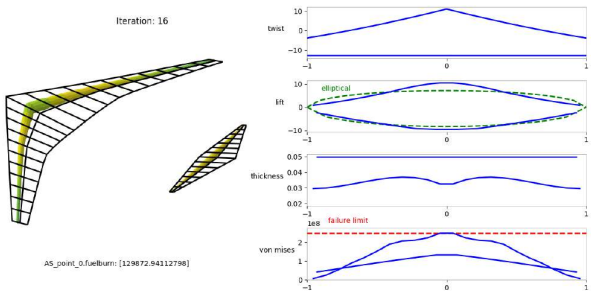
TÉCNICO LISBOA



Multidisciplinary Aircraft Design and Trajectory Control Optimization

Lourenço Manuel Martins Lúcio

Thesis to obtain the Master of Science Degree in

Aerospace Engineering

Supervisor: Prof. André Calado Marta

Examination Committee

Chairperson: Filipe Szolnoky Ramos Pinto Cunha

Supervisor: André Calado Marta

Member of the Committee: Pedro da Graça Tavares Álvares Serrão

December 2019

Here is to the end of an era...

Acknowledgments

First and foremost, I must thank professor André Marta for his unending patience towards my erratic work ethic. His guidance and adaptability proved invaluable in my many times of struggle, and this thesis would have never come to be without his help.

To my closest friends and family, who supported and pushed me through this seemingly impossible gauntlet; and time and time again pulled me out of the bottomless pits I dug myself into - thank you.

Finally, to Sofia, for refusing to give up on me.

Resumo

De modo a melhorar o desempenho esperado duma aeronave, podemos adoptar uma abordagem de optimização multidisciplinar ao invés de optimização sequencial. Desta forma, ao considerar os resultados das análises das várias disciplinas simultaneamente, é possível obter melhores resultados.

Este trabalho consiste no desenvolvimento de uma ferramenta numérica de baixa fidelidade para efectuar optimização multidisciplinar numa aeronave comercial - o *B777-300* - que considere análises de aerodinâmica, estruturas, propulsão e trajectória. A análise aerodinâmica das superfícies é feita com recurso ao método de vórtices discretos. As estruturas, mais especificamente as longarinas, serão modeladas por elementos finitos, com a forma de tubos cilíndricos ocos. A força propulsiva será calculada recorrendo a um modelo empírico baseado nos motores que pretendemos simular. Por fim, após efectuar optimização para condições de cruzeiro, será efectuada optimização de controlo dessa mesma configuração para outras fases da missão. A métrica de desempenho que pretendemos optimizar neste trabalho será a quantidade de combustível consumido pela aeronave.

Os processos de optimização descritos foram concluídos com sucesso, tendo o primeiro gerado a configuração da asa e estabilizador horizontal optimizados para a situação de cruzeiro, e o segundo fornecido os valores das variáveis de controlo para a fase de descida. Os valores obtidos foram validados por comparação com os valores do *B777-300* original, incluindo o combustível consumido - tendo-se verificado que esta quantidade diminui à medida que mais disciplinas são consideradas.

Palavras-chave: Optimização multidisciplinar, Optimização de controlo de trajectória, Optimização aero-estrutural, Método de vórtices discretos, Método de elementos finitos

Abstract

In the pursuit of increasing aircraft performance, one approach which can yield better results than a conventional design process is a multidisciplinary optimization process. In this paradigm, a design architecture is established so that the analyses for the several disciplines pertinent to the problem are handled simultaneously, rather than sequentially.

In this work, a numerical tool was developed in order to perform low-fidelity multidisciplinary optimization upon a commercial airliner - the *B777-300* - considering models for aerodynamics, propulsion, structures and trajectory. For the aerodynamic analysis, a vortex-lattice method (VLM) is employed. The lifting surface structures were modeled by finite elements with the shape of hollow tubular spars. For the propulsion system, a model based on empirical data collected from the target engines was utilized. Finally, the system was optimized for cruise conditions, and then control optimization was performed on the resulting configuration for additional mission phases. The performance metric optimized in this work was be the amount of fuel burnt by the aircraft in order to complete its mission.

The described optimization processes were successfully carried out, the former outputting the cruise-optimized wing and tail configurations, and the latter providing the optimized control parameter values for descent flight conditions. These values were validated by means of comparison with those of the original *B777-300*, including that of the performance metric - which improved as more disciplines were considered.

Keywords: Multidisciplinary optimization, Trajectory control optimization, Aerostructural optimization, Vortex lattice method, Finite element method

Contents

List of Tables

List of Figures

Nomenclature

Greek symbols

α	Angle of attack.
δ_T	Engine throttle.
η	Engine parameter.
η_p	Propulsive efficiency.
Γ	Circulation.
γ	Angle of ascent.
λ	Lagrange multiplier.
ϕ_T	Propulsion angle.
ρ	Density.

Roman symbols

$\mathcal{L}$	Lagrange function.
C_D	Coefficient of drag.
C_L	Coefficient of lift.
C_M	Coefficient of moment.
p	Pressure.
$\boldsymbol{V}$	Velocity vector.
u, v, w	Velocity Cartesian components.

Subscripts

∞	Free-stream condition.
i, j, k	Computational indexes.
n	Normal component.

x, y, z Cartesian components.

Superscripts

* Optimal.

T Transpose.

Chapter 1

Introduction

1.1 Motivation

With the consistently increasing demand for commercial flights in the last decades, the rate of fuel burn (i.e., the efficiency) of the employed aircraft has been a metric of great interest for several decades. Over the past few years, however, engineering advancements and consequent reductions of consumed fuel have stagnated when compared to previous decades, as illustrated in fig. 1.1 [1].

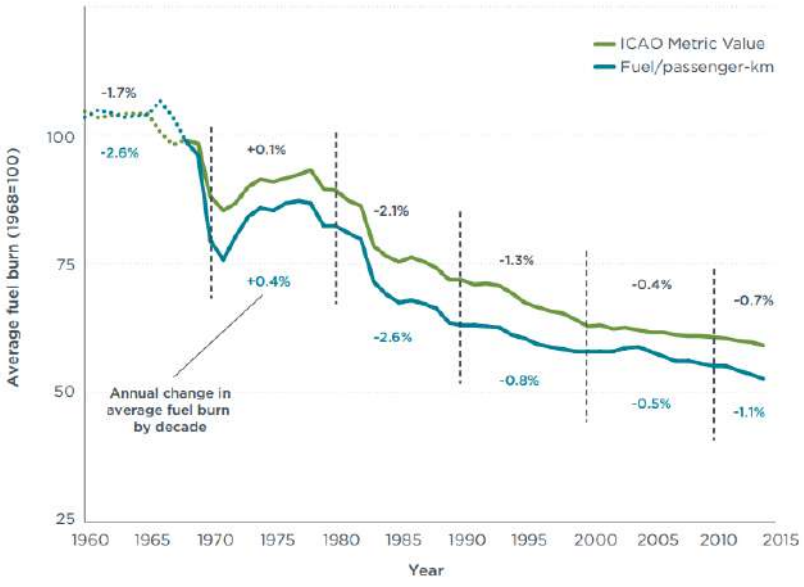


Figure 1.1: Evolution of fuel consumption in commercial aviation from 1960 to 2014 [1]).

Since so many factors go into minimizing fuel consumption, several efforts have been made in the many areas that the problem at hand comprises. Within the aircraft design process alone, fields such as aerodynamics, propulsion or structures play extremely important roles in determining the amount of fuel that the aircraft will require in order to complete its mission(s). Another complex problem which has consistently been improved upon is the allocation problem. Using the national airline TAP as an example, the company carries out about 2500 flights per week between 76 different destinations, with

their fleet of nearly 80 aircraft [2]. Allocating each of the aircraft to specific routes in order to maximize profit (i.e., minimize losses) is a topic extensive enough by itself to warrant papers being written on it alone [3], [4].

Focusing on any of these areas in specific is not the scope of this work, however. As aforementioned, since the progress within the separate disciplines is not as rampant as before, attention has turned to alternative routes through which to improve the fuel consumption based on currently available technology. Some of the ways in which this situation has been approached have been, for example, the possibility of hybrid and electric aircraft [5] or the research of more exotic configurations such as the blended wing-body design (an example of which is shown in fig. 1.2) [6].

Figure 1.2: Hybrid, blended-wing body NASA N3-X concept design [7].

A field of great interest in assessing the possibilities of these new configurations as well as improving the currently available aircraft has been Multidisciplinary Design Analysis and Optimization (MDAO). In an MDAO-based framework, the main goal is to take into account several disciplines of the aircraft development and usage during the design process simultaneously.

1.2 Aircraft Design and Trajectory Optimization

In this section, a brief overview of recent developments in the optimization processes for both aircraft design and trajectory analysis is presented. Some examples of how MDAO can be (and has been) integrated into the conceptual design process are also discussed.

Aircraft Design

Historically, the design process of an aircraft would be approached in a multiple team system, where each one would focus on a specific area of expertise - control, aerodynamics, noise, etc. These areas and their respective analyses, however, are tightly coupled - a change in the aircraft's design would affect more than one of these teams, even though they all perform computations in parallel.

As an example, let us look at a simplified design process for a wing. Saying we started by solving the aerodynamic problem, we would seek to minimize drag while producing enough lift to support the

aircraft, and would obtain the corresponding load profile. These loads in turn causes displacements on the wing, determined by the structural analysis. Then, the wing would be tweaked so as to meet the structural constraints (non-failure or displacement restrictions). Changing the wing shape, however, would change the results of the aerodynamic analysis, given the two disciplines' tight relationship.

This sequential approach provides non-optimal results, which an MDAO approach seeks to rectify. An example of this non-optimality can be observed in fig. 1.3, where it can be seen that all the design points computed by the multidisciplinary approach are strictly superior to those obtained via sequential optimization. Other examples of how the sequential approach is inferior to integrated aerostructural optimization include Chittick and Martins [8] or Grossman et al. [9], the latter of which dates as far back as 1988.

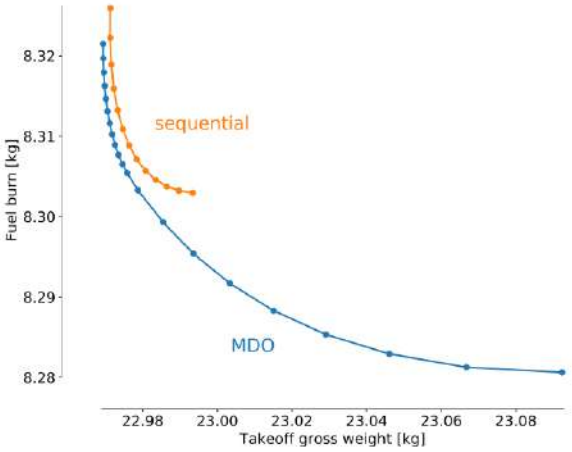


Figure 1.3: Comparison between Pareto fronts for sequential and multidisciplinary optimization [10].

An optimization approach is also fundamentally different in how it is formulated:

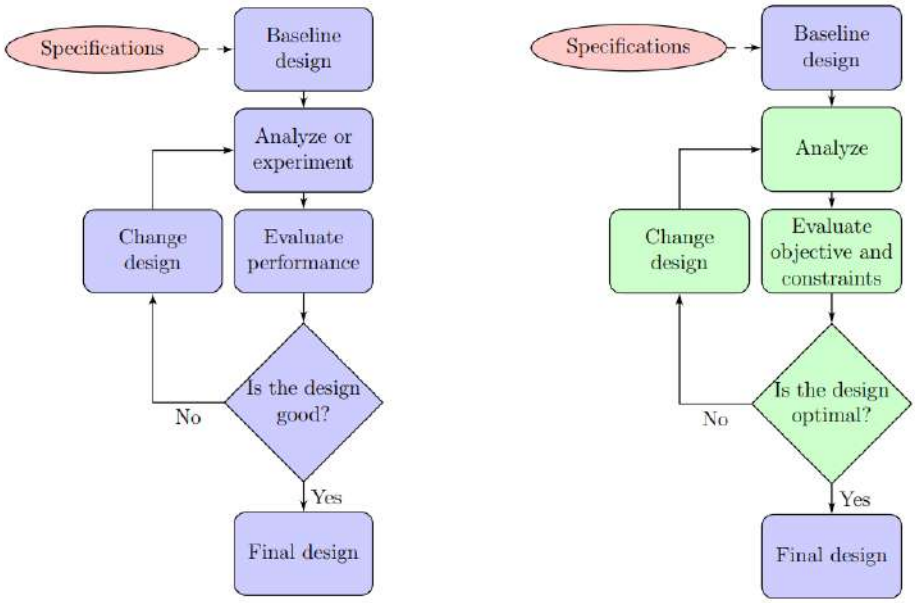


Figure 1.4: Decision flowchart for conventional (left) and optimized (right) design processes [11].

Because every aspect of the aircraft's performance is analytically defined as either a necessity (constraint) or that which we are looking to optimize (be it range, fuel burnt, lift-to-drag ratio, etc.), it is possible to ascertain whether the current design point is the optimal solution in the surrounding design space given the stipulations we have defined for what "optimal" is. A commonly used metric is the Breguet equation [12], [13], which will also be discussed in this work (see section 2.5). Another possibility is to consider a composite objective [14], meaning more than one function is weighted when computing the optimality of the design.

It is also important to note that the example provided only covers two disciplines within the aircraft design - the aerostructural problem. It should be easy to see, however, how this can be expanded to as many disciplines as one would want. A few examples include Moore and Ning [15] or Hwang and Ning [16], who carried out MDO to assess the design of electric propulsion systems; or Gray and Falck [17], who performed optimal control and trajectory optimization.

Trajectory Optimization

The process through which the trajectory is usually worked into the optimization process is transverse to the discussion of disciplines and design method. Rather than adding another discipline per se to the process, be it sequential or optimized, a more common approach is to consider several flight conditions for the same current design - a multipoint approach [17], [18], [19].

The mission profile is an interesting consideration when tackling aircraft design, since not only does it define the conditions under which the plane will have to operate, we can also consider how different flight phases carry different requirements for the aircraft. Figure 1.5 illustrates the typical trajectory breakdown for a commercial airliner.

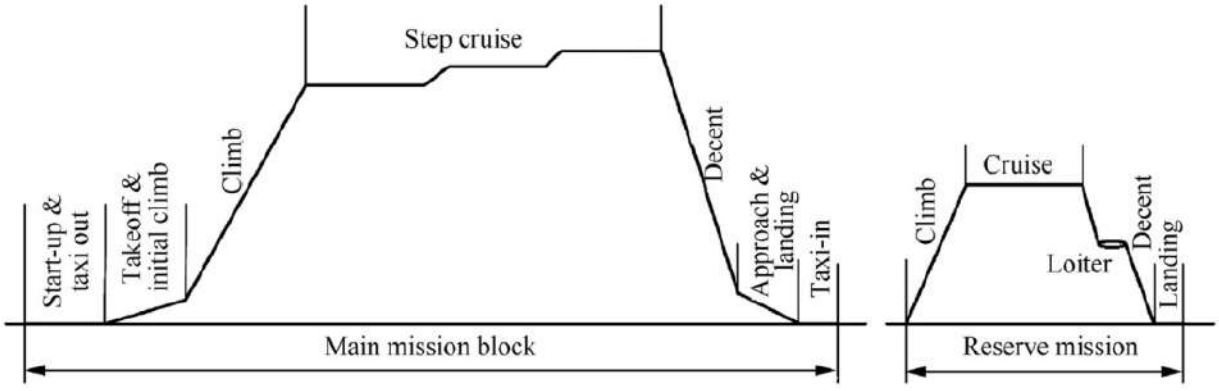


Figure 1.5: Standard mission profiles for an airliner [18].

This way, the aircraft design is subjected to a larger scope of conditions rather than just cruise, which is usually the choice when the trajectory is not considered in the optimization process [12].

MDAO Integration

On the implementation end, frameworks such as OpenMDAO [20] have been developed, which allows for the creation of MDAO schemes based on several data passing options within the code structure. Defining the flow and processing of data as the analyses of the several disciplines are carried out is the core of defining an MDAO scheme, and an extensive survey of these schemes, usually referred to as architectures, was published by Martins and Lambe [21]. These will be discussed in depth in section 2.1.2.

For a more visual reference, fig. 1.6 depicts an example architecture for a multidisciplinary analysis where three coupled disciplines are being considered.

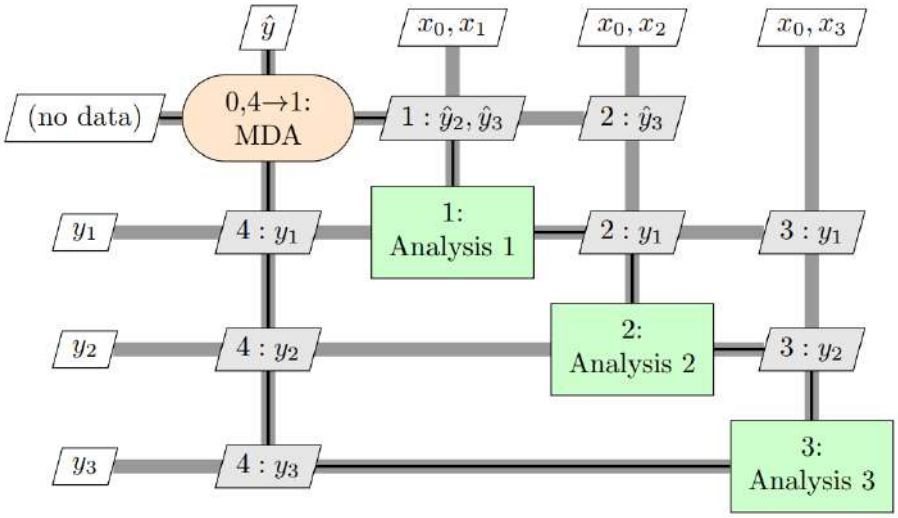


Figure 1.6: Visual representation of MDA architecture for a three-discipline problem [21].

These advancements, however, are not entirely incompatible with the more traditional design approach described at the beginning of this section. In order to facilitate the implementation of MDAO within a more conventional industrial setting, architectures have been developed where design teams work in parallel. These distributed architectures present less of a clash in between design processes than fully centralized ones [21], [22]. Section 2.1.2 covers a collaborative solution such as these in more detail.

Within our context (aircraft design and trajectory optimization) in specific, we can find optimization solutions that have been developed in varying degrees of fidelity, depending on what the main goal is. Roy and Crossley [23], for example, utilized a low-fidelity aircraft design model whilst studying the simultaneous allocation and design problem. The MDAO aerostructural problem has also been extensively documented with high-fidelity approaches [13], [12] and freeware code can even be found to solve it in low-fidelity fashion [10]. This makes that same problem a good starting point.

1.3 Objectives

As such, the motivation for this dissertation is to explore the potentiality of an MDAO approach to the aircraft design problem. To this end, the discipline of propulsion and the mission profile planning will be coupled to the aircraft disciplines of aerodynamics and structures. To do so, pre-existing MDAO aerostructural code will be improved upon to contemplate the two former areas in the optimization process.

This means that new constraints, as well as more design variables, will be added to the problem. These might worsen or improve the originally obtained solutions, respectively, but a more realistic and mission-compliant design should always be achieved.
In sequential and direct terms, this work aims to:

- Develop and implement a propulsion model, providing different engine configurations and subsequent control variables;
- Define adequate conditions to describe the several stages of the mission profiles;
- Perform coupled aircraft design and optimal control and compare its results to its previous iteration, as well as the chosen aircraft's original fuel consumption.

The main and overall contribution of this work will be the developed code, which will allow for first-stage, quick assessment of the scenario at hand, potentially providing insight into the treatment of the contemplated variables, constraints and other important performance metrics during the conceptual design phase.

1.4 Thesis Outline

Firstly, some background will be glossed over pertaining to the main disciplines within the scope of the work - aerodynamics, structures, propulsion, considerations regarding the discretization of the mission profile, and multidisciplinary optimization itself.

After discussing the fundamental equations at play, the computational implementation of said disciplines will be described in chapter 3, as well as other definitions necessary to properly formulate the optimization problem.

In chapter 4, the chosen optimization architecture will be looked into in greater detail, so as to provide the reader a better understanding of how data is being processed throughout the optimization. This chapter also includes a brief discussion of how the aircraft's performance is being assessed, both in terms of constraints being applied and metrics which may act as minimization objectives.

Once the full problem and respective optimization framework have been fully covered, its outputs will be discussed in chapter 5:

- In section 5.4 optimization will be carried out on the overarching case: a cruise-only situation. This will serve the purpose of validation and will provide us our base aircraft;

- In section 5.5 we will be performing control optimization upon other flight phases, utilizing the aircraft specifications obtained in the cruise-only optimization.

As previously described, optimization will be performed upon several disciplines. However, to better assess the impact each area has on the aircraft's design, the optimization process will first be carried out contemplating just a few of the disciplines. The problem's complexity will then be progressively incremented as more analyses and conditions are added.

Finally, in chapter 6, the work's achievements will be summed up, evaluating whether the original goals were met; and suggestions for further developments will be made.

Chapter 2

Theoretical Overview

The models used to carry out the analyses in the several disciplines are well-documented approaches based on relatively simple equations, which in turn facilitates their analytic implementation. This is of utmost importance, since it allows the usage of analytic derivatives, granting the optimization process much of its strength (as will be discussed further later in this chapter).

2.1 Multidisciplinary Design Analysis and Optimization

Let us start by defining some of the most essential concepts in multidisciplinary design analysis and optimization (MDAO).

2.1.1 Basic Concepts

- Objective function: this function surmises the fundamental goal of the whole optimization. We are looking to minimize some quantity (or maximize, the problem is exactly the same as we need only minimize the negative function), and in order to do so, a function of said quantity is required. In our situation, examples of suitable objective functions could be the amount of fuel burnt or the C_L/C_D ratio.

- Design variable: the way in which the solver is capable of changing the value of the objective function is through the defined design variables; these are the parameters of the problem which are allowed to change. Therefore, the optimization problem can be thought of as finding the set of values for the defined set of design variables which yields a minimum of the objective function. Examples for our case include the angle of attack (α), wing span or the throttling variable (δ_T).

- Discipline analysis: for each of the disciplines considered in an MDO problem, we must define the system of equations which describes its functioning. In the remaining sections of the chapter some background will be provided regarding the analysis (numerical) methods employed in this work, and these will be covered more in-depth in chapter 3.

- State variable: the outputs of the several analyses which constitute the current state of the system (i.e., at that iteration). Examples include the flow pressure or the structural displacement distributions. These variables may or not be used in the optimization process, and some are required for other analyses, which are described next.

- Coupling variable: these constitute the state variables which are necessary for other analyses; for example the aerodynamic forces applied upon the wing structure.

- Constraint: so that our problem yields applicable results, constraints must be applied. These can be simple as impositions on the bounds of some design variable (limiting $-10^{\text{o}} \leq \alpha \leq +10^{\text{o}}$ or $0.0 \leq \delta_T \leq 1.0$) or more complex relations between metrics (enforcing $L = D$ or the non-failure of any of the finite elements).

In the following sections, many of these notions will be referred to. This work will borrow the terminology from A. B. Lambe and J. R. R. A. Martins [21], as described in table 2.1:

Table 2.1: Notation in MDO problem formulation

Symbol	Definition
$\boldsymbol{x}$	Vector of design variables
$\boldsymbol{y}$	Vector of coupling variables
$\overline{\boldsymbol{y}}$	Vector of state variables
f	Objective function
$\boldsymbol{c}$	Vector of design constraints
$\boldsymbol{c^c}$	Vector of consistency constraints
$\mathcal{R}$	Governing equations of a discipline analysis in residual form (discipline analysis constraints)
N	Number of disciplines
$n_{()}$	Length of a given variable vector
$m_{()}$	Length of a given constraint vector
$()_0$	Functions or variables that are shared by more than one discipline
$()_i$	Functions or variables that apply only to discipline i
$()^*$	Functions or variables at their optimal value
$\widetilde{()}$	Approximations of a given function or vector of functions
$\widehat{()}$	Independent copies of variables distributed to other disciplines

2.1.2 Architectures

Multidisciplinary optimization methods can be classified in many different categories by several factors. One of the most important classifications in which they can be broken down is regarding their overarching architecture. As explained in [21], there is a certain degree of interchangeability between the term ”architecture” and other similar terms, such as ”procedure” or ”strategy”. The following subsections will borrow several notions and definitions from [21], namely the definition for this term: a proper

representation of a given architecture should leave no doubt as to how the optimization should be set up.

An MDO architecture describes how said method handles the coupling of the several disciplines (i.e., their models). Within this designation, two main architectures can be defined: monolithic architectures, and distributed architectures.

Monolithic Architectures

In a monolithic architecture framework, the MDO problem is approached as a conventional nonlinear programming problem: that is, it is solved as a single optimization problem, where we seek out the design variable values (within the constraints) which minimize our objective function. To provide an example of such an architecture, let us analyze the formulation of a Simultaneous Analysis and Design (SAND) architecture based problem,

$$
\begin{aligned}
&\text{minimize} && f_0(\boldsymbol{x}, \boldsymbol{y}) \\
&\text{with respect to} && \boldsymbol{x}, \boldsymbol{y}, \overline{\boldsymbol{y}} \\
&\text{subject to} && \boldsymbol{c}_0(\boldsymbol{x}, \boldsymbol{y}) \geq 0 \\
& && \boldsymbol{c}_i(\boldsymbol{x}_0, \boldsymbol{x}_i, \boldsymbol{y}_i) \geq 0, \text{ for } i = 1, 2, \ldots, N \\
& && \mathcal{R}_i(\boldsymbol{x}_0, \boldsymbol{x}_i, \boldsymbol{y}, \overline{\boldsymbol{y}}_i) = 0, \text{ for } i = 1, 2, \ldots, N
\end{aligned}
\tag{2.1}
$$

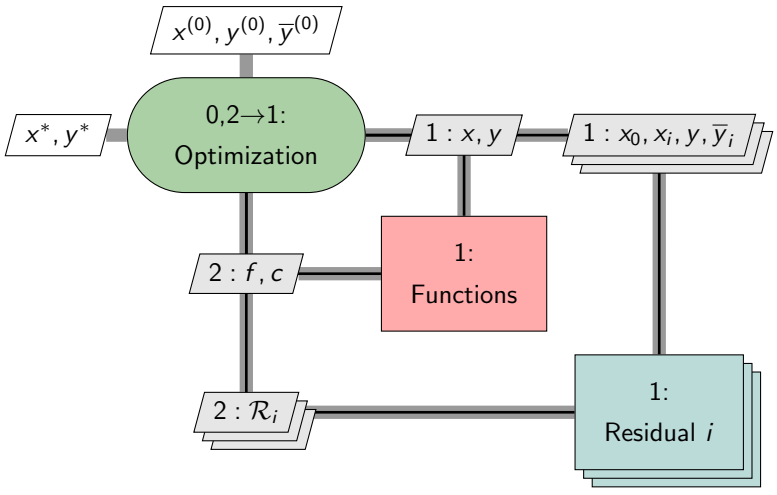


Figure 2.1: Extended design matrix for a SAND architecture.

Figure 2.1 depicts the eXtended Design Structure Matrix (XDSM) for such an architecture. This terminology will be used throughout this work, and is borrowed from A. B. Lambe and J. R. R. A. Martins [24]. Inputs are represented above the process itself (such as the initial conditions $\boldsymbol{x}^{(0)}$, $\boldsymbol{y}^{(0)}$ and $\overline{\boldsymbol{y}}^{(0)}$ in this example), whilst the outputs are placed to the left (usually the optimal conditions, as is the case here: $\boldsymbol{x}^*$ and $\boldsymbol{y}^*$). Also, in these diagrams the processes are broken down into stages, as sequenced by the numerals within each box: the process is initialized on (0) with the initial conditions; constraints,

objective function and discipline analyses are performed on (1); and its results fed back on (2). Loops are symbolized by $\rightarrow$, meaning that for this example the process will return to stage (1) if it has not yet converged on (2). This numbering system allows us to understand what processes are carried out in parallel in the architecture.

Let us now have a look at a Multidisciplinary Feasible (MDF) architecture, namely the one we will be using during this work:

$$
\begin{aligned}
&\text{minimize} && f_0\left(\boldsymbol{x}, \boldsymbol{y}(\boldsymbol{x}, \boldsymbol{y})\right) \\
&\text{with respect to} && \boldsymbol{x} \\
&\text{subject to} && \boldsymbol{c}_0\left(\boldsymbol{x}, \boldsymbol{y}(\boldsymbol{x}, \boldsymbol{y})\right) \geq 0 \\
& && \boldsymbol{c}_i(\boldsymbol{x}_0, \boldsymbol{x}_i, \boldsymbol{y}_i(\boldsymbol{x}_0, \boldsymbol{x}_i, \boldsymbol{y}_{j \neq i})) \geq 0, \text{ for } i = 1, 2, \ldots, N
\end{aligned}
\tag{2.2}
$$

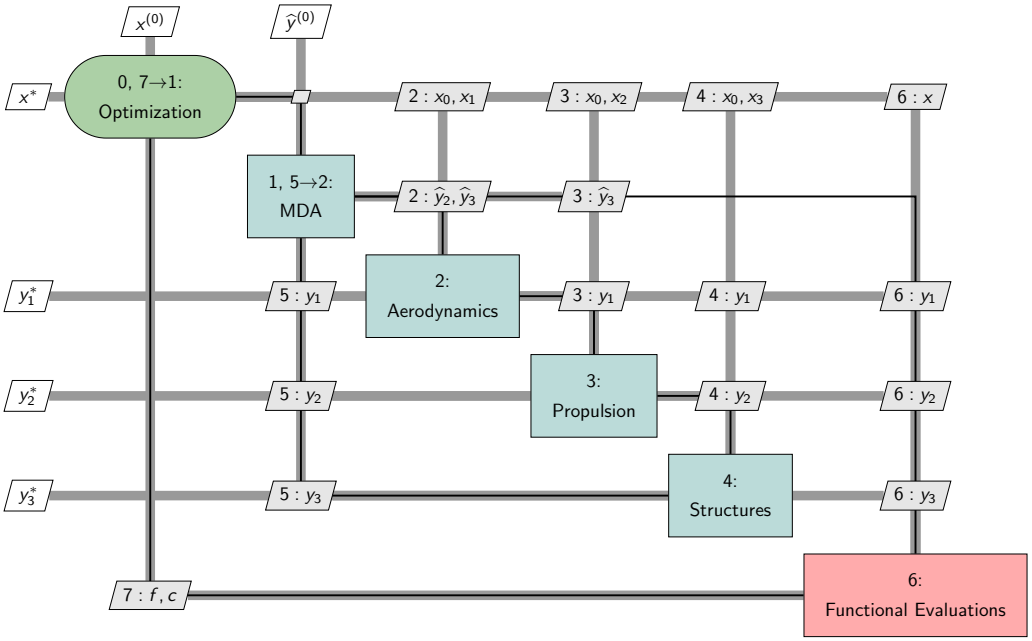


Figure 2.2: XDSM for a MDF architecture with a Gauss-Seidel MDA solver.

Following the sequential numeral flow in fig. 2.2, we can see that the MDA iterates through all of the discipline analyses until it converges, meaning that at every optimization iteration we have a consistent set of feasible coupling variables. The feasibility is a result of the design choice behind a feasible architecture, be it for an individual discipline or several: the removal of the discipline analysis constraints ($\mathcal{R}_i$) which is achieved by applying the implicit function theorem to these constraints. We can also see these differences in the problem formulation [21][25].

This can be advantageous from an engineering point-of-view, if the process must be terminated before the optimization is fully complete. Notably, however, we can see that the design constraints are only evaluated (and iterated upon) after the MDA has converged, meaning the same cannot be said regarding these constraints.

While this can be a very desirable attribute for an architecture, it does mean the entire MDA must be converged at each optimization iteration, which essentially requires the development of two iterative procedures. Namely one for MDA alone [21].

Distributed Architectures

For the sake of completion, let us now have a quick look at distributed architectures. As described at the beginning of this subsection, monolithic architectures tackle the optimization problem as a single one. Distributed architectures instead split it into smaller subproblems, which yield the same problem when put back together. The main reason why this might be a more desirable approach lies with two main factors, a mathematical one and a practical one:

From a mathematical standpoint, such an approach might be adequate when dealing with systems which are strongly decoupled on some level. This might mean, for example, a problem such as:

$$\begin{aligned} &\text{minimize} && \sum_{i=1}^{N} f_i\left(\boldsymbol{x}_i\right) \\ &\text{with respect to} && \boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \\ &\text{subject to} && \boldsymbol{c}_0\left(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\right) \geq 0 \\ & && \boldsymbol{c}_1\left(\boldsymbol{x}_1\right) \geq 0 \\ & && \ldots \\ & && \boldsymbol{c}_N\left(\boldsymbol{x}_N\right) \geq 0 \end{aligned} \tag{2.3}$$

In a problem as this, the separate functions which add up to our objective function depend on entirely independent variables. The coupling only arises from the global constraints, c_0, and as such this problem is what is described as one with complicating constraints. In a similar fashion, we could define a problem with complicating variables instead, wherein it is exclusively from the shared design variables, x_0, that coupling arises. Decomposition techniques have been developed in order to bridge the gap from solving N independent problems [26].

Besides these technical circumstances, it is also important to take into account how the overall problem is tackled on an industrial level. Teams are usually assigned to different tasks, which essentially can be looked at as our definition of discipline. With this in mind, it can usually be more practical to adopt an architecture which allows for asynchronous processing, especially considering different disciplines will have different computational costs [21].

Let us have a quick overview of what the XDSM for a distributed architecture looks like:

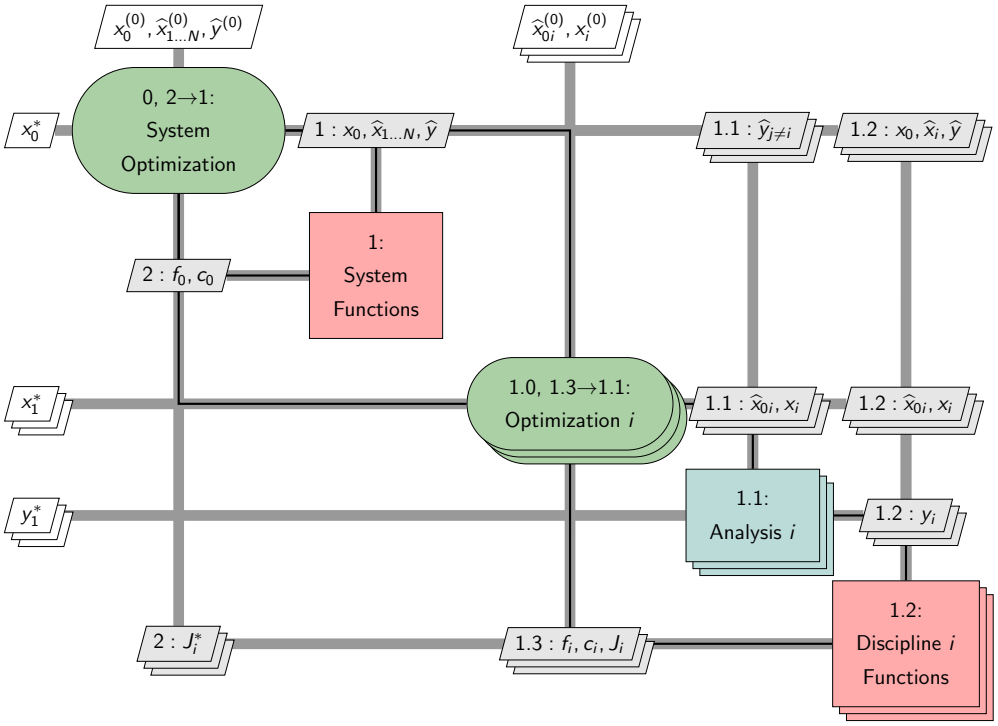


Figure 2.3: XDSM for a CO distributed architecture.

This is an example of a Collaborative Optimization (CO) architecture, where we can see some of the main attributes discussed previously: namely, the usage of two optimization loops. The outer loop, which controls the full system optimization, still looks to minimize the global objective function, f_0,

$$\begin{aligned} &\text{minimize} && f_0(\boldsymbol{x}_0, \widehat{\boldsymbol{x}}_1, \ldots, \widehat{\boldsymbol{x}}_n, \widehat{\boldsymbol{y}}) \\ &\text{with respect to} && \boldsymbol{x}_0, \widehat{\boldsymbol{x}}_1, \ldots, \widehat{\boldsymbol{x}}_n, \widehat{\boldsymbol{y}} \\ &\text{subject to} && \boldsymbol{c}_0\left(\boldsymbol{x}_0, \widehat{\boldsymbol{x}}_1, \ldots, \widehat{\boldsymbol{x}}_n, \widehat{\boldsymbol{y}}\right) \geq 0 \\ & && J_i^* = 0, \text{ for } i = 1, \ldots, N \end{aligned} \tag{2.4}$$

Within the inner loop, we can see that the optimization process is carried out in parallel between the several disciplines (as represented by the stacked display). These analyses receive $\widehat{\boldsymbol{x}}_{0i}$ as an input, which contains copies of the shared design variables values from the latest iteration of the outer optimization routine. Also, a new set of functions has been introduced in the inner loop, J_i. These are defined as,

$$J_i = \left\|\widehat{\boldsymbol{x}}_{0i} - \boldsymbol{x}_0\right\|_2^2 + \left\|\widehat{\boldsymbol{x}}_i - \boldsymbol{x}_i\right\|_2^2 + \left\|\left(\widehat{\boldsymbol{y}}_i - \boldsymbol{y}_i\left(\widehat{\boldsymbol{x}}_{0i}, \boldsymbol{x}_i, \widehat{\boldsymbol{y}}_{j\neq i}\right)\right\|_2^2 \right. \tag{2.5}$$

meaning they describe the difference between the aforementioned copied values of the shared design variables and those computed by the disciplines within the inner optimization loop. Minimizing these will be the objective of the discipline optimization, as seen in the formulation of the problem for the inner loop analyses,

$$
\begin{aligned}
&\text{minimize} && J_i\left(\widehat{\boldsymbol{x}}_{0i}, \boldsymbol{x}_i, \boldsymbol{y}_i(\widehat{\boldsymbol{x}}_{0i}, x_i, \widehat{\boldsymbol{y}}_{j\neq i})\right) \\
&\text{with respect to} && \widehat{\boldsymbol{x}}_{0i}, \boldsymbol{x}_i \\
&\text{subject to} && \boldsymbol{c}_i\left(\widehat{\boldsymbol{x}}_{0i}, \boldsymbol{x}_i, \boldsymbol{y}_i(\widehat{\boldsymbol{x}}_{0i}, \boldsymbol{x}_i, \widehat{\boldsymbol{y}}_{j\neq i})\right) \geq 0
\end{aligned}
\tag{2.6}
$$

As such, the objective of the discipline-level optimization is keep the system consistent; and only when that is achieved may the system optimizer iterate, updating the global design variables $\boldsymbol{x}_0$ [21].

2.1.3 Algorithms

Within the context of our problem, optimization algorithms can be divided into zeroth order methods, also referred to as gradient-free methods; and gradient-based methods [12]. Another important notion is that of sensitivity analysis, which describes how the optimization method performs its derivative computations.

Gradient-free

Gradient-free optimization algorithms comprise approaches such as grid-searching and genetic algorithms, in which the progress made from one iteration to the next is not based on any derivatives, but exclusively on evaluations of the objective function. At their core, the logic behind these algorithms is to move from a design point i to the following point $i+1$ solely because the objective function presents a lower value at the latter. These methods are more adequate when dealing with noisy functions, discrete search spaces or non-differentiable functions [27]. They can also be suitable when dealing with problems with a small amount of design variables, as the low amount of function evaluations required in these circumstances may justify its usage [12] (the number of function evaluations translates into computational cost and this is a determining factor in selecting an algorithm). However, this is not our case, and therefore such methods will not be further covered in this work.

Gradient-based

These are algorithms which define a search direction in which to progress from point i to the next. The direction is based on both evaluations of the objective function and its derivatives with respect to the design parameters, meaning these solvers utilize more information in order to navigate the design space. These algorithms require smooth objective functions within the design space to operate, but then are generally faster as they require fewer function evaluations [12]. They also differ amongst themselves in terms of the degree of derivation utilized. The steepest descent method, which simply follows the gradient vector, utilizes solely first order derivatives; Newton-based methods, for example, utilize the Hessian matrix in their computations (second-order information) [27].

To bypass the computation the second-order derivatives, however, several methods have been developed which fall under the quasi-Newton designation. These algorithms, such as Broyden–Fletcher––Goldfarb–Shanno (BFGS) or Davidon–Fletcher–Powell (DFP), only utilize first-order information, and

approximate any necessary Hessian information based on function values and derivatives from previous iterations [27].

The optimization algorithm employed in this work is SNOPT [28], which is a sequential quadratic programming (SQP) algorithm. The idea behind SQP methods is to break down the original optimization problem (a non-linear programming problem) into a quadratic programming subproblem at design point $\boldsymbol{x}_i$. By solving this significantly simpler subproblem, one where the objective function is quadratic and the constraints are linear functions, the algorithm then proceeds to design point $\boldsymbol{x}_{i+1}$ [27], [29].

In order to provide a brief overview of how this subproblem is built, let us define a few important entities. First, assuming a simple and generic constrained optimization problem as,

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{x}) \\ &\text{with respect to} && \boldsymbol{x} \\ &\text{subject to} && \boldsymbol{c}_n\left(\boldsymbol{x}\right) = 0, \text{ for } n = 1, \ldots, N, \end{aligned} \tag{2.7}$$

we can now define the Lagrange function of this problem based on the objective function f and the constraints $\boldsymbol{c}_n$ as,

$$\mathcal{L}(x, \lambda) = f(x) + \lambda^T c(x) \tag{2.8}$$

where λ is the vector of Lagrange multipliers, a set of scalars. The reason why these multipliers and function are important is because they allow us to express important conditions as,

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial x} = \frac{\partial f(x)}{\partial x} + \sum_{i=1}^{N} \lambda_i \frac{\partial c_i(x)}{\partial x} = 0 \\ \frac{\partial \mathcal{L}}{\partial \lambda_i} = c_i(x) = 0, \text{ for } i = 1, \ldots, N \end{cases} \tag{2.9}$$

where we can see the first equation expresses two optimality conditions: $\nabla f = \nabla c_n = 0$ (the first-order Karush–Kuhn–Tucker (KKT) conditions) and the second equation expresses the constraints of the original problem. In this way, the Lagrange formulations allow us to reduce the original constrained optimization problem into an unconstrained one, where the amount of design variables has increased by N, the number of constraints [27].

Going back to the SQP formulation, let us once again assume an optimization problem such as the one just mentioned. At design point $\boldsymbol{x}_i$, let us assume the quadratic subproblem

$$\begin{aligned} &\text{minimize} && \frac{1}{2}\boldsymbol{p}^T \nabla^2 \left(f(\boldsymbol{x})\right)_i \boldsymbol{p} + \nabla \left(f(\boldsymbol{x})\right)_i \boldsymbol{p} \\ &\text{subject to} && \nabla \left(\boldsymbol{c}_n(\boldsymbol{x})\right)_i \boldsymbol{p} + \left(\boldsymbol{c}_n(\boldsymbol{x})\right)_i = 0, \text{ for } n = 1, \ldots, N, \end{aligned} \tag{2.10}$$

where $\boldsymbol{p}$ is the factor which will help us determine the next point. Because we are evaluating all functions at point $\boldsymbol{x}_i$, there are no unknowns besides $\boldsymbol{p}$.

Let us now define the following Lagrange function for this problem,

$$\mathcal{L}(x, \lambda) = f(x) - \lambda^T c(x), \tag{2.11}$$

and perform the first-order derivatives of this function as previously discussed,

$$\begin{cases} \nabla^2 \left(f(x)\right)_i p + \nabla \left(f(x)\right)_i - \nabla \left(c_n(x)\right)_i^T \lambda = 0 \\ \nabla \left(c_n(x)\right)_i p + \left(c_n(x)\right)_i = 0, \text{ for } n = 1, \ldots, N \end{cases} \tag{2.12}$$

we can now express this system in matrix form,

$$\begin{bmatrix} \nabla^2 \left(f(x)\right)_i & -\nabla \left(c_n(x)\right)_i^T \\ -\nabla \left(c_n(x)\right)_i & 0 \end{bmatrix} \begin{bmatrix} p \\ \lambda \end{bmatrix} = \begin{bmatrix} -\nabla \left(f(x)\right)_i \\ -\left(c_n(x)\right)_i \end{bmatrix}, \tag{2.13}$$

where we can see the system takes the same form as a traditional Newton approach, where the step to the next design point is obtained from solving a system of the form $H \times step = -\nabla \left((f(x)\right)$. In this way, the (p, λ) array in eq. (2.13) can be interpreted as:

$$\begin{bmatrix} p_i \\ \lambda_{i+1} \end{bmatrix}, \tag{2.14}$$

with which we can update the design point,

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \boldsymbol{p}_i \, . \tag{2.15}$$

The computed λ in eq. (2.14) presents the subscript $i + 1$ since it already is the updated value [27]. It should also be mentioned that most applications of SQP methods, including the utilized SNOPT algorithm, utilize Hessian approximations for the second-order terms, making these Quasi-Newton optimization methods.

Sensitivity Analysis

As was introduced in the previous section, the computation of derivatives of quantities such as the objective function or the constraints w.r.t. the variables of the problem is an important aspect of how the problem is approached. This is what is called the sensitivity analysis, i.e., how are the performance metrics of our solution sensible to the parameters that drive the design process.

Several methods of sensitivity analysis exist, amongst which some of the most popular include complex-step derivatives or finite differences, which use formulae as simple as

$$\frac{\partial f}{\partial x} = \frac{f(x+h) - f(x)}{h} + \mathcal{O}(h) \, , \tag{2.16}$$

where h is the step size and $\mathcal{O}(h)$ denotes that the truncation error for this particular expression is proportional to h.

These are methods used to expressly to compute partial derivatives: the influence of one variable on a given function of interest. In order to analyze a more complex MDO problem such as the ones tackled in this work, methods have been developed to efficiently compute all the derivatives involved [30], [31].

How the sensitivity analysis is performed within a specific problem relies deeply on how data is being processed from an implementation point-of-view. In our situation, the main goal of methods such as algorithmic or monolithic differentiation [30] is to provide a way to compute derivatives across the several components which make up the optimization structure. Essentially, rather than just considering the impact that a given variable has on a specific function, we also want to factor in the influence that same variable has on state or coupled variables.

To this end, the MDAO framework utilized in this work employs a Modular Analysis and Unified Derivatives (MAUD) sensitivity analysis architecture [31]. From a theoretical perspective, what a MAUD system does is concatenate all design variables, state/coupled[1] variables and output functions (such as the objective function and constraints) into a single vector $\boldsymbol{u}$ as

$$\boldsymbol{u} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_n, \boldsymbol{f}_1, \ldots, \boldsymbol{f}_p] \,, \tag{2.17}$$

and assemble residual functions of the form

$$\boldsymbol{R}_i(\boldsymbol{u}) = \boldsymbol{x}_i - \boldsymbol{x}_i^*, \text{ for } i = 1, \ldots, m \tag{2.18}$$

for all design variables,

$$\begin{cases} \boldsymbol{R}_{m+i}(\boldsymbol{u}) = \boldsymbol{y}_i - \boldsymbol{y}_i^*(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_n) & \text{if } \boldsymbol{y}_i \text{ is explicitly defined} \\ \boldsymbol{R}_{m+i}(\boldsymbol{u}) = -\mathcal{R}_i((\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_n) & \text{if } \boldsymbol{y}_i \text{ is implicitly defined} \end{cases} \text{, for } i = 1, \ldots, n \tag{2.19}$$

for state/coupled variables, and

$$\boldsymbol{R}_{m+n+i}(\boldsymbol{u}) = \boldsymbol{f}_i - \boldsymbol{f}_i^*(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_n) \text{ , for } i = 1, \ldots, p \tag{2.20}$$

for all output functions. For eqs. (2.18–20), the design variable values marked with an asterisk, x^*, represent the point at which we are evaluating the expressions. As for y^* and f^*, these represent the function which yields their values, evaluated at x^*.

We may now, similarly to what was done in eq. (2.17), concatenate all of these residual equations into a single system, obtaining

$$\begin{cases} \boldsymbol{R}_1(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n) = 0 \\ \qquad\qquad \vdots \qquad\qquad\quad \Rightarrow \boldsymbol{R}(\boldsymbol{u}) = 0 \,. \\ \boldsymbol{R}_n(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n) = 0 \end{cases} \tag{2.21}$$

We can now define the vector $\boldsymbol{r}$, which will designate the system residual vector. This means that we want to solve the system laid out in eq. (2.21) at $\boldsymbol{r} = 0$.

Under these circumstances we may combine $\boldsymbol{u}$ from eq. (2.17) and the system from eq. (2.21) in

[1] In the brief description presented here the distinction between state and coupled variables is not pertinent, and both are represented by the $\boldsymbol{y}_n$ vectors.

order to define

$$\frac{d\boldsymbol{u}}{d\boldsymbol{r}} = \frac{\partial(\boldsymbol{R}^{-1})}{\partial \boldsymbol{r}} \,, \tag{2.22}$$

evaluated at $\boldsymbol{r} = 0$. We have also assumed that $\partial \boldsymbol{R}/\partial \boldsymbol{u}$ is invertible at the solution. Equation (2.22) allows us to, by the inverse function theorem, define

$$\frac{\partial(\boldsymbol{R}^{-1})}{\partial \boldsymbol{r}} = \left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{u}}\right]^{-1} \,, \tag{2.23}$$

which we can now combine with eq. (2.22) to obtain

$$\frac{d\boldsymbol{u}}{d\boldsymbol{r}} = \left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{u}}\right]^{-1} \Leftrightarrow \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{u}}\frac{d\boldsymbol{u}}{d\boldsymbol{r}} = \mathcal{I} = \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{u}}^{T}\frac{d\boldsymbol{u}}{d\boldsymbol{r}}^{T} \,. \tag{2.24}$$

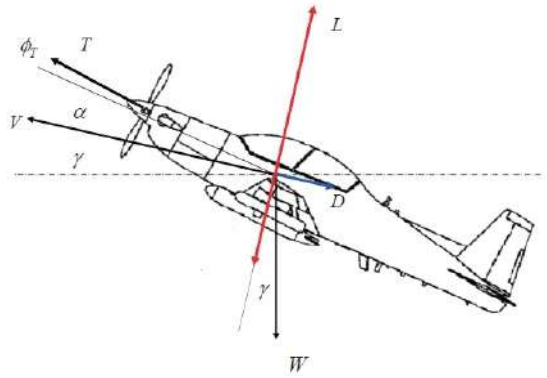
Equation (2.24) is designated as the unifying derivative equation. The main takeaway from this introduction to this particular analysis is that we have reduced the computation of all relevant derivatives to solving the system laid out in eq. (2.24). A full discussion and derivation of the MAUD architecture is presented in Martins and Hwang [31].

From a practical point, the great advantage of this architecture is that for each of the logic blocks in which we will be performing computations we only need to define the partial derivatives of the outputs w.r.t. the inputs for that block in specific. The assembling and computation of the full model derivatives is carried out automatically [31].

2.2 Dynamics & Equilibrium

As a starting point to this section, let us have a look at the three equilibrium equations to be defined for longitudinal in-plane motion: vertical forces, horizontal forces, and moment. In doing so, we will establish a foundation for what other quantities we will need to define and obtain from the following sections.

Let us derive a dimensional version of these equations from the simplified diagram represented in fig. 2.4, assuming steady flight conditions:



$$L - W cos(\gamma) + T sin(\alpha + \phi_T) = 0 \tag{2.25a}$$

$$T cos(\alpha + \phi_T) - D - W sin(\gamma) = 0 \tag{2.25b}$$

$$M = 0 \tag{2.25c}$$

Figure 2.4: Free body diagram for trim.

Equations (2.25a–c) describe the system in broad terms and allow for a simplified overview of the

equilibrium conditions. In order to obtain a more specific formulation of these conditions, let us consider the more comprehensive diagram in fig. 2.5, where the relative positions of both surfaces and the engine are discretized.

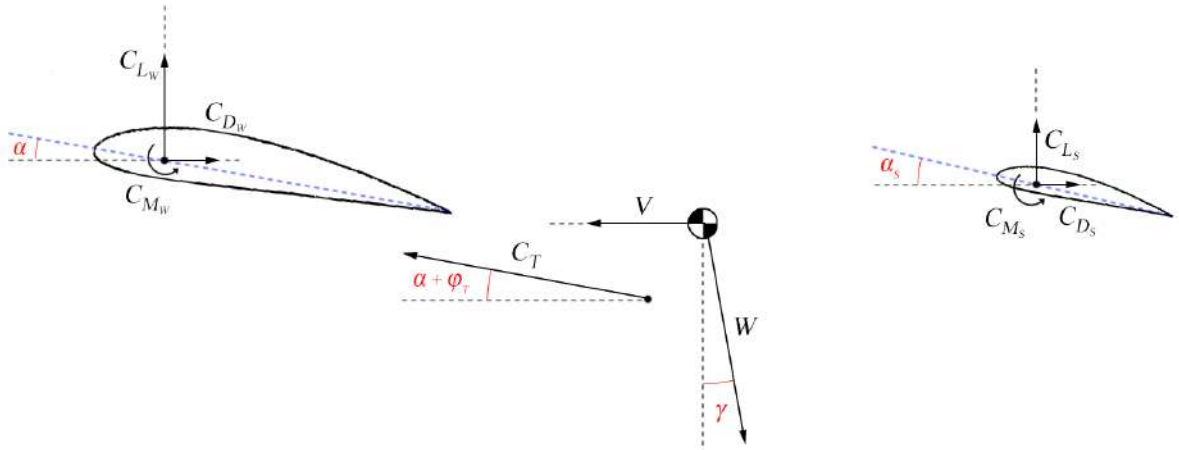


Figure 2.5: Exhaustive free body diagram.

The forces and moments have also been replaced by their respective dimensionless coefficients. With this diagram, we may now define the same equations contemplating all separate contributions.
The two force-equilibrium equations become:

$$\frac{1}{2}\rho V^2 S_W \left(C_{L_W} + \frac{S_S}{S_W} C_{L_S}\right) + \frac{1}{2}\rho^2 (S_W + S_S) C_T sin(\alpha + \phi_T) - W cos(\gamma) = 0 \tag{2.26a}$$

$$\frac{1}{2}\rho^2 (S_W + S_S) C_T cos(\alpha + \phi_T) - \frac{1}{2}\rho V^2 S_W \left(C_{D_W} + \frac{S_S}{S_W} C_{D_S}\right) - W sin(\gamma) = 0 \tag{2.26b}$$

In both eq. (2.26a) and eq. (2.26b), the corrective term applied to the coefficients of the stabilizer arises from the fact that these coefficients are defined w.r.t. the area of the stabilizer. It can also be seen that the effective angles of attack are no longer the same from the wing to the stabilizer. This difference arises from the downwash effect, and the tail incidence angle, a design variable which will be covered later on. The different normalization applied to the thrust coefficient, C_T, will be discussed in section 2.5.
We can also define expressions relating the full-aircraft lift and drag forces used in eq. (2.25a) and eq. (2.25b) with these coefficients:

$$L = \frac{1}{2}\rho V^2 S_W \left(C_{L_W} + \frac{S_S}{S_W} C_{L_S}\right) \tag{2.27a}$$

and

$$D = \frac{1}{2}\rho V^2 S_W \left(C_{D_W} + \frac{S_S}{S_W} C_{D_S}\right) \tag{2.27b}$$

For the moment equilibrium equation, let us take moments about the aircraft's center of mass (CM), and let $\boldsymbol{r}_w$, $\boldsymbol{r}_s$, and $\boldsymbol{r}_t$ be the positions of the aerodynamic center of the wing, the aerodynamic center of the stabilizer, and the point of application of the thrust force, respectively, in relation to the CM. Also, let $\bar{c}$ be the mean aerodynamic chord of the wing,

$$\frac{\boldsymbol{r}_t}{\bar{c}} \times \boldsymbol{C}_T + \left[\boldsymbol{C}_{M_W} + \frac{\boldsymbol{r}_w}{\bar{c}} \times \boldsymbol{C}_{L_W} + \frac{\boldsymbol{r}_w}{\bar{c}} \times \boldsymbol{C}_{D_W}\right] + \left[\boldsymbol{C}_{M_S} + \frac{\boldsymbol{r}_s}{\bar{c}} \times \boldsymbol{C}_{L_S} + \frac{\boldsymbol{r}_s}{\bar{c}} \times \boldsymbol{C}_{D_S}\right] = \boldsymbol{0}\,. \quad (2.27c)$$

Equations (2.26a), (2.26b) and (2.27c) represent the final scheme which we look to implement.

2.3 Aerodynamics

For the sake of completion, a brief derivation of some of the most essential proprieties and equations of fluid dynamics pertaining to the task at hand will be presented in this section. There is extensive bibliography covering these topics; in this document, we will be following methodologies described in Anderson [32] and Bertin, Cummings & Reddy [33].

2.3.1 Fundamental Equations

To begin to define the models we will be working with, the most fundamental equations must be established. In this scenario, the first one will arise from the conservation of mass: the continuity equation.

Let us regard fig. 2.6, depicting a control volume to be used in the subsequent developments.

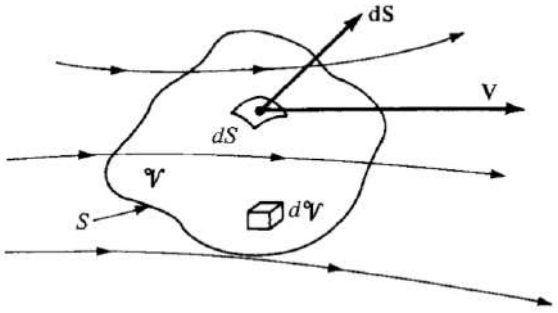


Figure 2.6: Control volume for a generic flow.

By the fundamental principle that mass can neither be created nor destroyed we can derive that, for the control volume dV, the amount of mass flowing in and out of its boundaries must be the rate of variation of the total mass within the volume, yielding

$$\frac{\partial}{\partial t} \iiint_{\mathcal{V}} \rho d\mathcal{V} + \iint_S \rho \boldsymbol{V} \cdot \boldsymbol{dS} = 0\,. \quad (2.28)$$

It should be noted that equation eq. (2.28) is expressed as a sum rather than a difference since, by convention, the product $\boldsymbol{V} \cdot \boldsymbol{dS}$ represents an outflow when positive. This means we are adding the rate of mass increase within the control volume to the outflow rate, which must cancel out.

Considering that the control volume is independent from time, invoking the Gauss theorem and some algebraic manipulation, we obtain the differential version of the continuity equation,

$$\iiint_{\mathcal{V}} \frac{\partial \rho}{\partial t} d\mathcal{V} + \iiint_{\mathcal{V}} \nabla \cdot (\rho \boldsymbol{V}) d\mathcal{V} = 0 \iff \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{V}) = 0 \tag{2.29}$$

which can be used to evaluate any point of the flow, rather than restraining its scope to an arbitrarily defined control volume. In order to simplify this equation further and arrive at the starting point of the employed model, we need now to embed assumptions regarding the flow and reference frame. Firstly, it should be noted that the problem to be solved in the aerodynamic analysis (fully developed flow around an airfoil) is a steady flow situation, meaning any derivatives w.r.t. time can be removed. Also, as only subsonic flow situations will be covered in this work, we will be considering the air density constant (i.e., $\rho =$ constant). Unlike the largely inconsequential steady flow assumption, considering the flow incompressible is very much a trade-off situation: it allows the usage of easier to implement methods, but is a rather crude approximation,

$$\frac{d\rho}{\rho} = -M_{\infty}^2 \frac{dV}{V} \tag{2.30}$$

This equation arises from algebraic manipulation of the conservation of momentum and isentropic flow equations. A full derivation can be consulted on [34]. Equation (2.30) shows us that $d\rho \propto M_{\infty}^2$, providing insight into the bearing that the Mach number will have in terms of compressibility.

Applying both of these conditions to eq. (2.29), we obtain:

$$\rho \nabla \cdot (\boldsymbol{V}) = 0 \iff \nabla \cdot (\boldsymbol{V}) = 0 \tag{2.31}$$

Furthermore, as mentioned in Anderson ([32], Sec. 2.12), the subsonic flow across an airfoil is fundamentally irrotational, which means

$$\xi = \nabla \times (\boldsymbol{V}) = 0 \,. \tag{2.32}$$

As such, the velocity field of the problem at hand can be defined as that gradient of a scalar ϕ, the velocity potential. Combining this relation with eq. (2.29) yields

$$\begin{cases} \boldsymbol{V} = \nabla \phi \\ \nabla \cdot (\boldsymbol{V}) = 0 \end{cases} \Rightarrow \nabla \cdot (\nabla \phi) = 0 \Rightarrow \nabla^2 \phi = 0 \,. \tag{2.33}$$

The end result of eq. (2.33) is an instance of Laplace's equation, whose pertinent solutions to our problem are discussed in the following section.

Because we will be dealing with Mach numbers above the commonly defined threshold for incompressible flow ($M_{\infty} \leq 0.3$), a corrective method will be employed: the Prandtl-Glauert transformation. Let us assume an xyz frame where the freestream is aligned with the x-axis, such as the one illustrated in fig. 2.7, and define the velocity field as the sum of the freestream velocity, V_{∞}, and a perturbation velocity field,

$$V = V_{\infty} + \widehat{V} \,, \tag{2.34}$$

denoted by $\widehat{V}$.

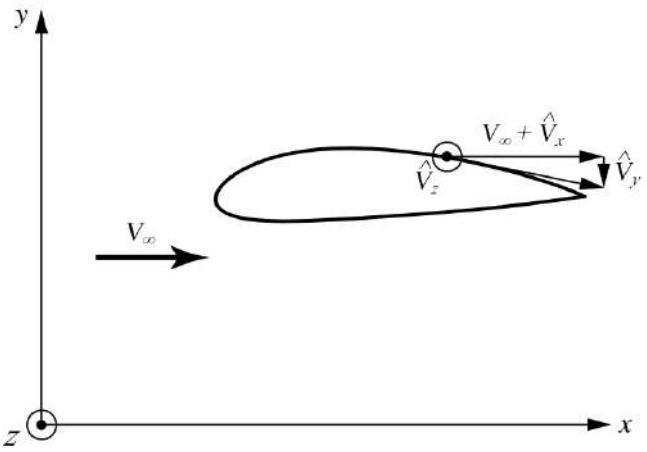


Figure 2.7: Reference frame for the Prandtl-Glauert transformation.

The basis for this approach then starts from the linearized compressible potential equation,

$$\beta^2 \nabla^2_{xx} \widehat{\phi} + \nabla^2_{yy} \widehat{\phi} + \nabla^2_{zz} \widehat{\phi} = 0 \,, \tag{2.35}$$

where β is the Prandtl-Glauert factor defined as

$$\beta^2 = 1 - M^2_\infty \,, \tag{2.36}$$

and $\widehat{\phi}$ is the perturbation velocity potential.

A full derivation for eq. (2.35) can be consulted in [32]. Most importantly, it is applicable when two additional conditions are met:

- The perturbation velocities are small, meaning the lifting surfaces are slender and at a small angle-of-attack;
- The flow is subsonic, namely $M_\infty \leq 0.8$.

Both of these assumptions are adequate in our situation, and express the increased Mach number range we wish to achieve with the correction.

The correction itself consists of defining a new coordinate system, $\tilde{x}\tilde{y}\tilde{z}$, as

$$\begin{cases} \tilde{x} = x \\ \tilde{y} = \beta y \\ \tilde{z} = \beta z \end{cases} \,, \tag{2.37}$$

and scaling the angle of attack, α, and perturbed velocity potential, $\widehat{\phi}$, to α_s and $\widehat{\phi}_s$ respectively:

$$\begin{cases} \alpha_s = \beta \alpha \\ \widehat{\phi}_s = \beta^2 \widehat{\phi} \end{cases} \,. \tag{2.38}$$

Converting eq. (2.35) to the new frame and replacing $(\alpha, \widehat{\phi})$ with $(\alpha_s, \widehat{\phi}_s)$ yields

$$\nabla^2_{\tilde{x}\tilde{x}}\widehat{\phi}_s + \nabla^2_{\tilde{y}\tilde{y}}\widehat{\phi}_s + \nabla^2_{\tilde{z}\tilde{z}}\widehat{\phi}_s = 0 \Leftrightarrow \nabla\widehat{\phi}_s = 0\,, \tag{2.39}$$

which is, once again, a formulation of the aforementioned Laplace's equation (eq. (2.33)). This means we can solve the compressible flow problem utilizing incompressible approaches, by first converting the problem as described. The results obtained in the new frame, $\tilde{x}\tilde{y}\tilde{z}$, can then be converted back to the original coordinate system.

2.3.2 Laplace's Equation - Vortex-based Flow

Since Laplace's equation is linear, solutions for more complex flow situations can be devised by combining simpler elements, i.e. if ϕ_i are all solutions to Laplace's equation on their own, then

$$\phi = \sum_{n=i}^{n} \phi_i \tag{2.40}$$

is also a solution.

As such, let us now introduce the vortex flow element, which will be the most essential tool in modelling the lifting surfaces. It is defined as a point about which we have concentric streamlines of constant velocity.

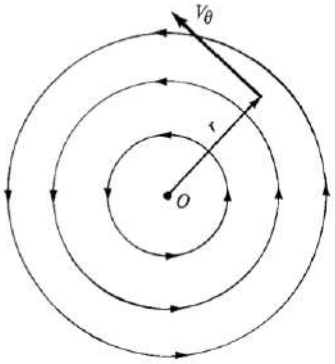


Figure 2.8: A vortex flow element.

The induced radial velocity induced by the element is inversely proportional to its origin; additionally, no radial velocity is imposed,

$$\begin{cases} V_\theta = \frac{C}{r} \\ V_r = 0 \end{cases}\,. \tag{2.41}$$

With this, we can now compute the flow's circulation, one of its most important metrics. Circulation is defined as

$$\Gamma = \oint \boldsymbol{V} \cdot ds \Rightarrow \Gamma = -V_\theta(2\pi r) \Leftrightarrow V_\theta = -\frac{\Gamma}{2\pi r}\,. \tag{2.42}$$

In eq. (2.42) not only is the value of the circulation inferred, but it can also be seen that it is, effectively, a measure of the vortex's strength. In fact, by merely knowing a vortex's circulation, we can compute the velocity it induces at any point. Introducing this flow element is crucial as it is the sole building block for several wing-modelling schemes. In order to move from a simple vortex to something more applicable, let us now discuss the concept of vortex filament.

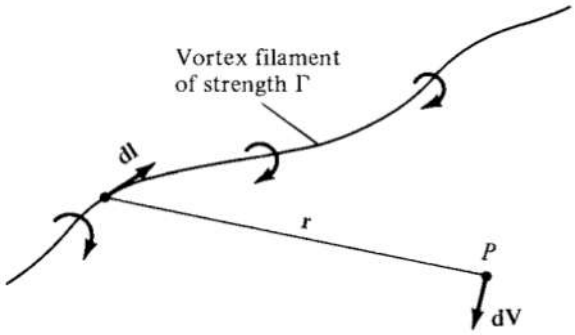


Figure 2.9: An arbitrarily-shaped vortex filament of strength Γ.

As suggested in fig. 2.9, a vortex filament consists of a continuous line about which a vortex flow is induced. Given the nature of this flow element, we can resort to the Biot-Savart Law in order to determine the velocity induced at point P by the filament,

$$d\boldsymbol{V} = \frac{\Gamma}{4\pi}\frac{\boldsymbol{dl}\times\boldsymbol{r}}{\|\boldsymbol{r}\|^3}\,. \tag{2.43}$$

By integrating eq. (2.43) over the length of the filament, the induced velocity is obtained.

Let us now introduce two theorems pertaining to the study of inviscid, incompressible flow via vortex filament-modelling. There were established by Helmholtz, and as such are widely known as the Helmholtz's vortex theorems. These state that:

1. The strength of a vortex filament (i.e., its circulation) is constant along its length;

2. A vortex filament cannot end in a fluid; it must extend to the boundaries (be they finite or infinite) of the fluid or form a closed path.

2.3.3 The Lifting-Line Method

With these conditions in mind, we can now establish a wing model based on a single vortex filament, for the sake of demonstration (as illustrated in fig. 2.10).

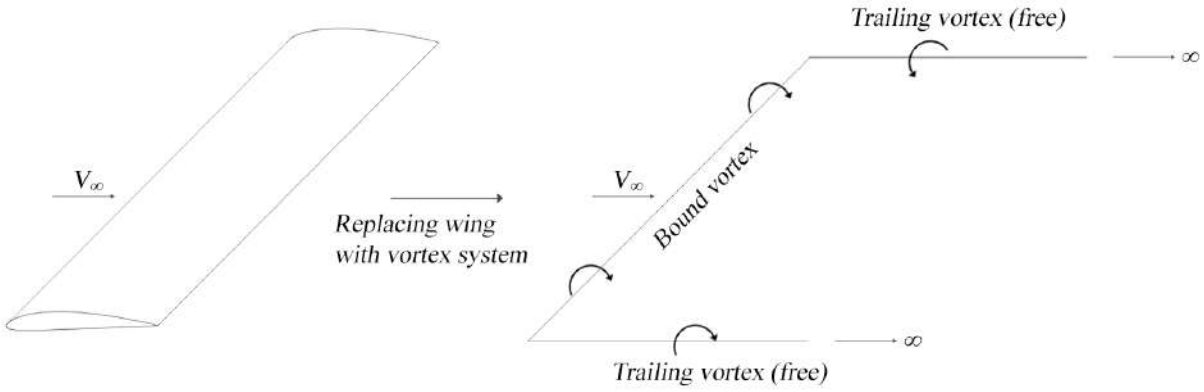


Figure 2.10: Diagram for a simple vortex wing-modelling scheme.

The segment denoted as bound is fixed in its location, and represents the wing itself. The trailing vortexes presented above and below the bound one arise from Helmholtz's theorems; namely from the fact that a vortex filament cannot end in fluid. As such, the bound vortex trails off to infinity in the form of free vortexes, maintaining its circulation. This is called a horseshoe vortex.

Expanding upon this approach, we can now model the wing by superimposing a larger number of horseshoe vortexes.

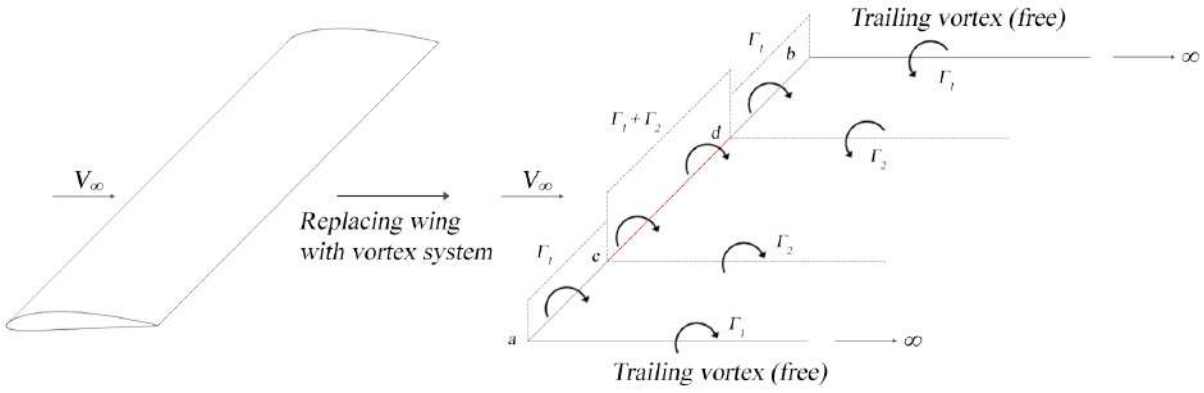


Figure 2.11: A lifting line-based scheme with two horseshoe vortexes.

Figure 2.11 displays a two bound vortex system, and allows us to see what the main features of this approach are as we add more vortexes:

1. Only a single bound vortex acts on the entirety of the wing's span (i.e., the lifting line). In fig. 2.11 we can see that Γ_1 is the only effecting vortex on segments $\overline{ac}$ and $\overline{db}$. It is also the circulation that then trails off at the ends of the wing;

2. On $\overline{cd}$, Γ_2 is superimposed, and we can see that as a result, the strength of the bound vortex on this segment is $\Gamma_1 + \Gamma_2$. Also, because the vortex is bound to $\overline{cd}$, it will trail off to infinity from these two points.

Describing the surface as a single lifting line comprising several bound vortexes constitutes what is known as a lifting-line theory (*LLT*) model. Such a model, however, only allows discretization in the

spanwise direction; by means of increasing the number of bound vortexes as demonstrated going from figs. 2.10 and 2.11.

2.3.4 The Vortex-Lattice Method

In order to enable a two-dimensional description of the surface, let us superimpose several lifting-lines (similar to the ones presented thus far) along the chordwise direction. By constructing a mesh of horse-shoe vortexes as such, we have defined a grid (or lattice) of control points. For each of these points, a velocity will be induced by each of the vortexes we have discretized (whose value we can obtain via eq. (2.43)), and it is therefore possible to assemble a system relating all the vortex circulations (Γ_n) to the induced velocities on all control points (v_n), which can be described as

$$v_n = \sum_{m=1}^{N} \{C_{n,m}\}\Gamma_m \,, \tag{2.44}$$

where N represents the total number of vortexes defined and $C_{n,m}$ represents a row of the aerodynamic influence coefficient (AIC) matrix. These coefficients represent the influence of each vortex on each control point. In eq. (2.44), we are essentially determining what the induced velocity at control point n is, by adding the contributions of all N vortexes.

This extension of the LLM is called the Vortex Lattice Method (VLM).

Let us have a closer look at a simple example, so we may better visualize what this process entails.

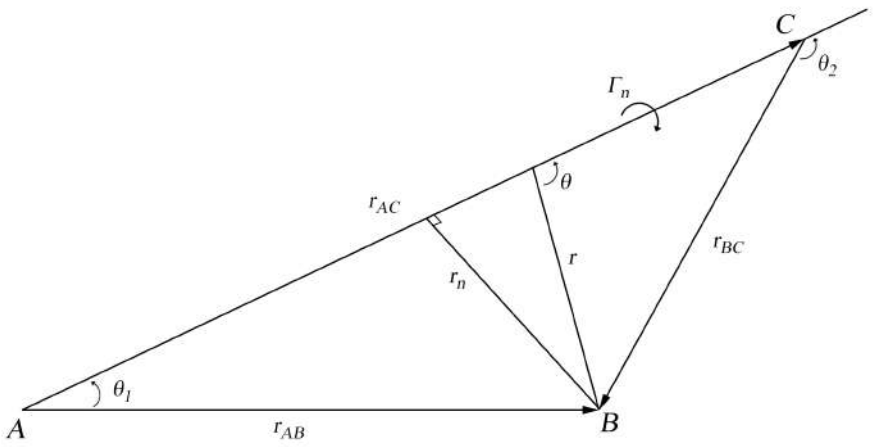


Figure 2.12: Influence of vortex Γ_n on control point B.

Infig. 2.12, a vortex of intensity Γ_n is applied along the line segment $\overline{AC}$. Point B represents one of the control points upon which the influence of the several vortexes will be evaluated. Resorting to eq. (2.43), we can compute the impact of Γ_n on B. Let us start by developing the cross-product between $\boldsymbol{dl}$ and $\boldsymbol{r}$,

$$d\boldsymbol{V} = \frac{\Gamma_n}{4\pi}\frac{\boldsymbol{dl}\times\boldsymbol{r}}{\|\boldsymbol{r}\|^3} \Leftrightarrow dV = \frac{\Gamma_n}{4\pi}\frac{dl\cdot r\cdot\sin\theta}{\|\boldsymbol{r}\|^3} \Leftrightarrow dV = \frac{\Gamma_n}{4\pi}\frac{\sin\theta\cdot dl}{\|\boldsymbol{r}\|^2}\,. \tag{2.45}$$

Now, let us change from dl to $d\theta$, so we may integrate from θ_1 to θ_2 (note that $dl = rd\theta$),

$$d\boldsymbol{V} = \frac{\Gamma_n}{4\pi}\frac{\sin\theta\cdot d\boldsymbol{l}}{\|\boldsymbol{r}\|^2} \Leftrightarrow d\boldsymbol{V} = \frac{\Gamma_n}{4\pi}\frac{\sin\theta\cdot d\theta}{\|\boldsymbol{r}\|}\,, \tag{2.46}$$

We can now integrate, keeping in mind that the normal distance from B to the vortex line $\overline{AC}$ is r_n, yielding

$$V = \frac{\Gamma_n}{4\pi\,\|r_n\|}\int_{\theta_1}^{\theta_2}\sin\theta\cdot d\theta \Leftrightarrow V = \frac{\Gamma_n}{4\pi\,\|r_n\|}(\cos\theta_1 - \cos\theta_2)\,, \tag{2.47}$$

and thus we have an expression for the magnitude of the induced velocity at B. We can also fully define the vector $\boldsymbol{V}$ as a function of $\boldsymbol{r}_{AB}$, $\boldsymbol{r}_{BC}$, $\boldsymbol{r}_{AC}$ and $\boldsymbol{r}_n$ by applying the computed V in its direction (normal to both $\boldsymbol{r}_{AB}$ and $\boldsymbol{r}_{BC}$); and expressing both $\cos\theta_1$ and $\cos\theta_2$ as sub-results of vector cross products,

$$\begin{cases} \|r_n\| = \frac{\|\boldsymbol{r}_{AB}\times\boldsymbol{r}_{BC}\|}{\|\boldsymbol{r}_{AC}\|} \\ \cos\theta_1 = \frac{\boldsymbol{r}_{AC}\cdot\boldsymbol{r}_{AB}}{\|\boldsymbol{r}_{AC}\|\|\boldsymbol{r}_{AB}\|} \\ \cos\theta_2 = \frac{\boldsymbol{r}_{AC}\cdot\boldsymbol{r}_{BC}}{\|\boldsymbol{r}_{AC}\|\|\boldsymbol{r}_{BC}\|} \\ \boldsymbol{i} = \frac{\boldsymbol{r}_{AB}\cdot\boldsymbol{r}_{BC}}{\|\boldsymbol{r}_{AB}\|\|\boldsymbol{r}_{BC}\|} \end{cases} \Rightarrow \boldsymbol{V} = \frac{\Gamma_n}{4\pi}\frac{\boldsymbol{r}_{AB}\times\boldsymbol{r}_{BC}}{\|\boldsymbol{r}_{AB}\times\boldsymbol{r}_{BC}\|^2}\left(\boldsymbol{r}_{AC}\cdot\left(\frac{\boldsymbol{r}_{AB}}{\|\boldsymbol{r}_{AB}\|} - \frac{\boldsymbol{r}_{BC}}{\|\boldsymbol{r}_{BC}\|}\right)\right) \tag{2.48}$$

where $\boldsymbol{i}$ represents the unit vector of the direction in which $\boldsymbol{V}$ is applied. This formulation allows us to quickly compute the influence of a vortex on a given control point resorting to no other points/vectors besides the ones already defined in our grid (lattice).

Lastly, we impose the boundary condition of tangency on the lifting surface we have defined. That is, the normal component of any induced velocity in regards to the surface must be cancelled by the corresponding freestream velocity in the same direction (u_n represents the normal component of the induced velocity vector),

$$u_n + V_\infty sin(\alpha) = 0\,. \tag{2.49}$$

Solving the aforementioned system in eq. (2.44), we obtain the circulation distribution for the entire surface. From here, we can obtain the force distribution across the wing by the Kutta-Joukowski theorem,

$$\boldsymbol{F}_i = \rho(\boldsymbol{V}_\infty + \boldsymbol{v}_i)\Gamma_n \times \boldsymbol{l_i}\,, \tag{2.50}$$

where v_i and l_i are, respectively, the velocity induced by the vortexes and the vector describing the bound vortex for each station.

The obtained force distribution can then be decomposed into both the lift and drag distributions across the lifting surface, which can be integrated in order to compute the total aerodynamic forces.

Because the VLM allows for chordwise discretization, it is more accurate and adequate than a LLT-

based model for the description of low aspect ratio wings (as mentioned in [32], Section 5.5.5). Neither of these two methods is, computationally speaking, very intensive. As such, so as to not limit the design space on which we may operate, a VLM scheme is employed in the developed MDAO framework.

2.4 Structures

In the previous section we established a system to compute the aerodynamic effect of the lifting surfaces, i.e., we described these in a way which allowed us to determine the generated aerodynamic force-more specifically, it allowed us to obtain the enacted forces as a load distribution, $\boldsymbol{F}$.

Now, we must define a scheme which allows us to obtain the stiffness of our structure. This way we can achieve our end-goal for the structural analysis, which is to compute the deformation the lifting surfaces undergo. This will be accomplished via Hooke's Law, formulated as

$$[K] \times \boldsymbol{u} = \boldsymbol{F} \, . \tag{2.51}$$

Now let us define a scheme so we may analyze the structural behaviour of the lifting surfaces.

To this end, a finite element method (FEM) approach is employed. At its core, what this method aims to do is break the domain (in our case, the lifting surfaces) into smaller, easier to model pieces (elements). These elements are, in general, objects for which carrying out analytical computations is significantly easier. As such, the goal of these methods is to solve the relevant equations on a local reference frame for each element and then convert these results to a global reference frame (which pertains to the entire structure). Combining the contributions from each element in this way allows us to determine the behaviour of the complete surface.

2.4.1 Base Elements

When applying finite element methods, one of the most important decisions is which element to use. In our situation we are looking to model the lifting surfaces of the aircraft, which usually present a wingbox-type structure, with a thin-walled spar and hollow interior. In the interest of simplicity, easy-to-describe geometric proprieties and of capturing this structural layout, one close element we can use to model the surfaces would be that of a beam; specifically, a circular cross-section, thin-walled beam.

Depending on what application and stresses we expect to be applied to our structure, we can define different elements; namely in terms of their degrees of freedom (DOFs) and number of nodes.

Let us now have a look at an element with two nodes, and one DOF at each node, as illustrated in fig. 2.13. This DOF will be the displacement in the x axis, meaning such an element would be used to study the axial extension/compression of its system.

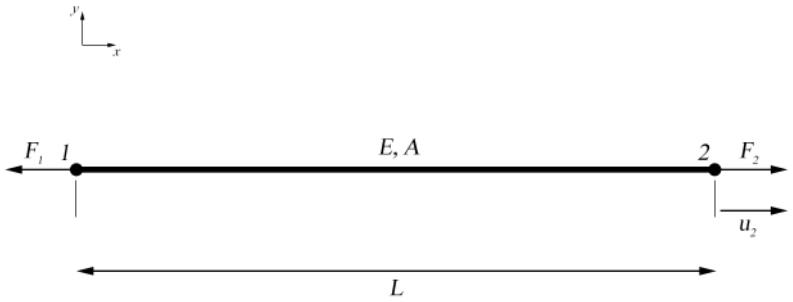


Figure 2.13: Finite element with DOFs for axial displacement analysis.

For this element, we can define the stiffness matrix

$$[k]_{axial} = \frac{EA}{L}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{2.52}$$

and respective displacement vector,

$$\boldsymbol{u} = \begin{bmatrix} u_1 & u_2 \end{bmatrix} . \tag{2.53}$$

Where E is the material's Young modulus, A is the element's cross-sectional area, L is its length, G is its shear modulus, J is the polar moment of inertia and finally, I_y and I_z are the second moments of inertia about their respective axes.

Similarly, we can define a beam element for torsion (rotation in the x axis), shown in fig. 2.14.

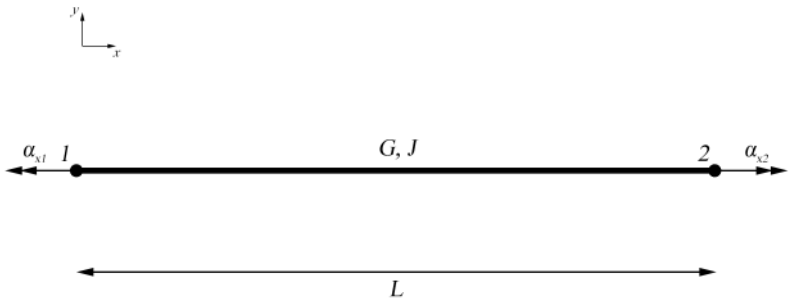


Figure 2.14: Finite element with DOFs for torsional analysis.

The stiffness matrix for this element is given by

$$[k]_{torsion} = \frac{GJ}{L}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{2.54}$$

and its displacement vector is

$$\boldsymbol{u} = \begin{bmatrix} \alpha_{x1} & \alpha_{x2} \end{bmatrix} . \tag{2.55}$$

Lastly, we can describe a pure-bending element that consists of a two node beam where we consider displacement and rotational DOFs on both nodes on a non-x direction, as shown in fig. 2.15.

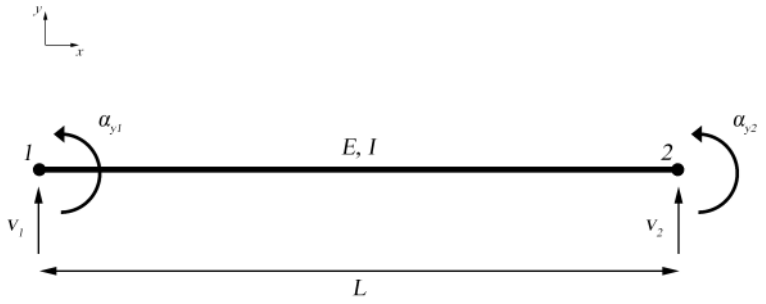


Figure 2.15: Finite element with DOFs for bending (buckling) analysis.

Such an element will have a stiffness matrix

$$[k]_{bending} = \frac{EI_z}{L}\begin{bmatrix} 12 & 6L & -12 & 6L \\ 6L & 4L^2 & -6L & 2L^2 \\ -12 & -6L & 12 & -6L \\ 6L & 2L^2 & -6L & 4L^2 \end{bmatrix} \tag{2.56}$$

and displacement vector

$$\boldsymbol{u} = \begin{bmatrix} v_1 & \alpha_{y1} & v_2 & \alpha_{y2} \end{bmatrix} . \tag{2.57}$$

In this example, DOFs have been considered in the y axis. We can now assemble the full element we will be using in the structural model.

2.4.2 Employed Element

Finally, based on the simpler elements described in the previous section, we can define a spatial beam element which encompasses all possible DOFs covered thus far. This will be accomplished by "super-imposing" two bending beam elements (one for the y and another for the z direction DOFs), one axial and one torsion element (which model the displacement and rotation in the x axis, respectively). The resulting element is depicted in fig. 2.16.

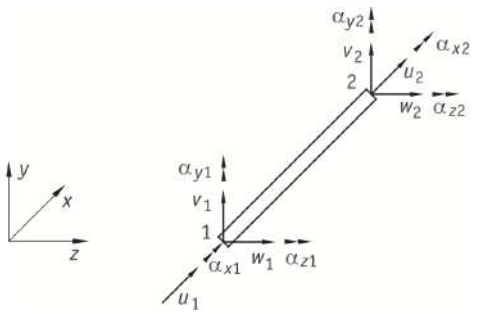


Figure 2.16: 12-DOF beam element.

To obtain the stiffness matrix of the element in fig. 2.16 we only need to combine the stiffness matrices of the simpler elements. Assuming a displacement/rotation vector given by

$$\boldsymbol{u} = \begin{bmatrix} u_1 & v_1 & w_1 & \alpha_{x1} & \alpha_{y1} & \alpha_{z1} & u_2 & v_2 & w_2 & \alpha_{x2} & \alpha_{y2} & \alpha_{z2} \end{bmatrix}, \tag{2.58}$$

The corresponding stiffness matrix can be assembled as

$$[k]_e = \begin{bmatrix} \frac{EA}{L} & 0 & 0 & 0 & 0 & 0 & -\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\ & \frac{12EI_z}{L^3} & 0 & 0 & 0 & \frac{6EI_z}{L^2} & 0 & \frac{-12EI_z}{L^2} & 0 & 0 & 0 & \frac{6EI_z}{L^2} \\ & & \frac{12EI_y}{L^3} & 0 & \frac{-6EI_y}{L^2} & 0 & 0 & 0 & \frac{-12EI_y}{L^3} & 0 & \frac{-6EI_y}{L^2} & 0 \\ & & & \frac{GJ}{L} & 0 & 0 & 0 & 0 & 0 & -\frac{GJ}{L} & 0 & 0 \\ & & & & \frac{4EI_y}{L} & 0 & 0 & 0 & \frac{6EI_y}{L^2} & 0 & \frac{2EI_y}{L} & 0 \\ & & & & & \frac{4EI_z}{L} & 0 & \frac{-6EI_z}{L^2} & 0 & 0 & 0 & \frac{2EI_z}{L} \\ & & & & & & \frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & \frac{12EI_z}{L^3} & 0 & 0 & 0 & \frac{-6EI_z}{L^2} \\ & & & \text{Symmetric} & & & & & \frac{12EI_y}{L^3} & 0 & \frac{6EI_y}{L^2} & 0 \\ & & & & & & & & & \frac{GJ}{L} & 0 & 0 \\ & & & & & & & & & & \frac{4EI_y}{L} & 0 \\ & & & & & & & & & & & \frac{4EI_z}{L} \end{bmatrix}. \tag{2.59}$$

This is a fairly standard element (and respective matrix) used in structural analysis. Because of the DOF-order in which we defined our displacement vector, the first six rows pertain to the DOFs of the first node, and the remaining to the DOFs of the second node (as can be seen in the anti-symmetry of both these blocks). The components of each of the aforementioned elements can be seen in: axial (rows 1 and 7), bending (2, 3, 5, 6, 8, 9, 11 and 12) and torsional (4 and 10).

In essence, our approach will consist of determining the element's properties at a local level (i.e. on a local reference frame) and then converting these to a global frame, obtaining the global stiffness matrix $[K]$ for the entire structure. To this end, a matrix such as in eq. (2.59) is assembled for each element, and these are then combined to form a three-dimensional matrix (with dimensions $[n_y \times 12 \times 12]$, where n_y is the number of finite element nodes we impose) which encompasses the entire structure.

As discussed in the previous section, the aerodynamic analysis provided us with a set of loads, $\boldsymbol{f}$ applied to the lifting surfaces. Now that we have also defined the structure's stiffness, we may solve the aforementioned linear system in equation eq. (2.51), which yields $\boldsymbol{u}[1 \times 12]$. This vector is the displacement and rotation vector enacting on the structure. With this, we can now iterate upon the architecture, computing the new set of loads applied to the updated mesh.

2.5 Propulsion

The need for a propulsion force to be introduced into the existing code structure arises from one of the end goals of this work: performing trajectory control optimization.

In its original form, the MDAO framework estimates the fuel burnt in a given route by means of the

Breguet range equation,

$$R = \frac{V}{g}\left(\frac{L}{D}\right)\frac{1}{TSFC}\ln\frac{W_i}{W_f}\,, \tag{2.60}$$

which approximates this quantity based on the cruise conditions and aircraft specifications, for a certain range (R).

Because we are looking to implement trajectory control optimization functionalities, the thrust force variable becomes a very pertinent component of the system, as it only matches the drag when the aircraft is in cruise (see eq. (2.25b)), assuming we are also disregarding α. As such, handling the propulsive force "by omission" (as per eq. (2.60), which assumes $T = D$) becomes inadequate. Therefore, a propulsion system which allows us to explicitly obtain the thrust from a throttling variable independent from the other systems will be required.

Let us have a look at the most commonly employed propulsion solutions in commercial and general aviation. Before we do, however, let us define an important metric: the propulsive efficiency. This can be defined as [35],

$$\eta_p = \frac{\text{Thrust power applied to the aircraft}}{\text{Net mechanical power applied to the engine's air flow}}\,. \tag{2.61}$$

If we now define V_0 and V_e as the air flow speed at the inlet and exhaust, respectively, and assume a constant mass flow $\dot{m}$ throughout the engine, eq. (2.61) becomes

$$\eta_p = \frac{\dot{m}(V_e - V_0)V_0}{\dot{m}\left(\frac{V_e^2}{2} - \frac{V_0^2}{2}\right)} \Leftrightarrow \eta_p = \frac{2V_0}{V_e + V_0}\,. \tag{2.62}$$

Finally, since we can define the flow velocity at the exhaust as $V_0 + \Delta V$ (where ΔV is the speed increment enacted upon the flow by the engine) we arrive at

$$\eta_p = \frac{V_0}{V_0 + \frac{\Delta V}{2}}\,, \tag{2.63}$$

which demonstrates one crucial principle: when designing an engine, we should aim to achieve our thrust by enacting the smallest ΔV possible, so as to have an higher propulsive efficiency. In more practical terms, we should aim to accelerate large mass flows by a small amount, rather than the other way around.

2.5.1 Aircraft Propulsion

Jet Engines

The basic premise behind all jet engines is the same: draw in air, compress it and perform combustion (in order to increase its temperature), and finally expand the exhaust gases (through turbines and/or nozzles) so as to extract work. In essence, a jet engine performs the same sequence as a gas turbine (i.e. the Brayton cycle, as illustrated in fig. 2.17) and both topics are usually covered in tandem [35].

Thrust is therefore generated due to two effects: the difference in momentum between the flow at the exhaust and admission; and the difference in pressure between the far-field and the exhaust nozzle.

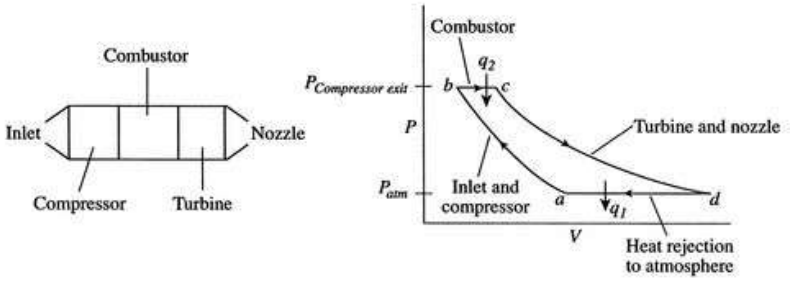
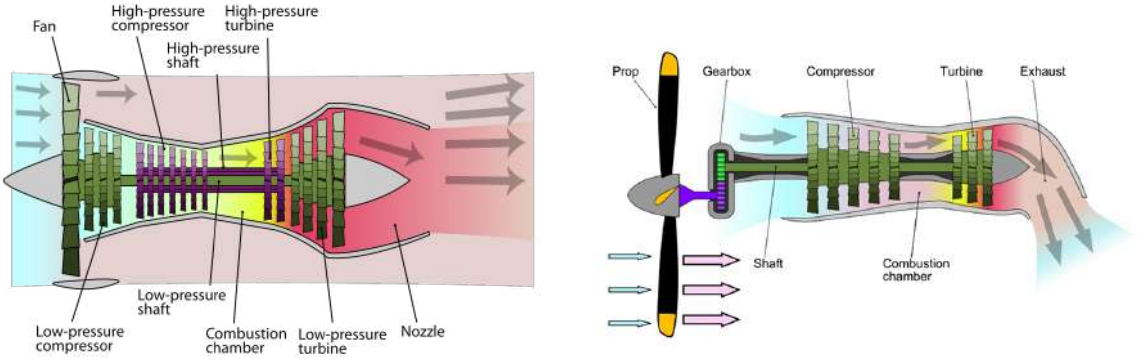


Figure 2.17: Simplified diagram for a generic jet engine (left) and $P - V$ plot of the Brayton cycle (right).

In subsonic conditions[2], there are three main jet engine designs which cover most applications: the turbojet, the turbofan and the turboshaft (similar to the turboprop).



(a) Turbofan.

(b) Turboprop.

Figure 2.18: Section view of two jet engines.

A turbojet engine is arguably the simplest iteration of the jet engine, as it is fundamentally similar to the engine depicted in fig. 2.17 (left): the flow is compressed and combustion occurs in the chamber(s) downstream. The gas then expands and powers the turbine, which in turn powers the compressor upstream (via the represented shaft).

In a turbofan engine, illustrated in fig. 2.18a, we have an additional low-pressure stage (both in the compressor and turbine ends), whose main purpose is to power the namesake fan at the very beginning of the engine. This design introduces a very important parameter, the bypass ratio, which is the ratio between the airflow around the generator proper and the airflow fed into the compressor. In this way, the central jet (duct) operates at a lower speed, meaning a greater propulsive efficiency (due to what was discussed regarding eq. (2.63)) and lower noise level when compared to a turbojet solution. Larger bypass ratios also lead to larger (heavier) engines. [35].

The turboshaft designation describes an engine design in which the exhaust gases (downstream from the combustion chamber(s)) are expanded to power and additional element (shaft) besides the turbine used to work the compression stage. When the shaft is employed in order to power a propeller

[2]The most notable omission from this list would the ramjet engine. Since this configuration is mostly unfeasible for subsonic flight conditions [35], it will not be covered.

(via gearbox), the engine is considered a turboprop (see fig. 2.18b). These engines (in aeronautical applications) are most frequently utilized to power helicopter rotors or propellers in general aviation aircraft and, in both cases, the system is limited to the occurrence of supersonic speeds at the blade tips. As such, aircraft outfitted with turboshaft engines are limited to operating Mach numbers around $0.5 - 0.6$, but operate at high efficiency in that range [35].

Piston Engines

Besides the turboprop engines described in the previous section, another common solution in general aviation aircraft would be to employ an engine not dissimilar from what is commonly found in cars: a piston engine. These engines (also referred to as reciprocal engines) are coupled to a propeller and present a trade-off between high efficiency and low maximum power output when compared to turboprop engines. This alone makes these engines unsuitable for any situation besides small general aviation aircraft, where higher thrust (and, therefore, power output) values are required [36].

Figure 2.19: Piston engine (Lycoming Thunderbolt).

Figure 2.20: Electric motor prototype (Siemens [37]).

Electric/Hybrid Propulsion

Another approach to aircraft propulsion is via electric motors (once again, coupled to a propeller) powered by batteries or fuel-cells. The main design limitation pertaining these configurations lies in the low energy density of the currently available batteries when compared to conventional fuel used in internal combustion engines [5]. These solutions are found in two situations: small UAVs, and recent general aviation designs. In UAVs, fully electrical solutions are common since an electric motor is significantly simpler and lighter than an internal combustion engine [38]. As for small passenger aircraft, there are several possible architectures as described in [5], which differ in how the powertrain is assembled.

The main advantages of electric motors when compared to normal combustion ones are the amount of noise and overall efficiency of the powertrain (since turbine-based engines dissipate around $30\% - 50\%$ in heat during the Brayton cycle alone). The differences in size and number of parts is also visible comparing fig. 2.19 and fig. 2.20.

It should also be noted that eq. (2.60) does not apply for an aircraft with an electric propulsion system.

This is due to the fact the aircraft's mass remains constant throughout its flight, since batteries do not lose mass. As such, following a derivation similar to that involved in obtaining eq. (2.60) yields

$$E_{\text{bat}} = P_{\text{bat}} t \Leftrightarrow E_{\text{bat}} = \frac{P_{\text{thrust}}}{\eta_{\text{powertrain}}} t \Leftrightarrow E_{\text{bat}} = \frac{DV}{\eta_{\text{powertrain}}} t \Leftrightarrow E_{\text{bat}} = \frac{DR}{\eta_{\text{powertrain}}} \,, \tag{2.64}$$

where P_{thrust} is the power required to generate the necessary thrust and $\eta_{\text{powertrain}}$ is the efficiency of the entire propulsive system (from stored energy to kinetic energy). From here, assuming once again $L = W = m_{\text{total}} \times g$ (with m_{total} being the aircraft's total mass and g being the gravitational acceleration), and expressing the total energy contained in the battery packs as $E_{\text{bat}} = e_{\text{bat}} \times m_{\text{bat}}$, where e_{bat} is defined as the specific energy (energy/mass) of the packs and m_{bat} is the batteries' mass, we finally obtain the range equation for electric battery aircraft,

$$R = e_{bat} \frac{\eta_{\text{powertrain}}}{g} \frac{L}{D} \frac{m_{\text{bat}}}{m_{\text{total}}} \,. \tag{2.65}$$

2.5.2 Propulsive Model

As seen in the previous sections, there is no shortage of engine design solutions; the problem arises in the parameterization of said engines. Besides the aforementioned bypass factor, several other parameters are crucial in the formal definition of a jet engine, such as the number of compression stages (and respective compression ratios), duct dimensions and expansion stages, and that is assuming a jet engine solution from the start.

Also, our aim is to employ a generic model, capable of estimating thrust from generic parameters rather than committing to an engine type. As such, let us now look at an alternative propulsive model.

At its simplest form, the thrust can be computed solely by means of a throttle coefficient, δ_T, as

$$T = \delta_T \times T_{max}. \tag{2.66}$$

A solution such as this one, however, leaves the propulsion system totally independent from the remaining disciplines, providing results which rely solely on the engine parameters defined at the start.

In order to develop the system considering more parameters, a general model such as the one described in Stengel [39] is adopted. In doing this, we begin by breaking the thrust force into its dimensionless and reference factors,

$$T = C_T(V, \delta_T) \frac{S\rho^2}{2} \,, \tag{2.67}$$

and describe the dimensionless factor C_T as a function of speed, V, and the throttle coefficient, δ_T, as well as three empirical coefficients. The factor is defined as

$$C_T = (k_0 + k_1 V^\eta)\delta_T \,, \tag{2.68}$$

where k_0 is the maximum-throttle thrust coefficient at $V = 0$, k_1 is a measure of the speed weighting,

and η is a parameter used to describe the engine type. These three coefficients will also be important in order to define the relevant derivatives for the C_T factor.

As will be discussed in the following section and was mentioned in section 2.3, only fully-developed, steady points in time will be analyzed. As such, we may define the relevant derivatives in relation to a nominal point as [39]

$$\frac{\partial C_T}{\partial V} = \eta k_1 V_0^{\eta-1} \delta_{T_0} \tag{2.69a}$$

and

$$\frac{\partial C_T}{\partial \delta_T} = \left(k_0 + k_1 V^{\eta}\right). \tag{2.69b}$$

By reintroducing eq. (2.68) into eq. (2.69a), we can obtain the alternative formulation,

$$\frac{\partial C_T}{\partial V} = \frac{\eta C_T}{V_0}\left(1 - \frac{k_0 \delta_{T_0}}{C_T}\right). \tag{2.70}$$

For the three previous equations, (V_0, δ_{T_0}) correspond to the nominal conditions under which C_T is computed.

The values (or range) of the empiric parameters themselves will be discussed in the implementation of the propulsive system, in section 3.5.

With this approach, since we now know the explicit thrust values exerted by the aircraft's propulsive system, computing the fuel burn rate for a given route can be done through the thrust-specific fuel consumption ($TSFC$),

$$\dot{m}_{fuel} = TSFC \times T \tag{2.71}$$

which can be done for all considered flight stages, adding up to the total amount of fuel spent.

2.6 Trajectory

2.6.1 General Formulation

The trajectory optimization problem can be described (in its most generic form) as the minimization of an objective function composed by a boundary term, J, and a integrand w. Such problem takes the form [40]

$$\min_{t_0, t_f, \boldsymbol{x}(t), \boldsymbol{c}(t)} J\left(t_0, t_f, \boldsymbol{x}(t_0), \boldsymbol{x}(t_f)\right) + \int_{t_0}^{t_f} w\left(\tau, \boldsymbol{x}(\tau), \boldsymbol{c}(\tau)\right) d\tau\,, \tag{2.72}$$

where t_0 and t_f represent two given times, while the vectors $\boldsymbol{x}(t)$ and $\boldsymbol{c}(t)$ give us the state and control, respectively.

In addition to eq. (2.72), boundary conditions need to defined in order to fully describe the optimization problem. These are:

System dynamics, which describe the system's response, and how it changes in time; in the form

$$\dot{\boldsymbol{x}} = \boldsymbol{f}\left(t, \boldsymbol{x}(t), \boldsymbol{c}(t)\right) \; ; \tag{2.73a}$$

Path constraints, which define restrictions along the trajectory (i.e., in any of the mid-way points of the trajectory); in the form

$$\boldsymbol{h}\left(t, \boldsymbol{x}(t), \boldsymbol{u}(t)\right) \leq 0 \; ; \tag{2.73b}$$

Boundary constraints, which specifically restrict the final and initial state conditions of the system; in the form

$$\boldsymbol{g}\left(t_0, t_f, \boldsymbol{x}(t_0), \boldsymbol{x}(t_f)\right) \leq 0 \, . \tag{2.73c}$$

Which specifically restrict the final and initial state conditions of the system.

Besides these impositions on the generic functions h and g, constraints may be applied directly to the state and control of the system, both at the initial/final times in specific, as well as on the remaining knot points:

Path state constraints:

$$\boldsymbol{x_{min}} < \boldsymbol{x}(t) < \boldsymbol{x_{max}} \tag{2.73d}$$

Path control constraints:

$$\boldsymbol{u_{min}} < \boldsymbol{u}(t) < \boldsymbol{u_{max}} \tag{2.73f}$$

Initial state constraints:

$$\boldsymbol{x_{0,min}} < \boldsymbol{x}(t_0) < \boldsymbol{x_{0,max}} \tag{2.73e}$$

Final state constraints:

$$\boldsymbol{x_{f,min}} < \boldsymbol{x}(t_f) < \boldsymbol{x_{f,max}} \tag{2.73g}$$

Lastly, we may also constraint the time elapsed by the system in performing its function, as

$$t_{low} \leq t_0 < t_f \leq t_{max} \, . \tag{2.73h}$$

Another important step in setting up the trajectory discipline is considering the physical domain in which the problem will be solved. Starting once again from the most generic formulation, our trajectory would be a three-dimensional path from point A to B, as depicted in fig. 2.21.

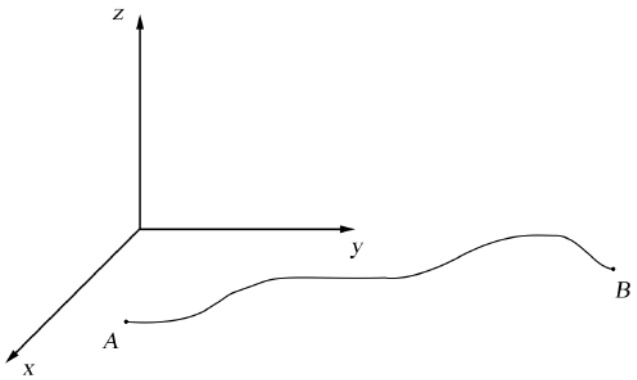


Figure 2.21: 3D trajectory from start to finish.

The first step towards a system more alike the one we will be using is to take into account what was discussed in section 2.2; namely that our model will only consider longitudinal motion. Therefore, we would be left with trajectories such as that exemplified in fig. 2.22.

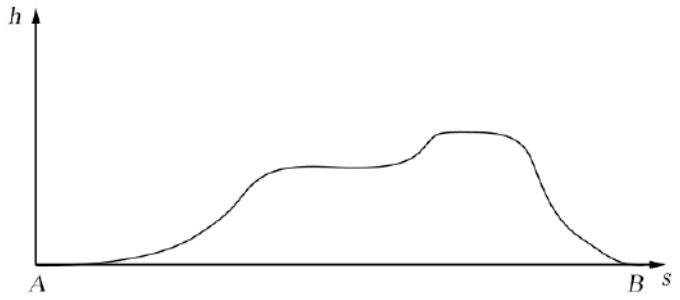


Figure 2.22: 2D trajectory from start to finish

Let us now discuss the final step from a trajectory such as fig. 2.22 to one workable with the selected approach.

2.6.2 Average/Sectioned Mission Profile

The adopted approach will not followed the mold defined by eq. (2.72), as no integration will be performed. Instead, the mission profile will be broken down into a prescribed number of points (multipoint approach). As such, only fully-developed situations will be considered in each of the points of our analysis. This way, a trajectory similar to that of fig. 2.22 could be broken down into five phases as shown in fig. 2.24.

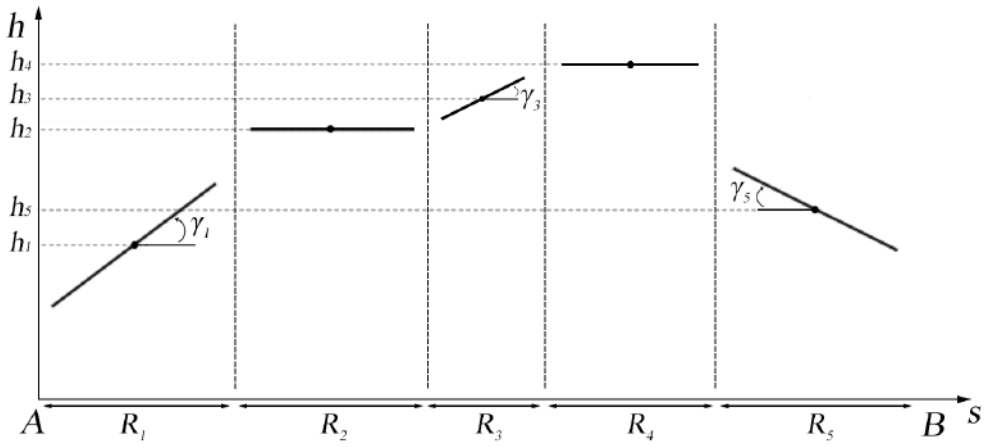


Figure 2.23: Steady multipoint trajectory evaluated at 5 points.

Much like in section 2.5, our goal is to reduce every aspect of the problem into simple, actionable parameters. As seen in fig. 2.24, each phase of the mission profile is defined by three clear parameters: its angle of ascent γ_i, its height h_i, and the ground distance covered R_i. It is, in fact, this description that defines our generic trajectory point from a design standpoint.

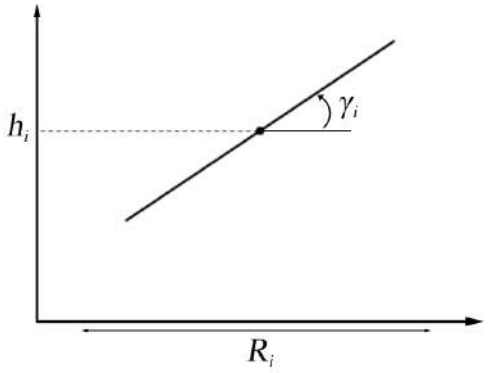


Figure 2.24: Generic trajectory point.

Different constraints may and will be applied to the different points in order to define the trajectory phases we want to optimize upon. In essence, for each of the trajectories, we optimize for a cruise situation, and then optimize the control variables to other conditions representative of different flight phases.

Chapter 3

Numerical Discipline Models

In this chapter we will take a closer look at how the models/approaches described in chapter 2 will be implemented in the MDAO software framework. For the analysis of each discipline, let us define which (design and/or coupled) variables it is receiving and which relevant outputs it produces. It is also important to recall how the evaluation of constraints, the objective function and other performance metrics is handled from a data flow perspective. Because most of these functions depend on outputs from more than one discipline, their computation only occurs after the MDA solver has converged (as explained in section 2.1.2).

The simplified diagrams presented in this chapter cover the inputs and outputs for each analysis, differentiating between outputs necessary for the convergence of the MDA system and those utilized in performance and constraint computations.

3.1 Mesh and Lifting Surfaces

Before delving into the actual disciplines, let us discuss the starting point. In this work, the only structures of interest will be the lifting surfaces, namely the wing and the horizontal stabilizer of the aircraft. Both of these surfaces are considered to be symmetric on all performed computations.

To start the computations, we must first define initial structural configurations for both of these components, and grids upon which to perform all the required analyses. This step in defining our approach is instrumental since it will greatly affect the load and displacement transfer scheme, i.e., how the loads are transferred from the VLM mesh upon which they are computed onto the finite element structural model. This scheme will be discussed in greater detail in section 3.4.

Some variables (which we will also consider as design variables in the optimization process) will be relevant in defining the geometry of the lifting surfaces: span, and twist distribution. In the tail, the twist distribution of the horizontal stabilizer will represent something different. Because the incidence of the incoming flow on the horizontal stabilizer is an important factor so that the aircraft can be trimmed, its twist distribution will be used to model this parameter. As such, it will be allowed to change between flight phases, acting as a control variable (see eq. (2.26b)).

The generated mesh will also depend on how fine we wish it to be: a set of variables ('num_x', 'num_y') is specified for each the lifting surfaces. The y-axis represents the spanwise direction, while the x-axis represents the chordwise direction. It is possible to specify alternative meshes but, in this work, we will stick to a conventional uniform grid point distribution.

Figure 3.1 synthesizes the inputs and outputs of the mesh generation process.

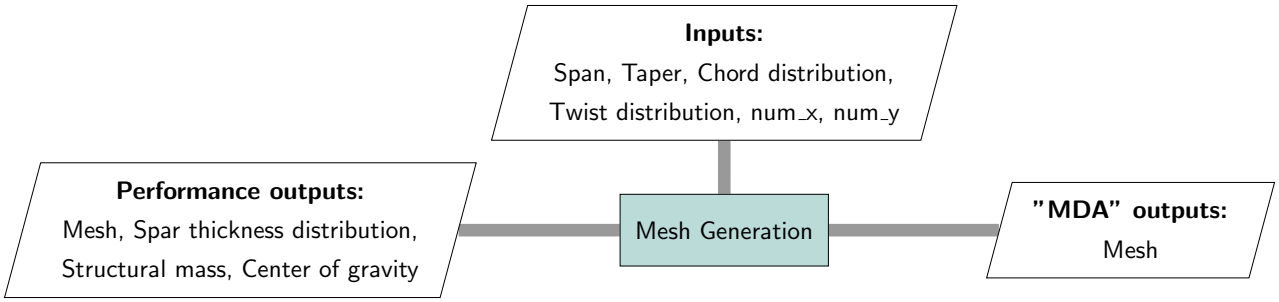
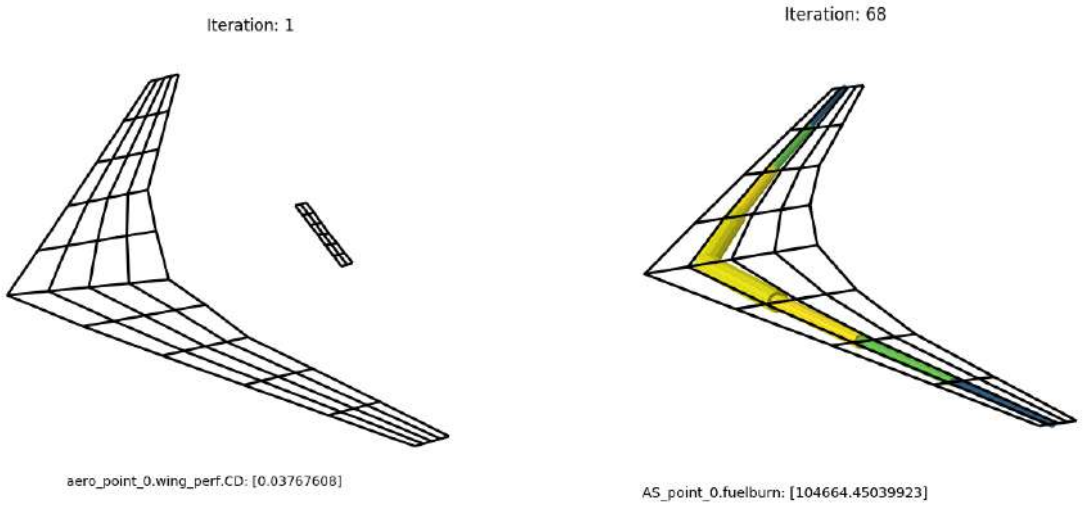


Figure 3.1: Inputs and outputs of the mesh generation.

This process is, in computational terms, located outside the MDA loop previously illustrated in the XDSM (see fig. 2.2). Still referring to the same figure, the mesh generation process essentially takes place in sequential step (1), meaning the base mesh is updated by the optimizer before each MDA loop. It is upon this base mesh that displacement and loads will be iterated in steps $(2) \rightarrow (5)$.

Regarding the finite element structure, due to how the fluid/structure computations will be approached, we will have a number of elements equal to the number of spanwise stations defined for the VLM grid (i.e. 'num_y'). Simple examples of these grids can be seen in fig. 3.2: in fig. 3.2a we see the output a simple aerodynamic-only analysis (VLM grid alone); and in fig. 3.2b this grid is overlapped with the FE structure.



(a) VLM grid for wing and horizontal stabilizer.

(b) VLM grid and FE structure for wing-only configuration.

Figure 3.2: Examples of VLM grids and a finite element structure.

In fig. 3.2a we can see a good example of the base shapes we will be employing to model the lifting surfaces in this work. For the tail, we will be using a simple rectangular shape. As for the wing, we will be adopting a version of a NASA Common Research Model (CRM) [41]. This model has been developed to perform at transonic speeds, specifically $M_\infty = 0.85$, which is slightly out of the range we established in section 2.3, meaning some error will be incurred as a result of our subsonic method.

We now present a simple rundown of how the information is processed as we proceed through the various disciplines.

3.2 Aerodynamics

As shown in section 2.3.4, we will be assembling a vortex-lattice in order to perform all aerodynamic computations. With this in mind, fig. 3.3 illustrates the necessary inputs to perform the analysis, and respective outputs.

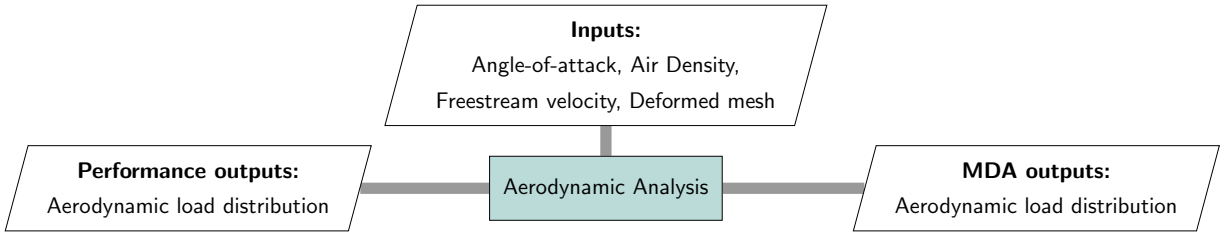


Figure 3.3: Inputs and outputs of the aerodynamic analysis.

The aerodynamic analysis is the first one to be carried out within the MDA iterative process. Upon initiating the optimization problem the main input of this discipline is the initial mesh, as described in section 3.1. The same data is output for both performance evaluations and the MDA process (i.e., this distribution will be sent to the following discipline analyses), however, it is handled very differently, as it will be seen in sections 3.3 and 3.4.

3.3 Structures

As mentioned previously, several structural variables will play an important part in defining the lifting surfaces.

As we have already defined the finite element mesh, let us synthesize the data flow through this discipline.

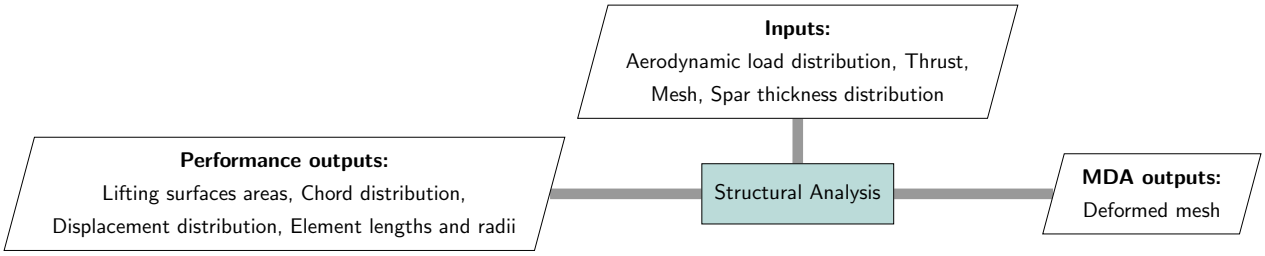


Figure 3.4: Inputs and outputs of the structural analysis.

The structural analysis, as depicted in fig. 3.4, receives loads from both the aerodynamic and propulsion analyses. After combining both inputs into a single load array, $\boldsymbol{F}$, the displacement computations are carried out as described previously (see eq. (2.51)).

We can also see that the outputs are vastly different for MDA and performance evaluation purposes. In terms of converging the MDA solver, the only necessary information is the deformed mesh, as this is the direct, practical result of the analysis which affects the remaining disciplines. However, in order to evaluate the constraints associated with the structure, we can see that a greater level of detail is necessary.

3.4 Load and Displacement Transfer

One important aspect to consider when dealing with more than one mesh within an MDO framework is how information is transferred between them. In our case, we need to apply the load distribution computed upon the aerodynamic mesh to the finite element structure. The implemented transfer scheme must satisfy two requirements: it must be consistent and conservative [10, 42].

In order for the load and displacement transfer method to be consistent, the sum of all the loads and moments applied on the nodes of the structure must be the same as the force and moment resultant of the integration of the computed aerodynamic loads. This means an infinite amount of transfer methods provide consistency, since we are integrating a continuous distribution over a a smooth region. Then, so that the scheme is conservative, we need to apply the principle of virtual work. Namely, the virtual work done by the forces on both meshes over the virtual displacements the scheme predicts must be the same.

Let us demonstrate the process for transferring the load from a single aerodynamic mesh panel to the finite element structure.

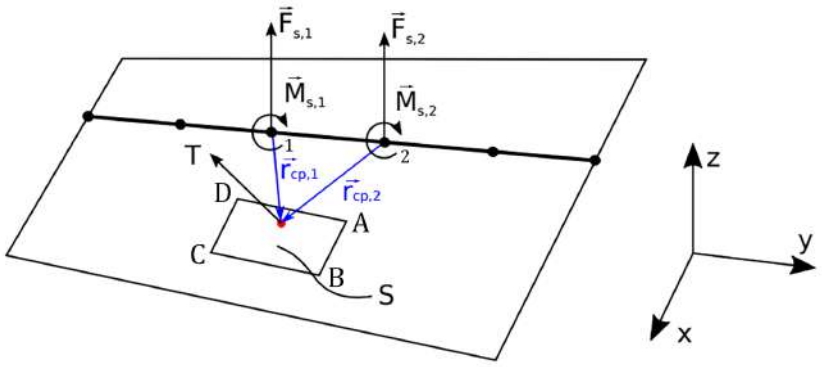


Figure 3.5: Scheme for transferring aerodynamic load $\vec{T}$ to structural nodes 1 and 2 [10].

Figure 3.5 illustrates an aerodynamical panel $ABCD$, the distributed load $\vec{T}$, structural nodes 1 and 2, the position vectors from the panel's center of pressure (cp) to those same nodes, and the force and moment pairs that have been transferred onto them. The subscript s indicates which quantities are tied with the structural mesh.

As mentioned in section 3.1, segments $\overline{AB}$ and $\overline{CD}$ line up with structural nodes 1 and 2 in the chordwise direction (x). Because $\vec{T}$ is being applied at the middle of the panel in the spanwise direction (y), defining the forces and moments at the nodes as

$$\begin{cases} \vec{F}_{s,i} = \int_{\text{ABCD}} \frac{1}{2}\vec{T}dS = \frac{1}{2}\vec{T}S \\ \vec{M}_{s,i} = \int_{\text{ABCD}} \vec{r}_i \times \frac{1}{2}\vec{T}dS = \frac{1}{2}\vec{r}_{cp,i}\vec{T}S \end{cases} \tag{3.1}$$

means the scheme is conservative since we are equating the transferred forces and moments to the resultant of the aerodynamic load distribution integrated over the panel surface. Because the center of pressure is equidistant from both nodes, the loads are factored by $\frac{1}{2}$.

Now that we have established the loads to be applied on the structural mesh, we need to define how the ensuing structural displacements are transferred to the aerodynamic mesh. Let us designate the structural displacement on node i as $\vec{u}_{s,i}$ and the transferred deformation on the aerodynamic panel as $\vec{u}_p$. The transfer scheme is defined as

$$\vec{u}_p = \frac{1}{2}\sum_{i=1}^{2}\left(\vec{d}_{s,i} + \vec{\theta}_{s,i} \times \vec{r}_i\right) , \tag{3.2}$$

where $\vec{d}_{s,i}$ and $\vec{\theta}_{s,i}$ are the translational and rotational components of the structural displacement, respectively, and $\vec{r}_i$ points from node i to where $\vec{u}_p$ is being evaluated. Now that the displacement transferal has been defined, we need to verify that the virtual work performed by $\vec{T}$ over $\vec{u}_p$ matches that of $\left(\vec{F}_{s,i}, \vec{M}_{s,i}\right)$ over $\vec{u}_{s,i}$, proving consistency. The virtual work effected on the structural mesh, δW_s, is given by

$$\delta W_s = \sum_{i=1}^{2}\left(\vec{F}_{s,i} \cdot \vec{d}_{s,i} + \vec{M}_{s,i} \cdot \vec{\theta}_{s,i}\right) , \tag{3.3}$$

where we can now introduce the load transfer (eq. (3.1)), resulting

$$\delta W_s = \frac{1}{2}\sum_{i=1}^{2}\left(\vec{T}\cdot\vec{d}_{s,i} + \vec{r}_{cp,i}\times\vec{T}\cdot\vec{\theta}_{s,i}\right) . \tag{3.4}$$

Now, to make ends meet, let us compute the virtual work effected on the aerodynamic mesh, δW_a, starting with

$$\delta W_a = \int \vec{T}\cdot\vec{u}_p dS , \tag{3.5}$$

where we can now insert the displacement transfer (eq. (3.2)), yielding

$$\delta W_a = \frac{1}{2}\sum_{i=1}^{2}\int\left(\vec{T}\cdot\vec{d}_{s,i} + \vec{T}\cdot\vec{\theta}_{s,i}\times\vec{r}_i\right) . \tag{3.6}$$

From where we may integrate as was done in eq. (3.1), making it so we are evaluating the displacement at the center of pressure (this is valid since $\vec{T}$, $\vec{d}_{s,i}$ and $\vec{\theta}_{s,i}$ are all constant over the panel). Doing so, we obtain

$$\delta W_a = \frac{1}{2}\sum_{i=1}^{2}\int\left(\vec{T}\cdot\vec{d}_{s,i} + \vec{T}\cdot\vec{\theta}_{s,i}\times\vec{r}_{cp,i}\right) , \tag{3.7}$$

which is algebraically equivalent to

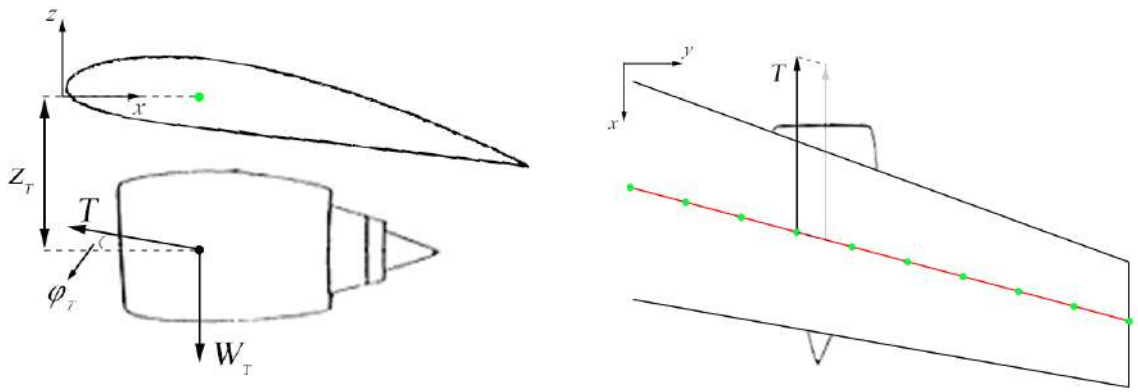
$$\delta W_a = \frac{1}{2}\sum_{i=1}^{2}\left(\vec{T}\cdot\vec{d}_{s,i} + \vec{r}_{cp,i}\times\vec{T}\cdot\vec{\theta}_{s,i}\right) . \tag{3.8}$$

Comparing eqs. (3.4) and (3.8), we can see that the virtual work applied on the structural mesh matches that which is applied on the aerodynamic mesh, confirming this transfer scheme (eqs. (3.1) and (3.2)) is consistent.

3.5 Propulsion

Stripping it down to the fundamental, we are simply looking to apply a force (the engines' thrust) at its designated location. To this end, we will require an estimation of where that point of application will be; as well as the magnitude and orientation of the thrust vector.

Let us start by discussing where it will be applied and in which direction.



(a) Side view. **(b)** Top view.

Figure 3.6: Application of the propulsive force.

Following the diagrams x, y, z defined in fig. 3.6, several details regarding the application of the thrust force can be defined:

- The force is parallel to the incoming flow, meaning only the aforementioned ϕ_T needs to be defined (i.e. there will be no component in the y-direction). For the sake of simplicity, this angle will be considered null in this work: $(\phi_T = 0)$;
- As a simplification, it may be considered that the force will be applied on a point whose spanwise coordinate matches that of one of the nodes of the mesh (denoted in green). Doing so simplifies the determination of the equivalent force-moment system upon transferring the force to the structure proper;
- When applying the engine force, a vertical component must be added to represent the weight of the engine. It is, therefore, assumed that both weight and thrust are applied at the same location;
- The vertical distance at which the force will be applied w.r.t. the wing (z_t) will be defined as half of the engine's diameter.

At this point, the conditions under which the thrust will be applied (i.e., the point of application and its orientation) have been fully defined. Let us now discuss its magnitude.

As introduced in section 2.5, the thrust coefficient will be computed via a model such as eq. (2.68). The three empirical factors, (k_0, k_1 and η) define the type of engine we wish to emulate. Referring once again to Stengel [39], the following assumptions will be made

$$\begin{cases} k_0 = 0 \\ \eta = -3 \end{cases} . \tag{3.9}$$

For all mentioned idealized models k_0 is considered to be 0. Furthermore, setting this value for η describes a constant power setting, representative of high propulsive efficiency systems such as high-bypass turbofans and turboprop coupled to a propeller (see eq. (2.63)).

We are left with the k_1 parameter, which will be used to fit our model to specific engines. As a default setting, we will be using values from a specific aircraft: the *B777-300*. As such, for purposes of defining k_1, the values from the *GE90* turbofan will be used - an engine commonly paired with this aircraft.

Finally, we may once again sum up the data flow for this analysis, illustrated in fig. 3.7:

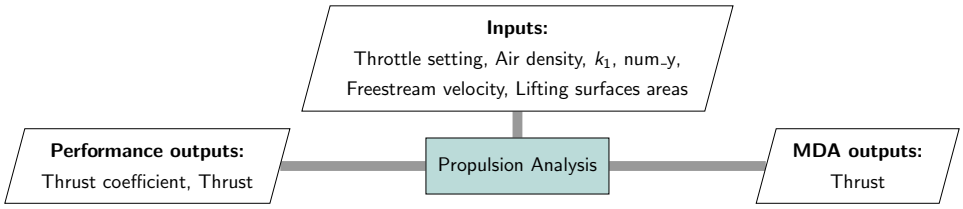


Figure 3.7: Inputs and outputs of the propulsion analysis.

3.6 Trajectory

The trajectory implementation is unlike that of the remaining disciplines. As covered in section 2.6, what our pseudo-multipoint analysis aims to accomplish is to perform coupled aircraft design in a cruise setting and optimal control across several separated sets of constraints. Each one of these sets represents a point in our mission profile, as the flight conditions for that phase are essentially imposed by these constraints. Because of this, it does not make sense to define constraints in the way we have defined them in the previous sections of this chapter; instead, let us present the mission profiles we will cover in this work, and the type of constraints associated to each flight phase.

3.6.1 First Mission Profile - Cruise

Let us start by defining the base line mission profile, that of a cruise condition situation, illustrated in fig. 3.8, where we plot the altitude, h, against the distance covered, s.

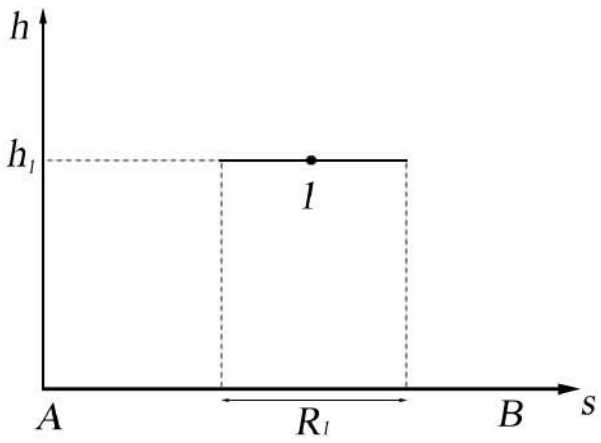


Figure 3.8: Simple cruise mission profile.

For cruise points, we impose that

$$\gamma = 0\,. \tag{3.10}$$

This is the baseline case, which serves as validation since the Breguet range equation (see eq. (2.60)) is valid and adequate for this situation. In fig. 3.8, R_i describes the distance covered in the i-th flight phase, and h_i the altitude at which computations for said phase are carried out. In this single-point situation, the phase distance matches the entire mission range.

3.6.2 Second Mission Profile - Cruise and Descent

Next, we have a two point profile with two of the main flight phases of a commercial mission profile, as depicted in fig. 3.9.

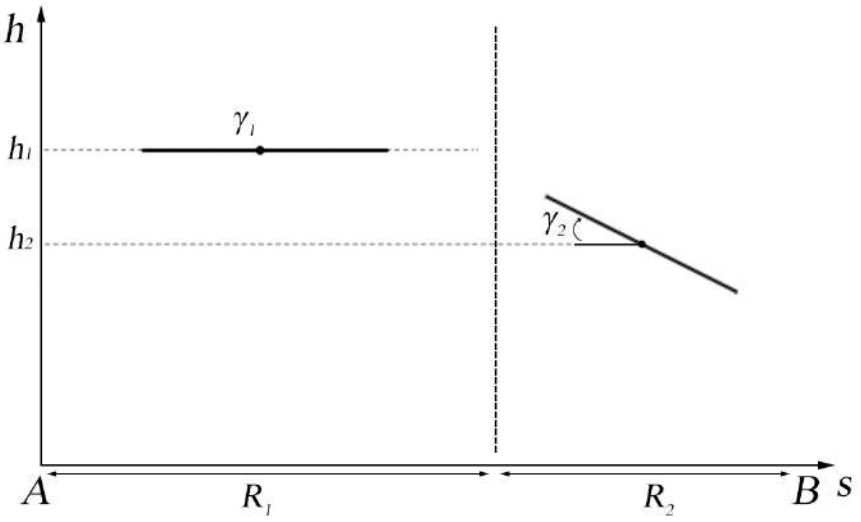


Figure 3.9: Two point mission profile with cruise and descent.

This profile comprises all the main features of our approach, and serves as the final case study of this work. For the point of descent, two aspects need to be considered: the altitude, h_i, and the value of the flight path angle, γ_i. This phase covers a large span of altitudes but, as an approximation, it will be considered that the altitude for this point is given by the average of their origin and target altitudes. As such, h_2 in this case is given by

$$h_2 = \frac{h_1}{2}\,, \tag{3.11}$$

where h_1 is the predefined cruise altitude, based on usual values observed for the case in point.

As for the flight path angle γ_i, the variable will be constrained to an adequate range of values for the designated flight phase. Because we are looking to emulate commercial airline missions performed by aircraft such as the aforementioned *B777-300*, we will be adopting the industry standard for the angle of descent [43], therefore forcing this value to be -3^{o}. As such, the employed constraints on these angles will be

$$\begin{cases} \gamma_1 = 0^{\text{o}} \\ \gamma_2 = -3^{\text{o}} \end{cases}\,. \tag{3.12}$$

Chapter 4

Optimization Framework

In this chapter we will cover how the disciplines interact with one another on a conceptual and computational level. We will also discuss how the constraints and objective function described in previous sections will be implemented.

4.1 Design Variables

Let us first formally define the design variables we will be exploring in this work, so that all the main parts of the optimization problems are covered. Most of these can be sorted into a single discipline, meaning we include them in an optimization process if that respective discipline analysis is being considered. The exceptions are the variables which affect the surfaces shapes, in our case their taper λ and span b, which are shared between the aerodynamic and structural analyses.

- Aerodynamics: angle-of-attack (α), twist distribution($\theta(y)$) for both wing (w) and tail (t);
- Propulsion: throttle setting (δ_T);
- Structures: spar thickness distribution ($t(y)$) on the wing (w);
- Trajectory: flight path angle (γ);
- Aerostructural: wing span (b_w).

It is also possible to contemplate only a few of these variables for specific results, such as optimizing upon the throttle setting, tail twist (incidence), α and γ alone for an optimal control situation.

Another important aspect regarding the design variables is the range of values within which these are allowed. For some of them, this range is conditioned by construction, namely

$$\begin{cases} -15^{\text{o}} \leq \alpha \leq +15^{\text{o}} \\ -15^{\text{o}} \leq \theta_w, \theta_t \leq +15^{\text{o}} \\ 0 \leq \delta_T \leq 1 \end{cases} . \tag{4.1}$$

since our aerodynamic analysis is based on the small angle approximation, and δ_T is a percentile multiplier.

The boundaries of the remaining design variables will be based on our reference aircraft, the *B777-300* [44]. As such, we will be imposing

$$\begin{cases} 50 \leq b_w \leq 70m \\ 0.01 \leq t_w \leq 0.05m \end{cases} . \tag{4.2}$$

The values for the span b_w of the wing were based on the wingspan found in the original aircraft [44]. As for the spar thicknesses, the boundaries have been defined so the final weight of the aircraft resembles that of the original aircraft. To this end, we can base ourselves on the operating empty weight (OEW). From [44] we know our OEW should be close to $1.7 \times 10^5 kg$.

One important note regarding the implementation of the spar thicknesses and twist (for the wing) is that these distributions are defined by a spline. Essentially, rather than defining the twist and spar thickness at every control point of the structure as a design variable, we may define any number of control points for these two distributions and define a B-spline curve from these. The values can then be computed from the curve. This does not apply to the tail since we will be not optimizing the tail spar thickness (and will be considering all of its spars to have the same thickness). Also, since its twist will be acting as the tail incidence, we will be using a single value for the tail twist, meaning no additional computations are required.

The values for the flight path angle γ were previously discussed in eq. (3.12).

4.2 Constraints and Performance Metric

Beside the boundaries on our design variables, we must also define additional constraints pertaining to the trim conditions and structural integrity of the aircraft.

Trim conditions

First, let us cover the equilibrium conditions discussed in section 2.2, starting with the vertical force equilibrium. This equation is expressed through an auxiliary variable as

$$L\_\text{equals}\_W = \frac{Wcos(\gamma) - L - Tsin(\alpha)}{W} = cos(\gamma) - \frac{L}{W} - \frac{Tsin(\alpha)}{W} = 0 . \tag{4.3}$$

The numerator on the right-hand side of eq. (4.3) represents eq. (2.25a) (with $\phi_T = 0$), which means the condition will be met if we force this variable to 0. It should be noted that the lift, L, in this variable has been computed as per eq. (2.27a). Similarly, we may enforce the horizontal force equilibrium condition through another variable,

$$T\_\text{equals}\_D = \frac{Wsin(\gamma) + D - Tcos(\alpha)}{W} = sin(\gamma) + \frac{D}{W} - \frac{Tcos(\alpha)}{W} = 0 , \tag{4.4}$$

where the numerator on the right-hand side of eq. (4.4) represents eq. (2.25b) (with $\phi_T = 0$), and the condition is imposed by equating this variable to 0. The drag, D, has also been computed as described in eq. (2.27b).

Both eq. (4.3) and eq. (4.4) have been constructed by dividing the respective equilibrium conditions by the aircraft's total weight, W. This is done in order to reduce the magnitude of the auxiliary constraint variables, and therefore scale the optimization problem better.

Analyzing both eq. (4.3) and eq. (4.4) and comparing them to the input/output diagrams discussed in chapter 3, we can see how the auxiliary variables we have defined will depend on outputs and design variables from all disciplines. Namely, the aerodynamic computations will dictate the behaviour of $\boldsymbol{\alpha}$, $\boldsymbol{L}$ and $\boldsymbol{D}$; the structural analysis will output $\boldsymbol{W}$; the propulsion analysis will output $\boldsymbol{T}$; and the trajectory conditions define γ. The coupling among the disciplines in the mulitdisciplinary analysis will be covered in the following section, but it is important to keep in mind that the constraints of the full system also depend on all analyses, acting as another level of interdependence.

In regards to the moment equilibrium condition, the total moment coefficient (C_M) is computed as per eq. (2.27c). As described previously, we will be considering complete symmetry across the xz-plane, meaning both roll and yaw moments are null by construction, and we only need to enforce the pitching moment

$$C_{M_y} = 0\,. \tag{4.5}$$

Structural Integrity

Other than ensuring trim conditions, we must guarantee the non-failure of all elements. A Von-Mises yield criterion is utilized to evaluate the integrity of the finite element structure. To perform these computations the displacements themselves are required [45], as well as specific geometric properties of the elements, namely the radii of the spars. Once the Von-Mises tensions, σ_{VM}, have been computed, the constraint is implemented as

$$\text{Failure} = 2\frac{\sigma_{VM}}{\sigma_y} - 1 \leq 0\,, \tag{4.6}$$

where σ_y is the material yield strength. The multiplicative factor in eq. (4.6) arises from the fact we will be using a safety factor of 2.0 in this work. Given the way in which this constraint is constructed, we will require only that this auxiliary variable be negative rather than necessarily null.

Rather than enforcing eq. (4.6) as a different constraint for each finite element, we will be utilizing a constraint aggregation method, based on the Kreisselmeier-Steinhauser equation [46]. This way, only a single failure constraint needs to be applied.

Lastly, a strictly computational constraint is set to ensure that the optimizer maintains the tubular structure geometrically feasible, in the form of

$$\text{Intersection} = \text{Thickness} - \text{Radius} \leq 0\,. \tag{4.7}$$

Performance Metric

As discussed previously, the metric we will be looking to minimize is the amount of fuel consumed in a prescribed mission. Because we explicitly compute the propulsive force for each flight phase, we may define our objective function as

$$m_{\text{fuel}} = \sum_{i=1}^{N} TSFC \frac{R_i}{V_i} T_i \,, \tag{4.8}$$

where we are considering a mission profile sectioned into N phases, and $TSFC_i$, R_i, V_i and T_i are respectively the thrust-specific fuel consumption (in $kg/N/s$ units), covered distance, speed and thrust of each phase. We can expand eq. (4.8) further by replacing the propulsive force with its model (as seen in eq. (2.67)) yielding

$$m_{\text{fuel}} = \sum_{i=1}^{N} TSFC_i \, R_i \, V_i^{\eta-1} k_1 \frac{S\rho_i^2}{2} \delta_{T_i} \,, \tag{4.9}$$

where we can more evidently see how the optimizer will attempt to minimize the amount of fuel spent: by means of the total reference area, S, and the throttle setting. Furthermore, because we are disregarding morphing wing solutions (where the lifting surface shape can change mid-flight) only the throttle variable will allow the optimizer to meet the thrust requirements throughout the flight.

Now that we have covered the main components of the optimization problem, let us synthesize the full data flow scheme.

4.3 Coupling Overview

On a conceptual level, we can illustrate the interaction between disciplines in the multidisciplinary analysis process as in fig. 4.1.

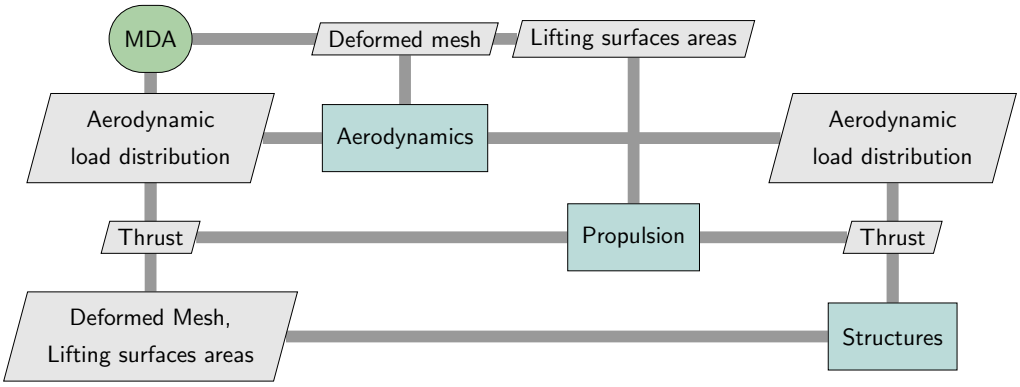


Figure 4.1: Coupling between the three disciplines in the MDA process.

For a more thorough layout of the actual computations, the N^2 diagram for the MDA in our problem is presented in fig. 4.2.

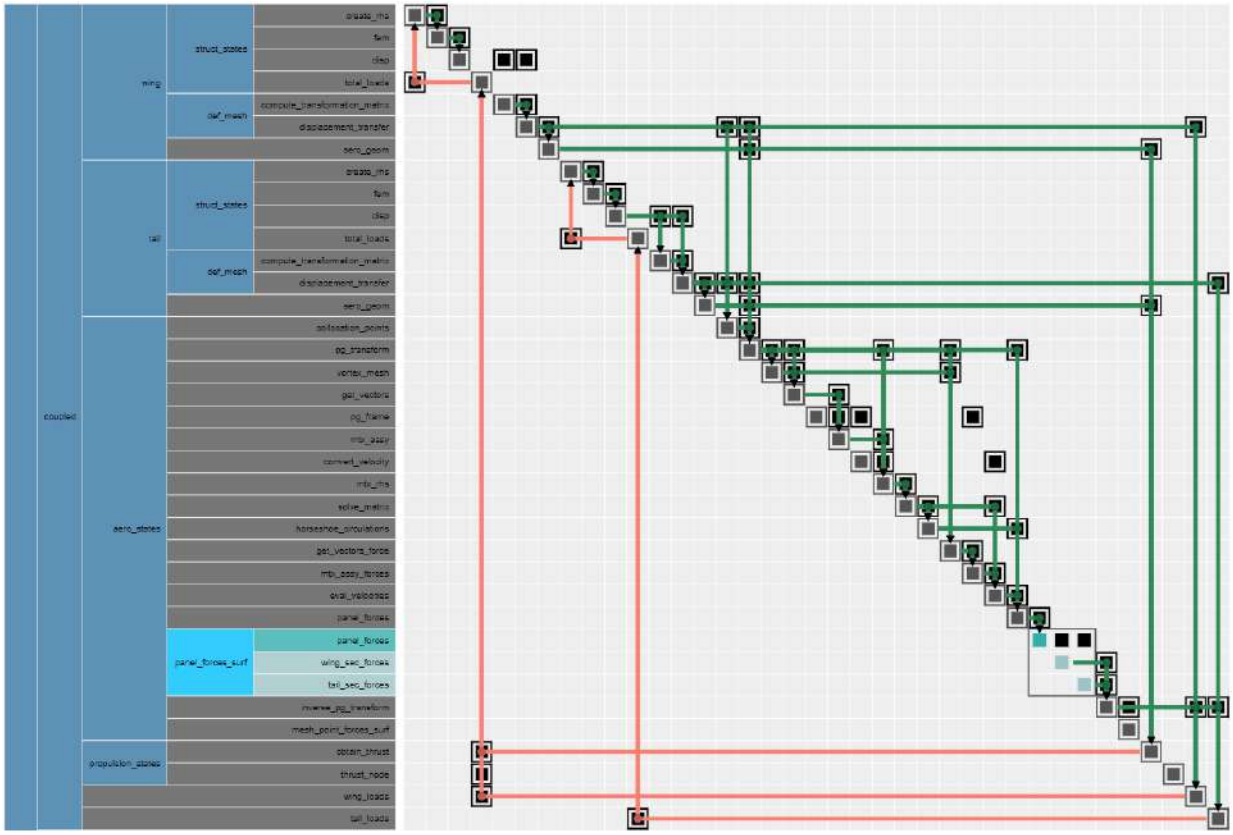


Figure 4.2: N^2 computational diagram for the MDA process.

Several groups are listed on the left side (represented in blue), where we find classes representing some computational step of the process (represented in grey). Data is fed forward on the upper triangular half of the matrix (outlined in green) and fed back on the lower triangular half (outlined in orange).

Comparing fig. 4.2 with fig. 4.1 we can see that the structural computations are performed within the 'coupled.wing' and 'coupled.tail' groups. Namely, the total loads applied on the elements are assembled within 'struct_states' and the displacement distribution is computed in 'struct_states.disp'. The deformed mesh and lifting surface area are then obtained in 'def_mesh.displacement_transfer' and 'aero_geom' respectively. This process is carried out for both the wing and the tail.

Next, we have the aerodynamic analysis within 'aero_states'. Both of the deformed meshes are fed into the top of the group, where the problem is converted according to the Prandtl-Glauert correction ('pg_transform'). The separation between wing and tail computations occurs within the classes; as seen in the expanded 'panel_forces_surf' class, where different arrays carry the data for each surface; hence why this analysis presents a different architecture in the matrix. Once the final aerodynamic load distribution has been converted back ('inverse_pg_transform'), both 'wing_loads' and 'tail_loads' are fed back to their respective structural classes.

Finally, we have the added propulsion discipline within 'propulsion_states'. The 'obtain_thrust' class receives the lifting surfaces areas, as described previously, and feeds the computed thrust force back, to be applied alongside the wing aerodynamic loads.

Chapter 5

MDO - Setup and Results

In this chapter we will be discussing the final details of the implementation, as well as the results we have obtained. Before we start performing full optimization processes however, let us research the convergence of the mesh for a single iteration of the MDA process.

5.1 Mesh Convergence Study

As previously discussed, the mesh for each lifting surface is defined by two parameters each, 'num_x' and 'num_y'. For the purpose of determining an adequate mesh for our problem, the MDA process was converged once with a (2×5) grid for the wing, and a (2×3) grid for the tail, yielding

$$\begin{cases} m_{\text{fuel}_{\text{ref}}} = 116150.8kg \\ t = 16.14s \end{cases}, \tag{5.1}$$

where t represents the time elapsed in the process.

In tables 5.1 and 5.2 we present the relative changes obtained for a selection of progressively finer grids in the x-direction for the consumed fuel and time elapsed, respectively. In each cell, the change is computed in relation to the previous result.

Table 5.1: Convergence in m_{fuel} (x-direction).

num_x	$\delta(m_{\text{fuel}})$ (%)
2	-
3	-0.00852
5	-0.00413
7	-0.00172
9	-0.00095

Table 5.2: Convergence in t (x-direction).

num_x	$\delta(t)$ (%)
2	-
3	$+1.61$
5	$+5.18$
7	$+2.90$
9	$+1.41$

We can see that refining the mesh in this direction yields negligible result changes on our metric (always below 0.01%), even with few control points. The time elapsed in the process, however, suffered

a more considerable change, up to 5%. In absolute terms, this is not very relevant considering it represents a change smaller than $1s$. However, it is important to consider that the results obtained in these computations only used 5 points in the y-direction. As we add more control points along the span (which will require several more control points), this computational cost will be exaggerated. For the sake of computational simplicity, only 3 control points will be considered in this direction.

Now, in tables 5.3 and 5.4 we present the results for similar computations in the y-direction.

Table 5.3: Convergence in m_{fuel} (y-direction).

num_y	$\delta(m_{\text{fuel}})$ (%)
5	-
7	-2.57
9	$+0.131$
11	-0.664
13	-0.0523
15	-0.0117
17	-0.126
19	-0.0348
21	-0.0387
23	-0.00851

Table 5.4: Convergence in t (y-direction).

num_y	$\delta(t)$ (%)
5	-
7	$+4.71$
9	$+0.362$
11	$+0.186$
13	$+0.653$
15	-0.124
17	$+6.21$
19	-1.98
21	-0.856
23	-0.175

The result sensibility in the y-direction is understandably greater, as num_y also controls the number of finite elements which compose the structural model. A relative change inferior to the aforementioned 0.01% is achieved with 23 control points across the entire span.

We have now established the grid we will employing for our cruise situations. In the following sections in this chapter we present and discuss the optimization results we obtained for different problem configurations.

5.2 Additional Optimization Parameters

Beside the design variables discussed in the previous chapter, there are other parameters which are relevant to the problem. These are listed in table 5.5, 5.6 and 5.7, and are utilized in all optimization problems unless stated otherwise.

Table 5.5: Spar material properties (Aluminum 7075).

Parameter	Value
$E(Pa)$	70×10^9
$G(Pa)$	30×10^9
$\sigma_y(Pa)$	500×10^9
$\rho(kg/m^3)$	2.8×10^3

Table 5.6: Tail parameters.

Parameter	Value
$b_t(m)$	24.22
λ_t	0.118
$l_t(m)$	0.05

Table 5.7: Cruise flight conditions and TSFC.

Parameter	Value
M	0.84
$h(m)$	10.7×10^3
$\rho(kg/m^3)$	0.38
Range (m)	11×10^6
$W_0(kg)$	1.125×10^5
TSFC $(kg/N/s)$	1.54×10^{-5}

Table 5.8: Numerical Parameters.

Parameter	Value
Iterative tolerance	10^{-7}
Optimizer tolerance	10^{-3}

The tail parameters were obtained by performing an optimization problem similar to the one which will be discussed in section 5.4, starting from a symmetrical rectangular surface. These values are not considered as design variables as it was observed that doing so greatly increased the problem's complexity for minimal objective function gains. For the optimization problems within this chapter, we will be considering the aircraft maximum range as well as its cruise altitude at all times [44]. The W_0 parameter factors in for the rest of the empty weight of the aircraft. As mentioned previously, part of our goal is to make sure the empty operating weight of our aircraft matches that of the original $B777-300$. Finally, regarding the parameters in table 5.8, the iterative tolerance applies to the system solving algorithm, i.e. the MDA analysis - it acts as a convergence threshold for each design point. On the other hand, the optimizer tolerance applies to the constraints we apply to the problem - acting as a convergence threshold for the entire problem.

5.3 Aerodynamic & Propulsion Optimization

5.3.1 Problem Setup

Let us start by performing single-point optimization considering the aerodynamic and propulsive systems in a cruise setting - an aeropropulsive (AP) problem. Referring back to the problem elements discussed in chapter 4, this optimization problem is defined as

$$\begin{array}{ll} \text{minimize} & \text{Fuelburn} \\ \text{with respect to} & \alpha, \gamma, \theta_t, \theta_w, \delta_T, b_w \\ \text{subject to} & L\_\text{equals}\_W = 0 \\ & T\_\text{equals}\_D = 0 \\ & C_{M_y} = 0 \\ & \gamma = 0 \end{array}, \tag{5.2}$$

where we consider all constraints and design variables except for those pertaining exclusively to the wing structure: the spar thicknesses and their failure.

Twist Curve Point Study

We will be optimizing upon the wing twist distribution which is defined, as described in section 4.1, by a B-spline constructed from a given number of points equally distributed across the wingspan. As such, we must define the number of points we will be employing. Let us perform an analysis similar to that of the previous section. Starting from a process considering a single point, we have

$$\begin{cases} m_{\text{fuel}_{\text{ref}}} = 115660.7kg \\ t = 20.06s \end{cases} . \tag{5.3}$$

Tables 5.9 and 5.10 list the relative changes from running the same MDA process with additional twist control points.

Table 5.9: Convergence in m_{fuel}.

B-spline points	$\delta(m_{\text{fuel}})$ (%)
2	-0.225
3	-0.00260
4	-0.00104

Table 5.10: Convergence in t.

B-spline	$\delta(t)$ (%)
2	$+1.994$
3	$+5.767$
4	$+1.802$

Following the same criterion we used in section 5.1, we will be employing a 3-point B-spline curve to model the wing twist.

Table 5.11 lists the initial values and allowed ranges for the design variables (DV).

Table 5.11: Design variable parameters for the AP cruise optimization problem.

DV	Initial value	Range
$\alpha(°)$	5	$[-15, +15]$
$\gamma(°)$	2	$[-15, +15]$
$\theta_t(°)$	-5	$[-15, +15]$
$\theta_w(°)$	$(-3.75, 1.5, 7)$	$[-15, +15]$
δ_T	0	$[0, 1]$
$b_w(m)$	60	$[50, 70]$

5.3.2 Results

Design Variables

Figures 5.1 to 5.6 illustrate the progression of the design variables through the optimization iterations.

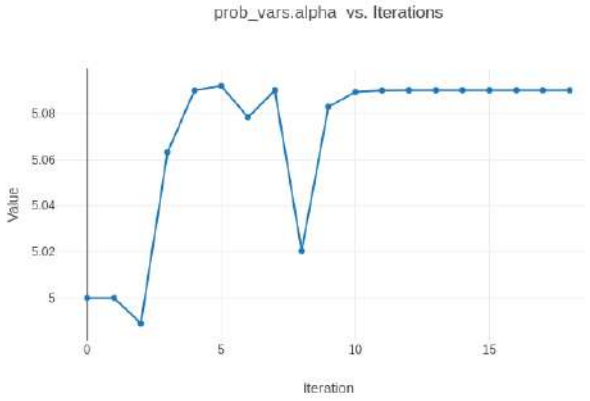


Figure 5.1: AP cruise optimization: α

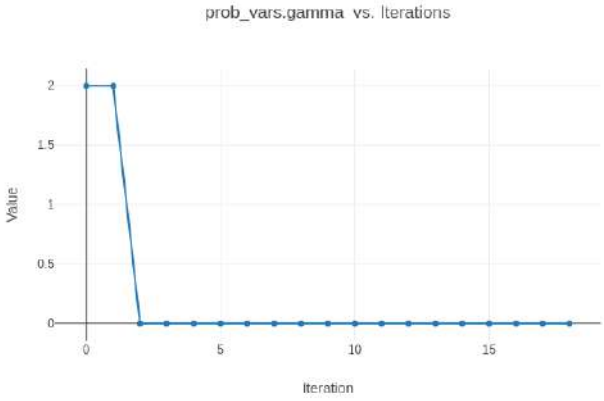


Figure 5.2: AP cruise optimization: γ

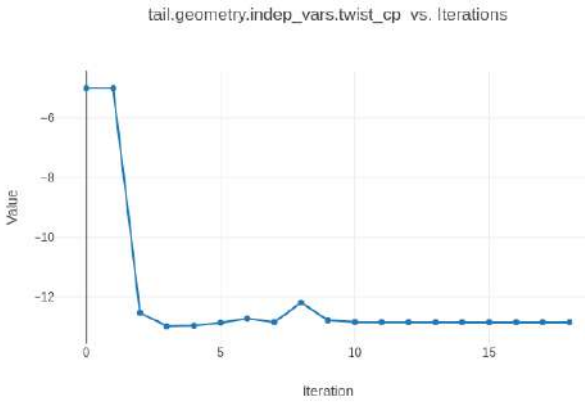


Figure 5.3: AP cruise optimization: θ_t

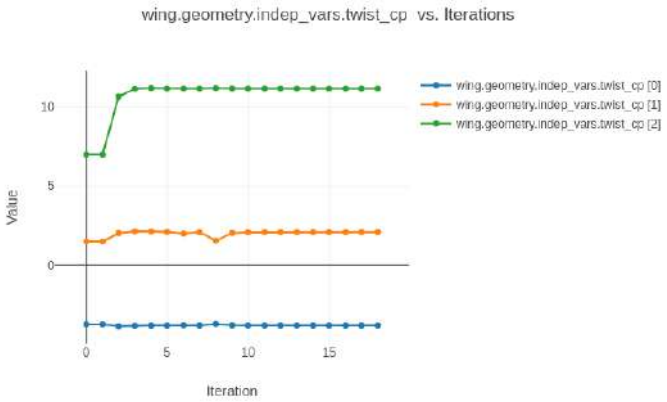


Figure 5.4: AP cruise optimization: θ_w

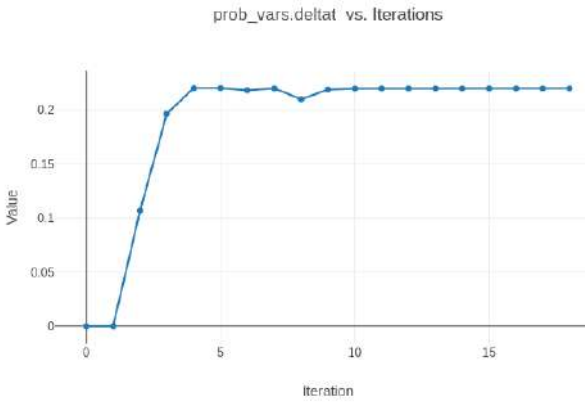


Figure 5.5: AP cruise optimization: δ_T

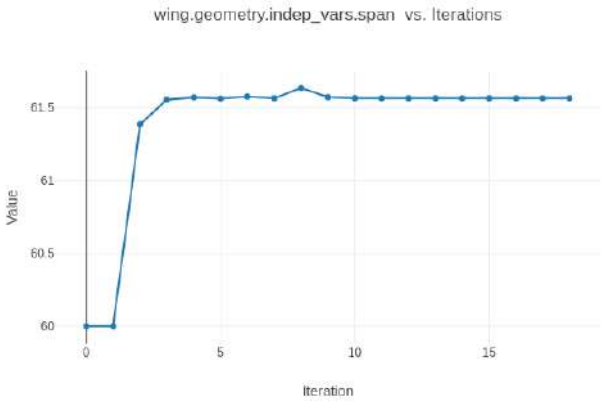


Figure 5.6: AP cruise optimization: b_w

And table 5.12 lists their final values.

Table 5.12: Final design variable values for the AP cruise optimization.

DV	Final value
$\alpha(°)$	5.09
$\gamma(°)$	0
$\theta_t(°)$	-12.8
$\theta_w(°)$	$(-3.81, 2.08, 11.2)$
δ_T	0.220
$b_w(m)$	61.6

Constraints and Objective Function

Figures 5.7 to 5.10 depict the progression of the trimming conditions as well as the amount of fuel burnt through the optimization iterations.

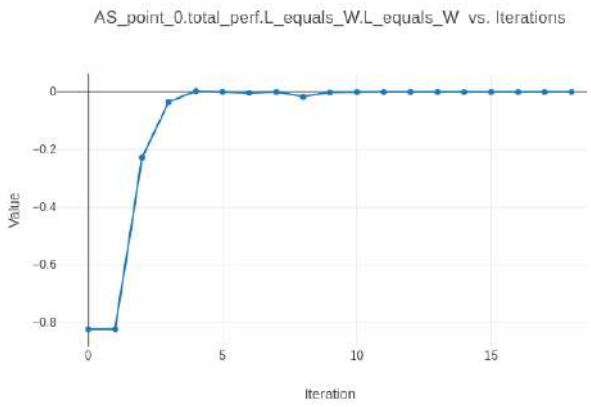


Figure 5.7: AP cruise optimization: L_equals_W

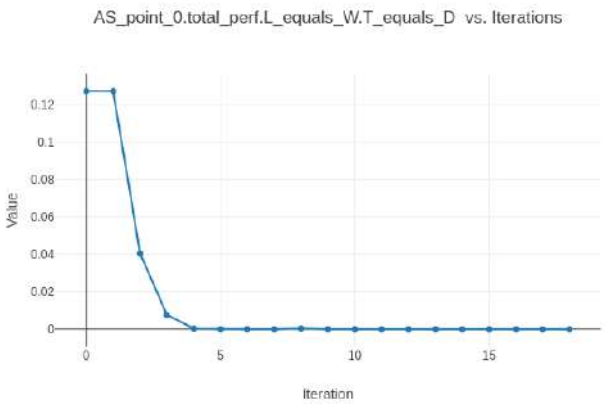


Figure 5.8: AP cruise optimization: T_equals_D

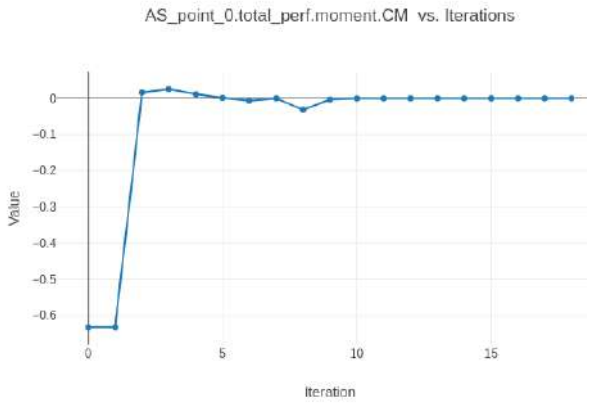


Figure 5.9: AP cruise optimization: C_{M_y}

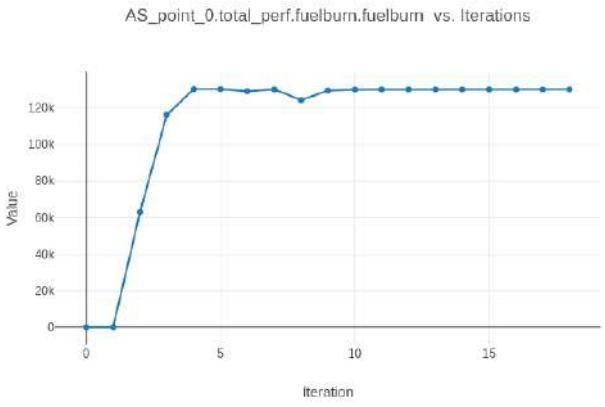


Figure 5.10: AP cruise optimization: Fuelburn

And table 5.13 lists their final values.

Table 5.13: Final constraint and objective function values for the AP cruise optimization.

Parameter	Final value
L_equals_W	-1.68×10^{-5}
T_equals_D	4.82×10^{-6}
C_{M_y}	-1.01×10^{-5}
Fuelburn (kg)	130193.0

Additionally, *OpenAeroStruct* allows us to plot the final structure as illustrated in fig. 5.11.

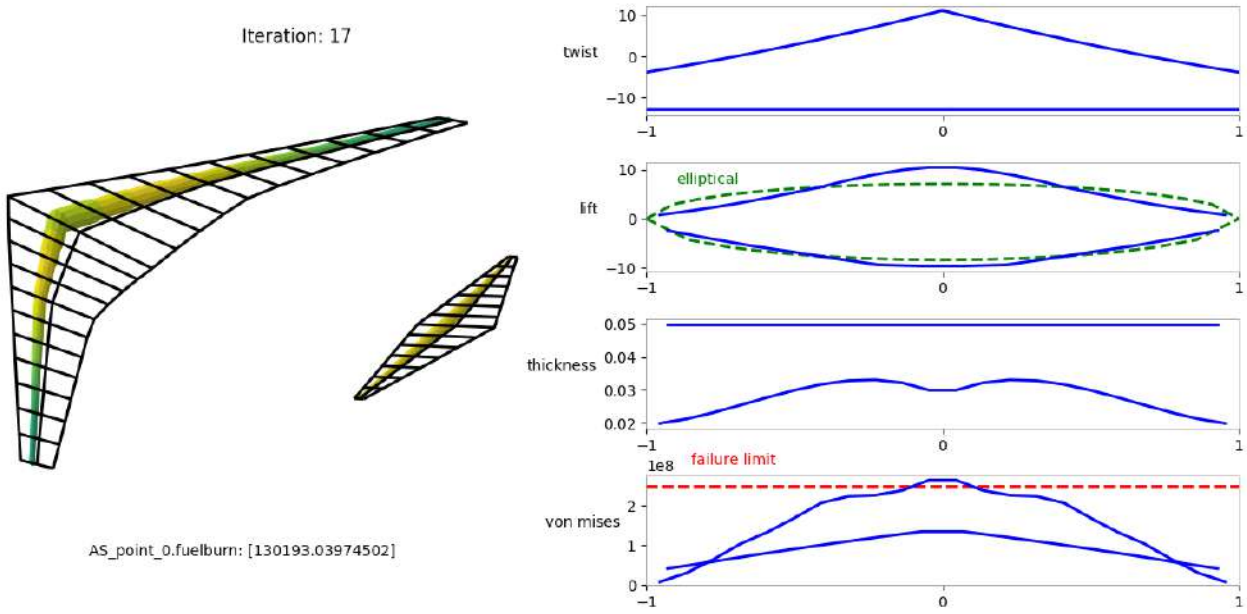


Figure 5.11: Optimized lifting surfaces for the AP cruise problem.

This optimization process elapsed 18 iterations, and was concluded in $127.16s$. We can see the process was successful, as all constraints are met (within their designated threshold) and all design variables concluded within their established ranges.

We may compare some of the values we have obtained with those from the actual aircraft [44]. Namely, the wingspan we have obtained $(61.6m)$ is close to that of the *B777-300* $(60.9m$, with a relative error of $1.15\%)$. Also, assuming a jet fuel density of around $800kg/m^3$ [47], the mass of fuel computed translates to

$$V_{\text{Fuelburn}} = \frac{130193}{800} = 162.7413m^3 = 162741.3l,$$

which is 4.93% lower the original aircraft maximum fuel capacity. This is adequate, since we are considering a maximum range mission and not factoring in any fuel reserves.

In fig. 5.11 we can see that the wing structure experienced failure (bottom-right plot). This was to be expected, since we did not enforce this constraint. Also in this figure, the twist plot displays a nearly linear progression across the wingspan (top-right plot), even though we optimized upon 3 B-spline points. This is congruent with the small variations described in table 5.9, suggesting the optimized twist distribution

is accurately described with just 1 or 2 curve points. Still in the same figure, the middle-right plot exhibits one of the hallmarks of MDO versus sequential optimization in aircraft design. Sequential optimization yields elliptic lift distributions [11], and we can see that our the lift distribution for the computed wing strays away from the green dashed line.

5.4 Aerostructural & Propulsion Optimization

5.4.1 Problem Setup

Now, let us add the structural elements to the optimization problem - a spar thickness curve for the wing spars as a design variable; their failure across the span and their physical feasibility as constraints. We now have a aerostructural-propulsive problem (ASP). Referring once again to the problem elements discussed in chapter 4, the new optimization problem is

$$\begin{array}{ll} \text{minimize} & \text{Fuelburn} \\ \text{with respect to} & \alpha, \gamma, \theta_t, \theta_w, \delta_T, b_w, t_w \\ \text{subject to} & L\_\text{equals}\_W = 0 \\ & T\_\text{equals}\_D = 0 \\ & C_{M_y} = 0 \\ & \gamma = 0 \\ & \text{Failure} \leq 0 \\ & \text{Intersection} \leq 0 \end{array} , \tag{5.4}$$

where we now consider all constraints and design variables.

Spar Thickness Curve Point Study

With the addition of the wing spar thickness distribution to the list of design variables, we are including another spline defined variable. This means we must once again ascertain the number of points we will be using in the optimization problem.

The MDA reference values for a single-point curve are

$$\begin{cases} m_{\text{fuel}_{\text{ref}}} = 115406.8kg \\ t = 18.87s \end{cases} . \tag{5.5}$$

Tables 5.14 and 5.15 list the relative changes from running the same MDA process with additional thickness B-spline control points.

Table 5.14: Convergence in m_{fuel}.

B-spline points	$\delta(m_{\text{fuel}})$ (%)
2	$+0.0250$
3	-0.0386
4	$+0.00581$
5	-0.00407

Table 5.15: Convergence in t.

B-spline points	$\delta(t)$ (%)
2	$+0.212$
3	$+1.16$
4	$+1.41$
5	-0.567

The 0.01% threshold is crossed with a 4-point B-spline, which we will be utilizing for the following optimization.

Table 5.16 lists the initial values and allowed ranges for the design variables in the updated problem.

Table 5.16: Design variable parameters for the ASP cruise optimization problem.

DV	Initial value	Range
$\alpha(^\circ)$	5	$[-15, +15]$
$\gamma(^\circ)$	2	$[-15, +15]$
$\theta_t(^\circ)$	-5	$[-15, +15]$
$\theta_w(^\circ)$	$(-3.75, 1.5, 7)$	$[-15, +15]$
δ_T	0	$[0, 1]$
$b_w(m)$	60	$[50, 70]$
$t_w(m)$	$(0.02, 0.0233, 0.04, 0.03)$	$[0.01, 0.05]$

5.4.2 Results

Design Variables

Figures 5.12 to 5.18 display the progression of the design variables through the optimization iterations.

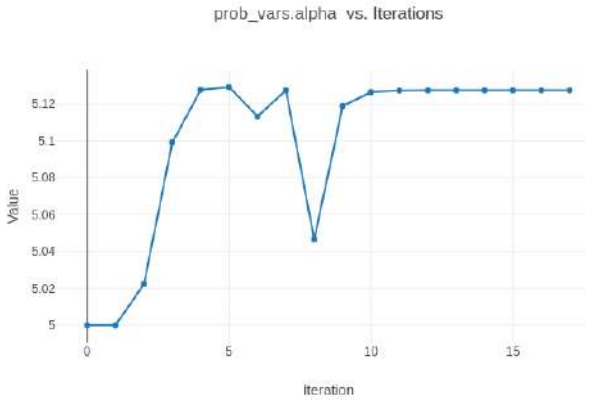


Figure 5.12: ASP cruise optimization: α

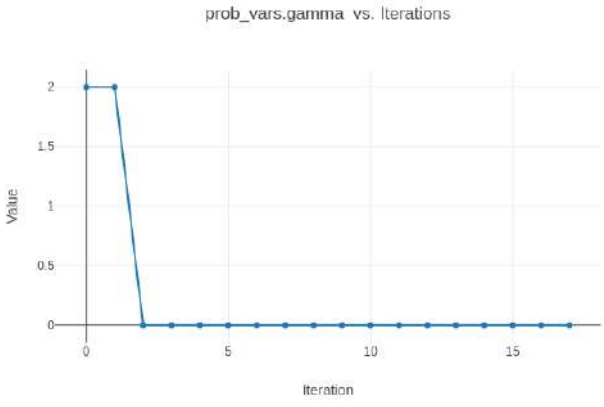


Figure 5.13: ASP cruise optimization: γ

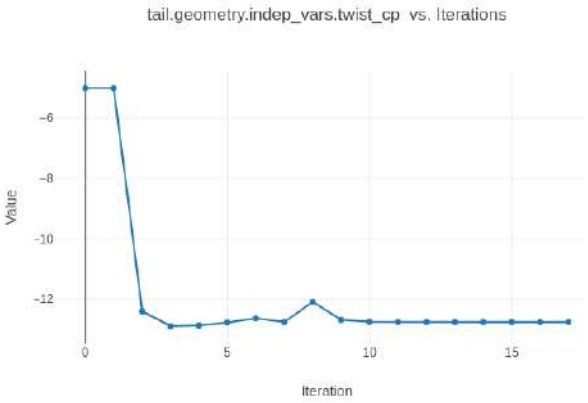


Figure 5.14: ASP cruise optimization: θ_t

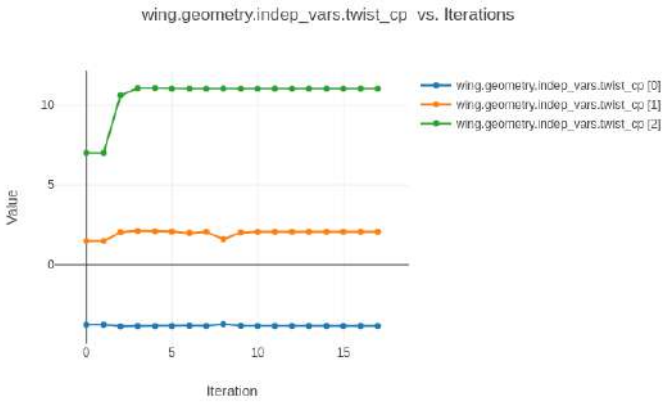


Figure 5.15: ASP cruise optimization: θ_w

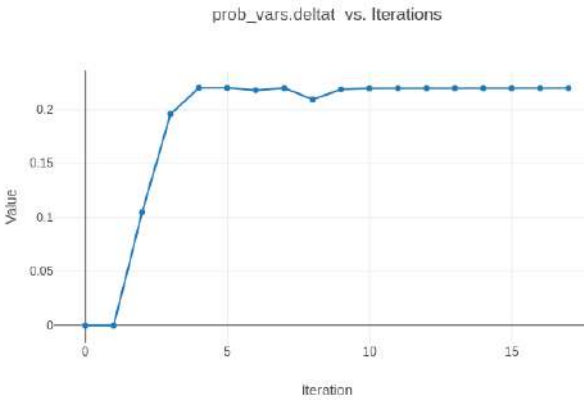


Figure 5.16: ASP cruise optimization: δ_T

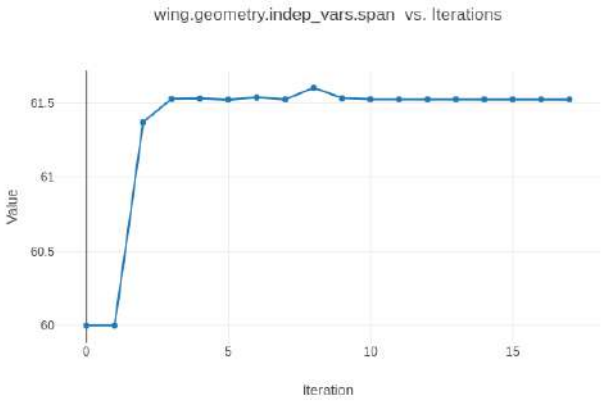


Figure 5.17: ASP cruise optimization: b_w

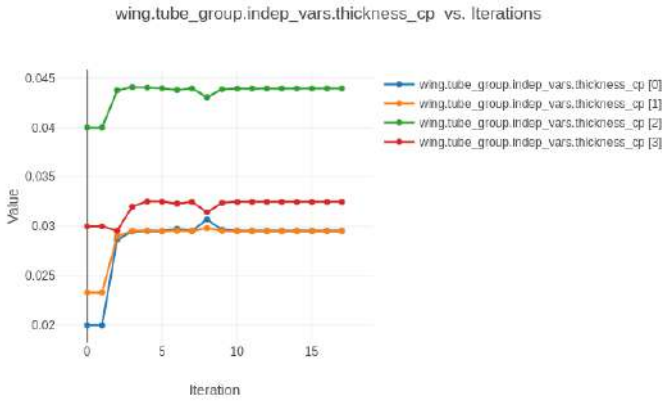


Figure 5.18: ASP cruise optimization: t_w

Table 5.17 lists their final values.

Table 5.17: Final design variable values for the ASP cruise optimization.

DV	Final value
$\alpha(°)$	5.13
$\gamma(°)$	0
$\theta_t(°)$	-12.8
$\theta_w(°)$	$(-3.82, 2.06, 11.03)$
δ_T	0.220
$b_w(m)$	61.5
$t_w(m)$	$(0.0295, 0.0295, 0.0440, 0.0325)$

Constraints and Objective Function

Figures 5.19 to 5.24 depict the progression of the trimming conditions, structural constraints, as well as the amount of fuel burnt through the optimization iterations.

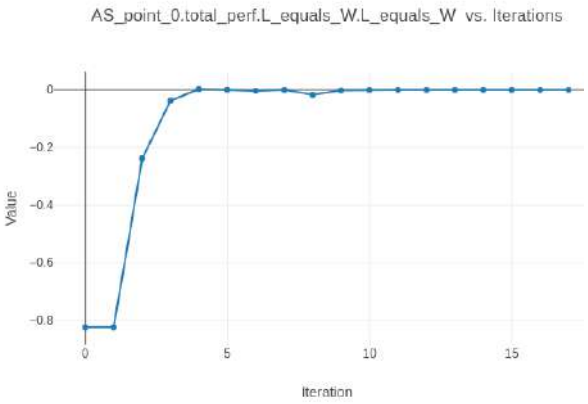


Figure 5.19: ASP cruise optimization: L_equals_W

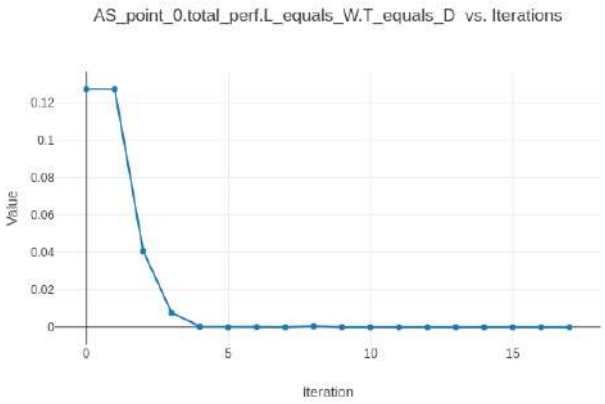


Figure 5.20: ASP cruise optimization: T_equals_D

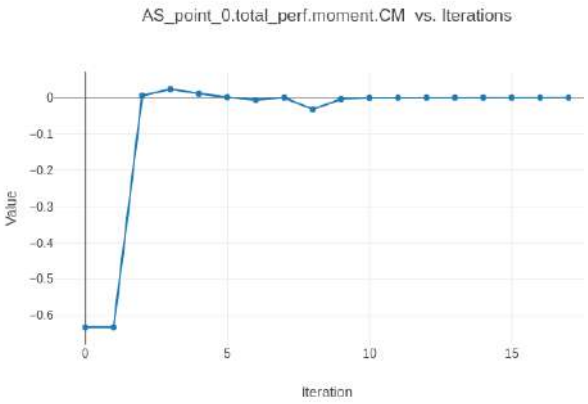


Figure 5.21: ASP cruise optimization: C_{M_y}

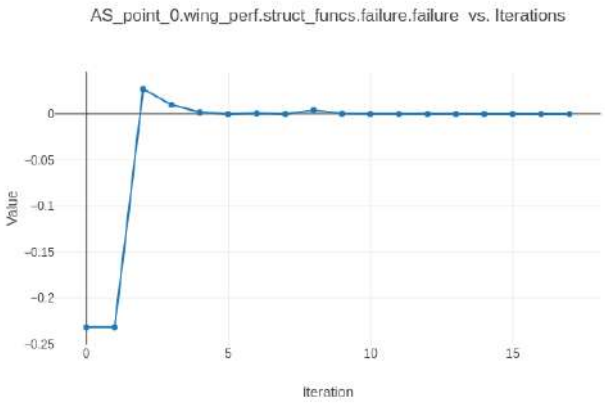


Figure 5.22: ASP cruise optimization: Failure

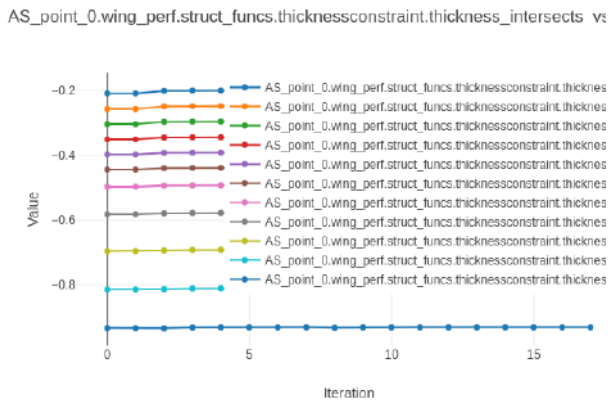


Figure 5.23: ASP cruise optimization: Intersection

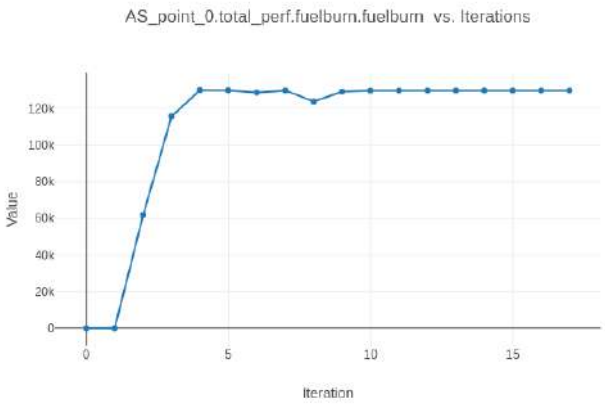


Figure 5.24: ASP cruise optimization: Fuelburn

And table 5.18 lists their final values.

Table 5.18: Final constraint and objective function values for the ASP cruise optimization.

Parameter	Final value
L_equals_W	-4.88×10^{-5}
T_equals_D	6.15×10^{-6}
C_{M_y}	-1.15×10^{-4}
Failure	-1.86×10^{-5}
Intersection	$(-0.209 \sim -0.933)$
Fuelburn (kg)	129872.9

Finally, our final wing is fully plotted in fig. 5.25.

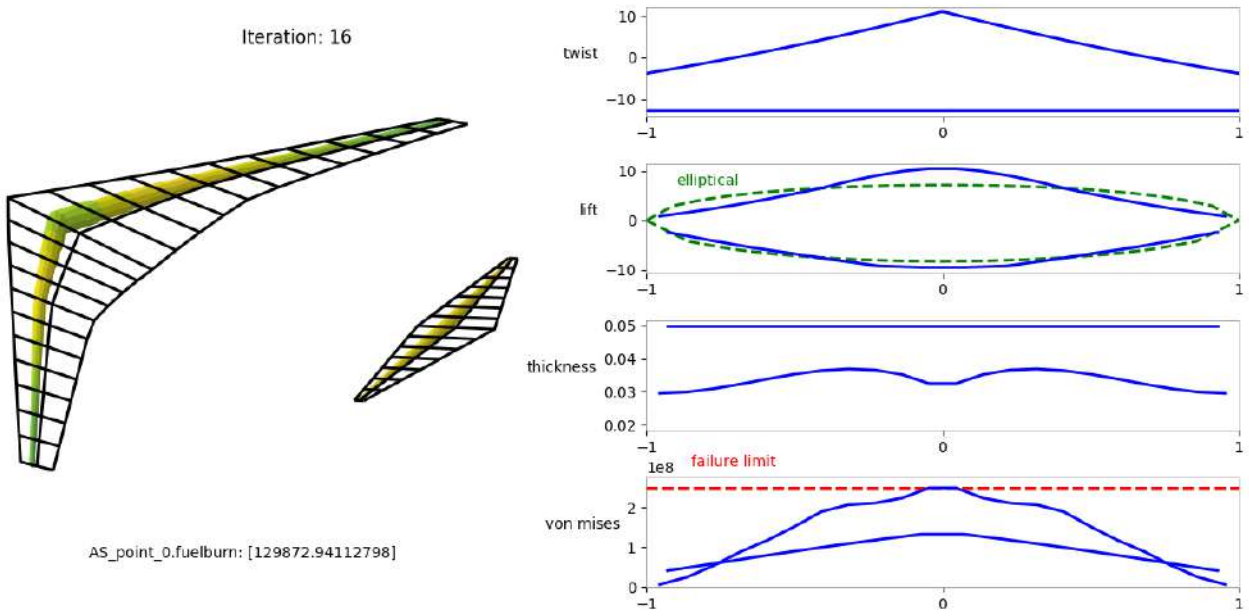


Figure 5.25: Optimized lifting surfaces for the ASP cruise problem.

This optimization was carried out in 17 iterations, over a time period of 138.59s. Once again, all of the constraints have been met, and all design variables remained within the established values. We have arrived at a very similar solution, as can be seen from the values in table 5.17, as well as the progression of most of the displayed plots.

When compared to the previous optimization solution, a marginal improvement has been attained in the amount of fuel spent, namely a decrease of

$$\delta(m_{\text{fuel}}) = \frac{129872.9 - 130193.0}{130139.0} = -0.246\%. \tag{5.6}$$

This change has also come at the cost of computational time, which increased by

$$\delta(t) = \frac{138.59 - 127.16}{127.16} = +8.99\%. \tag{5.7}$$

Similarly to what was mentioned in previous sections, $11s$ are, in absolute, a relatively irrelevant amount of time. However, these optimization cases are also significantly low-fidelity, and part of the purpose of these results is to gain a sensibility towards how the different elements of these optimization problems work, and how they might scale for larger problems. In relative terms, 9% is a significant time increase. Furthermore, it should be noted that in the AP optimization process carried out in the previous section the structural computations were still performed, we merely omitted all design variables and constraints pertaining exclusively to the discipline. The computational cost gap between both processes would certainly be greater if the previous process had strictly only performed aerodynamic and propulsive computations.

We have introduced both new design variables as well as new constraints, meaning there was no reason to necessarily expect either a better or worse solution. Some of the trends pointed out in the previous analysis are still present, such as the non-elliptical lift distribution and the nearly linear twist distribution.

Unsurprisingly, the greatest change pertains to the thicknesses of the spars. Comparing the thickness plots from both fig. 5.11 and fig. 5.25, we can see that the thickness of the spars near the center of the span has changed to accommodate the failure constraint, which is now met. More interestingly, the thicknesses of the spars near the wing tips have also increased, even though they do not seem to approach the failure threshold. There are a few reasons why this might be:

- The low constraint tolerance we have imposed in this work. The value of 10^{-3} is a compromise due to convergence robustness problems and as an effort to keep the optimization processes short and quickly repeatable. The reason for these problems might also be tied to the following point;
- How the propulsive force has been applied to the structure, considering that much of the strength of the original code is predicated on how the loads are distributed across the structure. Conversely, the implemented propulsive force and respective moment are applied on a single node, which might have also interacted poorly with the aggregated failure constraint approach mentioned in section 4.2.

Lastly, now that we have involved the wing structure in the optimization process, we can also compare the operating empty weight we have obtained with that of the original $B777-300$. The structural masses computed in this optimization process were $43632.3kg$ and $10558.3kg$ for the wing and tail, respectively. Adding these to the W_0 parameter described in section 5.2 yields

$$W_0 + W_{\mathsf{w}} + W_{\mathsf{t}} = 112500 + 43632.2 + 10558.3 = 166690.5kg. \tag{5.8}$$

This is another good fit to the original value of $160500kg$ [44], with a relative error of 3.86%.

5.5 Descent Trajectory Control

Now that we have obtained our cruise-optimized aircraft configuration, the last step is to optimize its control variables to the second flight phase we will be considering, the descent.

5.5.1 Problem Setup

As mentioned in previous sections, the main changes we will be employing in order to perform this optimization pertain to the flight conditions. Table 5.7 listed the cruise altitude and respective air density. As laid out in section 3.6.2, we will now be performing computations at half the altitude, meaning

$$\begin{cases} h_2 = 5350m \\ \rho_2 = 0.709kg/m^3 \\ c_2 = 319.1m/s \end{cases}, \tag{5.9}$$

are our new atmospheric conditions, and where c_2 is the speed of sound (at the new altitude).

Next, let us ascertain our new speed and Mach number. Our initial velocity (in cruise, that is) is $248.1m/s$. During the main descent phase, the $B777-300$ reduces its velocity to $154.3m/s$ [48].[1] Averaging both of these values yields

$$V_2 = \frac{248.1 + 154.3}{2} = 201.2m/s, \tag{5.10}$$

and therefore

$$M_2 = \frac{201.2}{319.1} = 0.63. \tag{5.11}$$

Furthermore, it was also established in section 3.6.2 that we would perform our descent at

$$\gamma = -3^{\mathrm{o}},$$

which means we can determine the ground distance we will be covering in this phase, since we assume

[1] According to [48], this is an indicated airspeed - it is an approximation to utilize this value without accounting for the instrumentation and wind factors.

we are descending from $h = 10700m$ to sea level. The distance covered, R_2, is given by

$$tan(3^{\circ}) = \frac{10700}{R_2} \Leftrightarrow R_2 = 204168.2m \tag{5.12}$$

Now, in order to perform the control optimization for the descent step, we solve the following problem

$$\begin{array}{ll} \text{minimize} & \text{Fuelburn} \\ \text{with respect to} & \alpha, \gamma, \theta_t, \delta_T \\ \text{subject to} & L\_\text{equals}\_W = 0 \\ & T\_\text{equals}\_D = 0 \\ & C_{M_y} = 0 \\ & \gamma = -3^{\circ} \end{array}, \tag{5.13}$$

over a range of $204168.2m$. We have omitted all non-control variables and constraints in eq. (5.13), i.e. any variables which would change the shape of our established wing. The initial (and constant) values for these absent parameters will be those which we obtained from the ASP optimization, listed in table 5.17.

5.5.2 Results

Control Variables

Figures 5.26 to 5.29 display the progression of the control variables through the optimization iterations.

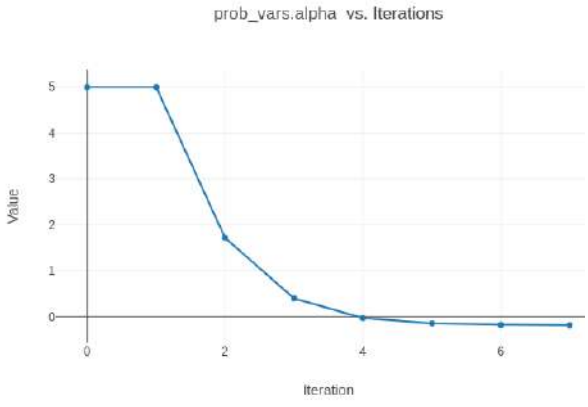


Figure 5.26: Descent control optimization: α

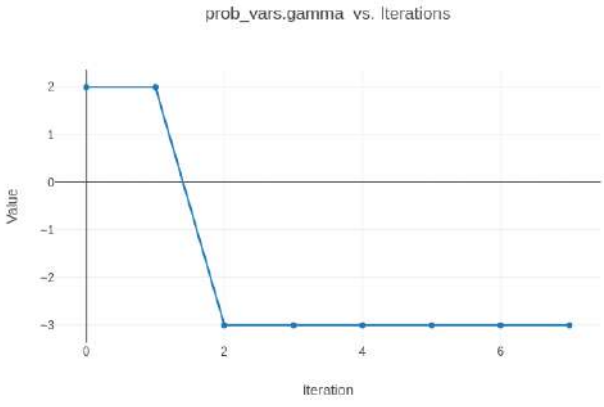


Figure 5.27: Descent control optimization: γ

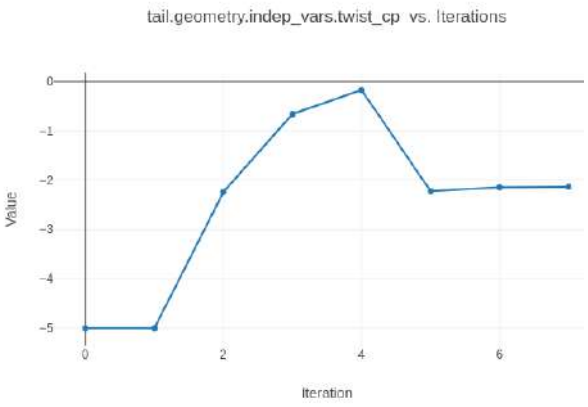


Figure 5.28: Descent control optimization: θ_t

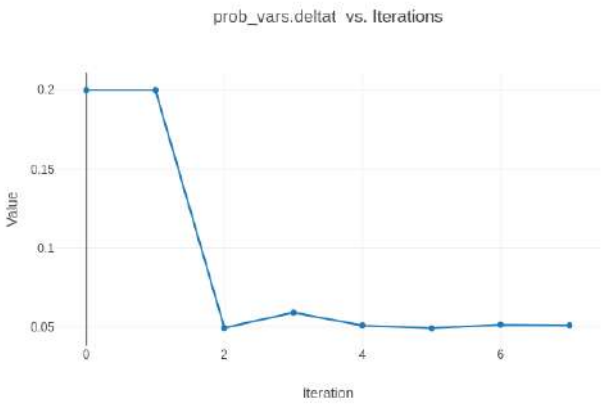


Figure 5.29: Descent control optimization: δ_T

And table 5.19 lists their final values.

Table 5.19: Final control variable values for the descent control optimization.

CV	Final value
$\alpha(^{\circ})$	-0.186
$\gamma(^{\circ})$	-3
$\theta_t(^{\circ})$	-2.13
δ_T	0.0510

Constraints and Objective Function

Lastly, figs. 5.30 to 5.33 depict the progression of the trimming conditions and the amount of fuel burnt through the control optimization iterations.

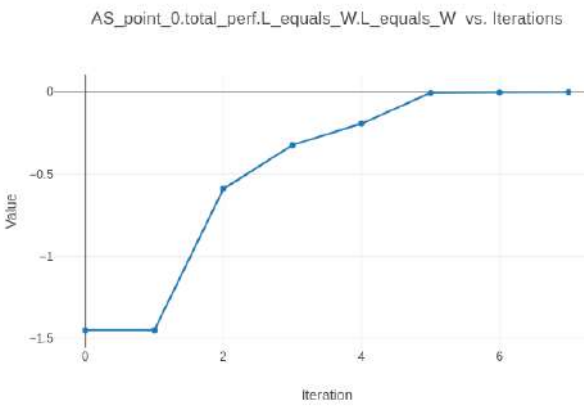


Figure 5.30: Descent control optimization: L_equals_W

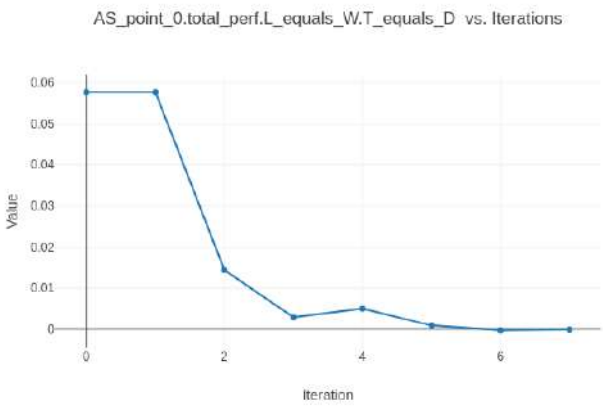


Figure 5.31: Descent control optimization: T_equals_D

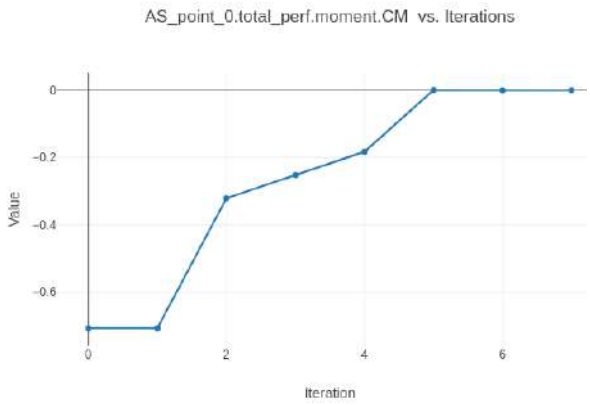


Figure 5.32: Descent control optimization: C_{M_y}

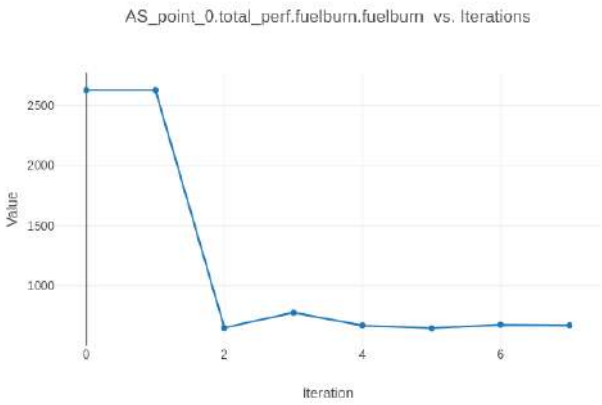


Figure 5.33: Descent control optimization: Fuelburn

Table 5.20 lists the final values for these.

Table 5.20: Final constraint and fuelburn values for the descent control optimization.

Parameter	Final value
L_equals_W	-4.51×10^{-4}
T_equals_D	-6.58×10^{-5}
C_{M_y}	-2.78×10^{-5}
Fuelburn (kg)	668.9

The final wing shape is the exact same as that depicted in fig. 5.25, and therefore there is no reason to display the full plot again.

This optimization process elapsed 7 iterations over the course of $72.96s$, which indicates a significantly lower computational load than the previous cases. This was to be expected since the control optimization problem was, initial conditions apart, strictly simpler than any of the previously discussed problems.

This approach, and therefore these results, cannot be completely separated from the optimization process which we performed previously, since we are attributing an additional amount of fuel necessary for the descent step which was not technically considered previously.

However, the ratio between the two fuel masses showcases why this is a reasonable approximation:

$$\frac{\text{Fuelburn}_{\text{Descent}}}{\text{Fuelburn}_{\text{Cruise}}} = \frac{668.9}{129872.9} = 0.0052. \tag{5.14}$$

The decision to optimize upon the control of the descent phase in specific was deliberate, as this phase incurs a relatively low fuel consumption when compared to other commonly defined flight phases.

With this process, we have showed that our design can handle these descent stipulations, and obtained the control variable values that should employed. As with the previous optimizations, all constraints have been met, including the control variable boundaries. For the sake of completion, we can even observe that the failure condition is not violated, as illustrated in fig. 5.34.

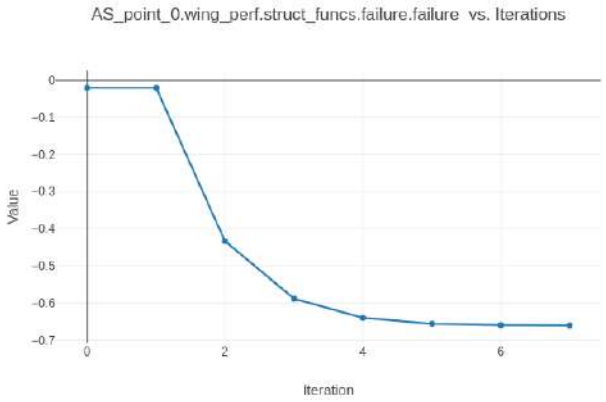


Figure 5.34: Descent control optimization: Failure

Chapter 6

Conclusions

6.1 Summary and Achievements

In this work, we have expanded upon existing aerostructural multidisciplinary optimization code by implementing a propulsive system. We successfully performed MDO considering the three disciplines and all their respective parameters for a cruise situation.

Several of the values we obtained in our final aircraft design closely resembled those of the original aircraft we tried to emulate and improve upon, the $B777-300$. These included the empty operating weight, wingspan and amount of fuel consumed in a maximum range situation.

We were also able to optimize the control variables of the system to perform on a different flight phase, the descent. This meant operating under significantly different atmospheric conditions and that the obtained aircraft was able to achieve trimming conditions with nothing else but its control inputs.

Overall, the final code managed to accomplish nearly everything it was expected to, though there are still several aspects in which it can be improved.

6.2 Future Work

As was mentioned in the discussion of the ASP optimization results, the developed tool lacks robustness for most of its parameters. It is quite sensitive to the initial values it is fed, and it was challenging to converge most of the cases presented in this work. Some suggestions were already mentioned in that same discussion, namely distributing the propulsive load and its moment across more than a single structural node. Conceivably such a solution would alleviate the need for such a large convergence tolerance, and allow for more interesting results.

Also regarding the propulsive force placement, the engine position in the span could have been parameterized, and acted as a design variable. This could be accomplished either as a discrete variable, assuming that the force would always be applied at a structural node (similar to what was done in this work); or as a continuous parameter, if some load/moment transfer scheme were devised to move the propulsive force to the neighbouring nodes. Another possibility would be a distributed propulsion

solution, considering more than one engine per wing.

It could also be interesting to optimize the control variables upon more flight conditions. Because, however, the initial cruise-only optimization becomes less and less accurate the more of these are tacked on, an important improvement to this code would be to implement a complete multipoint optimization scheme. This way, all of the design variables and all of the stipulated flight conditions would be simultaneously considered, and the process would yield the control values for all stages in tandem with an aircraft optimized upon all of the mission phases. This was an original goal for this work which could not be accomplished.

Bibliography

[1] A. Kharina and D. Rutherford. Fuel efficiency trends for new commercial jet aircraft: 1960 to 2014. *International Council on Clean Transportation*, Aug. 2015. doi:10.1080/10618560701678647.

[2] TAP Air Portugal - Star Alliance. http://www.staralliance.com/en/member-airline-details?airlineCode=TP [Accessed 27 June 2019].

[3] J. T. Hwang, S. Roy, J. Y. Kao, J. R. R. A. Martins, and W. A. Crossley. Simultaneous aircraft allocation and mission optimization using a modular adjoint approach. *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, pages 1–17, Jan. 2015. doi:10.2514/6.2015-0900.

[4] J. P. J. John T. Hwang and J. R. R. A. Martins. High-fidelity design-allocation optimization of a commercial aircraft maximizing airline profit. *Journal of Aircraft*, pages 1–15, Feb. 2019. DOI: 10.2514/1.C035082.

[5] E. National Academies of Sciences and Medicine. Commercial aircraft propulsion and energy systems research: Reducing global carbon emissions. Technical report, National Academy of Sciences, 2016. doi:10.17226/23490.

[6] H. Kim, D. Harding, D. T. Gronstal, M.-F. Liou, and M.-S. Liou. Design of the hybrid wing body with nacelle: N3-x propulsion-airframe configuration. *34th AIAA Applied Aerodynamics Conference*, June 2016. DOI: 10.2514/6.2016-3875.

[7] Hybrid wing body goes hybrid. NASA - N3-X. https://www.nasa.gov/content/hybrid-wing-body-goes-hybrid [Accessed 20 March 2018].

[8] I. R. Chittick and J. R. R. A. Martins. An asymmetric suboptimization approach to aerostructural optimization. *Optimization and Engineering*, 10(1):133–152, March 2009. doi: 10.1007/s11081-008-9046-2.

[9] G. J. S. W. M. E. B. Grossman, Z. Gurdal and R. T. Haftka. Integrated aerodynamic/structural design of a sailplane wing. *J. AIRCRAFT*, 25, 1988. doi: 10.2514/1.J052940.

[10] J. P. Jasa, J. T. Hwang, and J. R. R. A. Martins. Open-source coupled aerostructural optimization using python. *Structural and Multidisciplinary Optimization*, pages 1–13, Jan. 2018. doi: 10.1007/s00158-018-1912-8.

[11] J. R. R. A. Martins. Multidisciplinary design optimization. Presentation at 7th International Fab Lab Forum and Symposium on Digital Fabrication, August 2011.

[12] J. R. R. A. Martins. A coupled-adjoint method for high-fidelity aero-structural optimization. Master's thesis, stanford university, 2002.

[13] G. K. W. Kenway and J. R. R. A. Martins. Multipoint high-fidelity aerostructural optimization of a transport aircraft configuration. *Journal of Aircraft*, 51:144–160, 2014. doi: 10.2514/1.C032150.

[14] A. DeBlois and M. Abdo. Multi-fidelity multidisciplinary design optimization of metallic and composite regional and business jets. *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, 2010.

[15] K. Moore and A. Ning. Distributed electric propulsion effects on traditional aircraft through multidisciplinary optimization. 2018.

[16] J. T. Hwang and A. Ning. Large-scale multidisciplinary optimization of an electric aircraft for on-demand mobility. In *2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 1384, 2018.

[17] R. D. Falck and J. S. Gray. Optimal control within the context of multidisciplinary design, analysis, and optimization. In *AIAA Scitech 2019 Forum*, page 0976, 2019.

[18] X. Y. Xiao Chai and Y. Wang. Multipoint optimization on fuel efficiency in conceptual design of wide-body aircraft. *Chinese Journal of Aeronautics*, 2018. doi: 10.1145/3182393.

[19] R. P. Liem, G. K. W. Kenway, and J. R. R. A. Martins. Multimission aircraft fuel burn minimization via multipoint aerostructural optimization. *AIAA Journal*, 53, 2015. doi: 10.2514/1.J052940.

[20] J. Gray, K. Moore, and B. Naylor. Openmdao: An open source framework for multidisciplinary analysis and optimization. In *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, page 9101, 2010.

[21] J. R. R. A. Martins and A. B. Lambe. Multidisciplinary design optimization: A survey of architectures. *AIAA Journal*, Sept. 2013. DOI: 10.2514/1.J051895.

[22] C. H. A. S. o. Nickolay Jelev, Andy Keane. Rule based architecture for collaborative multidisciplinary aircraft design optimisation. *International Journal of Aerospace and Mechanical Engineering*, 1, 2017.

[23] S. Roy and W. A. Crossley. An ego-like optimization framework for simultaneous aircraft design and airline allocation. In *57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 1659, 2016.

[24] A. B. Lambe and J. R. R. A. Martins. Extensions to the design structure matrix for the description of multidisciplinary design, analysis, and optimization processes. *Structural and Multidisciplinary Optimization*, 46(2):273–284, Jan. 2012. doi:10.1007/s00158-012-0763-y.

[25] J. P. D. F. R. M. L. Evin J. Cramer, J. E. Dennis and G. R. Shubin. Problem formulation for multidisciplinary optimization. In *SIAM Journal on Optimization*, 1993.

[26] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8: 101–111, 1960.

[27] A. C. Marta. Aircraft optimal design - course notes. IST, 2018.

[28] W. M. Philip E. Gill and M. A. Saunders. Snopt: An sqp algorithm for large-scale constrained optimization. *SIAM Journal of Optimization*, page 979–1006, 2002.

[29] P. T. Boggs and J. W. Tolle. Sequential quadratic programming. *Acta Numerica*, 1995.

[30] J. R. R. A. Martins and J. T. Hwang. Review and unification of methods for computing derivatives of multidisciplinary computational models. *AIAA Journal*, 51:2582–2599, November 2013. doi: 10.2514/1.J052184.

[31] J. T. Hwang and J. R. R. A. Martins. A computational architecture for coupling heterogeneous numerical models and computing coupled derivatives. *ACM TOMS*, 44, 2018. doi: 10.1145/3182393.

[32] J. D. Anderson. *Fundamentals of Aerodynamics*. McGraw-Hill, 5th edition, 2010.

[33] R. M. C. John J. Bertin and P. V. Reddy. *Aerodynamics for Engineers*. Pearson Education, 6th edition, 2014.

[34] NASA. Role of mach number in compressible flows, . https://www.grc.nasa.gov/www/k-12/airplane/machrole.html[Accessed 30 March 2019].

[35] J. L. Kerrebrock. *Aircraft Engines and Gas Turbines*. MIT Press, 2nd edition, 1992.

[36] A. F. El-Sayed. *Fundamentals of Aircraft and Rocket Propulsion*. Springer, 2nd edition, 2016.

[37] Siemens exceptional electric aircraft motor. https://wordlesstech.com/siemens-exceptional-electric-aircraft-motor/ [Accessed 20 January 2019].

[38] M. Logan, J. Chu, M. Motter, D. Carter, M. Ol, and C. Zeune. Small uav research and evolution in long endurance electric powered vehicles. In *AIAA Infotech@ Aerospace 2007 conference and exhibit*, page 2730, 2007.

[39] R. F. Stengel. *Flight Dynamics*. Princeton University Press, 5th edition, 2004.

[40] M. Kelly. An introduction to trajectory optimization: How to do your own direct collocation. Technical report, SIAM Review, 2017.

[41] NASA. https://commonresearchmodel.larc.nasa.gov/, . [Accessed 27 June 2019].

[42] J. J. A. J.R.R.A. Martins and J. J. Reuther. A coupled-adjoint sensitivity analysis method for high-fidelity aero-structural design. *Optimization and Engineering*, page 33–62, 2005.

[43] C. R. Moren. Pilots, airplanes, and the tangent of three (3) degrees. The PUMAS Collection, December 1999.

[44] Boeing 777 specs, what makes this giant twin work? http://www.modernairliners.com/boeing-777/boeing-777-specs/ [Accessed 27 June 2019].

[45] R. M. Jones. *Deformation Theory of Plasticity*. Bull Ridge Publishing, 2009.

[46] J. Martins and N. M. Poon. On structural optimization using constraint aggregation. In *VI World Congress on Structural and Multidisciplinary Optimization WCSMO6, Rio de Janeiro, Brasil*, 2005.

[47] BP. Handbook of products. https://bit.ly/2N3bEda, 2000.

[48] Boeing performance data. https://www.skybrary.aero/index.php/B773. Accessed 10 October 2019.