flight loads are neglected during this analysis, as they are assumed relatively small in comparison to the other loads and require an aerodynamic model of the engine-pylon assembly including the pylons aerodynamic fairing and the aircraft wing. This is regarded out of the scope of this thesis project, but can be included in subsequent work in the form of a multidisciplinary optimization study.

Load cases CR-01 to CR-07 cover the maximum loads that the engine support structure can encounter during a normal cruise flight.

- Load case **CR-01** is derived from regulation item *CS 25.361(a)(1)(ii) - Engine and auxiliary power unit torque*, and represents the combination of maximum continuous thrust and maximum load factor.
- Load cases **CR-02** and **CR-03** represent the loads when the airplane executes either a pull-up or dive maneuver. These load cases are described in the regulations under items *CS 25.331 - Symmetric manoeuvring conditions* and *CS 25.337 - Limit manoeuvring load factors*. For the loads experience by the engine mounting structure during these maneuvers, Niu [8] prescribes an inertia load factor and gyroscopic loads resulting from a ±2.25 rad/s pitching input combined with a 1.5 maximum continuous thrust- and 3.75 vertical load. Finally, the gyroscopic moments caused by the maneuver are included into the load case per *CS 25.371 - Gyroscopic loads* and as recommended by Niu [8] and Niu [11].
- Load cases **CR-04** and **CR-05** represent the loads experienced when the aircraft executes a yawing maneuver. These load cases are described in the regulations under item *CS 25.351 - Yaw manoeuvre conditions*. For the loads experience by the engine mounting structure during yaw maneuvers, Niu [8] prescribes an inertia load factor and gyroscopic loads resulting from a ±2.25 rad/s yawing input combined with a 1.5 maximum continuous thrust- and 1.5 vertical load. Finally, the gyroscopic moments caused by the maneuver are included into the load case per *CS 25.371 - Gyroscopic loads* and as recommended by Niu [8] and Niu [11].
- Load cases **CR-06** and **CR-07** represent the loads experienced when the aircraft executes a roll maneuver. These load cases are described in the regulations under item *CS 25.349 - Rolling conditions*. For the loads experience by the engine mounting structure during roll maneuvers, Niu [11] recommends applying the inertia side loads resulting from roll or a 2.5g side load on engine weight, whichever is higher. The gyroscopic moments caused by the maneuver should be included into the load case per *CS 25.371 - Gyroscopic loads* but will be considered negligible per Niu [8] as the axes of rotation of the aircraft and engine are aligned during this maneuver.

Load cases TO-01 and TO-02 cover the maximum loads that the engine support structure can encounter during take-off.

Table 6.1: Limit load cases derived for the pylon structural sizing procedure.

ID	Name	Loading condition	Source
CR-00	Steady 1g cruise	1.0 $F_{cont,max}$ + 1.0 W_{eng}	N.A.
CR-01	Max. continuous thrust at limit loads from flight condition A	1.0 $F_{cont,max}$ + 3.0 W_{eng}	CS 25.361(a)(1)(ii) Engine and auxiliary power unit torque
CR-02	Pitch maneuver + (pull-up)	2.25 rad/s pitch + 1.5 $F_{cont,max}$ + 3.75 W_{eng} + M_{gyro}	CS 25.331 Symmetric manoeuvring conditions CS 25.337 Limit manoeuvring load factors CS 25.371 Gyroscopic loads Niu [8]/Niu [11]
CR-03	Pitch maneuver - (dive)	-2.25 rad/s pitch + 1.5 $F_{cont,max}$ + 3.75 W_{eng} + M_{gyro}	CS 25.331 Symmetric manoeuvring conditions CS 25.337 Limit manoeuvring load factors CS 25.371 Gyroscopic loads Niu [8]/Niu [11]
CR-04	Yaw maneuver +	2.25 rad/s yaw + 1.5 $F_{cont,max}$ + 1.5 W_{eng} + M_{gyro}	CS 25.351 Yaw manoeuvre conditions CS 25.371 Gyroscopic loads Niu [8]/Niu [11]
CR-05	Yaw maneuver -	-2.25 rad/s yaw + 1.5 $F_{cont,max}$ + 1.5 W_{eng} + M_{gyro}	CS 25.351 Yaw manoeuvre conditions CS 25.371 Gyroscopic loads Niu [8]/Niu [11]
CR-06	Rolling maneuver +	2.25 rad/s roll or 2.5 (side) (whichever is higher)	CS 25.349 Rolling conditions CS 25.371 Gyroscopic loads Niu [11]
CR-07	Rolling maneuver -	-2.25 rad/s roll or -2.5 (side) (whichever is lower)	CS 25.349 Rolling conditions CS 25.371 Gyroscopic loads Niu [11]
TO-01	Take-off (after lift-off) 1	3.0 $F_{T/O,max}$ + $T_{T/O,max}$ + 1.5 W_{eng}	CS 25.361(a)(1)(i) Engine and auxiliary power unit torque Niu [8]
TO-02	Take-off (after lift-off) 2	3.0 $F_{T/O,max}$ + $T_{T/O,max}$ + 3.0 W_{eng}	CS 25.361(a)(1)(i) Engine and auxiliary power unit torque Niu [8]
LA-01	Landing / touch-down 1	1.0 F_{idle} + 8.0 W_{eng}	CS 25.473 Landing conditions and assumptions CS 25.371 Gyroscopic loads Niu [11]
LA-02	Landing / touch-down 2	1.0 F_{idle} - 4.0 W_{eng}	CS 25.473 Landing conditions and assumptions CS 25.371 Gyroscopic loads Niu [11]
LA-03	Landing / Reverse thrust	3.0 $F_{rev,max}$	Niu [8]
LA-04	Landing / Reverse thrust with bump	3.0 $F_{rev,max}$ + 3.0 W_{eng}	Niu [8]
GR-01	Taxiing with turn +	1.0 F_{idle} + 5.0 W_{eng} (side) + $M_{gyro,\ taxi}$	CS 25.363 Side loads on engine and auxiliary unit mounts Niu [8]
GR-02	Taxiing with turn -	1.0 F_{idle} - 5.0 W_{eng} (side) + $M_{gyro,taxi}$	CS 25.363 Side loads on engine and auxiliary unit mounts Niu [8]
AIR-1	Vertical gust +	1.5 $F_{cont,max}$ + 6.5 W_{eng}	CS 25.341(a) Gust and turbulence loads CS 25.371 Gyroscopic loads Niu [8]
AIR-2	Vertical gust -	1.0 $F_{cont,max}$ - 3.5 W_{eng}	CS 25.341(a) Gust and turbulence loads CS 25.371 Gyroscopic loads Niu [8]
AIR-3	Lateral gust +	1.0 $F_{cont,max}$ + 3.0 W_{eng} (side)	CS 25.341(a) Gust and turbulence loads CS 25.371 Gyroscopic loads Niu [8]
AIR-4	Lateral gust -	1.0 $F_{cont,max}$ - 3.0 W_{eng} (side)	CS 25.341(a) Gust and turbulence loads CS 25.371 Gyroscopic loads Niu [8]

- Load cases **TO-01** and a **TO-02** represent the loads when the airplane is performing a take-off procedure. These load cases are described in the regulations under item *CS 25.361(a)(1)(i) - Engine and auxiliary power unit torque*. For the loads experience by the engine mounting structure during these maneuvers, Niu [8] prescribes an the combined loads resulting from a 3.0 maximum take-off thrust- and either a 1.5 engine weight load (covered in TO-01) or a 3.0 engine weight load (covered in TO-02). The effects of both vertical load cases should be investigated due to the load relieving effects of different combinations of loads.

Load cases LA-01 to LA-04 cover the maximum loads that the engine support structure can encounter during the landing procedure. The loads that the aircraft should be able to carry during the landing are described in the regulations under items *CS 25.373) - Landing conditions and assumptions*, *CS 25.479 - Level landing conditions*, and *CS 25.481 - Tail-down landing conditions*, but focus in these regulations is mostly on the wing and landing gear loads. Specific loads for engine mounts during landing are not described in these regulations.

- Load cases **LA-01** and **LA-02** represent the loads when the airplane during- and right after touch-down. For the loads experience by the engine mounting structure, Niu [11] prescribes investigating the structure under the loads resulting from an $8.0g$ downwards engine weight load (covered in LA-01) and an $4.0g$ upward engine weight load (covered in LA-02). Again, as per *CS 25.371 - Gyroscopic loads*, the gyroscopic moments caused by the flapping of the wings are included into the load case.

- Load cases **LA-03** and **LA-04** represent the loads after touch-down due to reverse thrust application. For the loads experience by the engine mounting structure, Niu [8] recommends investigating both a 3.0 reverse thrust load and a combined 3.0 reverse thrust and a $3.0g$ engine weight load.

Load cases GR-01 and GR-02 cover the maximum loads that the engine support structure can encounter during ground handling / taxiing. The loads which the engine mounting structure should be able to hold are not specified specifically in the regulations, but as explained in section 2.3 the loads experienced can become quite high.

- Load cases **GR-01** and **GR-02** represent the loads when the airplane makes a turn during taxiing. Niu [8] recommends applying $5.0/-5.0$ side loads for these load cases. Furthermore, since the turning rates of aircraft can be quite high during a classical 90 degree runway entry turn, the gyroscopic loads can not be neglected. For most airports, a maximum turning speed of 10 knots (about 18.5 km/h) is prescribed (in line with ICAO standards) [83], with a standard runway entry corner of 40 meters radius. This leads to the turn rate that is applied during this maneuver of $\omega_{turn} = 5.14/40 = 0.1285\ \mathrm{rad\,s^{-1}}$.

Load cases AIR-01 to AIR-04 cover the maximum loads that the engine support structure can encounter as a result of a (static) gust.

- Load cases **AIR-01** and **AIR-02** represent the effects of vertical positive and negative gusts. For these load cases, Niu [8] recommends applying a $6.5g$ downwards engine weight load (covered in AIR-01) and a $3.5g$ upwards engine weight load (covered in AIR-02). The gyroscopic moments caused by the vertical gusts are included into the load case per *CS 25.371 - Gyroscopic loads*.

- Finally load cases **AIR-03** and **AIR-04** represent the effects of lateral positive and negative gusts. For these load cases, Niu [8] recommends applying $3.0/-3.0$ side loads. Again, the gyroscopic moments caused by the lateral gusts are included into the load case per *CS 25.371 - Gyroscopic loads*.

6.2.2. FAN BLADE-OFF LOAD CASES

For the ultimate load cases, the dominant loads that the pylon should be designed for are developed during the occurrence of a fan blade-off event, as was explained in subsection 2.3.3. The loads for which the engine mounts and supporting structures should be designed for are explained as follows in section CS 25.362 of the certification specifications [50]:

"*For engine mounts, pylons and adjacent supporting airframe structure, an ultimate loading condition must be considered that combines 1g flight loads with the most critical transient dynamic loads and vibrations, as determined by dynamic analysis, resulting from failure of a blade, shaft, bearing or bearing support, or bird strike event. Any permanent deformation from these ultimate load conditions should not prevent continued*

safe flight and landing."

As was explained in subsection 2.3.3, dynamic analysis of an engine-pylon sub-assembly during an FBO-event is an extensive task that requires close cooperation between the engine manufacturer and the airframe manufacturer. It involves several steps that requires detailed knowledge of the engine under consideration in the form of a high-fidelity numerical analysis model of the engine which is usually only available to the engine manufacturer. The loads calculated by the engine manufacturer are then communicated to the airframe manufacturer for further analysis, which still requires a (coarse) numerical model of the engine, and a full rotordynamics analysis of the behavior of the engine and pylon assembly between the period directly after the FBO event up until windmilling speed is reached.

This poses several problems in the context if this thesis project. First of all, the development of a full rotordynamics model of the engine is an elaborate task that is clearly out of the scope of this thesis project, and a fully finished rotordynamics model that can be built into this analysis is not readily available. Moreover, the required initial values of the forces which are usually calculated by the engine manufacturer are also not available. Unfortunately, this means that a reliable implementation of all the forces and moments generated during the FBO event is deemed unrealistic. This is therefore left as a clear recommendation for future work. This being said, by applying Equation 2.9 it is actually possible to calculate one component of the FBO loads, namely the unbalance load in the rotor right after the release of the fan blade, assuming that the mass of the missing blade, the radius of the fan and the rotational velocity of the engine are known. This was done by Armendáriz *et al.* [59][15][60] in their work looking at the effects of the FBO event for a turboprop engine that does not have a casing, so impact loads and seizure loads could be neglected.

$$F_{unblc}(t) = mr\omega^2(t) \tag{2.9}$$

Looking at data provided by Heidari *et al.* [14] (see Figure 2.13) and Bettebghor *et al.* [26], is is clear that this load is highest during the first revolution after the blade is released, which makes sense as the rotational velocity is still at its maximum value. The resulting load can be applied for one full revolution to obtain an idea of the loads experienced by the pylon during the FBO event.

A second problem that requires attention, especially in the context of this thesis project, is the difficulty of integrating a nonlinear dynamic load case into a static linear response structural optimization schema as the one performed here. Note that even with the simplification made in this thesis that only the unbalance loads are taken into account as explained above, this still requires a dynamic simulation step to simulate the full revolution of the (broken) fan. Bettebghor *et al.* [26] describes the problems as being two-fold: first of all, sensitivity analysis on a nonlinear analysis mechanics response is difficult, and secondly the definition of a continuous optimization constraint can pose problems. And finally, not mentioned by Bettebghor *et al.* but equally important in the context of this study, dynamic analysis usually require significant computational resources to solve, which can significantly increase the computational time of the analysis to levels that far surpass those acceptable for a preliminary design method. To solve this problem, Bettebghor *et al.* proposes the use of *equivalent static load cases* that represent the maximum loads the pylon experiences at pre-specified moments in the dynamic process. This method has been applied and proven in several research projects, notably Cho and Choi [84], Kang *et al.* [85], Kim and Park [86], and Park [87]. In the case of the research by Bettebghor *et al.*, in which the researchers did have a full rotordynamics model of the engine to their disposal, four rotordynamic

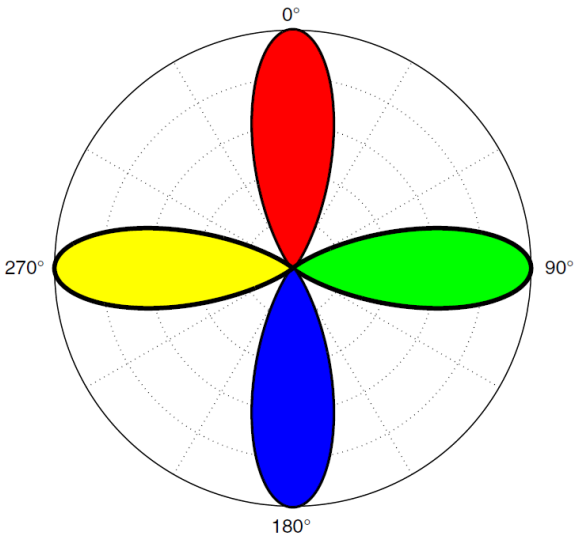


Figure 6.2: Blade release angles used to define equivalent static loads cases for FBO as proposed by Bettebghor *et al.* [26].

Table 6.2: Ultimate load cases derived for the pylon structural sizing procedure.

ID	Name	Loading condition	Source
FBO-01	Fan blade-off (FBO) at release angle 0°	$F_{cont,max} + W_{eng} + F_{unblc}$	CS 25.362 Engine failure loads CS 25.367 Unsymmetric loads due to engine failure Niu [8]
FBO-02	Fan blade-off (FBO) at release angle 90°	$F_{cont,max} + W_{eng} + F_{unblc}$	CS 25.362 Engine failure loads CS 25.367 Unsymmetric loads due to engine failure Niu [8]
FBO-03	Fan blade-off (FBO) at release angle 180°	$F_{cont,max} + W_{eng} + F_{unblc}$	CS 25.362 Engine failure loads CS 25.367 Unsymmetric loads due to engine failure Niu [8]
FBO-04	Fan blade-off (FBO) at release angle 270°	$F_{cont,max} + W_{eng} + F_{unblc}$	CS 25.362 Engine failure loads CS 25.367 Unsymmetric loads due to engine failure Niu [8]

analyses are performed for blade release angles at 0°, 90°, 180° and 270° (as shown in Figure 6.2), after which the researchers monitored the maximum and minimum forces experienced by each of the 11 DoFs defined in their analysis, leading to a total of 88 loadcases. Even though the execution of a rotordynamics simulation is unfortunately not possible in this project as mentioned, a similar approach is applied in this project in which the static response of the engine on the application of a maximum unbalance force at four each of the four blade release angles in combination with the maximum continuous thrust and the 1g weight of the engine is investigated, which forms the worst case scenario. This leads to four ultimate equivalent static load cases that will be applied during the analysis, which were named **FBO-01**, **FBO-02**, **FBO-03**, and **FBO-04** for blade release angles of 0, 90, 180 and 270 degrees respectively, as summarized in Table 6.2.

6.2.3. FATIGUE LOADS

Finally, in subsection 2.3.4, it was mentioned that several authors found fatigue to be a sizing load case for some specific components making up the box-beam construction in their projects. Mainly the flanges of the pylon and the corners of the box-beam where mentioned as prone to fatigue. However, with the limited time and resources available for this project in mind, it was decided that performing an extensive fatigue analysis on the structure is not required here as this is a relatively intensive task that is more suitable for the detailed design phase than for conceptual and preliminary design for which this tool is developed. Moreover, good load spectra suitable for analysis of engines and their support structures are currently not available, making analysis difficult. Future research might incorporate load cases that cover fatigue loads, if proper load spectra are found and analysis of the structure with these loads is deemed necessary.

6.3. TRANSLATING PARAPY GEOMETRY TO ABAQUS MODEL

With the modeling of the forces discussed and the relevant load cases identified, we now have all the information we need to discuss the implementation of the ParaPy/Abaqus API. Figure 6.3 shows a detailed flow diagram of the steps performed in the code. In this figure, rectangular blocks show the processes performed in a step, while data is shown using a *document*-symbol with a 'wavy' lower side. As indicated in the figure and discussed in the first paragraph of this chapter, the translation of the geometry into an input file that can be used by Abaqus is a process that requires several steps, which are executed in different scripts by code employing classes and functions from the ParaPy/Abaqus library as well as the ParaPy/Meshing library. As explained before, the main functionality and structure of the classes in these libraries was created by ParaPy, but unfortunately the functionality of the ParaPy/Abaqus library was quite limited at the time of delivery as it was tailor made for one specific use case, which meant many of the functions used in this thesis project had to be added to the library as part of this thesis project.

The result of this part of the Abaqus analysis procedure is an Abaqus *input file*, which as indicated before is one of the primary files needed for the Abaqus solver code and defines the entire problem that is to be solved. To write this code, a `Writer`-class is provided in the ParaPy/Abaqus library, which requires two data sets in order to write the .inp-file. First of all, an `<adaptor>`-object has to be provided, in which all *model-related information*, like nodes and elements but also boundary conditions, should be defined. To create this

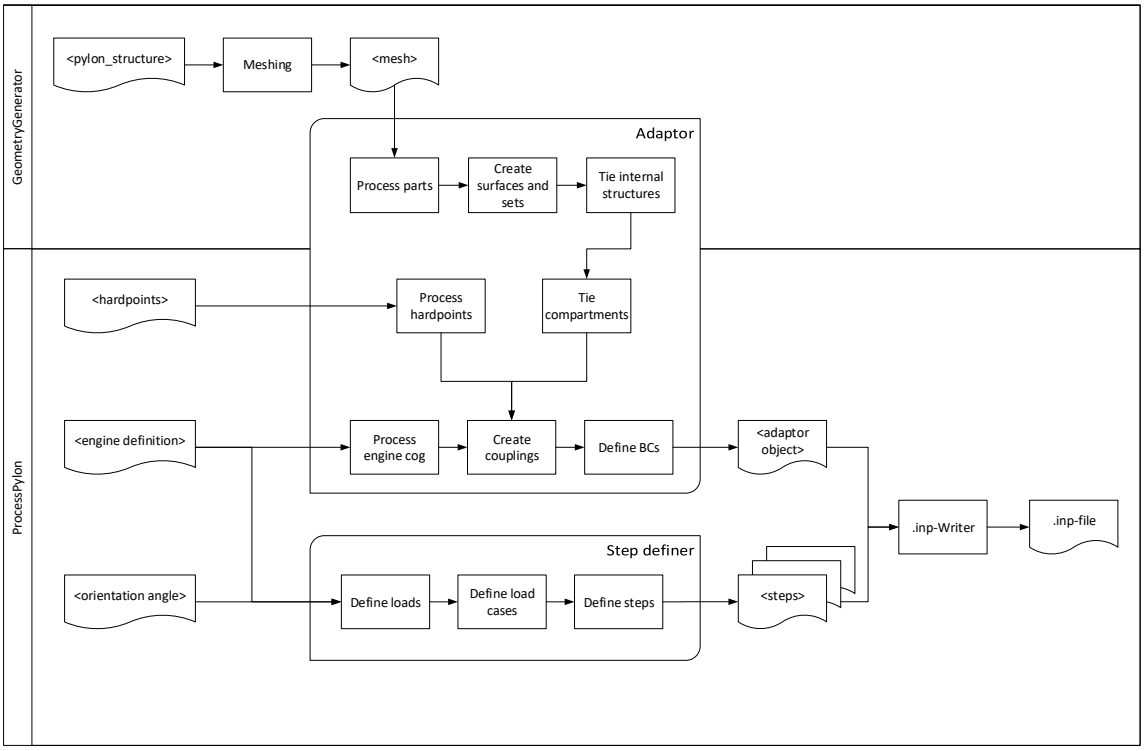


Figure 6.3: Process diagram showing the translation of the ParaPy geometry and engine characteristics into an input file that can be used by Abaqus for structural analysis.

<adaptor>-object, the ParaPy/Abaqus library provides an Adaptor-class with functions to import and process different types of model data. The second data-structure that has to be fed into Writer-class is a list of <step>-objects, which hold information about the analysis that is to be performed, the so-called *history data*. This covers the definition of the loads and loads cases that have to be applied in a specific analysis step, but also the required analysis type or solver procedure that is to be used in the step, the required output data, convergence criteria, etc. For this, a Step-class can be used, which in turn requires several separate data objects containing the step-specific data and properties.

As shown in Figure 6.3, only part of the information generated during this translation process is generated inside the ProcessPylon-class, which is used by the abaqus_input_file-attribute in the main PylonDesigner-class to create the .inp-file. This was also indicated in Figure 5.2 by the dashed arrow and the word *'use'*, indicating a dependency relation between the two elements. Both the meshing and part-level Adaptor-processes are performed inside the classes used in the GeometryGenerator part of the code, and imported into the ProcessPylon for further processing. The processing of the hardpoints, engine definition and orientation angle is performed in the ProcessPylon-class, which combines the final <adaptor>-object with the <step>-objects list to write the final .inp-file as explained.

6.3.1. PART DISCRETIZATION

As depicted in Figure 6.1, the first step in the process of transforming the geometry generated in ParaPy into a format that can be handled and analyzed by Abaqus, or the first step in any FEM analysis whatsoever, is the discretization of the geometry into nodes or 'finite elements' that can be used by the FEM-code to perform the required calculations.

Processing the mesh into a format that can be imported to Abaqus essentially consists of two steps. First, the discretization of the structure, which is performed inside the GeometryGenerator part of the code using the Salome SMESH meshing integration package in ParaPy. Here, the 3D geometry created in the GeometryGenerator is translated into a set of nodes in a pre-specified grid. And secondly element definition and assigning, which is done when the nodes are processed in the Adaptor during the 'part processing' step. Specialized classes were created in ParaPy to discretize each type of structural element in the box-beam. As such, the UML class-object diagram in Figure 5.3 can be extended with a 'meshing'-class and mesh-parts for each structural element.

Given that the tool developed in this thesis is meant to be used in the conceptual design phase of the aircraft design cycle, and that the structural analysis performed here is part of an optimization routine, keeping the complexity and thus the number of DoF's to a minimum is of vital importance. For that reason, the dimension of the mesh elements should be kept as low as possible, in accordance with what is suggested in literature on the Finite Element Method. Shell-like elements in the structure will be modeled exclusively using 2D elements with their thickness t assigned during the part-processing phase, while beam-elements are meshed using 1D elements, again with a section that is assigned automatically when processing the part in the Adaptor. Furthermore, the mesh density should vary such that it is as low as possible at locations where the stress variations are less, while being higher at locations where stresses concentrate (peak stresses) or the stress gradients are high. This usually is the case at locations where the geometry is non-uniform - for example if there is a hole in the structure, or a corner - or whenever there is an interface with another part.

The 'thin-walled', plate-like elements in the structure, meaning the upper, lower, and side panels and the ribs, but also the bulkheads, are meshed using 2D meshes employing a structured grid of quadrilateral elements. For the side panels, upper- and bottom spar, seeds are placed using cosine spacing along the longitudinal edges towards the plate interfaces with the bulkheads, and constant spacing along the lateral edges. The number of elements is controlled by varying the seed count on the lateral edge, with the number of seeds on the longitudinal edge chosen such that the aspect ratio of the resulting elements remains below 3, as recommended by [88], using the length-to-width ratio of the box under consideration. An example of a resulting mesh for a side panel is shown in Figure 6.4a.

In the case of the bulkheads, the seeds are placed on the edges of the bulkhead web and the flanges. The mesh density of the bulkheads in higher than that of the panels, as this is where the loads are introduced into the structure, meaning high stress peaks and gradients are expected. Figure 6.4b shows the resulting mesh

Table 6.3: Abaqus element types applied during part-processing and their characteristics.

Element name	Dimensionality	Element type	No. nodes	DoFs	Interpolation
B31	1D	Beam	2	6	Linear, 1st order
S4R	2D	Continuum shell	4	6	Linear, 1st order

for a flange density equal to the lateral mesh density of the panels, while the web density is chosen as 3 times the lateral density of the panels. The rib mesh density is set equal to the lateral mesh density of the panels, using equidistant spacing.

For the 1D stringers and longerons, mesh density is set equal to $n_{panel,long} + 2$, to ensure that the density of the beams is slightly higher than the longitudinal mesh density of the panels, which is required to tie the structure together using tie constraints in Abaqus [27]. Finally for the thrust-links, again a fixed number of seeds is used with equispacing to keep the model as simple as possible. Resulting meshes for the longerons, stringers and ribs are shown in Figure 6.4c.

6.3.2. PROCESSING PARTS

The next step in the translation process is the processing of the generated discretized parts in ParaPy using the `Adaptor`-class. As indicated in Figure 6.3, this step is also performed in the GeometryGenerator.

PROCESSING THE NODES AND DEFINING ELEMENTS

To completely process the parts making up the pylon structure into the Abaqus input-file, specialized functions exist (or are created) inside the `Adaptor`-class for each type of object that needs to be processed. This can either be a `1d_part`, a `2d_part`, or a `3d_part`, a `beam`-part or a `node`-part. The required information for each type of part differs slightly, but in all cases at least the `part`-object and it's related `mesh`-object needs to be provided.

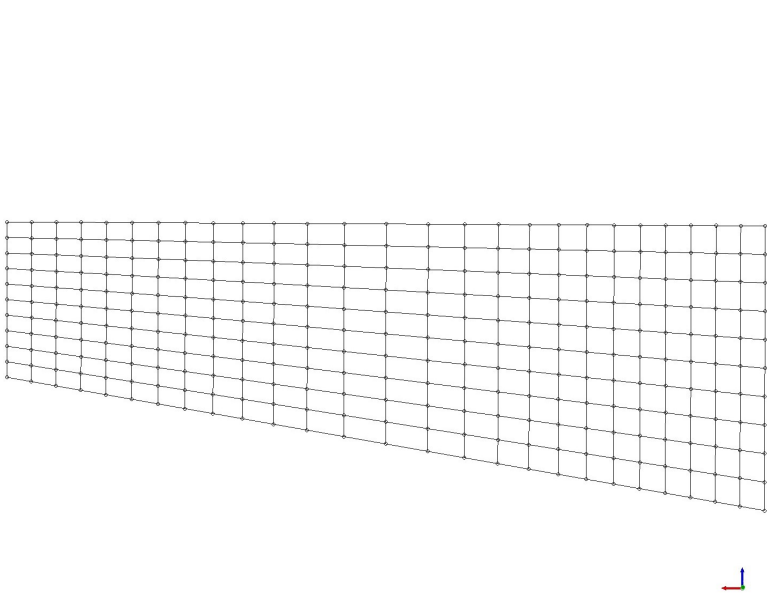
When creating a part in Abaqus/CAE, a geometry is drawn inside the GUI or imported using a .step-file from a different CAD-software, after which the user can assign properties and 'sections' to the geometry to define the structural properties of the part. Finally a mesh is created using the 'meshing' toolbox. The end result contains the nodes in the specified grid with assigned elements, section- and material properties. All this data is processed in the `Adaptor` using different data structures, using decorators like the `@property`-decorator or the `@dataclass`-decorator.

Using the mesh created in ParaPy, the nodes in Abaqus are defined. Then, the nodes that represent a part are grouped and assigned an 'element' type, which defines the characteristics of the grid. The element types used in this project and their characteristics are summarized in Table 6.3. For the 2D parts, 'S4R'-type elements are used, which are 4-node general-purpose shell elements employing reduced integration with hourglass control and finite membrane strains. A first-order linear interpolation method is used to calculate the stiffness- and mass matrices of each element. The thickness and material properties of each part are defined using a specific `ShellSectionProperties`-definition.

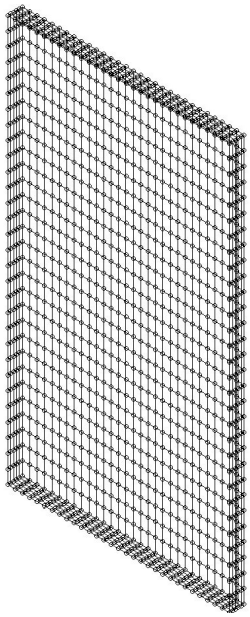
For the longerons, the stringers and the thrust links, 'B31' Timoshenko beam elements are used, with a beam cross-section and orientation defined during part creation using the ParaPy/Construction-library as explained in subsection 5.2.2. As indicated in the table, these elements have 6 DoFs — 3 displacements in x-, y- and z-directions and 3 rotations around the x-, y- and z-axis. As with the shell elements, first-order linear interpolation is used to calculate the stiffness- and mass matrices during the analysis.

MATERIAL DEFINITION

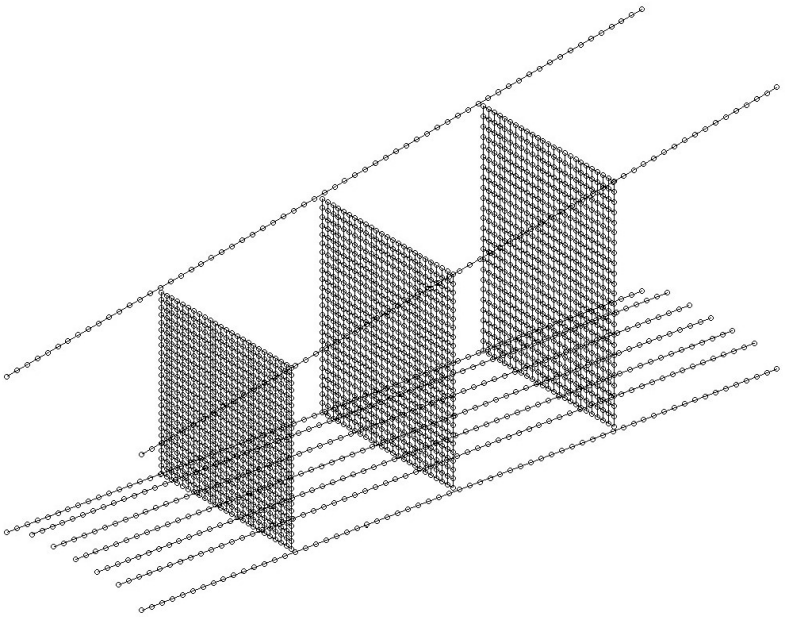
If we look at scientific literature regarding the design of pylons and struts, it is clear that nearly all structures employ one of two materials; either the pylon is made of aluminum, or titanium, or combinations of the two [8][89]. If the materials are combined, usually this means that titanium is used in the parts of the structure that carry the heaviest loads, like the bulkheads and mounts, or those parts that need to be extremely heat resistant, like firewalls, and aluminum is used for the remaining parts. In some cases, the lower web of the box-beam part between stations 1 and 2 acts as a firewall, which means this part can also be made of titanium. That being said, most examples found in literature use both fully aluminum and fully titanium structures (i.e.



(a) Side panel



(b) Bulkhead



(c) Ribs, longerons and stringers

Figure 6.4: 1D- and 2D-meshes for the structural elements making up the box-beam structure.

Table 6.4: Available materials for the purpose of this thesis and their properties [33].

	Aluminum 2024-T3	**Aluminum 7075-T6**	**Titanium Grade 2**	**Titanium Grade 3**	**Titanium Grade 4**	**Titanium 6Al-4V (Grade 5)**
Composition	Al: 94.7% Cu: 4.9% Mg: 1.8%	Al: 91.4% Zn: 5.1% Mg: 2.1%	Ti: 100%	Ti: 100%	Ti: 100%	Ti: 88% Al: 5.5% V: 3.5%
Density [kg/m3]	2,780	2,830	4,520	4,520	4,550	4,410
Young's Modulus [GPa]	75.7	76	105	105	120	110
Shear modulus [GPa]	29.4	28	39	39	45	40
Poisson's ratio [-]	0.343	0.335	0.37	0.37	0.37	0.31
Yield Strength [MPa]	372	530	360	470	570	786
Tensile strength [MPa]	510	580	490	610	690	860
Max. elongation [-]	0.15	0.1	0.26	0.27	0.25	0.1
Fatigue strength ($N = 10^7$ cycles) [MPa]	168	168	296	356	408	327

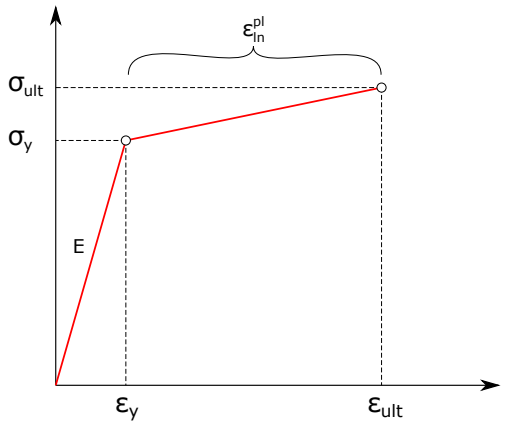


Figure 6.5: True stress-strain curve as employed in the bi-linear material model

[26]), for example the pylon of the Airbus A320 which is fully made of titanium.

For the purpose of this thesis, six material definitions representing frequently employed aluminum- and titanium alloys are implemented into the application and available for application. In case of the aluminum alloys, both the 2024-T3 and 7075-T6 alloys are implemented. Regarding the titanium variants, Grade 2, 3 and 4 pure titanium (used in several heavily loaded structural components like the mounts) is implemented, as well as the Ti-6Al-4V alloy, which is the most used alloys in titanium aircraft pylon structures [90][91][92]. The characteristics of the materials can be found in Table 6.4. Note that the yield stress, tensile stress and elongation presented in the table are engineering values, while Abaqus requires true stresses and logarithmic strain values. Therefore they need to be transformed to true values using the well-known equations as described in standard textbooks on strength of materials and the Abaqus Online User Manual [27] (Equation 6.1-6.2).

$$\sigma_{true} = \sigma_{eng}\,(1+\varepsilon_{eng}) \tag{6.1}$$

$$\varepsilon^{pl}_{ln,true} = \ln(1+\varepsilon_{eng}) - \frac{\sigma_{true}}{E} \tag{6.2}$$

All materials are modeled as elastoplastic since in the FBO loads analysis the failure criterion is based on ultimate loads, allowing for permanent (so plastic) deformation. Yet for simplicity reasons the materials are defined as being bi-linear and isotropic; in other words, both the yield as well as the hardening phases are assumed to be linear, as shown in Figure 6.5. And the strain in the material will be equally divided over the material, with no preferred strain direction.

Table 6.5: Surface types created for one box-beam element

Component	Mesh type	Input grids	Surface type	Orientation
Panels and spars surfaces	2D surface	'web'	Element-based	SPOS, SNEG
Panel and spars edges	Face edge	'web', 'edge'	Element-based	-
Stringers	1D line	'beam path'	Node-based	-
Longerons	1D line	'beam path'	Node-based	-
Bulkhead webs	2D surface	'web'	Element-based	SPOS
Bulkhead flanges	2D surface	'flange'	Element-based	SPOS, SNEG
Bulhead edges	Face edge	'flange', 'edge'	Element-based	-
Ribs edges	Face edge	'web', 'edge'	Element-based	-

6.3.3. TIEING THE INTERNAL STRUCTURES

With the part meshes processed, the next step is to implement kinematic relations between different parts in order to transform the 'loose' parts — which at this point are more or less floating in space without a context — into an actual structural assembly that works as one to redistribute the external forces across all components. In this thesis project, two types of kinematic constraints are implemented to couple the nodes of the structure to each other:

- **Surface-based tie constraints.** Used to tie two predefined surfaces and/or node sets together. By implementing the tie constraints, the translational and rotational DoFs of the nodes on the two surfaces are equated. This type of constraint is used to tie the structural components of one box-beam together, as well as to tie the box-beams compartments to each other during the next step of the process to create the full pylon assembly.
- **Surface-based coupling constraints.** Used to constrain a group of nodes located on a surface to one specific reference node or group of nodes. In this thesis, coupling constraints are used to couple relevant parts of the structure to the engine and aircraft hardpoints, replacing the sophisticated systems of links between the engine and pylon fittings at the forward- and aft engine-to-pylon and pylon-to-aircraft mounts.

With the focus at this point in the discussion still at the box-beam compartment level, the tie constraints are discussed here first. As explained, tie constraints are constraints that 'tie' together two surfaces that should be defined beforehand. This is visually represented in Figure 6.3 using two process blocks named 'Create surfaces and sets' and 'Tie internal structures'.

CREATING SURFACES

It is important to note that the term 'surface' in Abaqus is not solely reserved for 2D elements like faces, but can also refer to a 1D edge or even element (ends). In general the term 'surface' refers to a group of nodes that share a (geometric) context. Each surface has an associated area and orientation (positive or negative for 2D surfaces). Abaqus surfaces can be defined from elements, nodes or even using analytical relationships.

For each tie, two surfaces must be defined that represents the element- or node sets that need to be tied together. For example, to tie the stringers to the lower spar, first one 'lower spar surface' and then for each stringer a 'stringer surface' is created, which are then to be tied together using one tie constraint object per stringer. Similarly, to tie a longeron to the panels, one 'surface' needs to be created that represents the edge of the panel to which the longeron should be tied, and one 'surface' representing the longeron under consideration. To create the surfaces, the ParaPy/Abaqus library provides specialized functions for element-based surfaces, node-based surfaces and 'edge of plate'-based surfaces. An overview of the surface types created in this project to tie different parts of the structures together is given in Table 6.5.

TIEING SURFACES

The ties are created using a `process_tie()`-function in the `Adaptor`-class. The inputs of this function are the two Abaqus surfaces that need to be tied together, one of which is defined as the 'master/main' surface while the other is the 'slave/secondary' surface. The choice of main- and secondary surface depends on the surfaces tied together. In general, the Abaqus Online User Manual prescribes that the main surface should be the one that has the coarser mesh to obtain the highest accuracy. Schematic depictions of the ties created in

Abaqus for a box-beam element are depicted in Figure 6.6. The final result as visualized in Abaqus/CAE for a box-beam with three ribs and C-stringers is shown in Figure 6.7, where the ties are visualised using yellow circles.

For the element-based surfaces (depicted in red in Figure 6.6), the *'Surface-to-surface'* formulation is chosen to calculate the tie interpolation coefficients, which has optimized stress accuracy and averages the constraints across the elements rather than enforcing it at discrete points which is done in the 'node-to-surface' approach. An example of this type of tie is the constraint that ties the four longitudinal panels making up the box-beam to each other, shown in Figure 6.6a. Bulkheads are tied to the panels by tieing each edge of the flanges (which is represented by an element-based surface) to the respective complementary edge on the panel that lies adjacent to it (again represented by an element-based surface), as shown in Figure 6.6c.

To tie the stringers to the lower spar, as well as in the constraints that tie the longerons to the construction, the more traditional and less expensive *'Node-to-surface'* formulation is chosen (as prescribed for node-based surfaces), represented by blue dotted lines in Figure 6.6a and Figure 6.6b. More on this can be found in the Abaqus Online User Manual [27]. For the stringer ties, the element-based surface representing the lower spar is used as a main surface, while the node-based surface representing the beam is used as a secondary surface. The same is done for the longerons, where the element-based surface representing the panel edges are tied to the node-based surfaces representing the longerons. Finally a position tolerance is implemented that takes the thickness of the shells as well as the profile thicknesses of the beams into account to ensure that all nodes on the stringers and longerons are coupled to their closest node on the main surfaces.

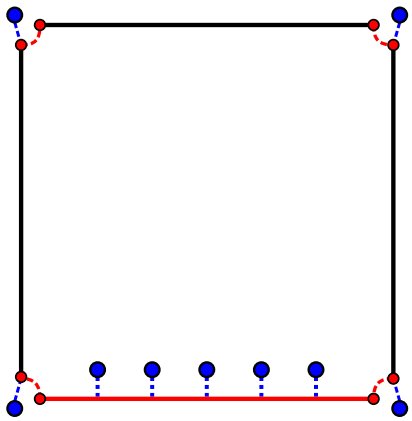
6.3.4. ASSEMBLING THE FULL PYLON STRUCTURE

Now that the compartment-based operations are performed, each processed box-beam compartment is imported into the `<adaptor>`-object created inside the `ProcessPylon`-class for further assembly-level processing. During this step, the box-beam compartments (and if needed the thrust links) need to be tied together, all other model information needs to be imported and processed and finally the resulting `<adaptor>`-object and `<step>`-objects need to be fed into the `Writer`-class to write the final .inp-file. The engine definition (with in it the engine performance characteristics and thermodynamic data, the masses and the cog locations) and orientation angle of the pylon are retrieved from the `ProcessPylon` input variables. The exact locations of the hardpoints where the pylon structure is connected to the engine and the aircraft are calculated in the GeometryGenetator in the `pylon_structure`-object from the hardpoint location parameters. Using this information, all information needed to complete the definition of the `<adaptor>`-object is available.

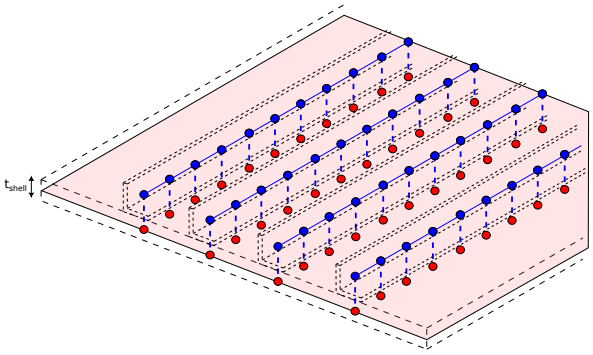
First, the hardpoints are processed using the `process_node()`-function and added to the geometry definition as *reference point* feature. To complete the geometry definition, the box-beam elements, and if implemented the thrust links, are tied together to form the full structural assembly. As previously, this involves firstly the creation of surfaces that can then be tied using the same `process_tie()`-function, but this time this is done inside the main `ProcessPylon`-class inside the `abaqus_adaptor` attribute that generates the `<adaptor>`-object. To tie the box-beam compartments together, ties are created between the bulkheads of adjacent box-beam elements, essentially transforming the bulkheads into I-flanges instead of C-flanges as they where before. As input surfaces, 'bulkhead web'-surfaces are created, as shown in Table 6.5. These are element-based surfaces, which can be used in a surface-based tie constraint between the 'outside' surfaces of the bulkheads, as it was found during the implementation of such surfaces helps in smoothing out the forces across the bulkheads better and lead to lower stress peaks.

To tie the thrust links to the rest if the structure, if available, again tie constraints are used. This time, the half surface of the lower flange is used as main surface, with the end node of the equivalent thrust link beam mesh used as secondary surface. To ensure that the tie constraint is enforced, the end of the thrust link is placed such that it is at the middle of the flange width in order to not interfere with the tie constraints enforced at the edges of the bulkhead to tie the bulkhead to the lower panel or secondary box-beam element, respectively. The resulting tie constraint is shown in Figure 6.8.

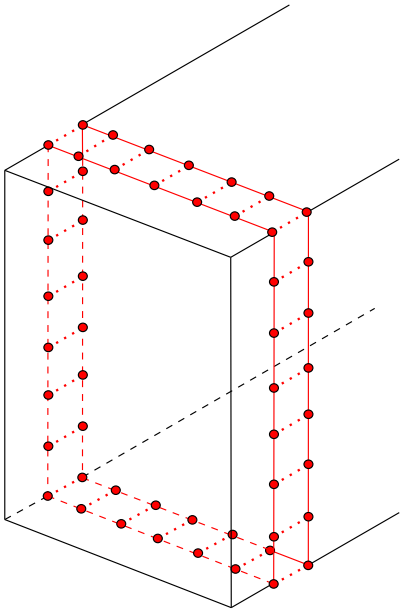
With these ties enforced, the full structural assembly is defined, save for the interactions with the outside



(a) front view with side panel-to-side panel and longeron-to-side panel ties



(b) stringers to lower spar ties



(c) bulkhead to sides ties

Figure 6.6: Schematic overview of tie constraints in a beam-box element. 'Node-to-surface' ties depicted in blue, 'surface-to-surface' ties depicted in red.

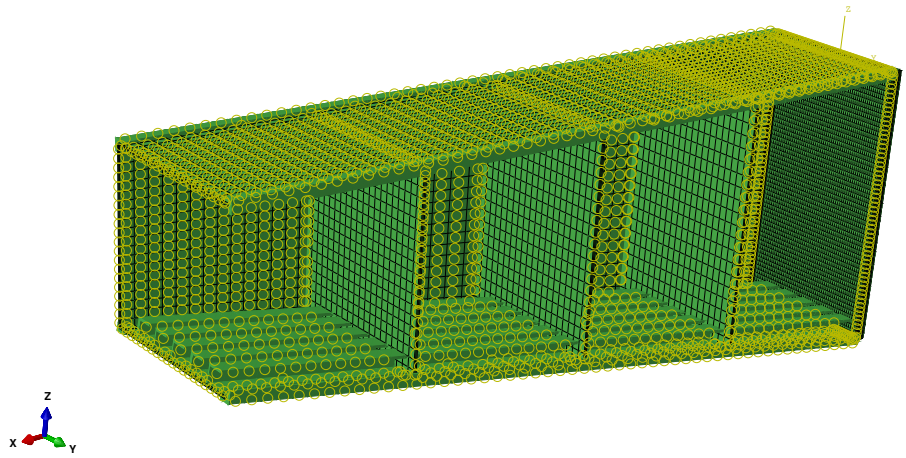


Figure 6.7: Box-beam element with constraints are visualized in Abaqus/CAE. Front bulkhead and side panel are removed for visibility reasons.

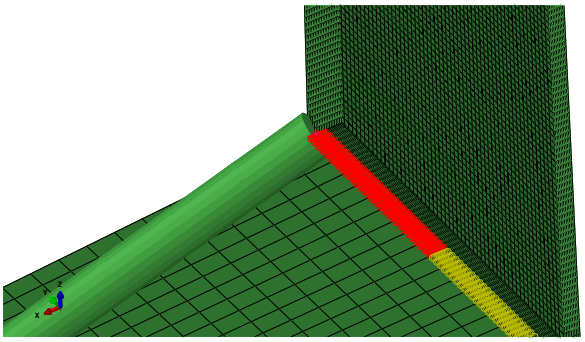


Figure 6.8: Tie constraint between the station 2 bulkhead left lower flange (red surface) and the left thrust link (pink surface).

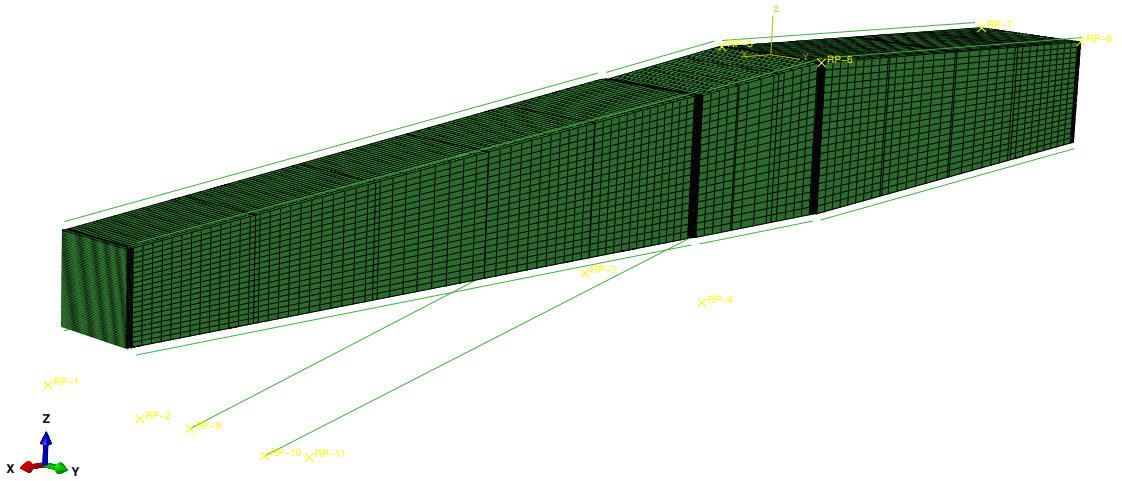


Figure 6.9: Full pylon model in Abaqus (thickness not rendered)

world at the pylon-to-engine and pylon-to-aircraft interfaces, which are defined in the following section. An image of the full structure for a conventional UWN pylon employing thrust links is shown now in Figure 6.9.

6.3.5. APPLYING COUPLINGS AND BOUNDARY CONDITIONS

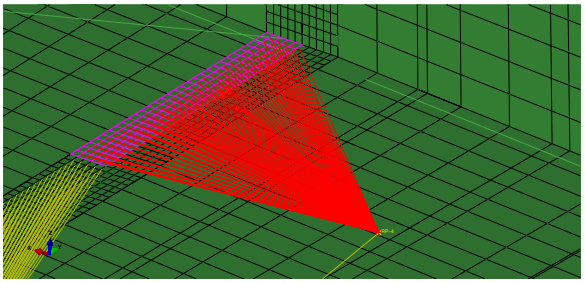
The final step in the definition of the model geometry is the definition of the structural behavior at the interfaces between the pylon and the engine- and aircraft, so at the hardpoints. By implementing the right constraints at these locations, the pylon structure is provided with the spatial context required to calculate the behavior of the structure under loads.

In section 4.1, some characteristics of the structural design of 'typical' engine-to-pylon- and pylon-to-aircraft connections were shortly discussed. The detailed design of these complex systems of links and fittings is considered out of the scope of this research project. Furthermore, the contribution of this structure to the overall weight of the pylon structure is assumed to be small. For that reason, these mounts are not included in the structural model of the pylon, but instead surface coupling constraints that translate the forces at the hardpoints into the front- and rear engine bulkheads are implemented. These constraints were already discussed briefly in subsection 6.3.3. As mentioned there, a coupling constraint 'couples' the DoFs of one or multiple nodes on a surface to the DoFs of a particular 'reference node' or 'control node'. This coupling can either be of type 'kinematic', which creates a rigid connection between the control node and the slave nodes and such that they follow the movement of the control node (much like how an 'RBE2 rigid element' behaves in Autodesk Nastran), or of type 'distributing', in which the rigid body movement of the control node is averaged across the slave nodes using a weighted distribution method and deformation among the coupling nodes is allowed (like in an 'RBE3 rigid element' in Autodesk Nastran). In all cases, the constructions needs to be statically determinate to ensure stability of the system.

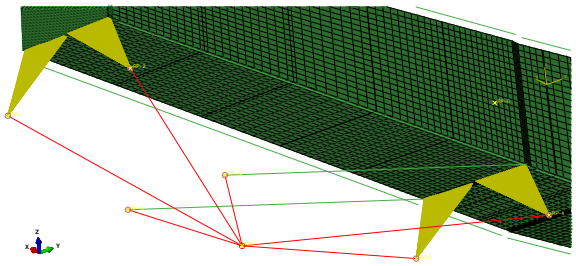
ENGINE-TO-PYLON CONNECTION

In case of the engine hardpoints-to-pylon structure coupling, a 'distributing' coupling constraint is created for each hardpoint that couples it to its closest bulkhead lower flange with the half flange surface nodes as coupling nodes using a 'uniform' averaging method. The bulkhead flange is chosen as a coupling target as this is the location where the forces are introduced into the system. An example of this coupling is shown in Figure 6.10a, which shows the coupling constraint implemented at the left aft engine-to-pylon mount.

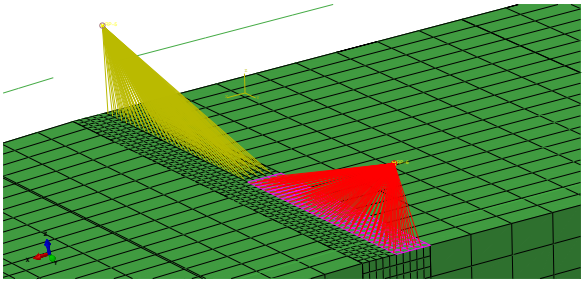
To couple the engine 'geometry' to the hardpoints, another coupling constraint is created, this time of type 'kinematic'. In this constraint, the engine hardpoints are coupled to the reference point representing the engine COG, which acts as a point mass representing the engine and nacelle, and is the point where all the forces are applied. This coupling is shown in Figure 6.10b. By implementing this coupling, the 'rigid body



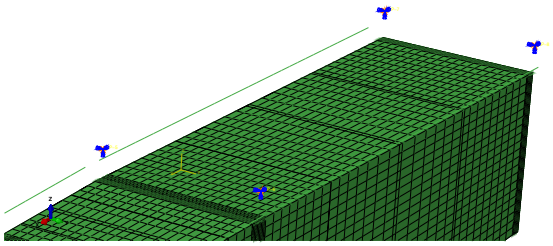
(a) Distributed coupling constraint coupling the left rear engine hardpoint to the left part of the lower flange of the bulkhead of box-beam compartment A at station 2.



(b) Kinematic coupling constraint coupling the engine hardpoints to the engine COG



(c) Distributed coupling constraint coupling the front right pylon-to-aircraft hardpoint to the right part of the upper flange of the bulkhead of box-beam compartment C at station 3.



(d) Boundary conditions implemented at the pylon-aircraft hardpoints

Figure 6.10: Detailed views showing the couplings and boundary conditions representing the interaction of the pylon structure with the engine and the aircraft

structure' of the engine is represented in a simple way without the need to model the full engine structure in Abaqus. As explained by Bettebghor *et al.* [26], this method of modeling the engine and nacelle is considered standard practice during the early phases of the engine integration design by the airframe manufacturer.

PYLON-TO-AIRCRAFT CONNECTION

For the pylon-to-aircraft interface, the same strategy was chosen of connecting the aircraft hardpoints to the pylon through distributed coupling constraints. Again, the bulkhead flanges are used, but this time at the top side. The full flange surface is constrained this way to distribute the forces as well as possible into the structure and prevent hour-glassing or peak loads as much as possible. An example of this for the bulkhead-to-aircraft coupling at station 3 is shown in Figure 6.10c.

Then, the boundary conditions are applied at the nodes representing the hardpoints that enforces a zero displacement and rotation at both the front- and rear aircraft mounting points. This is in line with the strategy for pylon-to-aircraft coupling reported mostly in literature. The resulting boundary conditions are shown in Figure 6.10d.

This step concludes the 'model definition' as defined at the start of this chapter using the `Adaptor`-class. Next, the 'history data' definition using the `Step`-class is defined.

6.4. DEFINING THE ANALYSIS

The second part of the .inp-file consists of the 'time-dependent' part of the analysis, or 'history data' in Abaqus. This part is defined using `<step>`-objects in which the analysis that is to be performed on the structure is defined.

6.4.1. APPLYING LOADS AND LOAD CASES

Throughout various steps, the different load cases that were defined in section 6.2 are prescribed onto the structure. All loads are defined as concentrated loads at the engine COG using a `CLoad`-class, with force components in x-, y- and z-direction as well as moments around those same axis derived using the perfor-

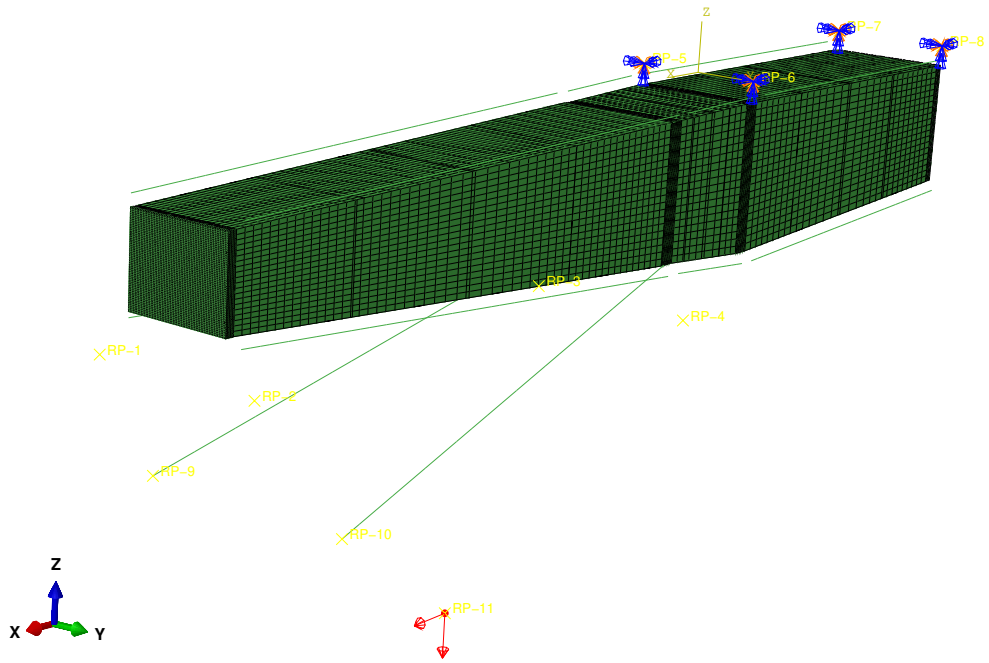


Figure 6.11: Loads application (in red) in the Abaqus model history data definition

mance characteristics of the engine as explained in section 2.3, in combination with the orientation angle ϕ which defines the direction in which the loads are applied. Load cases are applied as a combination of the forces and implemented as a list of `<loadcase>`-objects (generated using the `LoadCase`-class) that is fed into the `<step>`-object as an input. A graphical representation of loads application on the structure can be seen in Figure 6.11.

6.4.2. IMPLEMENTING ANALYSIS STEPS

As mentioned, steps form the basis of the analysis definition. They hold all the information that is required for Abaqus to execute the analysis, including the desired analysis procedure, initial conditions if applicable, the loads and load cases to which the structure is subjected, the 'analysis time' during the step including the time incrementation scheme, and the history- and field output data that Abaqus should save during the analysis.

Steps in Abaqus/Standard can either be **'general response analysis steps'** or **'linear perturbation steps'**. In the first, the analysis is executed as a full linear or non-linear analysis with a pre-specified time duration and incrementation, and all loads are applied directly to the structure, in full. In a linear perturbation analysis on the other hand, the loads are applied as 'linear perturbations' onto a *base state*, without adding physical time to the total analysis time. As such way, the numerical cost and therefore execution time of the analysis is greatly reduced. Furthermore, this enables the used of multiple load cases analysis inside one step, a feature that is only available in Abaqus when using linear perturbation steps. Since the number of load cases defined for this analysis is relatively high and moreover this FEM analysis is performed in the context of an optimization schedule, this is highly advantageous as is significantly reduces the computational cost of the analysis. An important note though is that the loads should not differ too much from those applied in the base step to maintain a proper level of accuracy.

Given the considerations explained, a multiple step analysis is proposed that combines linear perturbation steps to calculate the limit loads with general analysis steps for the ultimate load cases. An overview of all the step and the load cases applied per step is given in Table 6.6.

BASE STEP

The analysis starts with an initial step (step 1) in which the base load on the pylon is defined. In this analysis, this 'base state' is defined as an additional load case 'CR-00', or steady trimmed cruise at 1g at maximum continuous thrust as explained in subsection 6.2.1. As explained, this base step is implemented as a reference for the designer and the resulting state of the pylon will form the initial conditions of the following perturbation analysis. A simple static stress analysis procedure is implemented with a step time of 1 and non-linearity flag ON. Boundary conditions that have been defined in the `<adaptor>`-object will be applied in this first step as well and propagated to the following steps automatically.

STATIC PERTURBATION ANALYSIS STEP

Secondly, in step 2 a linear perturbation analysis step is performed that covers all limit load cases as defined in Table 6.1 and Table 6.6. For each load case defined in Table 6.1, a `<loadcase>`-object is defined that holds the definition of the loads and boundary conditions relevant for that load case. As is required for linear perturbation steps, the forces and moments applied in a load case are defined not as absolute values, but relative to the loads defined in the base step. Nonlinear geometry flag is turned off per definition for linear perturbation analysis steps.

Table 6.6: Step definition for the analysis.

Step No.	Load case	Analysis procedure	Type	General (G) / Lin. Pert. (P)	NLGEOM	Time
1.	CR-00	Static Stress Analysis	Static	G	ON	1
2.	CR-01	Static Stress Analysis	Static	P	OFF	0
	CR-02	Static Stress Analysis	Static	P	OFF	0
	CR-03	Static Stress Analysis	Static	P	OFF	0
	CR-04	Static Stress Analysis	Static	P	OFF	0
	CR-05	Static Stress Analysis	Static	P	OFF	0
	CR-06	Static Stress Analysis	Static	P	OFF	0
	CR-07	Static Stress Analysis	Static	P	OFF	0
	TO-01	Static Stress Analysis	Static	P	OFF	0
	TO-02	Static Stress Analysis	Static	P	OFF	0
	LA-01	Static Stress Analysis	Static	P	OFF	0
	LA-02	Static Stress Analysis	Static	P	OFF	0
	LA-03	Static Stress Analysis	Static	P	OFF	0
	LA-04	Static Stress Analysis	Static	P	OFF	0
	GR-01	Static Stress Analysis	Static	P	OFF	0
	GR-02	Static Stress Analysis	Static	P	OFF	0
	AIR-01	Static Stress Analysis	Static	P	OFF	0
	AIR-02	Static Stress Analysis	Static	P	OFF	0
	AIR-03	Static Stress Analysis	Static	P	OFF	0
	AIR-04	Static Stress Analysis	Static	P	OFF	0
3.	FBO-1	Static Stress Analysis	Static	G	ON	1
4.	FBO-2	Static Stress Analysis	Static	G	ON	1
5.	FBO-3	Static Stress Analysis	Static	G	ON	1
6.	FBO-4	Static Stress Analysis	Static	G	ON	1

FBO ANALYSIS STEPS

Finally for each equivalent static FBO load case, a separate general static analysis step is defined (steps 3-6). As explained in subsection 6.2.2, each step holds on static equivalent unbalance load for release angles 0°, 90°, 180° and 270°. Since these are ultimate load cases, the NLGEOM flag is set to ON again. To deal with possible instabilities and material failure, the Abaqus modified Riks method is used with automatic stabilization turned off. As for step 1, step time is set to 1, with the initial increment size set to 1.

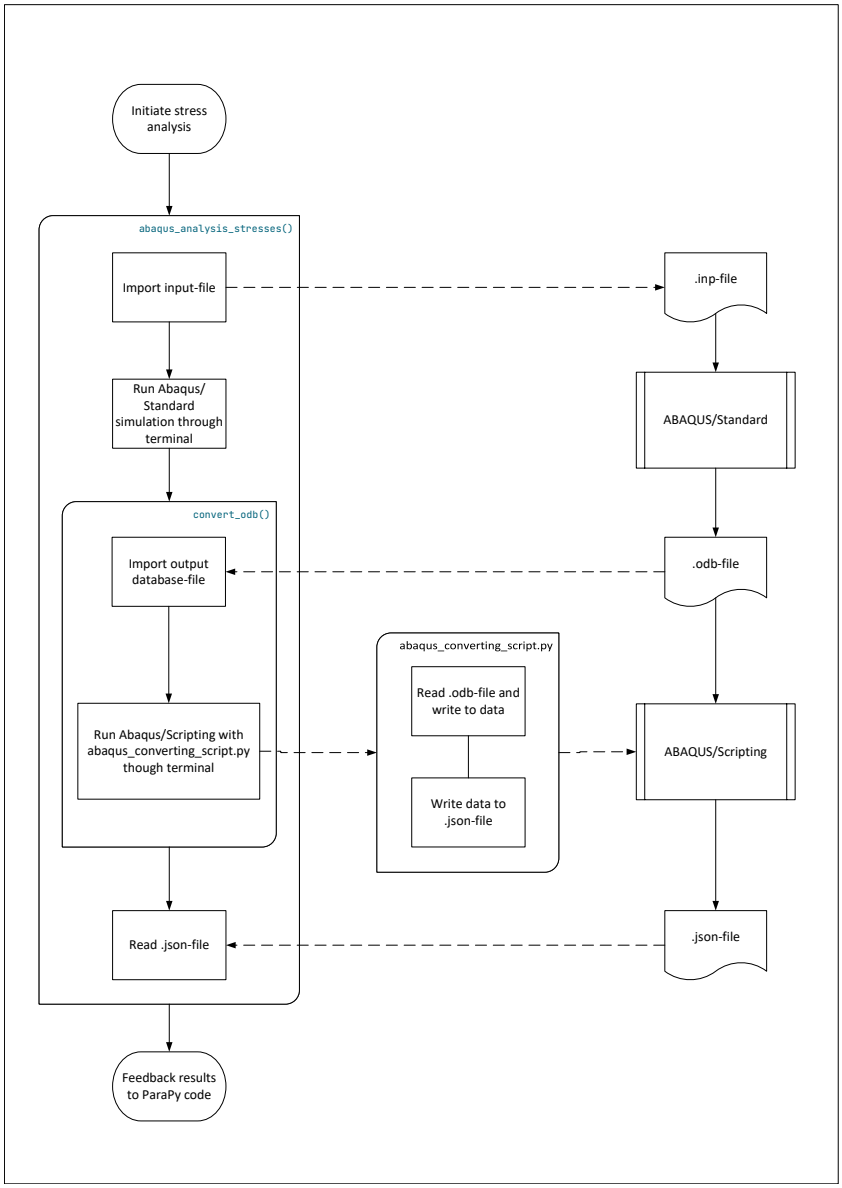


Figure 6.12: Flowchart showing the process of running an Abaqus simulation and post-processing the data to find the result data.

6.5. RUNNING THE ABAQUS SIMULATION AND READING OUTPUT DATA

With the model and history data are fully defined, and the input file written using the `Writer`-class from the ParaPy/Abaqus API, all information and files required to run the simulation are now available. To execute the simulation, a specific `abaqus_analysis_stresses()`-function is created, in which the simulation is ran and the resulting data is post-processed to obtain the data required for further analysis. The process executed in this script requires several steps, as shown in Figure 6.12 among which two Abaqus runs to run the simulation and to translate the results generated by Abaqus into a format that can be read by a secondary code like the one used in this project.

6.5.1. RUNNING THE SIMULATION

The first step in the process of obtaining the stress data for the model and analysis defined in the input file is to actually run the simulation for this input file. To do so, the Abaqus FEM solver is initiated using a terminal process, with the input-file path and the job name for the analysis as main input parameters. Other job settings, like options for parallel computing, memory usage, etc. can also be defined here if required.

To analyze the simulation cases defined in the input-file, the *'Abaqus/Standard'* solver is used. In Abaqus/Standard, an implicit finite element method simulation is performed, which is suitable for most general-purpose simulation procedures that do not require explicit time-integration to solve (like what is required in problems involving high non-linearity, high speed loading, or large deformations, like impact analysis or analyses where contact is important, all of which is omitted in this thesis project), including static analysis, steady-state dynamic analysis and linear buckling analysis. In an implicit analysis, the governing equations of motion are solved using an implicit direct time integration method, in the case of Abaqus the so-called Newmark-β method [93] or Hilbert-Hughes-Taylor method [94]. Using an implicit time integration method, the solution of the equations of motion at a time $t+\Delta t$ is found using the information at time t plus the derivative at time $t+\Delta t$. This requires an iteration scheme to find the right derivative, and works as follows:

The governing equations for a (transient) dynamic problem are given by:

$$M\ddot{\mathbf{u}}(t) + C\dot{\mathbf{u}}(t) + K\mathbf{u}(t) = \mathbf{p}(t) \tag{6.3}$$

with initial values:

$$\begin{cases} \mathbf{u}(t_0) = \mathbf{u}_0 \\ \dot{\mathbf{u}}(t_0) = \dot{\mathbf{u}}_0 \end{cases} \tag{6.4}$$

In the Newmark method, the solution for the displacement and velocity variables at timestep $t+\Delta t$ is assumed to be of the form of a Taylor series approximation [88]:

$$\mathbf{u}_{t+\Delta t} = \mathbf{u}_t + (\Delta t)\dot{\mathbf{u}}_t + (\Delta t)^2\left[\left(\frac{1}{2}-\beta\right)\ddot{\mathbf{u}}_t + \beta\ddot{\mathbf{u}}_{t+\Delta t}\right] \tag{6.5}$$

$$\dot{\mathbf{u}}_{t+\Delta t} = \dot{\mathbf{u}}_t + (\Delta t)\left[(1-\gamma)\ddot{\mathbf{u}}_t + \gamma\ddot{\mathbf{u}}_{t+\Delta t}\right] \tag{6.6}$$

Substituting this into Equation 6.3, we obtain:

$$M\ddot{\mathbf{u}}_{t+\Delta t} + C\left\{\dot{\mathbf{u}}_t + (\Delta t)\left[(1-\gamma)\ddot{\mathbf{u}}_t + \gamma\ddot{\mathbf{u}}_{t+\Delta t}\right]\right\} + K\left\{\mathbf{u}_t + (\Delta t)\dot{\mathbf{u}}_t + (\Delta t)^2\left[\left(\frac{1}{2}-\beta\right)\ddot{\mathbf{u}}_t + \beta\ddot{\mathbf{u}}_{t+\Delta t}\right]\right\} = \mathbf{p}_{t+\Delta t} \tag{6.7}$$

which can be written as:

$$K_{cm}\ddot{\mathbf{u}}_{t+\Delta t} = \mathbf{p}^{\text{residual}}_{t+\Delta t} \tag{6.8}$$

where:

$$K_{cm} = \left[K\beta(\Delta t)^2 + C\gamma\Delta t + M\right] \tag{6.9}$$

$$\mathbf{p}^{\text{residual}}_{t+\Delta t} = \mathbf{p}_{t+\Delta t} - K\left\{\mathbf{u}_t + (\Delta t)\dot{\mathbf{u}}_t + (\Delta t)^2\left(\frac{1}{2}-\beta\right)\ddot{\mathbf{u}}_t\right\} - C\left\{\dot{\mathbf{u}}_t + (\Delta t)(1-\gamma)\ddot{\mathbf{u}}_t\right\} \tag{6.10}$$

This can then be solved using the Newton-Raphson iteration method to find the accelerations $\ddot{\mathbf{u}}_{t+\Delta t}$ using the inverse stiffness matrix K_{cm}, from which the velocities and displacements are then found for the time step under consideration, as:

$$\ddot{\mathbf{u}}_{t+\Delta t} = K_{cm}^{-1}\,\mathbf{p}^{\text{residual}}_{t+\Delta t} \tag{6.11}$$

The advantage of using this implicit Newmark method is that it is unconditionally stable (if $\gamma \geq 0.5$ and $\beta \geq (2\gamma+1)^2/16$), meaning large time steps are possible, which brings down the required CPU time significantly. On the contrary, explicit analyses require many small time steps to maintain stability and remain accurate, making them unsuitable to solve most dynamic problems which do not involve high frequency loading and short response times. On the downside, the implicit analysis is more computationally intensive in terms of memory, since it requires inversion of the stiffness matrix K_{cm} to solve the equations, which requires multiple iterations and can become very computationally expensive if the problem gets large. Nevertheless, the use of implicit analysis procedures is advised for most problems unless explicit methods are absolutely required, due to the inherent stable behavior, higher accuracy, lesser CPU time and ability to deal with a wide range of problems, which is why it is applied in the analysis in this thesis project.

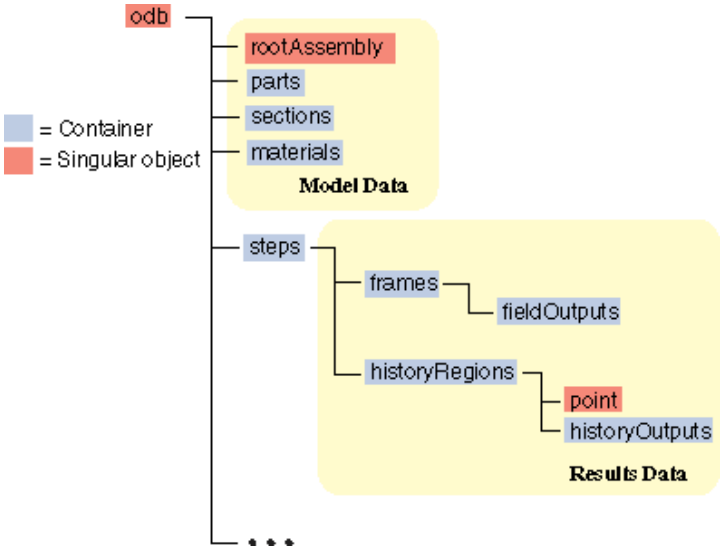


Figure 6.13: Output Database object structure [27].

6.5.2. READING OUTPUT DATA AND POST-PROCESSING

When the simulation is completed, Abaqus stores the obtained data from the simulation in a so-called 'Output Database'-file, with extension '.odb'. To read the data inside the .odb-file, which is written in binary format and therefore not suitable for direct post-processing, a dedicated script must be executed using the Abaqus Scripting Interface, an extension to the standard Python interpreter created by Abaqus, to translate the data in the ODB into something that can be imported and read using Python. This requires a run of the internal *'Abaqus Python'* kernel in Abaqus through the PC's terminal using the `abaqus python`-command. For this, a second function `convert_odb()` is created, which takes the .odb-file as an input and creates a .json-file with the same name in the same folder. Then, it runs the 'Abaqus Python' kernel though the terminal of the PC, in which a third script called `abaqus_converting_script.py` (written in Abaqus Python v2.7) is executed that can use the specialized Abaqus Python commands to read and manipulate the data in the .odb-file and write them into the .json-file, which can then be read again in the ParaPy Python v3.7 application for further processing. This rather laborious process is represented visually in the lower half of the image in Figure 6.12. More information and example scripts showcasing the use of the Abaqus Scripting Interface can be found in the book of Puri [95] and the Abaqus Online User Manual [27].

As shown in Figure 6.13, the Output Database is structured into two main sections. The model data contains the description of the model as it was defined in the input file. The results data contains the results of the simulation, and is further subdivided into steps, with field data per frame (representing a time increment) and history region. Note that for a 'perturbation step', each frame does not represent a time increment but instead defines a perturbation load case, as time incrementation is not defined for a perturbation step as explained in subsection 6.4.2. For the sake of this analysis, we require the maximum stress in each instance per load case as an output, which for the static load cases should be below the yield stress of the material and for the FBO load cases below the limit stress of the material. So, we have to loop through all the steps and all the frames, in order to find the maximum Von Mises stress in each part instance for both the static and dynamic load cases, and in what load case this maximum stress value was found. This data is then written to a .json-file for further analysis in the remain part of the `abaqus_analysis_stresses`–script, which outputs the data as an *Attribute* for further use, for example in the optimization as described in the upcoming chapter.

7

SIZING OPTIMIZATION

Having defined the pylon model and explained its implementation in the ParaPy KBE application, the final step of the process defined in Figure 3.1 is using the model to perform a structural sizing process in order to obtain a structure that is not only fit to carry all the loads which are exerted onto it by the engine, but can do so efficiently using a low weight pylon structure. This way, the resulting pylon structure will resemble the actual pylon structure as it would be designed in a real airplane as close as possible, resulting in a good estimation of the resulting pylon weight for the inputs provided.

7.1. PROBLEM DEFINITION

Structural sizing of a structural assembly involves picking the right set of structural parameters to generate a 3D structure that can hold all the loads and load cases for which the structure must be designed. In this case, the desire is to pick not just any set of parameters that can hold the specified loads, but those that lead to the lightest structure. As such, the problem posed here can be defined in the form of an optimization problem, with a pre-specified set of structural design parameters as the design vector, the weight of the structure as the objective and the maximum stresses of the applied material as constraints. In this section, the full problem is defined in mathematical terms.

7.1.1. DESIGN VECTOR

In chapter 4, the parametric model of the pylon used in this thesis project was described. An overview of the parameters was given in Table 4.5. Using this parameterization, the pylon structure can be defined in full, including both what was described in this chapter as the 'zero-thickness geometry' and the 'structural elements' definition. The translation of these parameters to the parameters used inside the `PylonDesigner`-app is given in Table C.1 for reference.

While the full parameterization as explained consist of a total of 80 parameters, a large part of these parameters are defined by the user during the initiation of the code. For the 'structural sizing' part of this thesis project, fortunately a smaller subset of parameters describing mainly the structural parameterization of the pylon can be used as a design vector in the optimization. First of all, the definition of the 'zero-thickness geometry', Table 4.1-4.3, was considered as an input for the application, as was also shown in Figure 3.2 and explained in chapter 5. The hardpoint definition is assumed to be set by the airplane- and engine head designers — the locations of the engine hardpoints are assumed to be a set parameter derived from the design of the engine used, while the aircraft hardpoints as well as the location and orientation of the engine with respect to the aircraft are assumed to be enforced by the top-level aircraft topology design. Finally, the outer-mold of the pylon can theoretically be chosen by the pylon designer. But in practice, the pylon structure is almost always designed in concurrence with the aerodynamic design of the pylon fairing, for example in a multidisciplinary design optimization project as was executed by Gazaix *et al.* [7], which was described in subsection 2.1.2. In that case, the aerodynamic fairing acts as a cover for the structural part of the pylon, thereby constraining the design of the pylon structure to the enclosed space inside it (see Figure 2.3a). With the pylon structure seeking to increase the Moment of Inertia of the pylon as much as possible in order to obtain the highest torsional stiffness, this space is assumed to not only constrain but in fact *set* the outer-mold

shape of the pylon structure.

This leaves the structural elements parameterization as the parameters that can be manipulated in a structural sizing optimization process. An overview of these parameters was given in Table 4.4. These parameters are defined per compartment, as described in section 4.3, with thrust link parameters of course defined separately.

The resulting set of parameters indicated above is unfortunately still not suitable for the purposes of this project, as it consists of both discrete variables — the numbers of ribs and stringers — and continuous variables, like thicknesses and flange heights. Solving an optimization where both discrete as well as continuous design variables are implemented is a difficult task that requires advanced optimization algorithms which are deemed outside the scope of this project. Therefore, the discrete structural elements parameters will not be considered in this optimization, but instead left as a design choice for the user.

Finally, it is common practice in optimization to normalize the design vectors, objective and constraint in order to obtain a more accurate result and eliminate convergence issues related to overly large step sizes. In the case of the design variables, the design variables are normalized with respect to the bounds that are defined for the design variable, such that:

$$-1 \leq \mathbf{x} \leq 1$$

where a value of -1 means that the design variable sits right on the lower bound, while a value of 1 means that the design variable is equal to the upper bound set for it. The normalized value is obtained by dividing the distance between the real value and the mean of the upper and lower bound by the half interval between the two bounds, as follows:

$$x_{avg} = \frac{\overline{x} + \underline{x}}{2} \tag{7.1}$$

$$x_0 = \frac{x - x_{avg}}{(\overline{x} - \underline{x})/2} \tag{7.2}$$

All together, this leaves the following design vector for the sizing optimization. Note that for reasons of readability, this vector is written in non-normalized form here.

Design Vector

The (non-normalized) design vector used in this optimization is the vector $\mathbf{x}$, consisting of 46 parameters describing the structural design of the 3 pylon compartments, and the diameter of the thrust links.

$$\mathbf{x} = \begin{bmatrix} \left\{ \begin{array}{c} t_{spar,upp}, \\ t_{spar,low}, \\ t_{web,left}, \\ t_{web,right}, \\ t_{blkh1,web}, \\ t_{blkh1,flanges}, \\ w_{blkh1,flanges}, \\ t_{blkh2,web}, \\ t_{blkh2,flanges}, \\ w_{blkh2,flanges}, \\ t_{longeron,flanges}, \\ h_{longeron,flanges}, \\ t_{ribs}, \\ t_{stringers}, \\ h_{stringers} \end{array} \right\}^{A...C} \\ d_{thrustlink} \end{bmatrix} \tag{7.3}$$

7.1.2. OBJECTIVE FUNCTION

Next, the objective of the structural optimization should be defined. Since we are interested in finding a realistic weight estimate of the pylon, the structure of the pylon should be such that is resembles that of the real pylon as it would be applied in reality as closely as possible, to the extent this is possible in this stage of the design. For most structural members that make up the aircraft, the value that most drives the design of the structure is weight, as was explained in subsection 2.4.1. As such, the sizing optimization procedures executed using this tool need to size the structural members that make up the pylon such that the lowest structural weight is achieved. As with the design vector, the objective function is normalized, this time using the initial weight for the optimization case under consideration as a normalizer. This leads to the objective function definition as expressed below.

Objective Function The objective function for this optimization can be mathematically expressed as:

$$\min_{\mathbf{x}} f(\mathbf{x}) = \frac{W_{struct}(\mathbf{x})}{W_{struct}(\mathbf{x_0})} \tag{7.4}$$

7.1.3. EQUALITY AND INEQUALITY CONSTRAINTS

Where the objective function of the sizing optimization is decided based on the weight of the pylon, which is extracted from the ParaPy CAD model, the constraints are based on the internal stresses inside the pylon structure, which is calculated by performing a finite element method simulation using Abaqus. The details of this simulation were explained in detail in the previous chapter. As was explained in subsection 6.5.2, the results of the structural analysis using Abaqus is a vector of maximum Von Mises stresses for each load case and in each part of the pylon structure in the form of a Python dictionary. In the optimization constraints, these stresses are compared with the so-called 'stress allowables' for each load case and each part, of which the value depends on the type of load case considered.

LIMIT LOAD CASES

For the limit load cases, the failure of the structure is defined as a situation in which the stresses in the pylon structure lead to permanent structural deformations. In other words, the structural members fail whenever the maximum Von Mises stress becomes bigger than the Yield Stress of the material. For extra security, the certification standards prescribe a safety factor of 1.5 for these load cases, which has been incorporated into the load factors that are applied during the simulation as was explained in subsection 6.2.1. This leads to the definition of the *yield criterion*, which is defined as follows:

$$\sigma_{maxMises} \leq \sigma_{y,material} \tag{7.5}$$

or rewritten into so-called *negative null form*:

$$\frac{\sigma_{maxMises}}{\sigma_{y,material}} - 1 \leq 0 \tag{7.6}$$

ULTIMATE LOAD CASES

For the ultimate load cases, permanent deformation of the structure is allowed since these loads are only supposed to occur once in the lifetime of the component, as long as it does not lead to a total breakdown of the structure and the pilot can still safely land the airplane. This means that permanent deformation due to inelastic deformation of the material and post-buckling are allowed, but the stress in the material should not surpass the ultimate tensile stress of the material at which the material breaks. Because of this criterion, post-buckling of the material is not investigated, meaning effects like local panel buckling and column buckling are not considered in this analysis. This means that for the ultimate load cases, the following criterion should be true:

$$\sigma_{maxMises} \leq \sigma_{ult,material} \tag{7.7}$$

$$\frac{\sigma_{maxMises}}{\sigma_{ult,material}} - 1 \leq 0 \tag{7.8}$$

These are the only constraints that are applicable in this optimization. No equality constraints are defined. This means that the full constraints definition for this optimization problem can be defined as follows.

Inequality constraints The inequality constraints for this optimization can be mathematically expressed as:

$$g_1(\mathbf{x}) = \frac{\sigma(\mathbf{x})_{maxMises}}{\sigma_{y,material}} - 1 \leq 0 \tag{7.9}$$

$$g_2(\mathbf{x}) = \frac{\sigma(\mathbf{x})_{maxMises}}{\sigma_{ult,material}} - 1 \leq 0 \tag{7.10}$$

Equality constraints The equality constraints for this optimization can be mathematically expressed as:

$$\mathbf{h}(\mathbf{x}) = [\mathbf{0}] \tag{7.11}$$

7.1.4. INITIAL DESIGN VECTOR AND BOUNDS

Finally, the initial design vector and the bounds need to be defined to fully define the optimization problem. Unfortunately, given that this is a parametric model for which each initial design can be different, defining one definite initial design vector is impossible. Likewise, it is not possible to set global bounds for the optimization, as each optimization case will require its own tailored values. It is left up to the discretion of the designer to choose the initial design vector, upper- and lower bounds that he or she considers appropriate for a given use case. As such, the initial design vector and upper- and lower bound vectors can only be defined in terms of their respective variables.

Initial Design Vector The initial design vector is defined as:

$$\mathbf{x}_0 = \begin{bmatrix} \left\{ \begin{array}{c} t_{spar,upp}, \\ t_{spar,low}, \\ t_{web,left}, \\ t_{web,right}, \\ t_{blkh1,web}, \\ t_{blkh1,flanges}, \\ w_{blkh1,flanges}, \\ t_{blkh2,web}, \\ t_{blkh2,flanges}, \\ w_{blkh2,flanges}, \\ t_{longeron,flanges}, \\ h_{longeron,flanges}, \\ t_{ribs}, \\ t_{stringers}, \\ h_{stringers} \end{array} \right\}_0^{A...C} \\ d_{thrustlink,0} \end{bmatrix} \tag{7.12}$$

Bounds The bounds are defined as:

$$\overline{\mathbf{x}} = \begin{bmatrix} \left\{ \begin{array}{c} \overline{t}_{spar,upp}, \\ \overline{t}_{spar,low}, \\ \overline{t}_{web,left}, \\ \overline{t}_{web,right}, \\ \overline{t}_{blkh1,web}, \\ \overline{t}_{blkh1,flanges}, \\ \overline{w}_{blkh1,flanges}, \\ \overline{t}_{blkh2,web}, \\ \overline{t}_{blkh2,flanges}, \\ \overline{w}_{blkh2,flanges}, \\ \overline{t}_{longeron,flanges}, \\ \overline{h}_{longeron,flanges}, \\ \overline{t}_{ribs}, \\ \overline{t}_{stringers}, \\ \overline{h}_{stringers} \end{array} \right\}^{A...C} \\ \overline{d}_{thrustlink} \end{bmatrix} \tag{7.13}$$

$$\underline{\mathbf{x}} = \begin{bmatrix} \left\{ \begin{array}{c} \underline{t}_{spar,upp}, \\ \underline{t}_{spar,low}, \\ \underline{t}_{web,left}, \\ \underline{t}_{web,right}, \\ \underline{t}_{blkh1,web}, \\ \underline{t}_{blkh1,flanges}, \\ \underline{w}_{blkh1,flanges}, \\ \underline{t}_{blkh2,web}, \\ \underline{t}_{blkh2,flanges}, \\ \underline{w}_{blkh2,flanges}, \\ \underline{t}_{longeron,flanges}, \\ \underline{h}_{longeron,flanges}, \\ \underline{t}_{ribs}, \\ \underline{t}_{stringers}, \\ \underline{h}_{stringers} \end{array} \right\}^{A...C} \\ \underline{d}_{thrustlink} \end{bmatrix} \tag{7.14}$$

7.1.5. OPTIMIZATION PROBLEM SUMMARY

Now that all components of the minimization problem have been found, it is time to summarize them in a complete optimization problem.

Optimization formulation The sizing optimization problem of the pylon box structure for minimum weight can be summarized as follows:

$$\min_{\mathbf{x}} f(\mathbf{x}) = \frac{W_{struct}(\mathbf{x})}{W_{struct}(\mathbf{x_0})} \tag{7.4}$$

subject to:

$$g_1(\mathbf{x}) = \frac{\sigma(\mathbf{x})_{maxMises}}{\sigma_{y,material}} - 1 \leq 0 \tag{7.9}$$

$$g_2(\mathbf{x}) = \frac{\sigma(\mathbf{x})_{maxMises}}{\sigma_{ult,material}} - 1 \leq 0 \tag{7.10}$$

$$\mathbf{h}(\mathbf{x}) = [\mathbf{0}] \tag{7.11}$$

where $\mathbf{x}$ is the vector of structural design variables as defined by Equation 7.3.

7.2. OPTIMIZATION IMPLEMENTATION

To implement the optimization problem in the `PylonDesigner` KBE-application, the SciPy Optimize Python library was used[1]. This is a well known Python library that provides many capabilities for scientific computations and data analysis. This optimization library provides several algorithms for optimization, including two that can perform constrained optimization. For the purpose of this thesis, the *Sequential Linear Quadratic Programming* (SLSQP) algorithm is used, that uses a Sequential Quadratic Programming approach to calculate the values of the Jacobian and Hessian to obtain the gradients of the objective function and constraint functions in order to find the direction of steepest descent and find the optimum. More information on the method can be found in the original paper that described the method on which the SLSQP algorithm

[1] Information on this library can be found via `https://docs.scipy.org/doc/scipy/reference/optimize.html`

Table 7.1: Options used for the implementation of the SLSQP algorithm using SciPy.

Nonlinear constrained optimization using SciPy Optimize	
Algorithm	SLSQP
ftol	1e-2
max. iterations	10
eps	default

was based, which was written by Kraft [96]. The settings used for the implementation of this algorithm are summarized in Table 7.1.

8

VALIDATION CASE STUDIES

Having discussed all four ingredients of the pylon structural design and weight estimation tool, in this final parts of the thesis the PylonDesigner tool is applied to a set of two case studies to validate the workings of the tool. The reference data for these use cases was provided by engineers at Safran Aircraft Engines, who co-sponsored this thesis project together with TU Delft. First, the pylon structure of the LEAP-1B engine as implemented on the Boeing B737-MAX is analyzed. Then, a similar study will be performed on for the pylon of the LEAP-1A engine as implemented on the Airbus A320neo. To facilitate these case studies, the PylonDesigner tool was implemented in the context of a higher-order case study analysis script in which an analysis class called `PylonAnalysis` was created. This class creates the pylon using the `PylonDesigner`-class, but also creates the engine geometry using the parametric engine design-tool GTpy by Proesmans [23] that was discussed in the literature study in section 2.5. The engine data is collected in the form of a Python dictionary called `engine_characteristics` that will act as an input for the `pylon`-part that holds the pylon structure. Furthermore, a rudimentary wingbox structure is generated. This script can be omitted when using the tool in a different context, for example when the code is implemented into an already existing aircraft design code like the MMG.

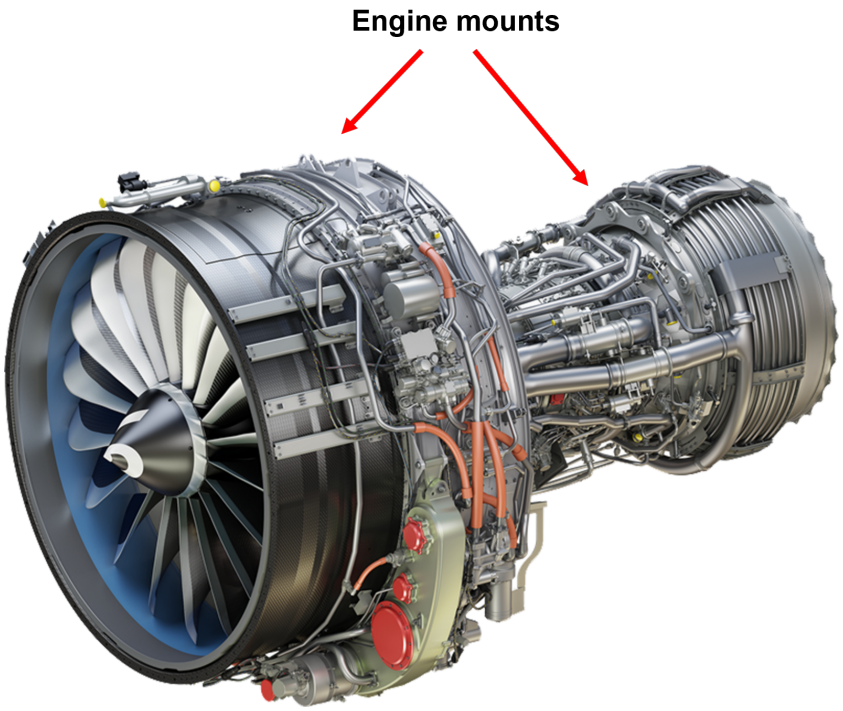
When using the script, the process starts with the user inputting the starting data for the analysis case, as mentioned above, including the required input data for GTpy, which requires an input file using the *CPACS*-file format [80]. For exact details on the GTpy input file and the inner workings of the GTpy code, the reader is referred to the thesis of Proesmans. The result of the GTpy execution is a fully sized outer geometry and flow path of a turbofan engine and a fitting nacelle geometry, including a set of characteristic engine parameters.

8.1. ENGINE INTEGRATION OF A LEAP-1B AIRCRAFT ON THE BOEING 737-MAX

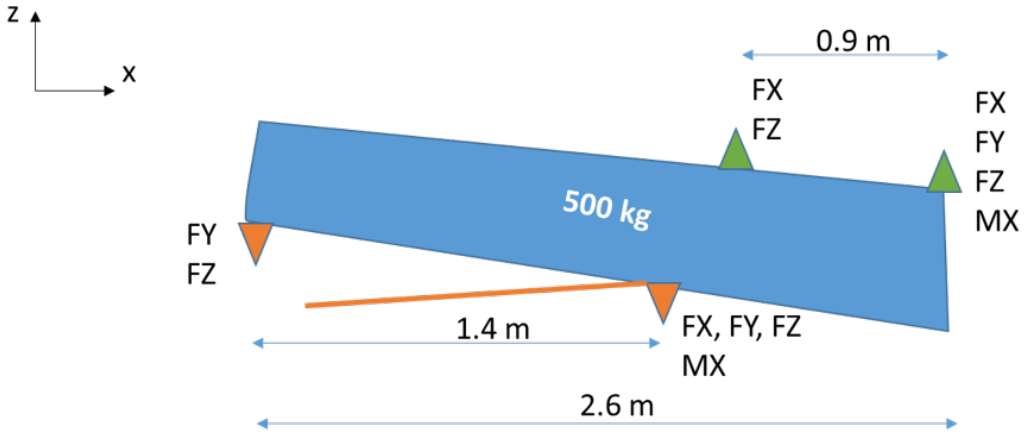
In the first case study, the pylon of the LEAP-1B engine as installed on the Boeing 737-MAX 10 is evaluated using the PylonDesigner KBE-tool.

8.1.1. PROBLEM DESCRIPTION

The LEAP-1B is a medium sized, dual rotor, high-bypass ratio turbofan engine designed by CFM International (a joint venture of GE Aviation and Safran Aircraft Engines) specifically for application onto the Boeing 737 MAX 7/8/9/10 family. It was designed as a successor to the popular CFM56-engine along with the LEAP-1A, who's installation is cover in section 8.2, and first entered service in May 2017. It features a BPR of at maximum approximately 9:1 and a maximum rated take-off thrust of 130,410 N or 29,317 pounds force. A schematic drawing of the engine is shown in Figure 8.1a, with the engine mounting points indicated. The characteristic data of the engine are summarized in Table 8.1. Note that as not all of the specific data of the engine is publicly available, some of the data is assumed to be equal to that of the LEAP-1A of which it was derived. This holds specifically for the thermodynamic data that is required to build the engine model in GTpy, most of which is assumed equal to that of the LEAP-1A provided by Holsteijn [98]. The weight and dimensions of the engine, as well as the rated thrust and N1 settings are described in the type certificate document of the LEAP-1B [99]. For the reverse thrust settings, the data reported by Noel and Boeker [100] is



(a) CFM LEAP-1B engine with engine mounts indicated (adapted from [97]).



(b) Pylon data (as provided by Safran Aircraft Engines)

Figure 8.1: LEAP-1B engine and its pylon as installed on the Boeing B737 MAX aircraft.

Table 8.1: LEAP-1B28 characteristic data

Parameter	Value	Unit	Source
	Dimensions		
Weight	2,780	kg	[99]
Length	3,147	mm	[99]
Diameter	2,421	mm	[99]
Mounting type	Fan	-	
	Thrust levels		
Max. continuous thrust	127,620	N	[99]
Max. continuous N1	100% at 4,397	rpm	[99]
Max. T/O thrust	130,410	N	[99]
Max. T/O N1	104% at 4,586	rpm	[99]
Max. reverse thrust	41.3% $F_{thr,max.cont}$	N	[101][100]
Max. reverse N1	73.5% $N1_{max.cont}$	rpm	[101][100]
	Thermodynamic data		
BPR	8.6:1	-	[102]
OPR	41.5:1	-	[102]
	Fan dimensions		
No blades	18	-	[103]
Blade material	3D woven resin transfer moulded (RTM) carbon fibre	-	[103] [104]
Blade mass	4.2	kg	[103]
Blade height		mm	GTpy calculation
	Engine COG location data		
X_eng	-2.0	m	Estimation
Y_eng	4.5	m	Estimation
Z_eng	-0.5	m	Estimation

applied. Idle thrust setting is assumed to be 24% rated rpm as described in the report by Yetter [101].

The engine is mounted in a traditional UWN-type configuration as is shown in Figure 8.2. For the purpose of this thesis, the integration of the engine onto the B737 MAX 10 was chosen as a reference, since this is the largest configuration of the B737 MAX family, meaning this is the worst case with respect to inertia forces. Note that for the purpose of this simulation, the left engine was taken as a reference both in this validation case as in the one for the A320neo.

A schematic drawing of the pylon that is used for this attachment was provided by Safran Aircraft Engines and is shown in Figure 8.1b. The pylon features a classical forward mounted box-beam type structure with thrust links. The pylon is fully made of titanium, and has a weight of approximately 500 kgs as per Safran.

8.1.2. INITIAL PROBLEM INVESTIGATION

In this first section, the pylon structure as generated using the initial design vector is analyzed.

GEOMETRY GENERATION VERIFICATION USING GEOMETRYGENERATOR

The data provided by Safran unfortunately is not sufficient to define the full analysis, meaning that part of the remaining measurements of the pylon have to be defined by the user. The dimensions of the outer shape of the pylon will then depend on the available space inside the fairing. In this case the dimensions are entered by the designer, leading to a full parametric definition of the pylon. The full zero-thickness definition applied

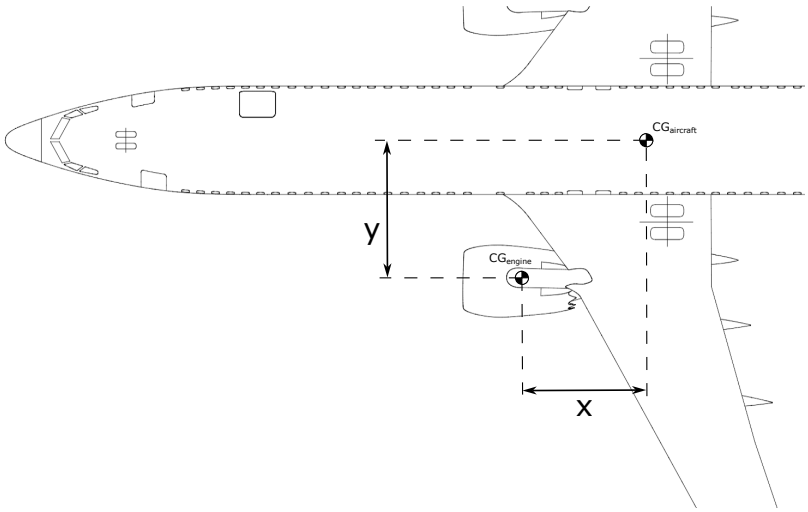


Figure 8.2: LEAP-1B installation on the B737 MAX 10 (adapted from [28]).

Table 8.2: Pylon zero-thickness geometry used for the analysis of the LEAP-1B installation on the B737

x_{12}	z_{12}	y_1	x_{23}	z_{23}	y_2	x_{34}	z_{34}	y_3	y_4	x_{52}	z_{52}	y_5
1.4	0.5	0.4	0.3	0.6	0.4	0.9	0	0.3	0.3	1.15	0.2	0.45
h_1	d_1	w_1	h_3	d_3	w_3	h_4	d_4	w_4	ϕ	$n_{stringers}$	$n_{ribs_{A,C}}$	n_{ribs_B}
0.2	0.1	0.25	0.25	0.05	y_3	0.3	0.05	y_4	0	4	4	2

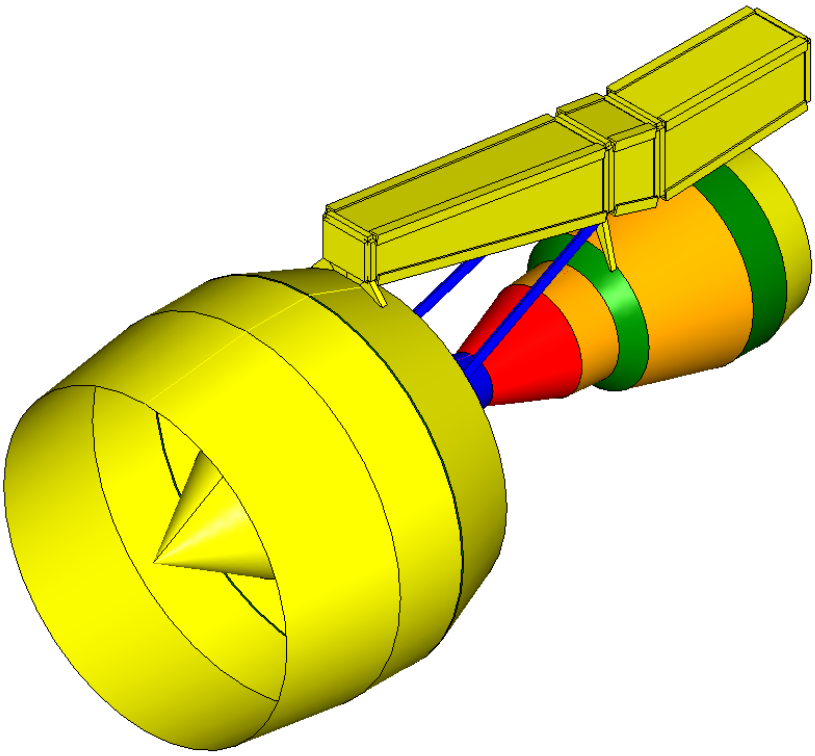
during this analysis can be found in Table 8.2. For the purpose of this analysis, the number of stringers was set to 0, and the number of ribs was set to 4 on compartments A and C and 2 in compartment B. As a material, Titanium 6Al4V was applied. The starting design vector for the simulation is shown in Table 8.3, together with the upper and lower bounds that were set for the design variables during this analysis.

The resulting initial pylon model generated by the GeometryGenerator in ParaPy is shown in Figure 8.3. As is visible in Figure 8.1a, the engine hardpoints where the pylon is attached in this configuration are at the end of the fan casing for the forward mount and right before the LPT for the aft engine mount. This closely represents the installation of the LEAP-1B engine onto the B737 as provided by Safran and shown in Figure 8.1a.

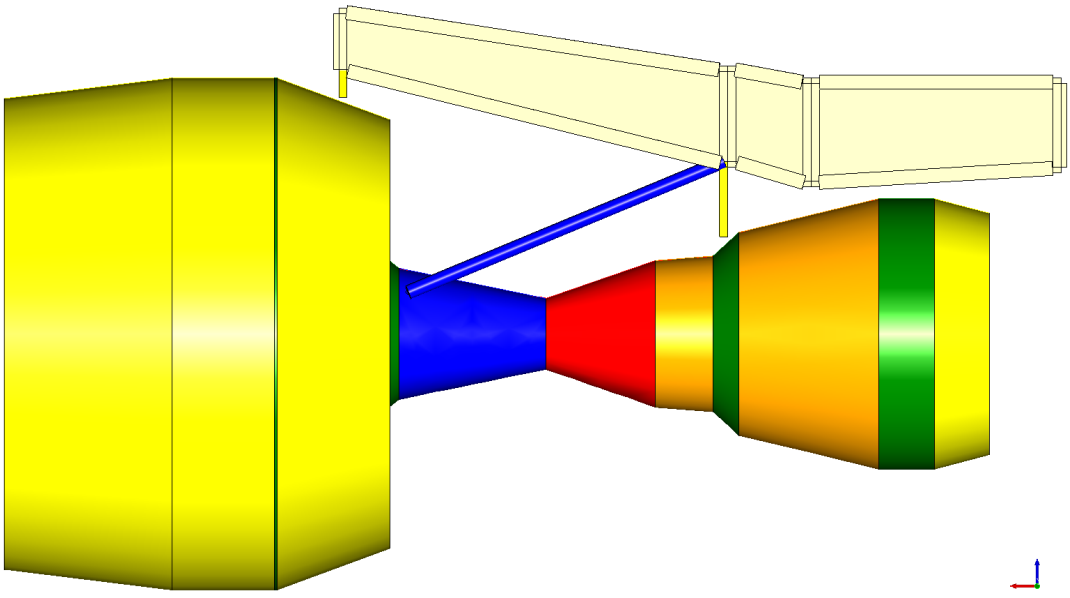
MESH SENSITIVITY STUDY

Before we can define an analysis use case, it is important to choose the right mesh density for the analysis, in order to make sure that the mesh is chosen such that it enables a satisfactory analysis of the pylon during the optimization, without leading to unworkable computational times. To check the effects of varying the mesh on the structure, the initial design of the pylon was analyzed using different mesh densities. The results of this study can be found in Figure 8.4.

In this mesh sensitivity study, the number of lateral elements in the beambox structure was slowly varied from a starting 12 lateral elements until finally 40 lateral elements. This represents a mesh elements total varying between 15072 and 162408 elements for the entire structure. At each step, the number of lateral elements is increased by 2, and the maximum Von Mises stress in the pylon structure is obtained. From this study, it was found that while the maximum Von Mises stress found in the beam structure varies between 617.6 MPa at 12 lateral elements to finally 800.1 MPa at 40 lateral elements the difference in maximum stress value found by Abaqus remains virtually the same after 22 lateral elements, only increasing by slightly less than 9 MPa over the remaining 9 mesh density increase steps. At the same time, the cost of increasing the



(a) Isometric view



(b) Side view

Figure 8.3: Initial geometry for the LEAP-1B pylon as generated by the GeometryGenerator and engine geometry generated using GTpy.

Table 8.3: Design variables and their bounds used in the analysis.

Parameter	Initial value	Lower Bound	Upper bound	Unit
$t_{ribs_{A...C}}$	10	1	50	mm
$h_{long,fl_{A...C}}$	50	10	100	mm
$t_{long,fl_{A...C}}$	10	1	50	mm
$h_{stringers_{A...C}}$	20	10	60	mm
$t_{stringers_{A...C}}$	5	1	30	mm
$t_{spar,low_{A...C}}$	30	1	50	mm
$t_{spar,upp_{A...C}}$	30	1	50	mm
$t_{web,left_{A...C}}$	30	1	50	mm
$t_{web,right_{A...C}}$	30	1	50	mm
$t_{blkh1,web_{A...C}}$	40	1	80	mm
$t_{blkh1,flanges_{A...C}}$	40	1	80	mm
$w_{blkh1,flanges_{A...C}}$	40	5	50	mm
$t_{blkh2,web_{A...C}}$	40	1	80	mm
$t_{blkh2,flanges_{A...C}}$	40	1	80	mm
$w_{blkh2,flanges_{A...C}}$	40	5	50	mm
$d_{thrustlinks}$	50	10	100	mm

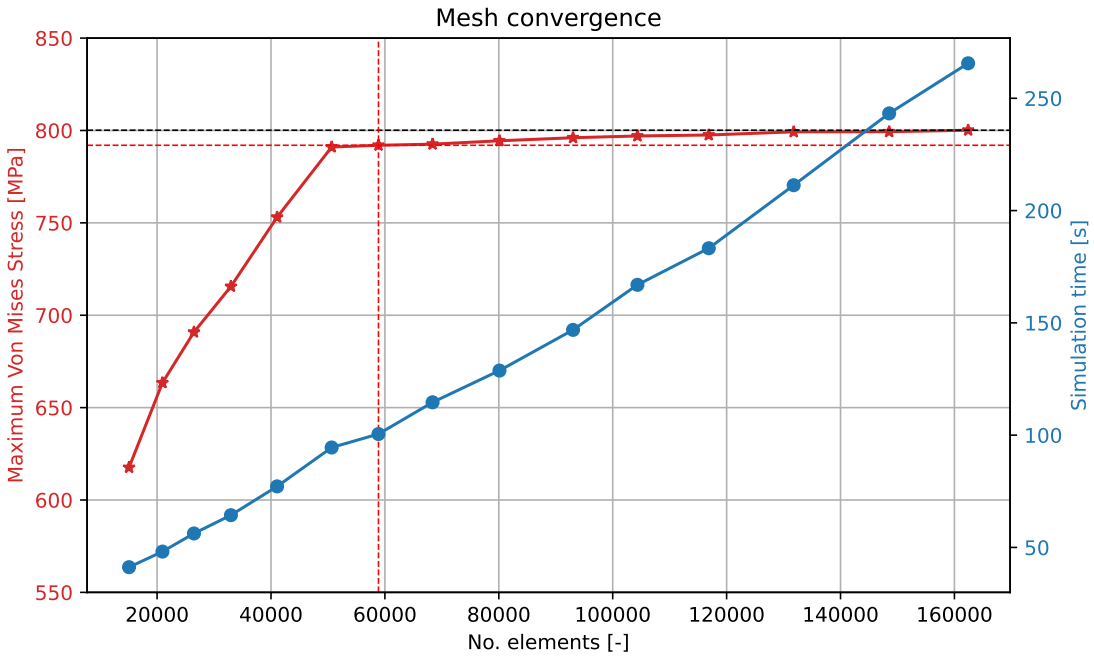


Figure 8.4: Maximum Von Mises stress and analysis execution time with varying mesh densities.

number of elements is hugely increased when increasing the number of lateral elements. At 12 lateral elements, the total clock time for the analysis (while employing parallel execution scheme using 4 CPUs) was only 25 seconds with a total CPU time of 60.7 seconds, while for the 40 lateral elements simulation time the total clock time increased to 227 seconds, but the CPU time increased even more to over 658 seconds.

From this, it can be concluded that while increasing the mesh density to over 22 lateral elements or a total of 50652 elements in the entire structure only increases the accuracy of the solution by a mere 1.14%, while the CPU time needed to solve the analysis is more than doubled, or even tripled if one would look only at CPU time. This is clearly out of proportion, especially given the importance of low computational time in this project due to the embedding of the FEM simulation in a sizing optimization scheme.

In the end, it was decided that for a good balance between calculation speed and accuracy, a lateral element density of 24 elements was appropriate (indicated by the red line in Figure 8.4). This is equivalent to a total of 58872 elements for the entire structure. According to the analysis, this leads to an underestimation of the maximum Mises stress in the structure of just under 1%, which is deemed acceptable for the purpose of this thesis project, while remaining the calculation time for an analysis low enough so that resonable optimization time runs can be expected.

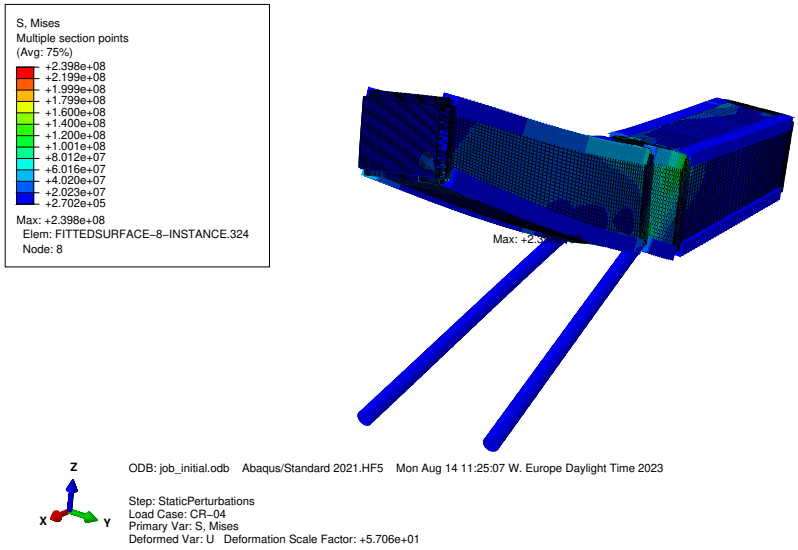
INITIAL DESIGN EVALUATION

Using the parametric values explained above and a mesh of 58872 elements, an initial weight and strength analysis was performed. This leads to a starting weight of **611 kg**. The initial maximum limit stress in the structure is found to be equal to **482 MPa** in load case CR-04, while for the maximum ultimate load cases a maximum stress of **799 MPa** was found in load case FBO-03. The limit maximum Von Mises load was found in instance *Bulkhead-3-instance*, which is one of the instances making up the bulkhead at station 2, where the rear engine mount forces enter the pylon structure. The maximum Von Mises stress for the ultimate load cases on the other hand was found to be in instance *Bulkhead-2-instance*, which is the part of the bulkhead of station 3 (at the wing front mounts) that is formed as a part of the middle compartment B. Both these load cases and their resulting stress profiles can be found in Figure 8.5.

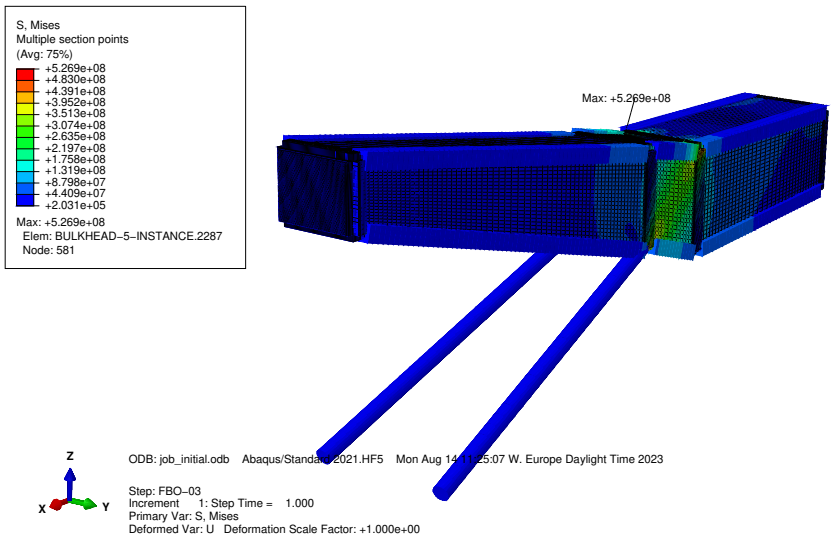
8.1.3. WEIGHT ESTIMATION USING PYLONDESIGNER APP

Finally, using the initial geometry, initial design vector and bounds as specified in the above, the sizing optimization case was run, using the optimization setting as explained in section 7.2 and Table 7.1. The optimization ran for a total duration of 67.9 hours on a 4-core 64GB ram PC provided by the TU Delft. The resulting convergence of the optimization is shown graphically in Figure 8.6. As can be seen in the plot, the optimization converged in 10 iterations, which was the maximum specified, obtaining a final optimum weight value of **236 kg** for an objective function value of 0.386. This is an improvement of roughly 61% with respect to the initial value. Table 8.4 shows the optimal design vector for which this design was obtained, including the distance to the lower and upper bounds. Note that due to an error in the code, the stringer parameters where not taken into account in the design vector, which means that their dimensions did unfortunately not change during the optimization. From this table, it is clearly visible that the main structural members that carry most of the loads are indeed the bulkheads, specifically the aft engine bulkhead, front wing bulkhead and the aft wing bulkhead. This is also where the constraints are approached to the closest, with a minimum constrain value of -3.71014492e-08 in *Bulkhead-3-instance*, which is located at the rear engine mounting location. From the loads analysis, it is found that for the limit load cases, a maximum Von Mises stress is found in load case CR-04 in this bulkhead, and likewise, for the ultimate load cases, the same bulkhead instance is found to be limiting, and the limiting load case in FBO-04. The full loads results for the optimized structure can be found at the end of this report in section D.1.

When we look at the result obtained by the sizing optimization, it is immediately clear that the obtained weight is significantly lower that the weight reported for the pylon by Safran. Where the reported weight lies around 500 kg, the weight which the optimizer finds in less than half of that at 236 kg. Though this might look a bit odd at first glance, the difference between the weight which the optimizer finds and the reported weight can actually be explained rather easily when considering the so-called *"fem weight-to-real weight"* ratio. By applying such a ratio, the weight found by analyzing a simplified structure using FEM is corrected for the necessary imperfections that are introduced into the system when integrating it into a real aircraft design. This can mean things like for example the weights of the rivets and bolts holding the structure together, or



(a) CR-04



(b) FBO-03

Figure 8.5: Resulting stress profiles in limit and ultimate load sizing load cases for the initial pylon geometry of the LEAP-1B.

Table 8.4: Optimal design variable and differences with respect to the initial value and the bounds.

Parameter	Optim vector	Diff initial DV	Diff LB	Diff UB
t_ribs_12	0.001	-90.00%	0%	98%
stringer_h_12	0.02	0.00%	100%	67%
stringer_t_12	0.005	0.00%	400%	83%
longeron_h_12	0.03313891	-33.72%	231%	67%
longeron_t_12	0.00519188	-48.08%	419%	90%
t_low_12	0.001	-96.67%	0%	98%
t_upp_12	0.00437317	-85.42%	337%	91%
t_left_12	0.00567886	-81.07%	468%	89%
t_right_12	0.00351611	-88.28%	252%	93%
t_bh1_12_web	0.01800112	-55.00%	1700%	77%
bh_flangew_st1_12	0.02904752	-27.38%	481%	42%
t_bh2_12_web	0.02371156	-40.72%	2271%	70%
bh_flangew_st2_12	0.02964536	-25.89%	493%	41%
t_ribs_23	0.001	-90.00%	0%	98%
stringer_h_23	0.02	0.00%	100%	67%
stringer_t_23	0.005	0.00%	400%	83%
longeron_h_23	0.05168971	3.38%	417%	48%
longeron_t_23	0.00703432	-29.66%	603%	86%
t_low_23	0.02397471	-20.08%	2297%	52%
t_upp_23	0.02677914	-10.74%	2578%	46%
t_left_23	0.0234127	-21.96%	2241%	53%
t_right_23	0.02342882	-21.90%	2243%	53%
t_bh1_23_web	0.03105048	3.50%	3005%	61%
bh_flangew_st1_23	0.02930597	-26.74%	486%	41%
t_bh2_23_web	0.01800237	-39.99%	1700%	77%
bh_flangew_st2_23	0.02977061	-25.57%	495%	40%
t_ribs_34	0.00391117	-60.89%	291%	92%
stringer_h_34	0.02	0.00%	100%	67%
stringer_t_34	0.005	0.00%	400%	83%
longeron_h_34	0.03828787	-23.42%	283%	62%
longeron_t_34	0.001	-90.00%	0%	98%
t_low_34	0.01110607	-62.98%	1011%	78%
t_upp_34	0.01390097	-53.66%	1290%	72%
t_left_34	0.01208976	-59.70%	1109%	76%
t_right_34	0.00943305	-68.56%	843%	81%
t_bh1_34_web	0.02971721	-0.94%	2872%	63%
bh_flangew_st1_34	0.03079453	-23.01%	516%	38%
t_bh2_34_web	0.02828727	-5.71%	2729%	65%
bh_flangew_st2_34	0.03146535	-21.34%	529%	37%
d_thrustlinks	0.01	-80.00%	0%	90%

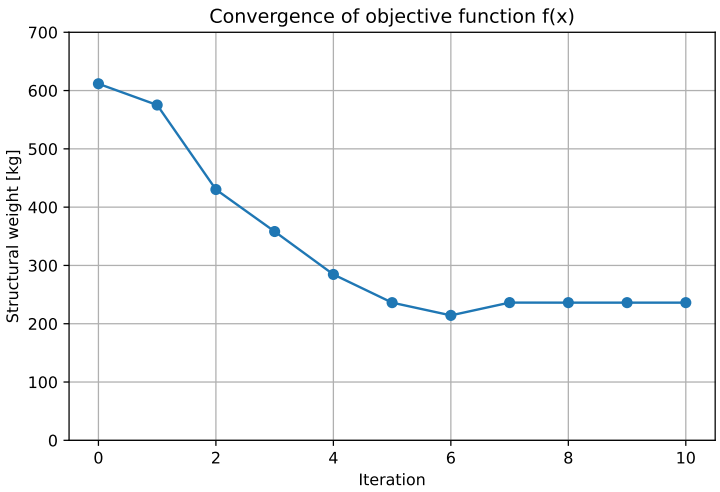


Figure 8.6: Convergence history of the objective function during sizing optimization of the LEAP-1B installation on the B737 MAX.

the holes that are needed to run fuel pipes and electricity through. Other installations that add weight to the structure can cover facilities related to fire safety and heat shields. Or facilities for ground handling and maintenance, like maintenance doors. In industry a fem-weight-to-real-weight factor in the order of 2 is not unusual, and applying this to the weight found using the PylonDesigner-app brings it much closer to the real value at **472 kg**, meaning the weight is underestimated by about 5.6%, well in the 10% range that was specified as acceptable for the purpose of this thesis project.

8.2. ENGINE INTEGRATION OF A LEAP-1A AIRCRAFT ON AN AIRBUS A320NEO AIRCRAFT

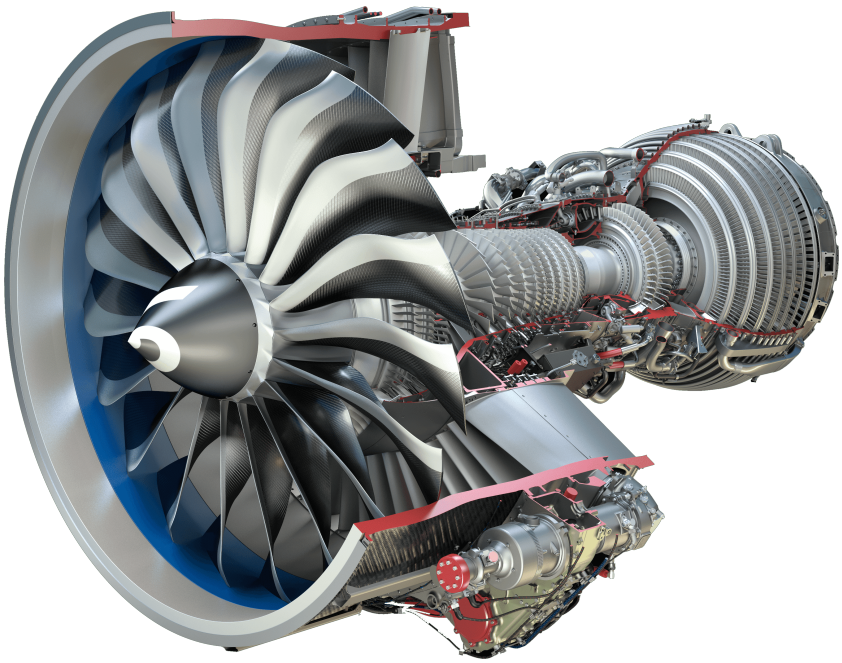
In the second case study that was performed in this thesis project, the pylon structure of the LEAP-1A engine as installed on the Airbus A320 neo is simulated. As for the LEAP-1B/B737 use case, the data for this analysis was provided by Safran Aircraft Engines.

8.2.1. PROBLEM DESCRIPTION

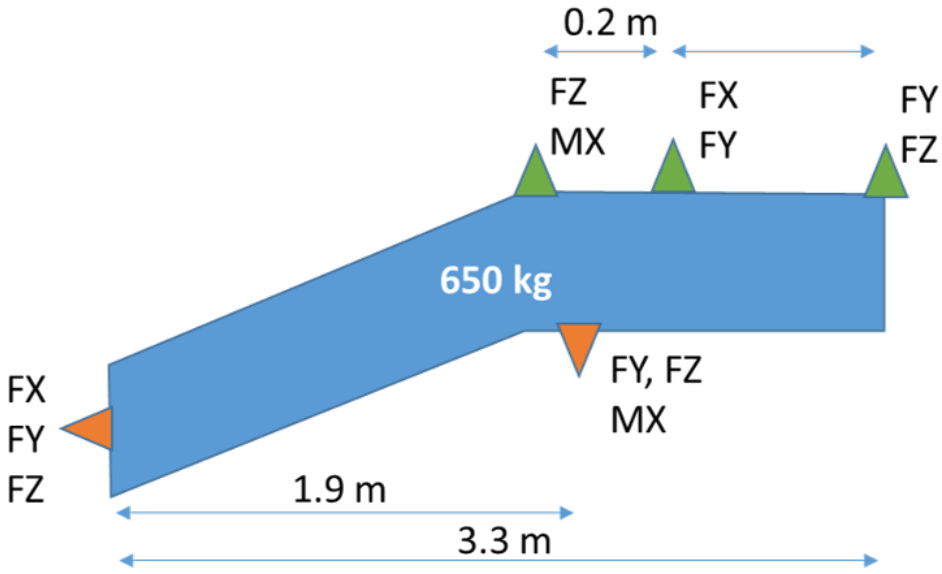
The LEAP-1A is the slightly larger brother of the LEAP-1B engine that was analyzed in the previous section, and powers the Airbus A319/320/321neo family. A cutaway view showing the engine is shown in Figure 8.7a. As is the case in the LEAP-1B, the LEAP-1A also features a two-spool turbofan configuration, although the fan of the LEAP-1A is slightly larger than that of the LEAP-1B, leading to a higher BPR but also a higher weight. Moreover, being the bigger engine, the LEAP-1A is also slightly more powerful than the LEAP-1B, with a maximum rated take-off thrust of around 143,000 N for the largest variant. The LEAP-1A produces this thrust while applying a BPR of up to 11:1 and an OPR of 38.5. As was done for the LEAP-1B, the characteristics of this engine are summarized in Table 8.5. Again, the engine integration follows the traditional UWN scheme, shown in Figure 8.8. The pylon that is applied by Airbus though applies a completely different design philosophy than the one taken by Boeing with their integration of the LEAP-1B. A schematic drawing of the LEAP-1A-A320neo pylon with data supplied by Safran is shown in Figure 8.7. As can be seen in the drawing, the engine installation applied by Airbus comes in the form of a core-mounted pylon that is slightly longer than the one used on the Boeing B737. Furthermore, the rear engine mount is placed aft of the front wing mounts, changing the layout of the pylon as was described in subsection 4.3.3. Finally, thrust links are omitted in this pylon design. As was the case for the Boeing pylon, this pylon is fully made of titanium and has a reported weight of around 650 kg.

8.2.2. INITIAL PROBLEM INVESTIGATION

Just like in the previous example, defining the full pylon geometry requires some input from the designer to complete the pylon geometry definition.



(a) LEAP-1A engine (cutaway view) [105]



(b) Pylon data (as provided by Safran Aircraft Engines)

Figure 8.7: LEAP-1A engine and its pylon as installed on the Airbus A320 neo aircraft.

Table 8.5: LEAP-1A30 characteristic data

Parameter	Value	Unit	Source
	Dimensions		
Weight	3,153	kg	[98][106]
Length	3,328	mm	[98][106]
Diameter	2.533	mm	[98][106]
Mounting type	Core	-	
	Thrust levels		
Max. continuous thrust	140,960	N	[98][106]
Max. continuous N1	100% at 3,856	rpm	[98][106]
Max. T/O thrust	143,050	N	[98][106]
Max. T/O N1	104% at 3,894	rpm	[98][106]
Max. reverse thrust	32.0% $F_{thr,max.cont}$	N	[101][100]
Max. reverse N1	68.4% $N1_{max.cont}$	rpm	[101][100]
	Thermodynamic data		
BPR	10.5:1	-	[102]
OPR	38.5:1	-	[102]
	Fan dimensions		
No blades	18	-	[103]
Blade material	3D woven resin transfer moulded (RTM) carbon fibre	-	[103] [104]
Blade mass	4.2	kg	[103]
Blade height		mm	GTpy calculation
	Engine COG location data		
X_eng	-2.0	m	Estimation
Y_eng	6.1	m	Estimation
Z_eng	-0.9	m	Estimation

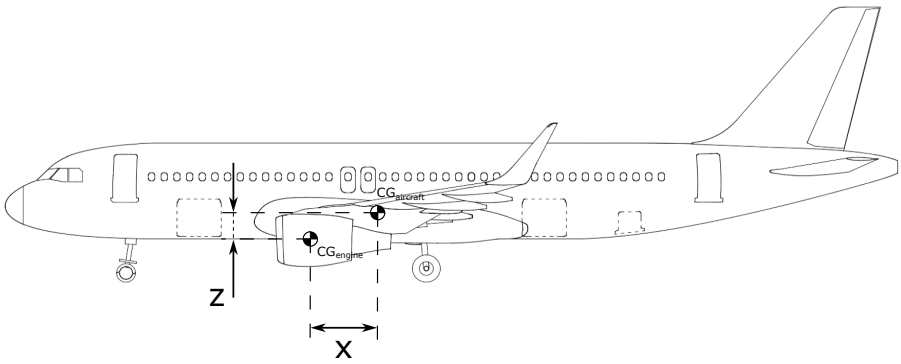


Figure 8.8: LEAP-1A installation on the A320 neo (adapted from [29]).

Table 8.6: Pylon zero-thickness geometry used for the analysis of the LEAP-1A installation on the A320

x_{12}	z_{12}	y_1	x_{23}	z_{23}	y_2	x_{34}	z_{34}	y_3	y_4
1.9	-0.1	0.4	-0.2	0.55	0.4	1.6	0	0.3	0.3
h_1	d_1	w_1	h_3	d_3	w_3	h_4	d_4	w_4	ϕ
0.2	0.1	0.25	0.25	0.05	y_3	0.3	0.05	y_4	0
$n_{stringers}$	$n_{ribs_{A,C}}$	n_{ribs_B}							
0	4	2							

GEOMETRY GENERATION USING THE GEOMETRYGENERATOR

As the pylon does not feature stringers, the number of stringers was set to 0 for the analysis. The number of ribs was set to 4 for pylon box 1 and the box-beam compartments, and again 2 for the middle compartment. All other definitions were kept at the same values that were applied in the LEAP-1B case study, leading to a full geometry definition as summarized in Table 8.6. The resulting initial pylon model generated by ParaPy is shown in Figure 8.9.

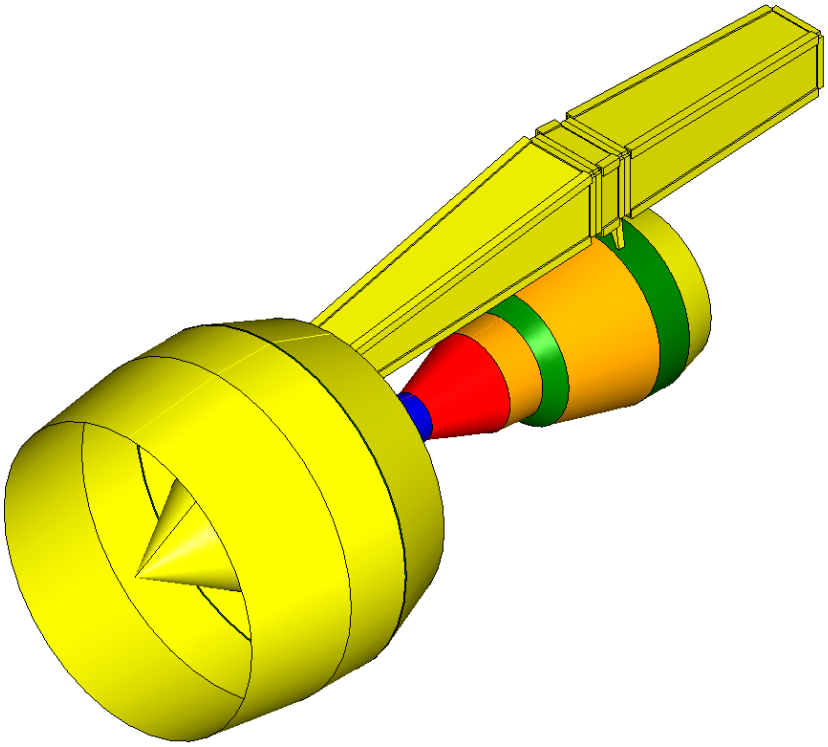
MESH SENSITIVITY STUDY

Similar as was described in the previous case study, a mesh sensitivity study was performed on the initial geometry to find the mesh density that is a good balance between calculation speed and accuracy of the solution. The results of this sensitivity study can be found in Figure 8.10.

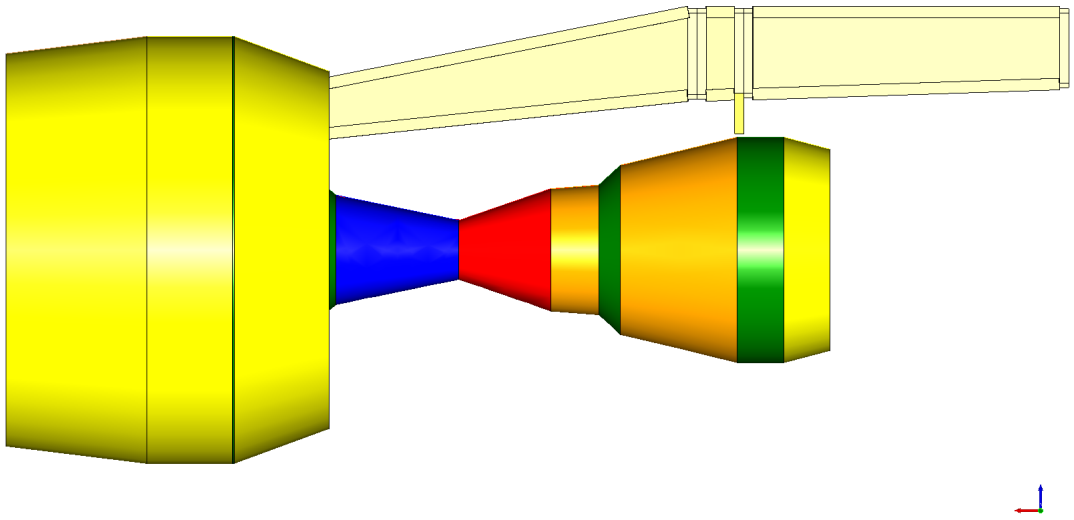
From this study, it is quite clear that obtaining mesh convergence for this case is especially difficult. Even with 40 lateral elements, the maximum stress found in the box-beam still seems to diverge, even though it looks like a platform is eminent. As the simulation time is growing equally rapid with every mesh densification step, finding the right mesh size proved to be quite difficult for this use case. Eventually a mesh density of 32 lateral nodes or 104320 elements was chosen for the analysis. Even though, it is clear that this could lead to a relatively large underestimation of the peak stress in the pylon structure.

8.2.3. WEIGHT ESTIMATION USING PYLONGENERATOR APP

Following the initial problem investigation and the mesh sensitivity study, a sizing optimization procedure was started for this use case. Unfortunately, it was found that the optimization was not converging due to failure of the underlying finite element simulation. Searching deeper into the underlying files, it was observed that after the third iteration, the simulation did not converge anymore, leading to extremely high computational times. Checking the data files that Abaqus generates during the simulation taught that the simulation was getting stuck at the solution of the second FBO-load case, with Abaqus prompting a “TIME INCREMENT REQUIRED IS LESS THAN THE MINIMUM SPECIFIED” error. This usually occurs when the simulation runs into local buckling effects, which seemed to be true for the ribs in this simulation. Eventually, it was decided to stop the optimization process prematurely as it was clear that it was not going to deliver satisfactory results. Investigating the observed behavior unfortunately is left as a recommendation for future work, given the time constraints on this thesis project.



(a) Isometric view



(b) Side view

Figure 8.9: Initial geometry for the LEAP-1A pylon as generated by the GeometryGenerator and engine geometry generated using GTpy.

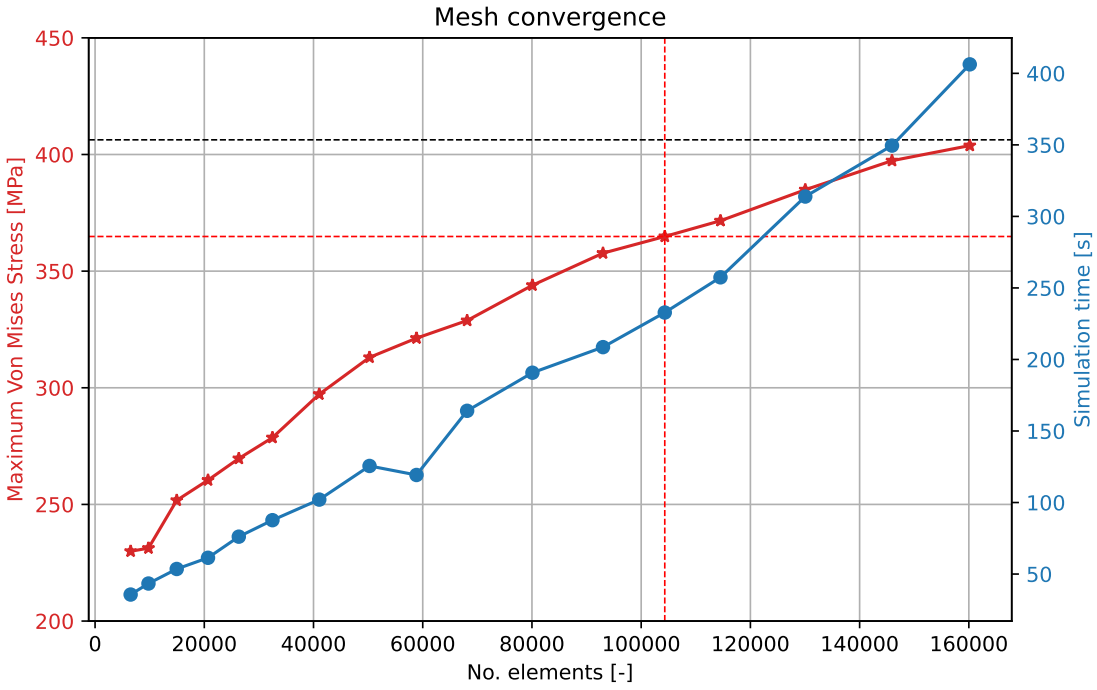


Figure 8.10: Maximum Von Mises stress and analysis execution time with varying mesh densities.

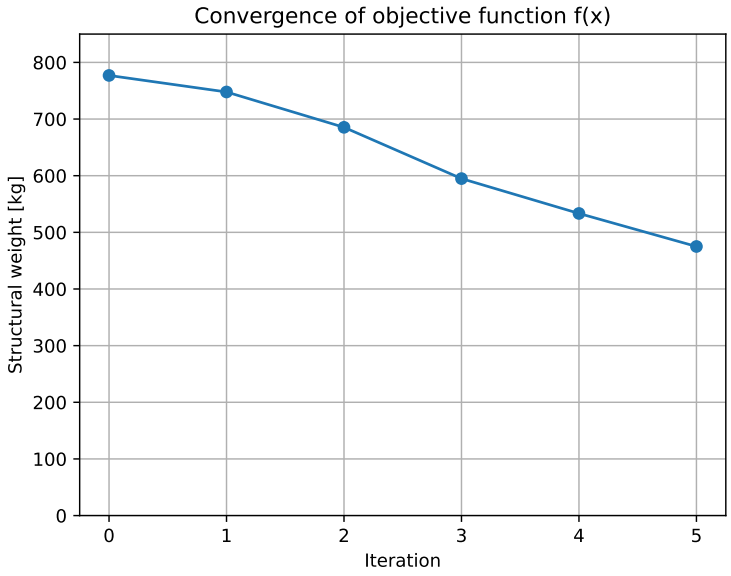


Figure 8.11: Convergence history of the objective function during sizing optimization of the LEAP-1A installation on the A320 neo.

9

Conclusion and Recommendations

9.1. Conclusion of the thesis project

At the end of this report, all information presented in the previous chapters is re-evaluated and summarized into a conclusion that reveals the final answers to the research questions posed at the start of this report. The goal of this thesis project was to develop a new, physics-based, design-sensitive engineering tool that can be applied during the preliminary design of engine-aircraft integrations, or to investigate the relations between design choices made during the early conceptual- and preliminary design phases of both engine and aircraft and their integration, and the effect of these design choices on the structural weight of the engine support structure.

In order to obtain the right knowledge that is required to develop such a tool, first an extensive literature review was executed. First, the general design process of an engine-integration in industry was investigated. It was found that although in early engine integration design, most emphasis was put on aerodynamics, modern engine integration design techniques like multidisciplinary design and optimization (MDO) and knowledge-based engineering (KBE) have developed into vital tools that are more and more applied in current day engine integration design. Focusing more on the structural details of the engine integration, several types of structures where found to have been applied over the past 50 years, yet it is clear that for modern pylons, the box-beam architecture has gained significant momentum and is now widely applied in engine support structures. The relevant loads for engine support structure were found to consist of mainly engine loads like thrust and weight, inertia loads and gyroscopic loads during maneuvers. One especially important load case for the development of pylon structures is the so-called *Fan Blade-off* event, which features the violent release of a fan blade following foreign object damage of fan blade fatigue. Unfortunately though, the analysis and simulation of this load case poses significant challenges and requires excellent cooperation between engine- and airframe manufacturers. Finally, the current state-of-the-art in conceptual- and preliminary weight estimation methods was investigated, which showed that most classical conceptual- and preliminary weight estimation method lack the flexibility to incorporate the effects of modern technologies. Physics-based class 2.5 weight estimation methods, especially when applied in the context of KBE applications, can form an important asset in the future, unfortunately with respect to engine integration, a proper physics-based class 2.5 weight estimation method is not yet available, which gives rise to the research performed during this thesis project.

To fill this capability and knowledge gap, a Python-based software tool employing knowledge-based engineering principles is proposed. At the basis of this software tool lies a design methodology that exists of four basic ingredients;. First of all, a mathematical representation of the engine by means of a parameterization needs to be defined. Then, this parameterization needs to be captured in a software app built on a proven ecosystem that can facilitate and supply the required functionality needed to incorporate KBE principles into the tool. The ParaPy KBE platform is used for this. Then, the geometry generated inside the KBE app needs to be evaluated using a finite element code to ensure that is is structurally sound. For that, a dedicated API is built that can communicate between the code in ParaPy and the chosen commercial finite element method software Abaqus. And finally, a sizing optimization scheme needs to be implemented to ensure that

the structure generated in the ParaPy app is not just any structure that can hold the relevant loads, but in fact that structure that would also be applied in reality when a airframe manufacturer is developing a new support structure for an engine and aircraft combination.

The project starts with the definition of the mathematical parameterization of the pylon. The proposed parameterization consists of two parts - first of all, the 3D pylon geometry is defined using a zero-thickness parameterization that is comprised of parameters describing the locations of the pylon's hardpoints, the section geometry at prescribed locations and finally the orientation angle of the pylon with respect to the aircraft. On top of this, the structural components are parameterized using specific parameters describing the length, width and thickness of the components. Finally, a special condition is proposed to define the topology of the pylon and the relation of the 5 identified section planes with respect to each other. This leads to a full parameterization of the pylon by in total 80 parameters, that fully define the geometry of the pylon structure.

Then, the structure of the KBE application in ParaPy is described. It consists of a set of ParaPy class and functions, with at the top the superclass `PylonDesigner` in which the entire method is codified. Inside this superclass, the pylon geometry is generated using a dedicated `GeometryGenerator` that can use several specialized Python classes to generate the required geometry. Furthermore, it encompasses functions to obtain the weight of the structure under consideration, perform structural analysis and finally execute the sizing optimization using SciPy Optimize.

For structural analysis, the `PylonDesigner`-app is coupled with Abaqus via an API that is developed as part of the thesis project, and uses a combination of classes from the ParaPy/Abaqus library developed by ParaPy and several especially created Python functions. The entire process of defining the geometry, meshing the geometry, processing the meshes, tieing the structures, applying boundary conditions and loads and finally executing the analysis is performed inside this Abaqus API. Furthermore, a complete set of load cases covering both limit and ultimate loads that are relevant for the design of the pylon is defined, leading to a total of 24 load cases is defined this way, of which 20 are limit load cases, and 4 cover the ultimate loads experienced by the pylon in case of an FBO-event. Finally, the stresses calculated by Abaqus are post-processed and made ready for further use by means of a separate Python script, which also had to be developed during the coarse of this thesis project.

The stresses obtained during the structural analysis with Abaqus are combined with the weight found in ParaPy in a sizing optimization procedure, in which the structural members of the pylon are sized for minimum structural weight. The stresses that are formed in the structure due to the limit and ultimate loads form the constraints for the optimization, and are applied in the form of inequality constraints.

Finally the methodology is applied in two case study to validate the working of the tools and identify possible points of attention. First, the pylon structure of the LEAP-1B engine as it is integrated into the Boeing B737 MAX is investigated. It is found that when applying a medium density mesh with a total of 58872 elements, the sizing optimization is able to converge and provide an estimation of the required pylon weight for that engine-aircraft combination. At the same time, it is noted that the weight obtained thus is significantly lower than the weight reported by Safran. This is explained by the concept of the 'fem weight to real weight' factor, which corrects the weight found by analyzing a simplified structure using FEM for the necessary imperfections that are introduced into the system when integrating it into a real aircraft design, like for example the weights of the rivets and bolts holding the structure together, holes that are needed to run fuel pipes and electricity through, installations for fire safety, ground handling and maintenance, and so on. In industry a fem-weight-to-real-weight factor in the order of 2 is not unusual, and applying this to the weight found using the PylonDesigner app brings it much closer to the real value.

Moreover, the method is also applied in a second case study that analyzes the installation of the LEAP-1A engine onto the Airbus A320 neo aircraft. Here, the simulation was unfortunately not able to converge, and closer inspection of the intermediate results indicated that the simulation was failing due to local instabilities in the panels due to local buckling, which was not taken into account in the development of the method.

With this, all the research questions posed at the beginning of this thesis in chapter 1 have been answered successfully. That concludes the research done in this thesis project.

9.2. RECOMMENDATIONS FOR FUTURE RESEARCH

Finally, based on the experiences gained during this thesis project, the assumptions made during the development of the tool and the difficulties experienced during the validation of the tool, some recommendations for future research and improvement of the tool can be made.

First of all, this tool was developed in the context of a master thesis project. Given the complexity of the material and the many different aspects that play a role in the design and analysis of the pylon structure, this setup is clearly far from ideal. This means that in order to finish the project in time, some assumptions had to be made that definitely require further validation in the future.

One important part of this is the modeling and implementation of a good Fan blade-off analysis model. In the current version of the application, the forces generated by the FBO model have been largely neglected, apart from the unbalance load which is in itself greatly simplified to incorporate this into a static structural optimization. The implementation of a good rotordynamics model that can be used prior to the sizing optimization procedure to find a better load set for the ultimate load cases is to be desired.

Speaking more generally, it is clear that the complexity of the (nonlinear dynamic) loads on the structure require more attention. For example, important failure modes like local buckling, stringer buckling and fatigue are currently not taken into account in the method. This has led to some significant problems during the validation, and many attempts to obtain a converged sizing optimization solutions have failed due to the occurrence of these types of local structural phenomena. Since this project was executed in the context of a master thesis by a student from the Flight Performance and Propulsion track, and not per se as purely a Structural Engineering thesis, the detailed implementation of these types of effects was considered secondary to the workings of the ecosystem. Nonetheless, it is recognized that this is a component of the methodology that clearly needs further work.

Finally, looking to the future, the application of this tool in the context of a research project investigating the effects of changing characteristic design variables of the engine and the engine installation onto the aircraft is considered an enormous opportunity. Unfortunately, further research into these effects was not possible during this thesis project due to lack of time and the unavailability of a proper engine model including the internal components of the engine. But this could definitely lead to some very interesting insights.

BIBLIOGRAPHY

[1] Airbus, *FAST 65 - Flight Airworthiness Support Technology - Airbus Technical Magazine*, edited by D. Buckler, no. 65 (2020).

[2] D. S. Lee, D. W. Fahey, A. Skowron, M. R. Allen, U. Burkhardt, Q. Chen, S. J. Doherty, S. Freeman, P. M. Forster, J. Fuglestvedt, A. Gettelman, R. R. De León, L. L. Lim, M. T. Lund, R. J. Millar, B. Owen, J. E. Penner, G. Pitari, M. J. Prather, R. Sausen, and L. J. Wilcox, *The contribution of global aviation to anthropogenic climate forcing for 2000 to 2018,* Atmospheric Environment **244**, 117834 (2021).

[3] J.-F. Brouckaert, F. Mirville, K. Phuah, and P. Taferner, *Clean Sky research and demonstration programmes for next-generation aircraft engines,* Aeronautical Journal **122**, 1163 (2018).

[4] D. P. Raymer, *Aircraft design : a conceptual approach*, 6th ed., edited by J. A. Schetz (American Institute of Aeronautics and Astronautics, Reston, VA, 2018) p. 729.

[5] S. Mouton, J. Laurenceau, and G. Carrier, *Aerodynamic and structural optimization of powerplant integration under the wing of a transonic transport aircraft,* in *Proceedings of 42th AAAF symposium on applied aerodynamics, Nice* (2007).

[6] S. Grihon, M. Meaux, A. Lucchetti, P. Sarouille, J. Laurenceau, G. Carrier, and S. Mouton, *Pylon Multidisciplinary Optimization,* in *Advances in Collaborative Civil Aeronautical Multidisciplinary Design Optimization*, Vol. 233, edited by E. Kesseler and M. D. Guenov (American Institute of Aeronautics and Astronautics, Reston, VA, 2010) Chap. 8, pp. 249–288.

[7] A. Gazaix, F. Gallard, V. Ambert, D. Guénot, M. Hamadi, S. Grihon, P. Sarouille, T. Y. Druot, J. Brézillon, V. Gachelin, J. Plakoo, N. Desfachelles, N. Bartoli, T. Lefebvre, S. Gürol, B. Pauwels, C. Vanaret, and R. Lafage, *Industrial Application of an Advanced Bi-level MDO Formulation to Aircraft Engine Pylon Optimization,* in *AIAA Aviation Forum 2019* (American Institute of Aeronautics and Astronautics (AIAA), Dallas, TX, 2019).

[8] M. C. Niu, *Airframe Structural Design*, 3rd ed. (Conmilit Press Ltd., Hong Kong, 1988) pp. 1–612.

[9] P. A. Ward, P. A. Mucklow, and K. Herbstritt, *Design for installation of new transport engines*, Tech. Rep. (SAE Technical Papers No. 680334, New York, 1968).

[10] A. A. Verri, F. L. S. Bussamra, T. R. Spinelli, and J. A. Hernades, *Design and Structural Analysis of a Pylon of an Aircraft With Engines At the Tail Cone,* in *COBEM 2009 - 20th International Congres of Mechanical Engineering* (Associação Brasileira de Engenharia e Ciências Mecânicas (ABCM), Gramado, Brazil, 2009).

[11] M. C. Y. Niu, *Airframe Stress Analysis And Sizing*, 3rd ed. (Hong Kong Conmilit press Ltd., Hong Kong, 2011).

[12] R.-R. Plc., *Trent-1000 cutaway,* (2019).

[13] T. Megson, *Aircraft Structures for Engineering Students*, 6th ed. (Elsevier Ltd, Kidlington, Oxford, UK, 2013).

[14] M. Heidari, D. L. Carlson, S. Sinha, R. Sadeghi, C. Heydari, H. Bayoumi, and J. Son, *An efficient multidisciplinary simulation of engine fan-blade out event using MD nastran,* in *AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference* (AIAA, Schaumburg, IL, 2008).

[15] I. Armendáriz, J. Olarrea, and J. García-Martínez, *Parametric analysis of a highly dynamical phenomena caused by a propeller blade loss,* Engineering Failure Analysis **57**, 528 (2015).

[16] L. R. Jenkinson, P. Simpkin, and D. Rhodes, *Civil jet aircraft design* (American Institute of Aeronautics and Astronautics, 1999) p. 418.

[17] E. Torenbeek, *Synthesis of Subsonic Airplane Design*, 9th ed. (Delft University Press, Kluwer Academic Publishers, Delft, The Netherlands, 1982) pp. 1–598.

[18] M. Stavreva, *Structural Sizing Method for Propulsive Empennage System*, Master thesis, Delft University of Technology (2020).

[21] G. La Rocca, *Knowledge Based Engineering Techniques to Support Aircraft Design and Optimization*, Doctoral thesis, Delft University of Technology, Delft (2011).

[22] G. La Rocca, *Knowledge based engineering: Between AI and CAD. Review of a language based technology to support engineering design,* Advanced Engineering Informatics **26**, 159 (2012).

[19] D. N. Mavrist, D. A. Delaurentis, O. Bandte, and M. A. Hale, *A stochastic approach to multi-disciplinary aircraft analysis and design,* 36th AIAA Aerospace Sciences Meeting and Exhibit (1998), 10.2514/6.1998-912.

[20] D. Zaccai, *Design Framework for Trailing Edge High-Lift Systems a Knowledge Based Engineering Application*, Master thesis, Delft University of Technology, Delft (2014).

[23] P.-J. Proesmans, *Preliminary Propulsion System Design and Integration for a Box-Wing Aircraft Configuration: A Knowledge-Based Engineering Approach*, Master thesis, Delft University of Technology, Delft, NL (2019).

[24] NTSB Bureau of Accident Investigation, *Aircraft accident report. American Airlines DC-10-10, N110AA. Chicago O'Hare International Airport. Chicago, IL, May 25 1979.*, Tech. Rep. (National Transportation Safety Board, Washington, DC., 1979).

[25] CFMI Customer Training Center, *CFM56-7B Nacelle Traning Manual*, Tech. Rep. (CFM International, 2003).

[26] D. Bettebghor, C. Blondeau, D. Toal, and H. Eres, *Bi-objective optimization of pylon-engine-nacelle assembly: weight vs. tip clearance criterion,* Structural and Multidisciplinary Optimization **48**, 637 (2013).

[27] Dassault Sysèmes, *SIMULIA User Assistance 2021 - ABAQUS Documentation,* (2021).

[28] The Boeing Company, *Airplane characteristics for airport planning B737,* (2023).

[29] Airbus S.A.S., *A320 Aircraft Characteristics Airport and Maintenance Planning,* (2022).

[30] *Flightglobal Image Archive,* (2011).

[31] R. Wanhill and A. Oldersma, *NLR Technical Publication*, Tech. Rep. (National Aerospace Laboratory (NLR), Amsterdam, The Netherlands, 1996).

[32] M. Stefanovic and E. Livne, *On the structural design synthesis of aircraft engine pylons at a certification level of detail,* in *AIAA Scitech 2020 Forum*, Vol. 1 Part F (American Institute of Aeronautics and Astronautics Inc, AIAA, Orlando, FL, 2020).

[33] ANSYS Inc., *Granta EduPack 2021 R2,* (2021).

[34] IATA, *Annual Review 2022*, Tech. Rep. (International Air Transport Association, Doha, 2022).

[35] International Energy Agency, *Energy Technology Perspectives 2020* (Paris, France, 2020).

[36] TU Delft and NLR, *Towards A Sustainable Air Transport System* (Amsterdam, The Netherlands, 2021).

[37] X. S. Zheng and D. Rutherford, *Fuel burn of new commercial jet aircraft: 1960 to 2019*, Tech. Rep. (International Council on Clean Transportation, Washington, DC, 2020).

[38] P. Petitcolin and S. Cueille, *Open Rotor Istres Press Update,* (2017).

[39] J. Roskam, *Airplane Design - Part I: Preliminary sizing of airplanes* (Roskam Aviation and Engineering Corporation, Ottawa, Kansas, 1985).

[40] J. Roskam, *Airplane Design - Part III: Layout Design of Cockpit, Fuselage, Wing and Empennage: Cutaways and Inboard profiles* (Roskam Aviation and Engineering Corporation, Ottawa, KS, 1986).

[41] D. L. Berry, *The Boeing 777 Engine/aircraft integration aerodynamic design process,* in *19th Congress of the International Council of Aeronautical Sciences (ICAS)*, Boeing Commercial Airplane Group (International Council of Aeronautical Sciences, Anaheim, CA, USA, 1994) pp. 1305–1320.

[42] W. P. Henderson, *Propulsion-Airframe Integration for Commercial and Military Aircraft,* SAE Transactions **96**, 1775 (1987).

[43] J. R. Carlson and M. Lamb, *Integration effects of pylon geometry and rearward mounted nacelles for a high-wing transport,* in *AIAA/ASME/SAE/ASEE 23rd Joint Propulsion Conference, 1987* (American Institute of Aeronautics and Astronautics, San Diego, CA, USA, 1987).

[44] D. Naik, *Innovative pylon concepts for engine-airframe integration for transonic transports,* in *AIAA 20th Fluid Dynamics, Plasma Dynamics and Lasers Conference, 1989* (1989).

[45] W. P. Henderson, *Airframe/propulsion integration at transonic speeds,* Journal of Engineering for Gas Turbines and Power **113**, 51 (1991).

[46] AGARD, *Aerodynamic Engine/Airframe Integration for High Performance Aircraft and Missiles,* in *AGARD conference proceedings 498*, edited by P. Sacher, R. Decuypere, L. Chan, M. Leynaert, H. Korner, G. Buccrantini, A. Elsenaar, J. Simon, P. Bignell, D. Bowers, J. Brandley, and I. Fottner (AGARD, Forth Worth, 1992).

[47] A. Magrini, E. Benini, H.-D. Yao, J. Postma, and C. Sheaf, *A review of installation effects of ultra-high bypass ratio engines,* Progress in Aerospace Sciences **119**, 100680 (2020).

[48] J. Sobieszczanski-Sobieski, A. Morris, M. J. van Tooren, G. La Rocca, and W. Yao, *Multidisciplinary Design Optimization Supported by Knowledge Based Engineering* (Wiley, Chichester, UK, 2015).

[49] Australian Transport Safety Bureau, *Engine pylon cracking involving Boeing 747-438 , VH-OJT*, Tech. Rep. (Hong Kong, CH, 2016).

[50] EASA, *CS-25 Easy Access Rules for Large Aeroplanes (Amendment 27),* (2022).

[51] S. Bogos, *The rotational velocities evaluation for the engine mounts gyroscopic loads,* INCAS BULLETIN **5**, 9 (2013).

[52] M. J. Stallone, V. Gallardo, A. F. Storace, L. J. Bach, G. Black, and E. F. Gaffney, *Blade Loss Transient Dynamic Analysis of Turbomachinery.* in *18th Joint Propulsion Conference* (American Institute of Aeronautics and Astronautics, Cleve, 1982).

[53] C. Lawrence, K. Carney, and V. Gallardo, *Simulation Structural of Aircraft Dynamics Engine - NASA/TM-2001-210957*, Tech. Rep. (NASA Glenn Research Center, Cleveland, OH, 2001).

[54] K. S. Carney, C. Lawrence, and D. V. Carney, *Aircraft Engine Blade-Out Dynamics,* in *7th International LS-DYNA Users Conference* (Livermore Software Technology Corporation, Detroit, MI, 2002) pp. 14–17.

[55] J. B. Husband, *Developing an Efficient FEM Structural Simulation of a Fan Blade Off test in a Turbofan jet engine*, Doctoral thesis, University of Saskatchewan, Saskatoon, Saskatchewan, Canada (2007).

[56] M. A. Heidari, D. L. Carlson, and T. Yantis, *Rotor-dynamics Analysis Process,* in *MSC Worldwide Aerospace Conference and Technology Showcase* (2001) pp. 1–16.

[57] S. K. Sinha and S. Dorbala, *Dynamic loads in the fan containment structure of a turbofan engine,* Journal of Aerospace Engineering **323**, 260 (2009).

[58] S. K. Sinha, *Rotordynamic analysis of asymmetric turbofan rotor due to fan blade-loss event with contact-impact rub loads,* Journal of Sound and Vibration **332**, 2253 (2013).

[59] I. Armendáriz, J. López, J. Olarrea, M. Oliver, and H. Climent, *Case study: Analysis of the response of an aircraft structure caused by a propeller blade loss,* Engineering Failure Analysis **37**, 12 (2014).

[60] I. Armendáriz, J. Olarrea, and J. García-Martínez, *Engine to wing structural design under critical loads caused by a propeller blade loss,* Engineering Structures **158**, 155 (2018).

[61] J. Roskam, *Airplane Design - Part II: Preliminary Configuration Design and Integration of the Propulsion system* (Roskam Aviation and Engineering Corporation, Ottawa, KS, 1985).

[62] A. Elham, *Weight Indexing for Multidisciplinary Design Optimization of Lifting Surfaces*, Doctoral thesis, Delft University of Technology (2012).

[63] J. Roskam, *Airplane Design - Part V: Component Weight Estimation* (Roskam Aviation and Engineering Corporation, Ottawa, KS, 1985).

[64] T. C. Corke, *Design of Aircraft* (Pearson, 2003) pp. 1–408.

[65] R. Schmitt, K. Foreman, W. Gertsen, and P. Johnson, *Weight Estimation Handbook for Light Aircraft* (Cessna Aircraft Company, 1959).

[66] L. Nicolai, *Fundamentals of Aircraft Design* (METS Inc., San Jose, CA, 1982).

[67] D. P. Raymer, J. Wilson, H. D. Perkins, A. Rizzi, M. Zhang, and A. R. Puentes, *Advanced Technology Subsonic Transport Study N+3 Technologies and Design Concepts - NASA/TM-2011-217130*, Tech. Rep. (NASA Glenn Research Center, Cleveland, OH, 2011).

[68] R. Staton, *Cargo/Transport Statistical Weight Estimation Equations,Vought Aircraft Rept. 2-59320 9R-50549*, Tech. Rep. (Vought Aeronautics Division, 1969).

[69] A. Jackson and R. Christian, *Preliminary Design Weight Estimation Program (SW-007). Report S11-009.*, Tech. Rep. (North American Rockwell. Aero Commander Division, Thousand Oaks, CA, USA, 1971).

[70] D. P. Wells, B. L. Horvath, and L. A. McCullers, *The Flight Optimization System Weight Estimation Method. NASA TM-2017-219627*, Tech. Rep. (NASA Langley Research Center, Hampton, VA, 2017).

[71] W. Liu, *Pylon and Nacelle Weight estimation. MEMO 0998*, Tech. Rep. (DAR Corporation, Lawrence, KA, 2014).

[72] A. Elham, G. La Rocca, and M. J. Van Tooren, *Development and implementation of an advanced, design-sensitive method for wing weight estimation,* Aerospace Science and Technology **29**, 100 (2013).

[73] A. Elham and M. J. Van Tooren, *Effect of wing-box structure on the optimum wing outer shape,* Aeronautical Journal **118**, 1 (2014).

[74] A. Elham and M. van Tooren, *Beyond Quasi-Analytical Methods for Preliminary Structural Sizing and Weight Estimation of Lifting Surfaces,* in *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference* (American Institute of Aeronautics and Astronautics, Reston, Virginia, 2015).

[75] A. Elham and M. Van Tooren, *Tool for preliminary structural sizing, weight estimation, and aeroelastic optimization of lifting surfaces,* Journal of Aerospace engineering **230**, 280 (2016).

[76] A. H. Van Der Laan, *Knowledge Based Engineering Support for Aircraft Component Design*, Doctoral thesis, Delft University of Technology (2007).

[77] G. La Rocca and M. J. van Tooren, *Enabling distributed multi-disciplinary design of complex products: a knowledge based engineering approach,* Journal of Design Research **5**, 333 (2007).

[78] F. Bertels, *Design framework for flap system kinematics; a knowledge based engineering application*, Master thesis, Delft University of Technology (2012).

[79] D. Zaccai, F. Bertels, and R. Vos, *Design methodology for trailing-edge high-lift mechanisms,* CEAS Aeronautical Journal **7**, 521 (2016).

[80] B. Nagel, D. Bohnke, V. Gollnick, P. Schmollgruber, J. Alonso, A. Rizzi, and G. La Rocca, *Communication in Aircraft Design: Can we establish a Common Language?* in *Proceedings of the 28th congress of the International Council of the Aeronautical Sciences*, edited by I. Grant (ICAS, Brisbane, 2012) pp. 1–13.

[81] W. Visser, *Generic Analysis Methods for Gas Turbine Engine Performance*, Doctoral thesis, Delft University of Technology (2015).

[82] A. B. Lambe and J. R. Martins, *Extensions to the design structure matrix for the description of multidisciplinary design, analysis, and optimization processes,* Structural and Multidisciplinary Optimization **46**, 273 (2012).

[83] N. Zhao, N. Li, Y. Sun, and Z. Gao, *4D Trajectory Planning of Aircraft Taxiing considering Time and Fuel,* Mathematical Problems in Engineering (2020), 10.1155/2020/9603968.

[84] S. Cho and K. K. Choi, *Design sensitivity analysis and optimization of non-linear transient dynamics. Part I - Sizing Dynamics,* International Journal for Numerical Methods in Engineering **48**, 351 (2000).

[85] B.-S. Kang, G.-J. Park, and J. S. Arora, *A review of optimization of structures subjected to transient loads,* Structural and Multidisciplinary Optimization 2006 31:2 **31**, 81 (2006).

[86] Y. I. Kim and G. J. Park, *Nonlinear dynamic response structural optimization using equivalent static loads,* Computer Methods in Applied Mechanics and Engineering **199**, 660 (2010).

[87] G.-J. Park, *Technical overview of the equivalent static loads method for non-linear static response structural optimization,* Structural and Multidisciplinary Optimization 2010 43:3 **43**, 319 (2010).

[88] G. R. Liu and S. S. Quek, *The Finite Element Method: A Practical Course: Second Edition*, 2nd ed. (Butterworth-Heinemann, Oxford, UK, 2013).

[89] D. T. Bran, C. F. Elefterie, and B. Ghiban, *Aeronautical Industry Requirements for Titanium Alloys,* IOP Conference Series: Materials Science and Engineering **209**, 1 (2017).

[90] I. Inagaki, T. Takechi, S. Yoshihisa, and N. Ariyasu, *Application and Features of Titanium for the Aerospace Industry,* Nippon Steel Sumitomo Metal Technical Report , 22 (2014).

[91] North Steel Company, *Titanium use in aeroplanes,* (2018).

[92] H. Khalid and A. Gomez-Gallegos, *Substituting Ti-64 with Aa2099 as Material of a Commercial Aircraft Pylon,* Advances in Materials Science **21**, 77 (2021).

[93] N. M. Newmark, *A Method of Computation for Structural Dynamics,* Journal of the Engineering Mechanics Division **85**, 67 (1959).

[94] H. M. Hilber and T. J. Hughes, *Collocation, dissipation and [overshoot] for time integration schemes in structural dynamics,* Earthquake Engineering Structural Dynamics **6**, 99 (1978).

[95] G. Puri, *Python Scripts for Abaqus - Learn by Example*, 1st ed. (2011).

[96] D. Kraft, *A Software Package for Sequential Quadratic Programming,* (1988).

[97] CFM International, *LEAP-1B schematic drawing,* (2017).

[98] M. R. V. Holsteijn, *Finding the Operating Limits and Optimal Configuration of an Electrically Assisted Turbofan*, Master thesis, Delft University of Technology (2018).

[99] EASA, *Type-certificate data sheet for engine LEAP-1B series engines (No. E.115),* (2022).

[100] G. J. Noel and E. Boeker, *Thrust Reverser Analysis for Implementation in the Aviation Environmental Design Tool (AEDT)*, Tech. Rep. (U.S. Department of Transportation Research and Innovative Technology Administration, 2007).

[101] J. A. Yetter, *Nasa Technical Memorandum 109158*, Tech. Rep. (NASA, Hampton, Virginia, 1995).

[102] ICAO, *ICAO Aircraft Engine Emissions Databank,* (2017).

[103] G. Gardiner, *3-D preformed composites: The leap into LEAP,* (2014).

[104] Safran Aircraft Engines, *LEAP-1B, a new-generation engine for the B737 MAX,* (2017).

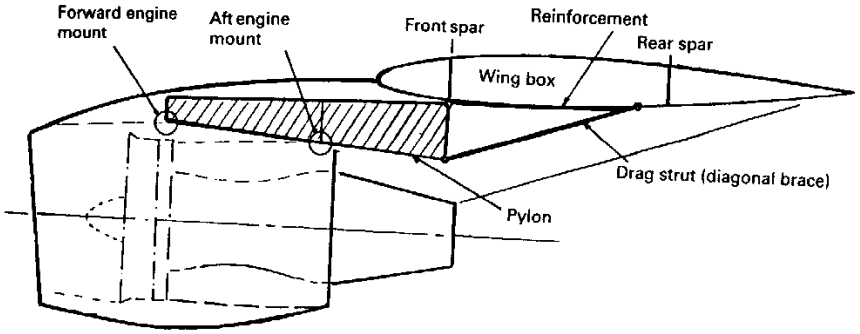
[105] CFM Internatonal, *LEAP-1A cutaway drawing,* (2018).

[106] EASA, *Type-certificate data sheet for engine LEAP-1A & LEAP-1C series engines (No. EASA.E.110),* (2022).

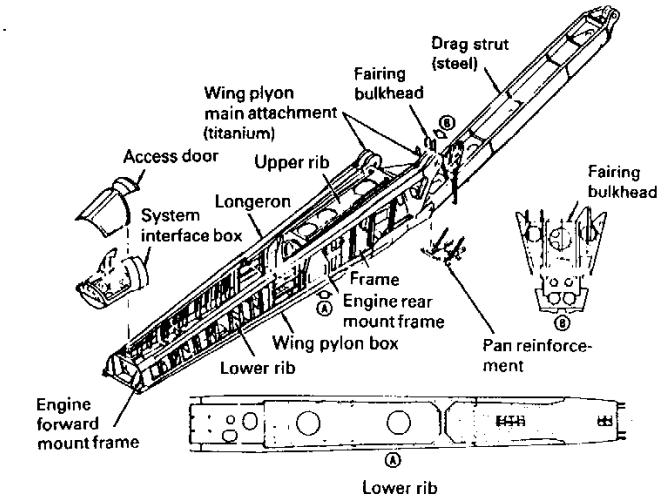
A

LITERATURE REVIEW SUPPLEMENT

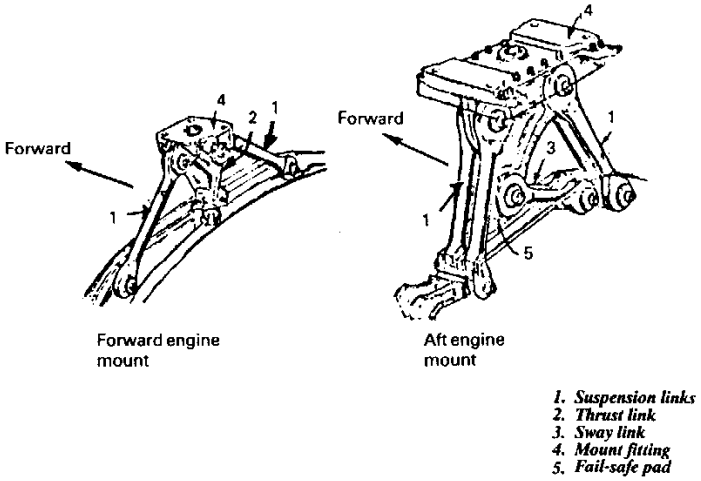
A.1. STRUCTURAL DESIGNS OF PYLONS FOR WING-MOUNTED ENGINES OF SELECTED AIRCRAFT



(a) Wing-pylon mount



(b) Pylon structure



(c) Engine mounts

Figure A.1: Wing-pylon mount configuration - Lockheed L-1011 Tristar [8].

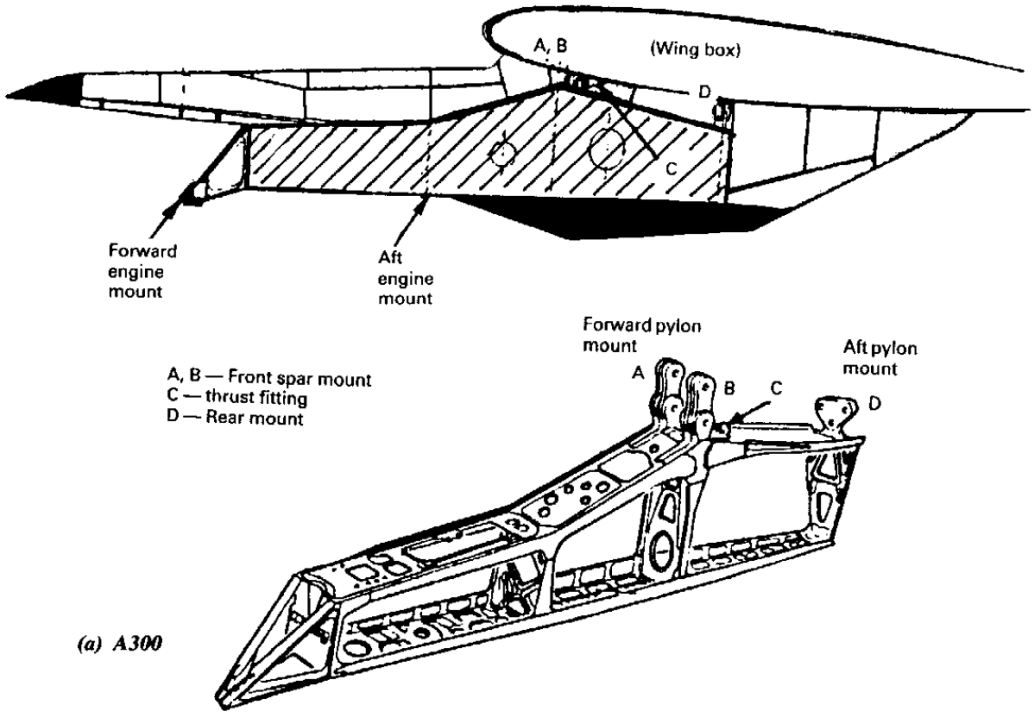


Figure A.2: Wing-pylon mount configuration - Airbus A300 [8].

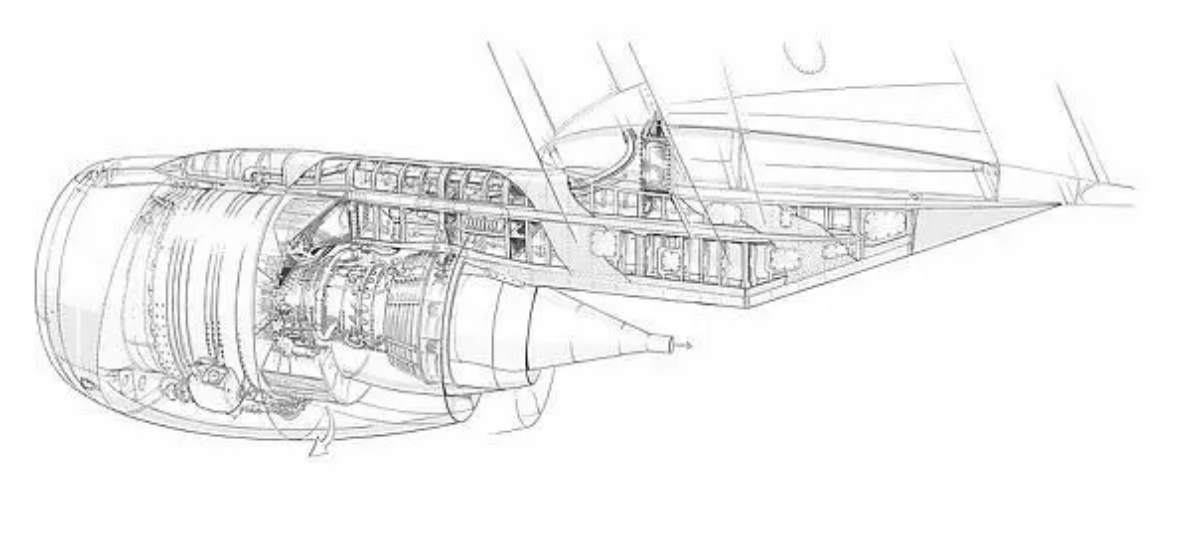


Figure A.3: Wing-pylon mount configuration links - Douglas DC-8 cutaway [30].

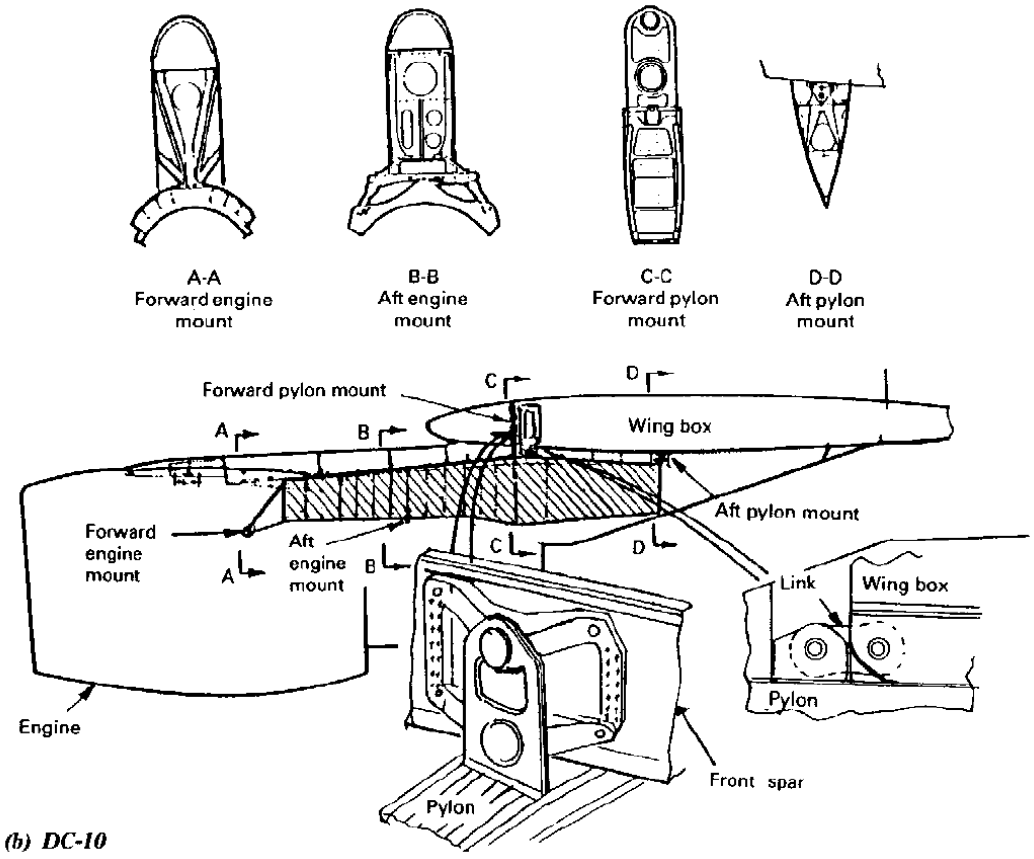


Figure A.4: Wing-pylon mount configuration - Douglas DC-10 [8].

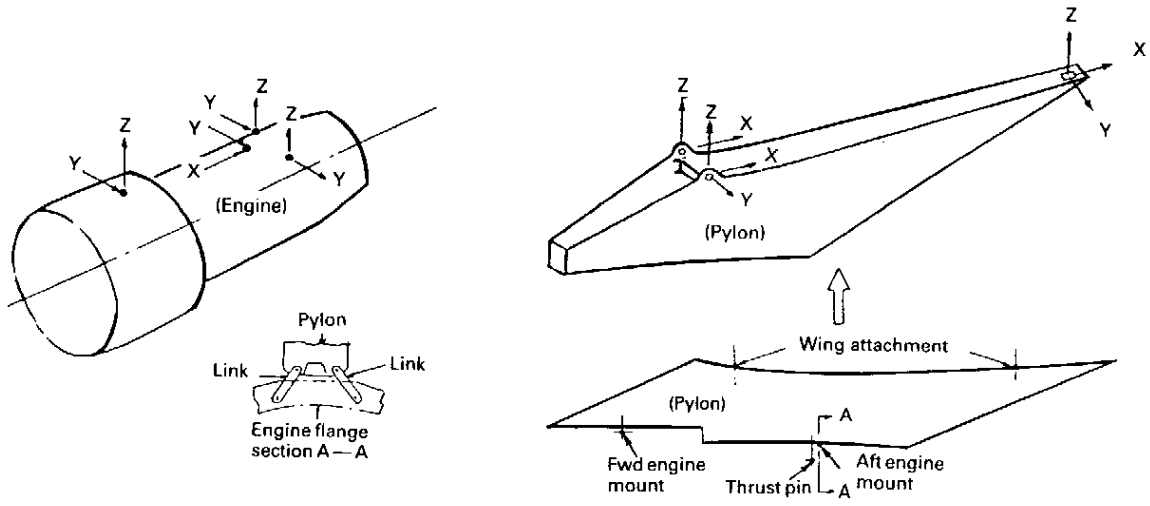


Figure A.5: Wing-pylon mount configuration - Lockheed S-3A Viking [8].

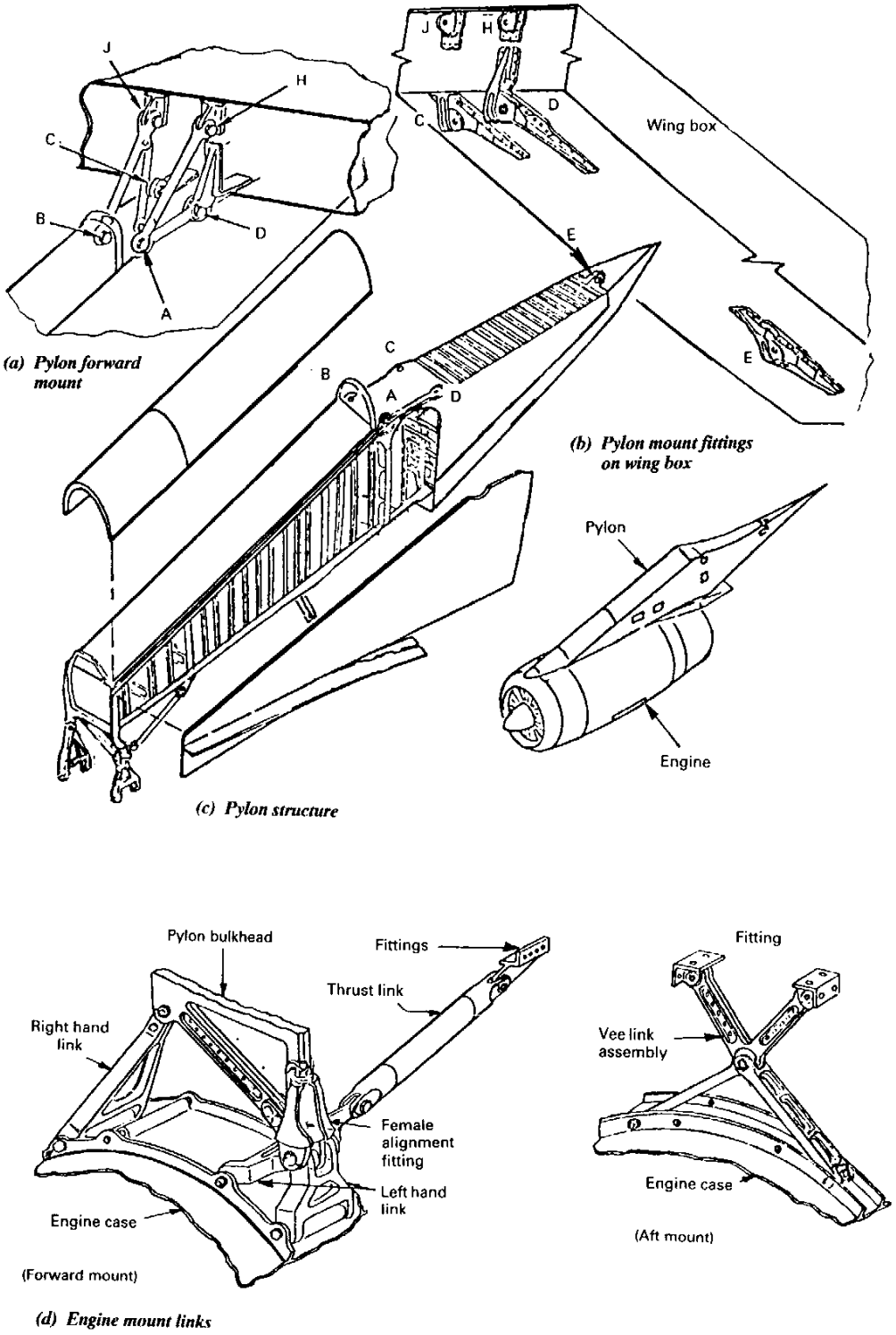
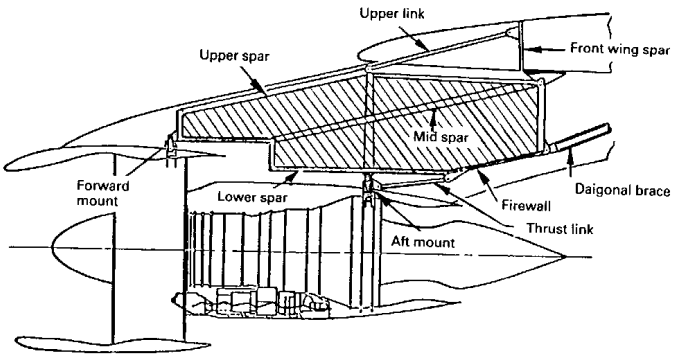
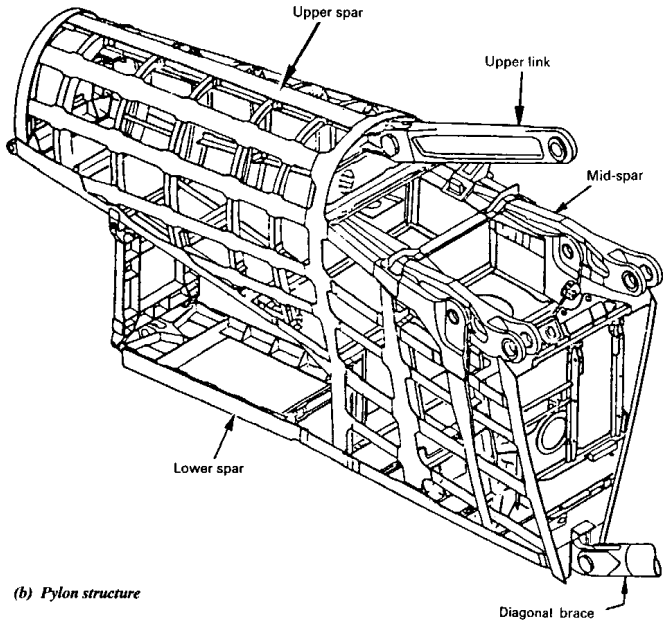


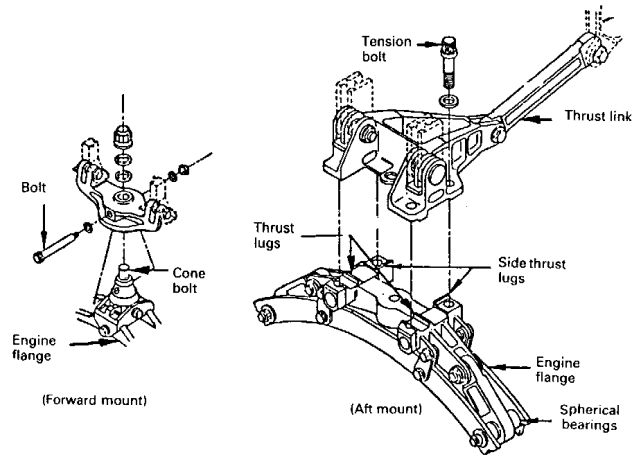
Figure A.6: Wing-pylon mount configuration - Lockheed C-141 Starlifter [8].



(a) Wing-pylon mount



(b) Pylon structure



(c) Engine mounts

Figure A.7: Wing-pylon mount configuration - Boeing 747 [8].

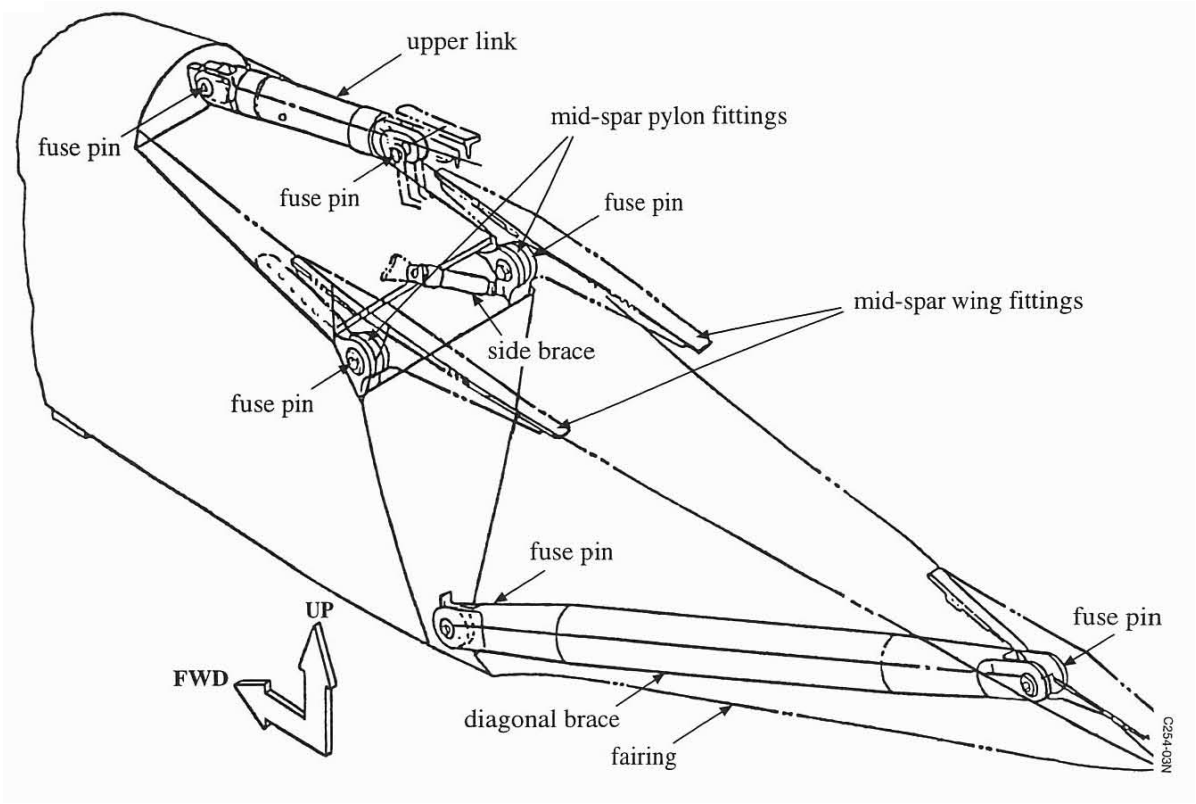


Figure A.8: Wing-pylon mount configuration links - Boeing 747-200 [31].

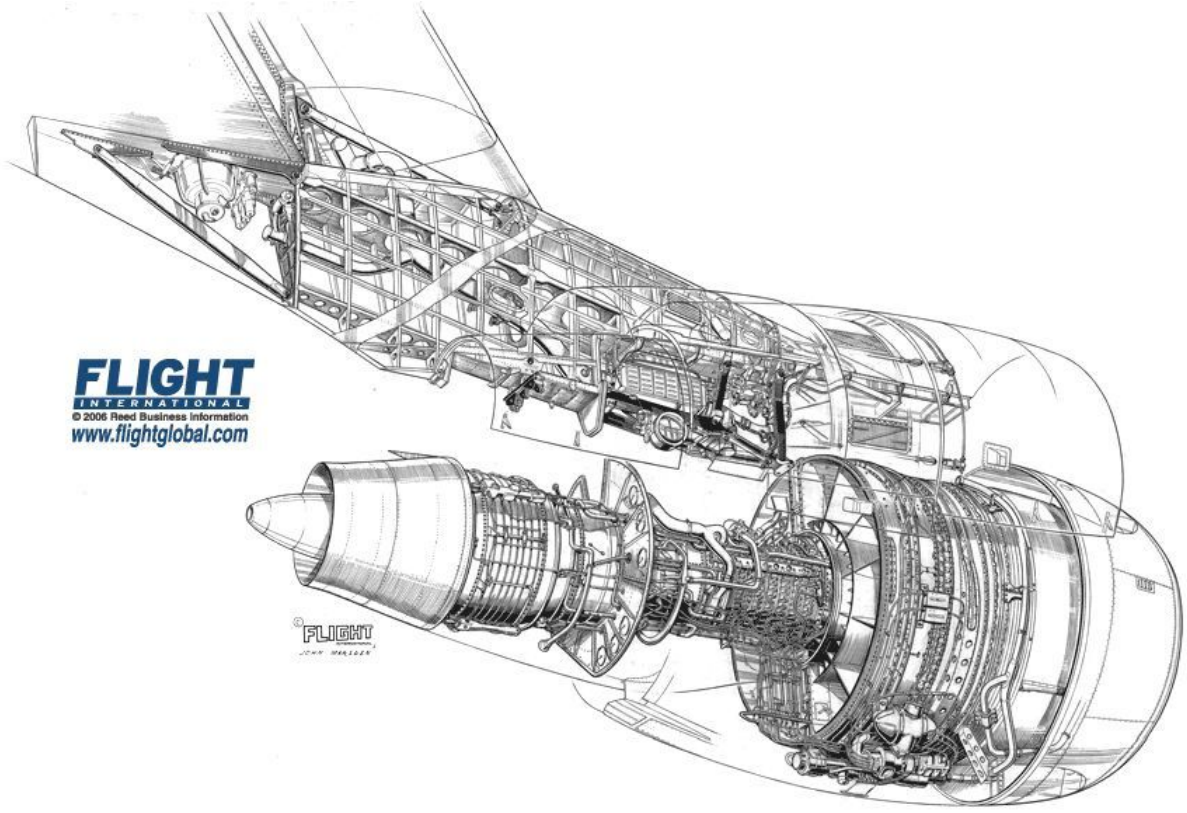


Figure A.9: Wing-pylon mount configuration links - Boeing 747-200 cutaway [30].

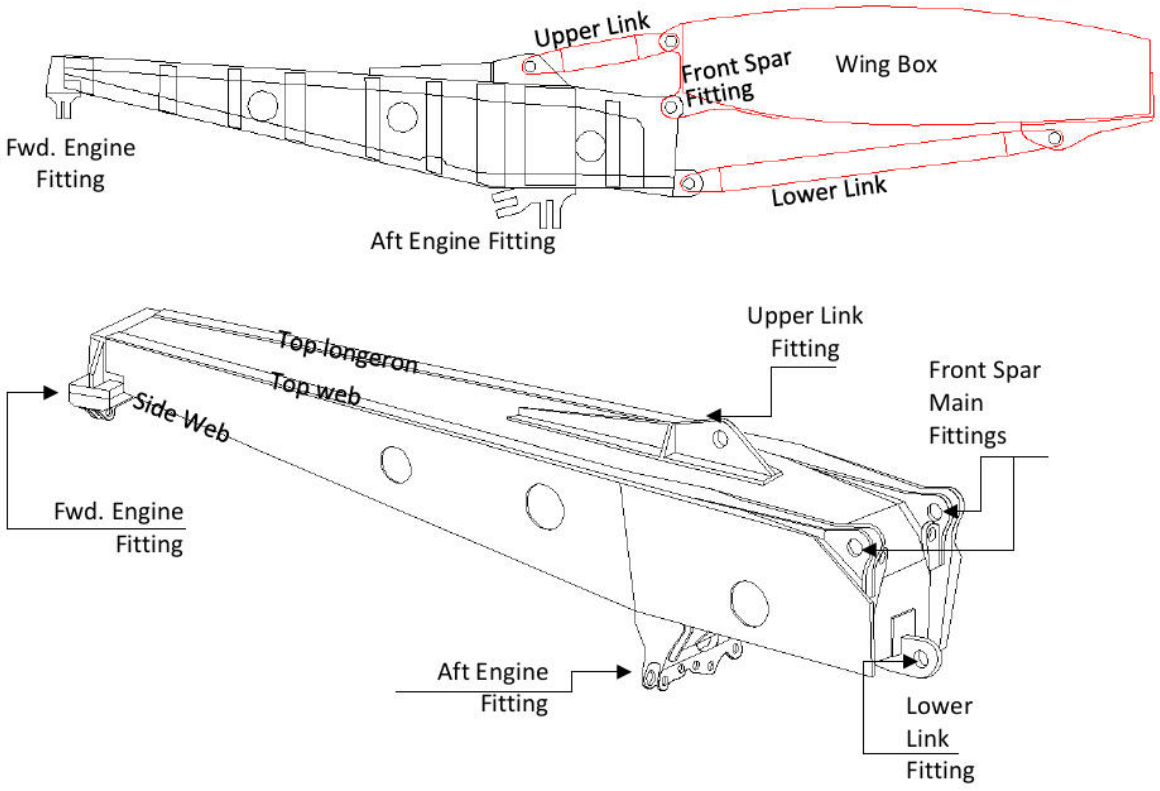


Figure A.10: Wing-pylon mount configuration links - Boeing 787 [32].

A.2. CS-25 REGULATIONS REGARDING ENGINE INSTALLATIONS

Taken from CS-25, Amendment 27 published June 2022 [50].

BOOK 1 - CERTIFICATION SPECIFICATIONS

SUBPART C - STRUCTURE

CS 25.341 - Gust and Turbulence loads

(a) ...

(b) ...

(c) *Supplementary gust conditions for wing mounted engines.* For aeroplanes equipped with wing mounted engines, the engine mounts, pylons, and wing supporting structure must be designed for the maximum response at the nacelle centre of gravity derived from the following dynamic gust conditions applied to the aeroplane:

1. A discrete gust determined in accordance with CS 25.341(a) at each angle normal to the flight path, and separately,

2. A pair of discrete gusts, one vertical and one lateral. The length of each of these gusts must be independently tuned to the maximum response in accordance with CS 25.341(a). The penetration of the aeroplane in the combined gust field and the phasing of the vertical and lateral component gusts must be established to develop the maximum response to the gust pair. In the absence of a more rational analysis, the following formula must be used for each of the maximum engine loads in all six degrees of freedom:

$$P_L = P_{L-1g} \pm 0.85 \sqrt{L_{V_i}^2 + L_{L_i}^2}$$

Where:
P_L = limit load;
P_{L-1g} = steady 1-g load for the condition;
L_V = peak incremental response load due to a vertical gust according to CS 25.341(a);
and
L_L = peak incremental response load due to a lateral gust according to CS 25.341(a).

CS 25.361 Engine and auxiliary power unit torque
(see AMC 25.361)

(a) For engine installations:

1. Each engine mount, pylon and adjacent supporting airframe structures must be designed for the effects of:–
(i) a limit engine torque corresponding to take-off power/thrust and, if applicable, corresponding propeller speed, acting simultaneously with 75% of the limit loads from flight condition A of CS 25.333 (b);
(ii) a limit engine torque corresponding to the maximum continuous power/thrust and, if applicable, corresponding propeller speed, acting simultaneously with the limit loads from flight condition A of CS 25.333 (b); and
(iii) for turbo-propeller installations only, in addition to the conditions specified in sub-paragraphs (a)(1) (i) and (ii), a limit engine torque corresponding to take-off power and propeller speed, multiplied by a factor accounting for propeller control system malfunction, including quick feathering, acting simultaneously with 1g level flight loads. In the absence of a rational analysis, a factor of 1.6 must be used.

2. The limit engine torque to be considered under sub-paragraph (1) must be obtained by:
(i) for turbo-propeller installations, multiplying mean engine torque for the specified power/thrust and speed by a factor of 1.25
(ii) for other turbine engines, the limit engine torque must be equal to the maximum accelerating torque for the case considered.

3. The engine mounts, pylons, and adjacent supporting airframe structure must be designed to withstand 1g level flight loads acting simultaneously with the limit engine torque loads imposed by each of the following conditions to be considered separately:
(i) sudden maximum engine deceleration due to malfunction or abnormal condition: and
(ii) the maximum acceleration of engine.

(b) ...

CS 25.362 Engine failure loads.
(See AMC 25.362)

(a) For engine mounts, pylons and adjacent supporting airframe structure, an ultimate loading condition must be considered that combines 1g flight loads with the most critical transient dynamic loads and vibrations, as determined by dynamic analysis, resulting from failure of a blade, shaft, bearing or bearing support, or bird strike event. Any permanent deformation from these ultimate load conditions should not prevent continued safe flight and landing.

(b) The ultimate loads developed from the conditions specified in paragraph (a) are to be:

1. multiplied by a factor of 1.0 when applied to engine mounts and pylons; and

2. multiplied by a factor of 1.25 when applied to adjacent supporting airframe structure.

CS 25.363 Side load on engine and auxiliary power unit mounts

(a) Each engine and auxiliary power unit mount and its supporting structure must be designed for a limit load factor in a lateral direction, for the side load on the engine and auxiliary power unit mount, at least equal to the maximum load factor obtained in the yawing conditions but not less than –

1. 1.33; or

2. One-third of the limit load factor for flight condition A as prescribed in CS 25.333 (b).

(b) The side load prescribed in sub paragraph (a) of this paragraph may be assumed to be independent of other flight conditions.

CS 25.367 Unsymmetrical loads due to engine failure

(a) The aeroplane must be designed for the unsymmetrical loads resulting from the failure of the critical engine. Turbo-propeller aeroplanes must be designed for the following conditions in combination with a single malfunction of the propeller drag limiting system, considering the probable pilot corrective action on the flight controls:

1. At speeds between V_{MC} and V_D, the loads resulting from power failure because of fuel flow interruption are considered to be limit loads.

2. At speeds between V_{MC} and V_C, the loads resulting from the disconnection of the engine compressor from the turbine or from loss of the turbine blades are considered to be ultimate loads.

3. The time history of the thrust decay and drag build-up occurring as a result of the prescribed engine failures must be substantiated by test or other data applicable to the particular engine-propeller combination.

4. The timing and magnitude of the probable pilot corrective action must be conservatively estimated, considering the characteristics of the particular engine propeller-aeroplane combination.

(b) Pilot corrective action may be assumed to be initiated at the time maximum yawing velocity is reached, but not earlier than two seconds after the engine failure. The magnitude of the corrective action may be based on the control forces specified in CS 25.397 (b) except that lower forces may be assumed where it is shown by analysis or test that these forces can control the yaw and roll resulting from the prescribed engine failure conditions.

CS 25.371 Gyroscopic loads

The structure supporting any engine or auxiliary power unit must be designed for the loads, including gyroscopic loads, arising from the conditions specified in CS 25.331, CS 25.341, CS 25.349, CS 25.351, CS 25.473, CS 25.479, and CS 25.481, with the engine or auxiliary power unit at the maximum rpm appropriate to the condition. For the purposes of compliance with this paragraph, the pitch manoeuvre in CS 25.331(c)(1) must be carried out until the positive limit manoeuvring load factor (point A2 in CS 25.333(b)) is reached.

SUBPART D - DESIGN AND CONSTRUCTION

CS 25.721 - Landing gear; General

(a) ...

(b) ...

(c) For configurations where the engine nacelle is likely to come into contact with the ground, the engine pylon or engine mounting must be designed so that when it fails due to overloads (assuming the overloads to act predominantly in the upward direction and separately predominantly in the aft direction), the failure mode is not likely to cause the spillage of enough fuel to constitute a fire hazard.

CS 25.865 Fire protection of flight controls, engine mounts, and other flight structure
Essential flight controls, engine mounts, and other flight structures located in designated fire zones or in adjacent areas which would be subjected to the effects of fire in the fire zone must be constructed of fireproof material or shielded so that they are capable of withstanding the effects of fire.

A.3. CLASS II WEIGHT ESTIMATION METHODS FOR PYLON AND NACELLE GROUP

Equation	Application	Covers	Notes	Source	Ref.
$W_{ng}\ [lbs] = 7.435\ N_{inl}\ [-]$ $\left(A_{inl}\ [ft^2]\ l_n\ [ft]\ p_2\ [psi]\right)^{0.731}$ (A.1)	Turbofan engines	Weight of all nacelles, engine external ducts, cowling and pylons	N_{inl}: number of inlets. A_{inl}: capture area per inlet. l_n: nacelle length from inlet lip to compressor face. p_2: max. static pressure at engine compressor face.	General Dynamics	[63]
$W_{ng}\ [lbs] = 0.065\ T_{T/O}\ [lbs]$ (A.2)	'High BPR' turbofan engines (BPR >2)	Nacelles, engine external ducts, cowling and pylon weight, including thrust reverser	No thrust reverser: −10%. Additional noise suppression material: +20%	Torenbeek	[17]
$W_{ng}\ [kg] = 6.8\ T_{T/O}\ [kN]$ (A.3)	Turbofan engines with $T_{T/O}$ <600 kN	Nacelle group mass, for all engines	No thrust reverser: −10%. Additional noise suppression material: +10%	Jenkinson-Simpson-Rhodes	[16]
$W_{ng}\ [kg] = 2760 + 2.2\ T_{T/O}\ [kN]$ (A.4)	Turbofan engines with $T_{T/O}$ >600 kN	Nacelle group mass, for all engines	No thrust reverser: −10%. Additional noise suppression material: +10%	Jenkinson-Simpson-Rhodes	[16]
$W_{ng}\ [lbs] = N_{eng}^{0.984}\ [-] \times 0.6725\ K_{ng}\ [-]\ l_n^{0.1}\ [ft]$ $w_n^{0.294}\ [ft]\ n_{ult}^{0.119}\ [-]\ W_{ec}^{0.611}\ [lbs]\ S_n^{0.224}\ [ft^2]$ (A.5)	Turbofan / turbojet / turboprop engines	All nacelles, including air induction and pylons	Recommended equation as per Vought. K_{ng} = 1.017 for pylon mounted nacelles. W_{ec}: weight of engine and contents.* S_n: Nacelle wetted area.**	Vought Aeronautics Division (commonly known as 'Raymer method')	[68][67]
$W_{ng}\ [lbs] = N_{eng}^{0.900}\ [-] \times 0.4836\ K_{ng}\ [-]\ l_n^{0.1}\ [ft]$ $d_n^{0.100}\ [ft]\ w_n^{0.337}\ [ft]\ n_{ult}^{0.100}\ [-]\ W_{ec}^{0.499}\ [lbs]\ S_n^{0.258}\ [ft^2]$ $T_e^{0.100}\ [lbs]\ l_{py}^{0.100}\ [ft]\ h_{py}^{0.100}\ [ft]$ (A.6)	Turbofan / turbojet / turboprop engines	All nacelles, including air induction and pylons	K_{ng} = 0.667 for pylon mounted nacelles. W_{ec}: weight of engine and contents.* S_n: Nacelle wetted area.**	Vought Aeronautics Division	[68]

$W_{ng}\,[lbs] = N_{eng}^{0.969}\,[-] \times 0.6535\,K_{ng}\,[-]\,l_n^{0.1}\,[ft]$ $d_n^{0.100}\,[ft]\,w_n^{0.239}\,[ft]\,n_{ult}^{0.111}\,[-]\,W_{ec}^{0.608}\,[lbs]\,S_n^{0.221}\,[ft^2]$ (A.7)	Turbofan / turbojet / turboprop engines	All nacelles, including air induction and pylons	K_{ng} = 1.040 for pylon mounted nacelles. W_{ec}: weight of engine and contents.* S_n: Nacelle wetted area.**	Vought Aeronautics Division	[68]
$W_{ng}\,[lbs] = N_{eng}^{0.927}\,[-] \times 0.3866\,K_{ng}\,[-]$ $n_{ult}^{0.153}\,[-]\,W_{ec}^{0.655}\,[lbs]\,S_n^{0.400}\,[ft^2]$ (A.8)	Turbofan / turbojet / turboprop engines	All nacelles, including air induction and pylons	K_{ng} = 1.012 for pylon mounted nacelles. W_{ec}: weight of engine and contents.* S_n: Nacelle wetted area.**	Vought Aeronautics Division	[68]
$W_{ng}\,[lbs] = 0.25 \times TNAC\,[-] \times DNAC\,[ft]$ $\times\, XNAC\,[ft] \times T_{T/O,scaled}^{0.36}\,[ft]$ (A.9)	Nacelles of transport aircraft	All nacelles or air induction systems	TNAC: Total number of nacelles plus 0.5 if there is a center-mounted engine. DNAC: Average diameter of the scaled engine nacelles.*** XNAC: Average length of the scaled engine nacelles.****	NASA FLOPS	[70]
$W_{py}\,[lbf] = 24.11\,N_n\,[-]\,S_{py}^{0.381}\,[ft^2]$ $\left[\frac{1.1\,K_{py}[-]\,n_{ult}\,[-]\,W_{eng}\,[lbf]\,l_n\,[ft]\,D_n\,[ft]}{10^6 \cos\Lambda_{py}\,[°]}\right]^{0.952}$ (A.10)	Large turbofan engines	Pylon mass	K_{py} = 1.46 for commercial aircraft, 1.0 for military aircraft. Λ_{py} is measures as the average angle between forward engine mount to forward wing attachment point and aft engine mount to aft wing attachment point.	DAR Corporation	[71]
$W_n\,[lbf] = 35.45\,N_n\,[-]$ $\left[\frac{1.1\,K_{eng}\,[-]\,W_{eng}\,[lbf]\,S_n\,[ft^2]\,R_d\,[-]}{10,000}\right]^{0.59}$ (A.11)	Large turbofan engines	Nacelle mass	K_{eng} = 2.33 for single subsonic turbofan engines, 1.0 for single subsonic turbojet engines, 2.15 for single supersonic turbojet engines.	DAR Corporation	[71]

* When not known, use:

$$W_{ec} = 2.331\, W_{eng}^{0.901}\, K_p\, K_{tr} \tag{A.12}$$

where: $\begin{cases} K_p = 1.4 & \text{for engines with propeller} \\ K_p = 1.0 & \text{otherwise} \end{cases}$ $\begin{cases} K_{tr} = 1.18 & \text{for jet engines with thrust reverser} \\ K_{tr} = 1.0 & \text{otherwise} \end{cases}$

** When not known, use:

$$S_n = 5.983\, l_n^{0.812}\, w_l^{0.750} \tag{A.13}$$

*** When not known, use:

$$DNAC = 0.04 \times \sqrt{T_{baseline}} \quad \text{or} \quad DNAC = \sqrt{\frac{T_{scaled}}{T_{baseline}}}$$

**** When not known, use:

$$XNAC = 0.07 \times \sqrt{T_{baseline}} \quad \text{or} \quad XNAC = \sqrt{\frac{T_{scaled}}{T_{baseline}}}$$

B

Parameterization overview

B.1. ZERO-THICKNESS PYLON GEOMETRY PARAMETERIZATION

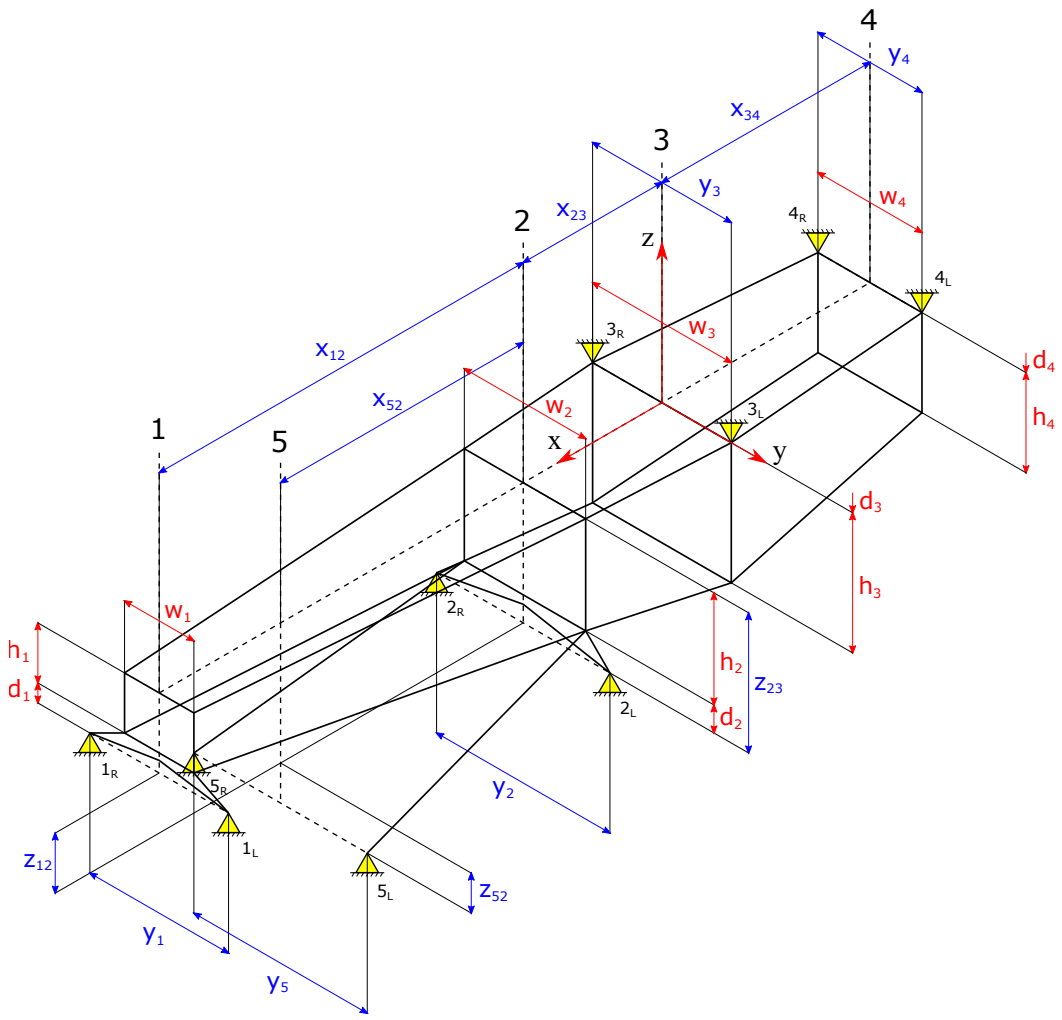


Figure B.1: Zero-thickness pylon parameterization, without orientation angle φ.

C

CODE DOCUMENTATION

C.1. INPUT VARIABLES FOR THE CLASS PYLONDESIGNER

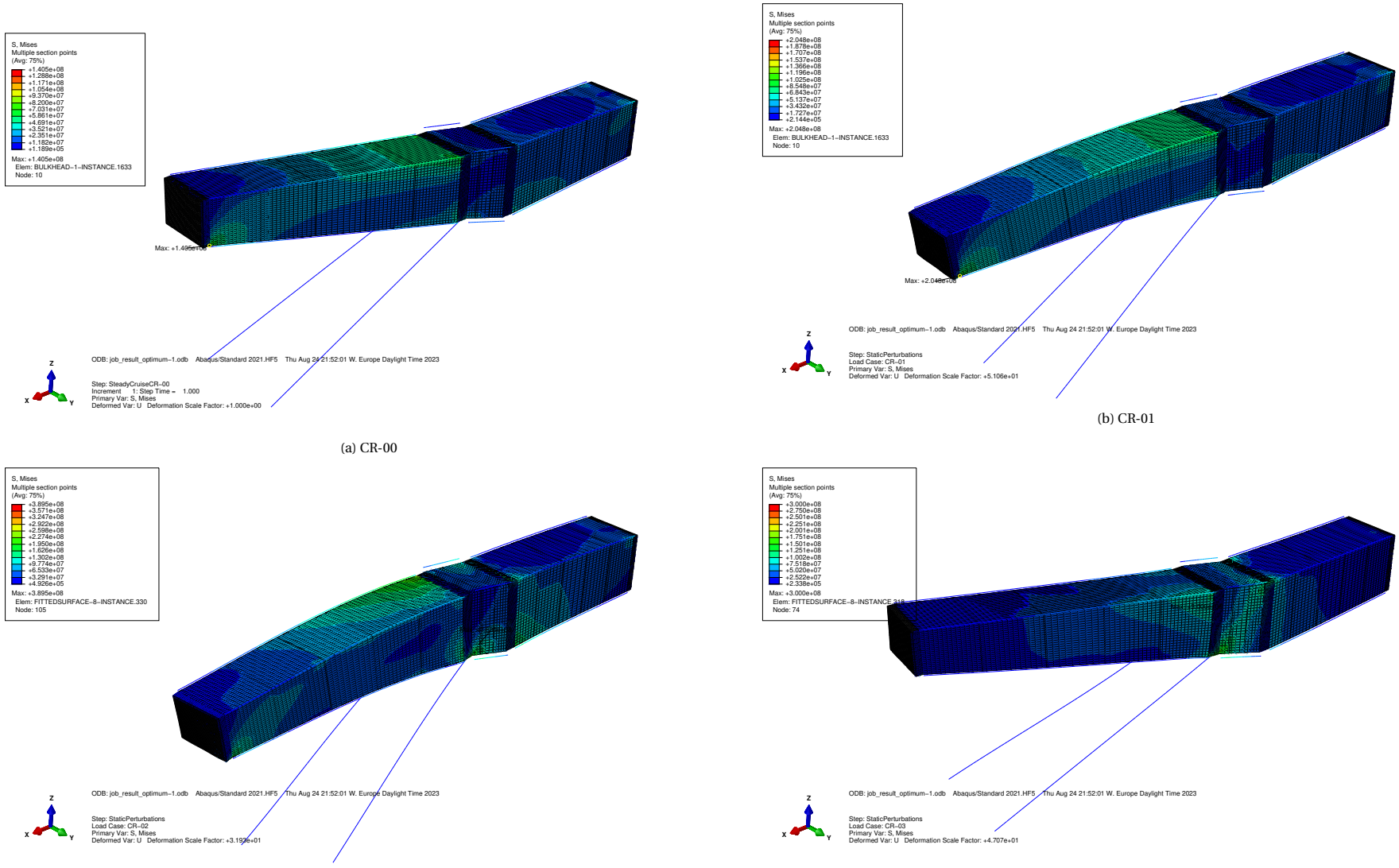
Table C.1: Input variables for the `PylonDesigner` class (for forward mounted engines)

Parameter	Chapter 4 parameterization equivalent	Type	Req. / Opt.	Remarks
`analysis_name`	n.a.	str	R	Name of the case
`longdist_enginemounts`	x_{12}	float	R	Positive when hps st 1 forward of hps station 2
`vertdist_enginemounts`	z_{12}	float	R	Positive when hps st 1 above hps station 2
`latdist_engfront`	y_1	float	R	Positive by definition
`longdist_reareng2frontacmounts`	x_{23}	float	R	Positive when hps st 2 forward of hps station 3
`vertdist_reareng2frontacmounts`	z_{23}	float	R	Positive when hps st 2 below hps station 3
`latdist_engrear`	y_2	float	R	Positive by definition
`longdist_acmounts`	x_{34}	float	R	Positive by definition
`latdist_acfront`	y_3	float	R	Positive by definition
`latdist_acrear`	y_4	float	R	Positive by definition
`longdist_thrustlinks`	x_{52}	float	R	Positive by definition, should be ≤ `longdist_enginemounts`
`vertdist_thrustlinks`	z_{52}	float	R	Positive when hps st 5 below hps station 2
`latdist_thrustlinks`	y_5	float	R	Positive by definition
`fronteng_boxheight`	h_1	float	R	
`fronteng_boxwidth`	w_1	float	R	
`fronteng_boxdist`	d_1	float	R	
`reareng_boxheight`	h_1	float	O	Value interpolated if not provided (for front-mounted engines)
`reareng_boxwidth`	w_1	float	O	Value interpolated if not provided (for front-mounted engines)
`reareng_boxdist`	d_1	float	O	Value interpolated if not provided (for front-mounted engines)
`frontac_boxheight`	h_3	float	R	
`frontac_boxwidth`	w_3	float	R	
`frontac_boxdist`	d_3	float	R	
`rearac_boxheight`	h_4	float	R	
`rearac_boxwidth`	w_4	float	O	Standard value = frontac_boxwidth
`rearac_boxdist`	d_4	float	O	Standard value = frontac_boxdist
`n_ribs`	n_{ribs}	int	R	
`n_stringers`	$n_{stringers}$	int	R	
`stringer_type`	n.a.	str	R	Type of stringers. Possible entries: `C-TYPE`, `L-TYPE`, `T-TYPE`, `I-TYPE`
`attachment_angle`	ϕ	float	R	
`pylon_material`	n.a.	str	R	Material used in the pylon (see Table 6.4). Options are: `Aluminium2024T3`, `Aluminium7074T6`, `TitaniumG2`, `TitaniumG3`, `TitaniumG4`, `Titanium6Al4V`
`pylon_type`	n.a.	str	R	Pylon type required. Possible entries are: `'Boxbeam'`, `'Dragstrut'`, `'RedundantLink'`. Only `'Boxbeam'` implemented during this thesis project.
`engine_characteristics`	n.a.	dict	R	Characteristics of the engine used for the analysis.

D

ABAQUS SIMULATION RESULTS

D.1. LEAP-1B WITH BOEING B737 MAX ENGINE INTEGRATION RESULTS



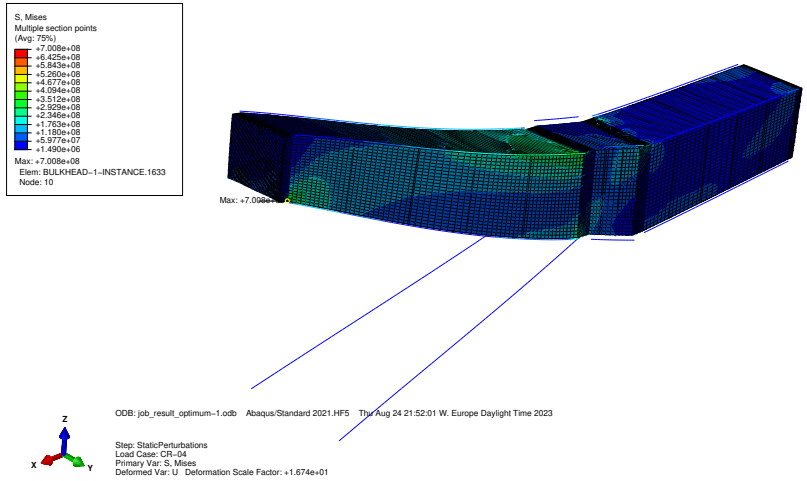
(a) CR-00

(b) CR-01

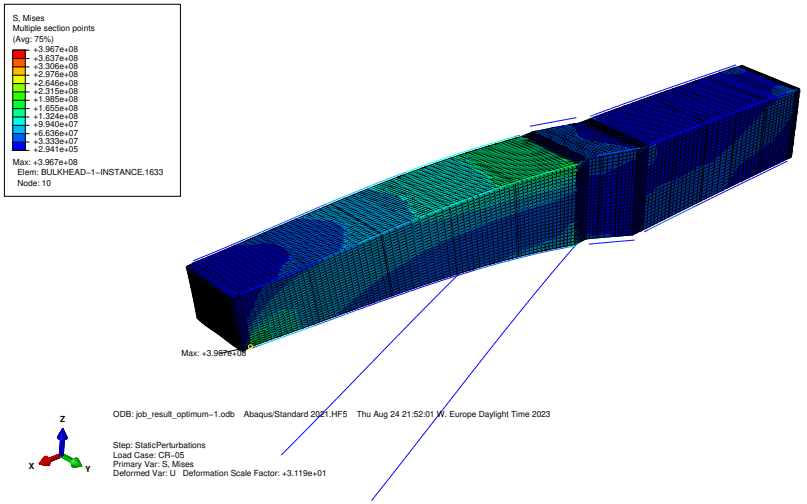
(c) CR-02

(d) CR-03

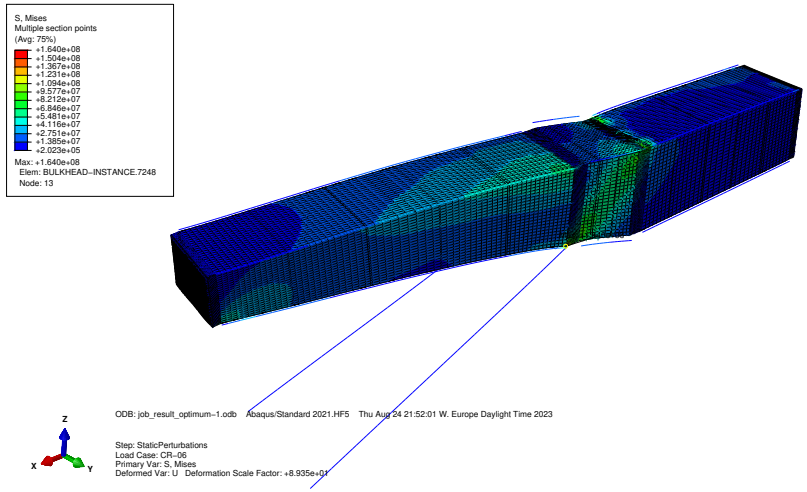
Figure D.1: Abaqus simulation results for the initial design vector model of the LEAP-1B integration onto the B737 MAX airplane



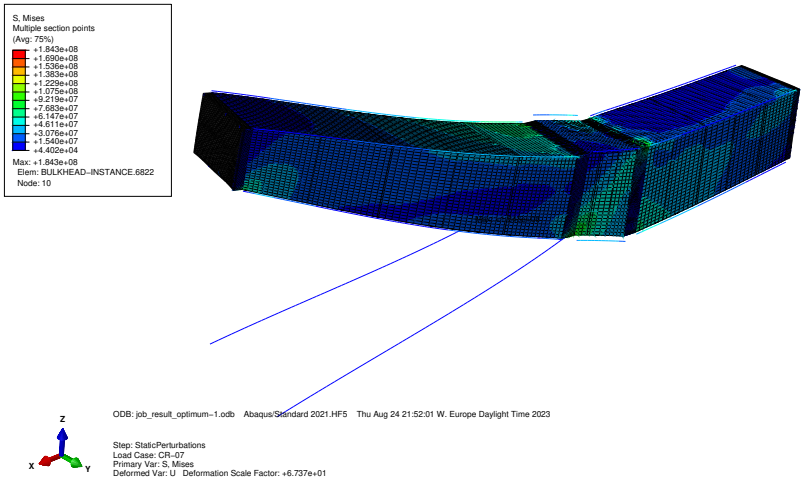
(e) CR-04



(f) CR-05

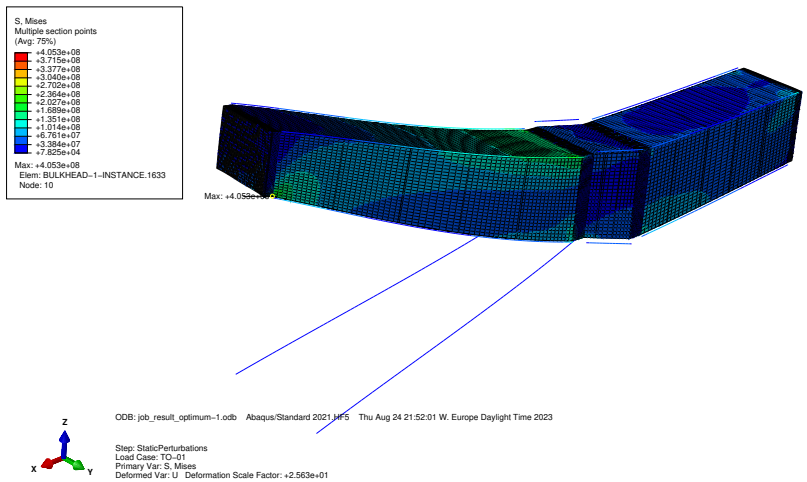


(g) CR-06

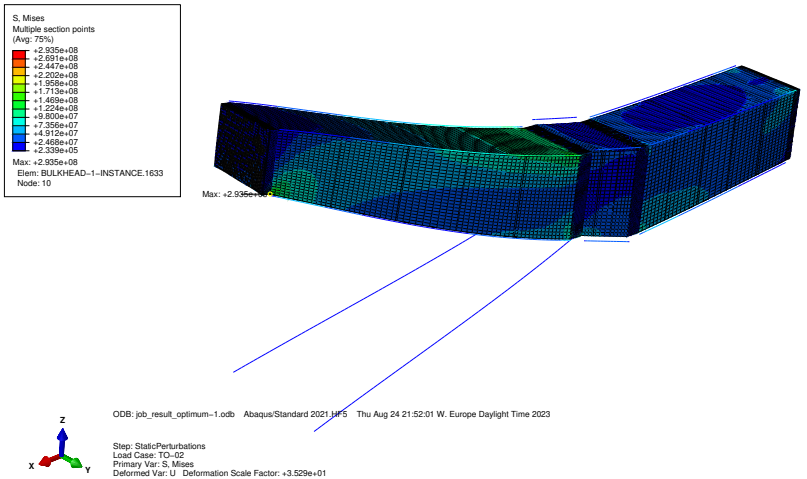


(h) CR-07

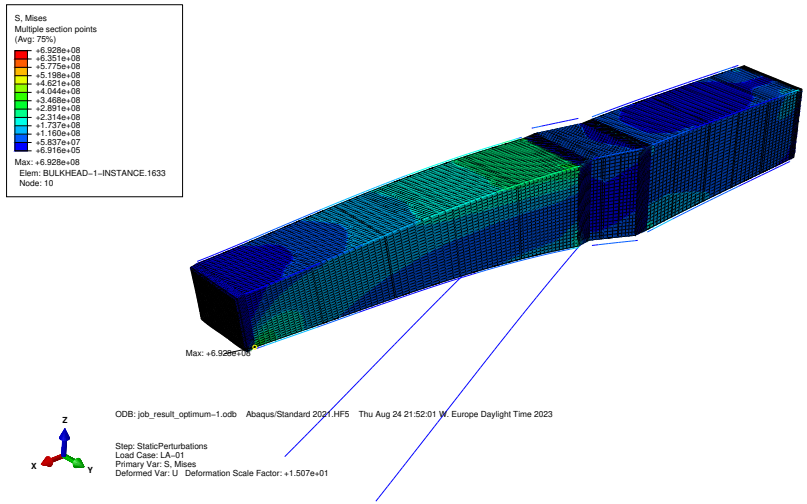
Figure D.1: Abaqus simulation results for the initial design vector model of the LEAP-1B integration onto the B737 MAX airplane (continued)



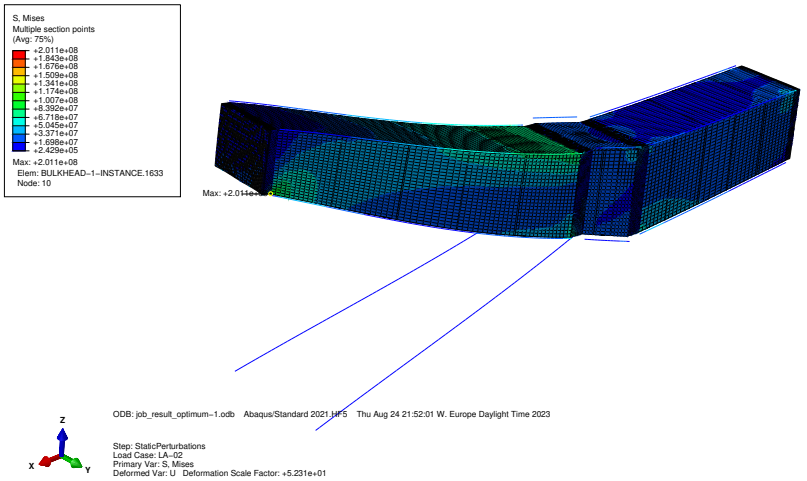
(i) TO-01



(j) TO-02

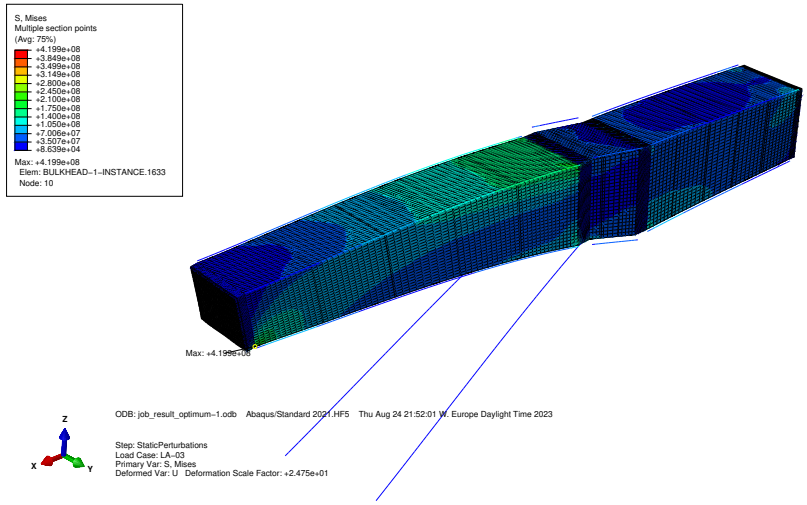


(k) LA-01

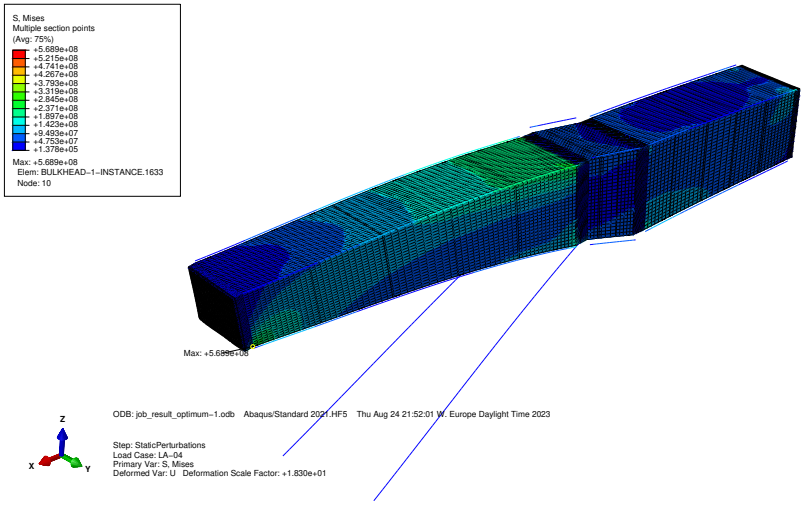


(l) LA-02

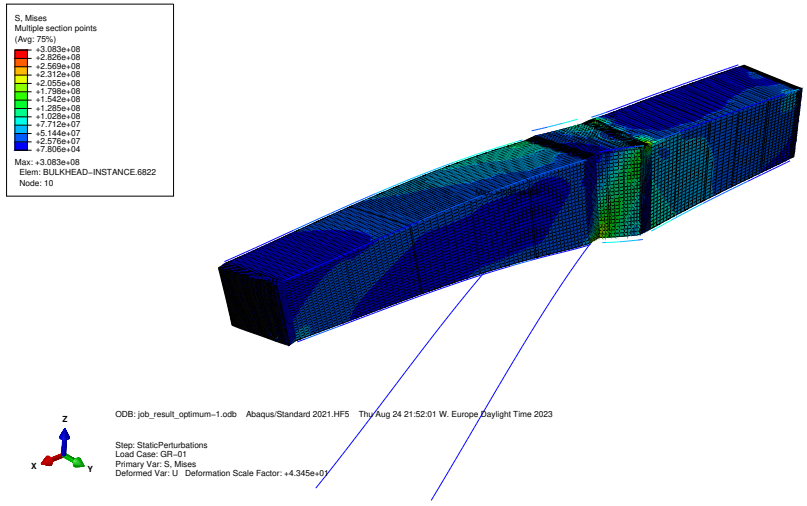
Figure D.1: Abaqus simulation results for the initial design vector model of the LEAP-1B integration onto the B737 MAX airplane (continued)



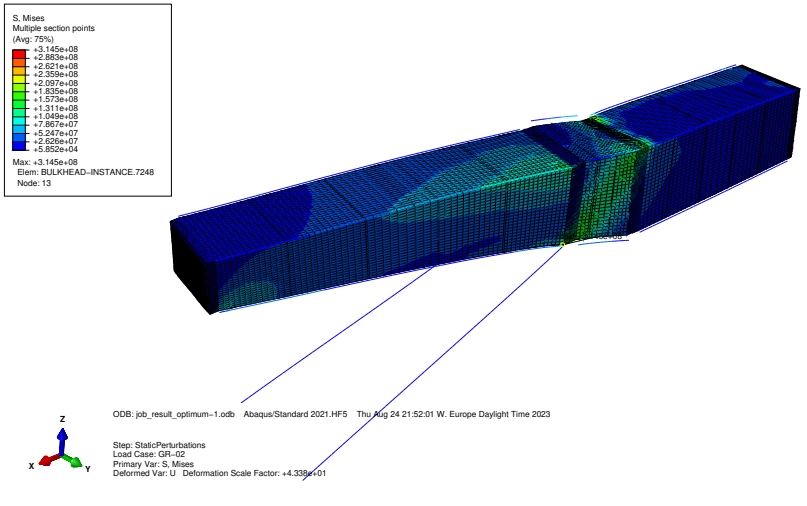
(m) LA-03



(n) LA-04

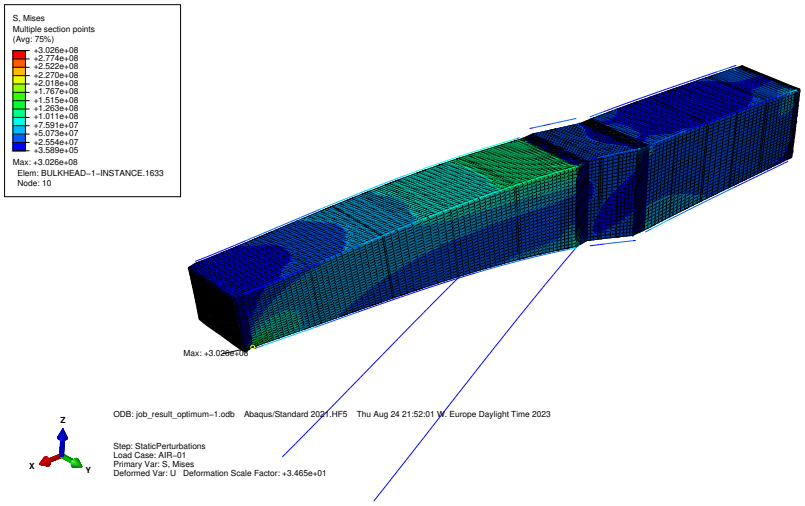


(o) GR-01

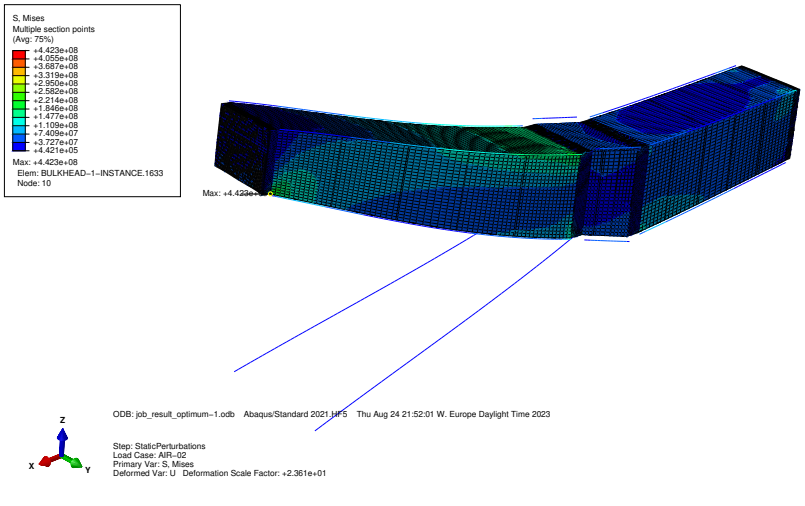


(p) GR-02

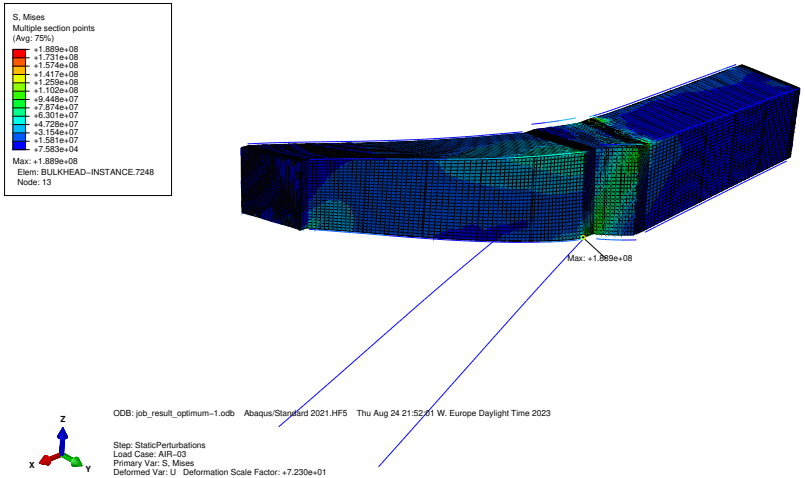
Figure D.1: Abaqus simulation results for the initial design vector model of the LEAP-1B integration onto the B737 MAX airplane (continued)



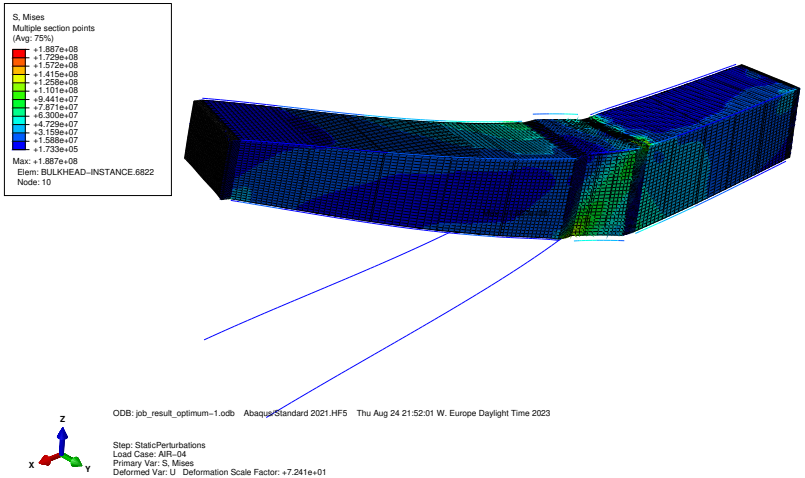
(q) AIR-01



(r) AIR-02

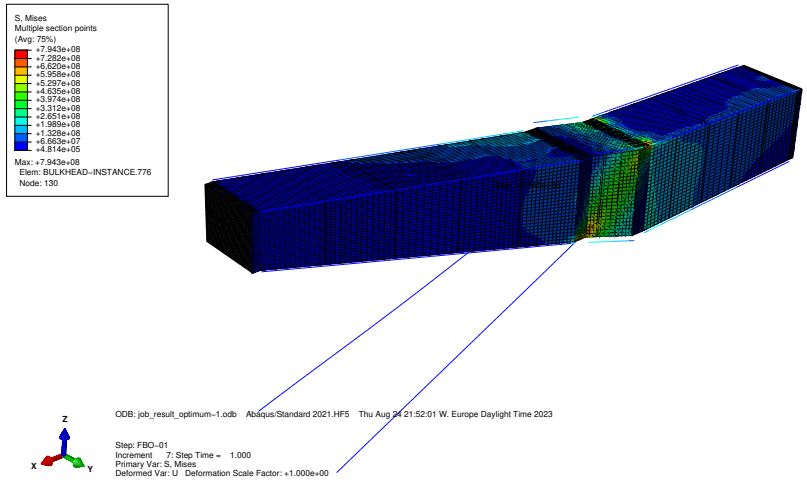


(s) AIR-03

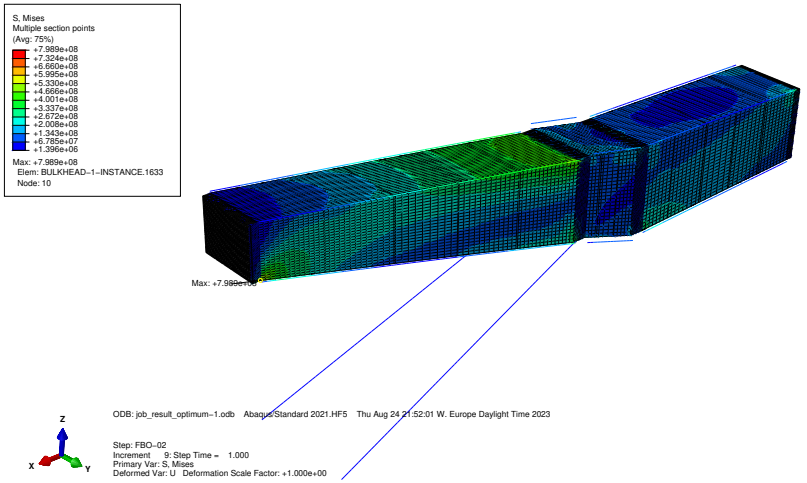


(t) AIR-01

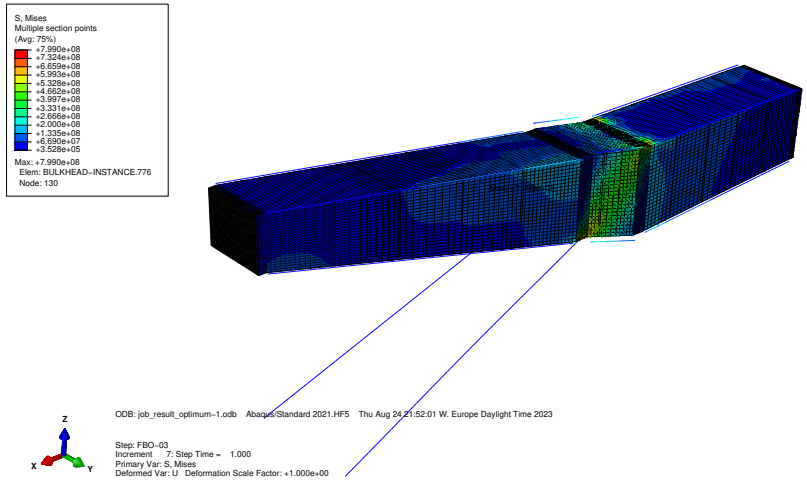
Figure D.1: Abaqus simulation results for the initial design vector model of the LEAP-1B integration onto the B737 MAX airplane (continued)



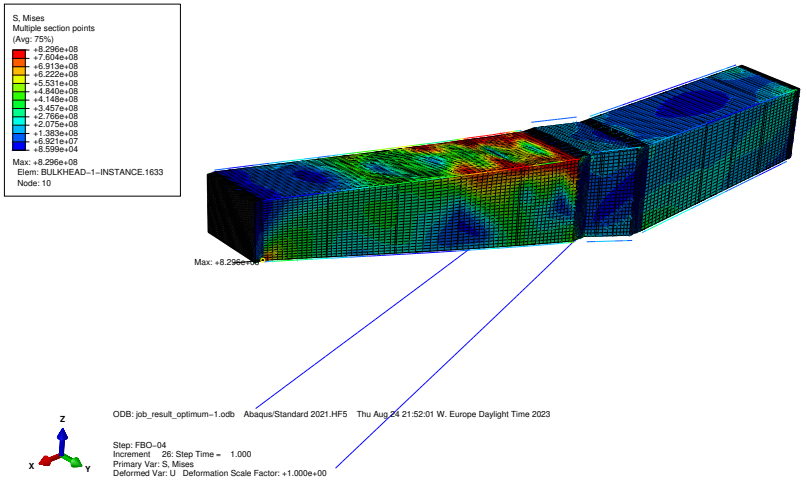
(u) FBO-01



(v) FBO-02



(w) FBO-03



(x) FBO-04

Figure D.1: Abaqus simulation results for the initial design vector model of the LEAP-1B integration onto the B737 MAX airplane (continued)