Integrated Vehicle and Mission Design using Convex Optimization

by

Beldon Chi Lin

B.S.E., University of Michigan (2018)

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2020

Author . . . .
Department of Aeronautics and Astronautics
May 19, 2020

Certified by . . . .
Olivier de Weck
Professor of Aeronautics and Astronautics and Engineering Systems
Thesis Supervisor

Certified by . . . .
Michele Carpenter
Draper Advisor
Thesis Supervisor

Accepted by . . . .
Sertac Karaman
Chairman, Department Committee on Graduate Education

Integrated Vehicle and Mission Design using Convex Optimization

by

Beldon Chi Lin

Submitted to the Department of Aeronautics and Astronautics
on May 19, 2020, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

Convex optimization is used to solve the simultaneous vehicle and mission design problem. The objective of this work is to develop convex optimization architectures that allow both the vehicle and mission to be designed together. They allow the problem to be solved very quickly while maintaining similar fidelity to comparable methods. Multiple architectures are formulated, and the architectures are implemented and evaluated for a sounding rocket design problem and a hydrogen aircraft design problem. The methodology proves successful in designing the sounding rocket while taking into account the optimal trajectory and control strategy and extended to a multi-mission design case. The hydrogen aircraft was successfully designed, allowing for both the cryogenic tank design to be chosen in conjunction with the mission profile. For the rocket design problem, the integrated vehicle and mission problem can only be combined into alternating and integrated approach, and the integrated architecture for convergence to solution in 50% computation time while reaching similar solution. For the hydrogen aircraft case, a 50+% decrease in fuel burn was able to be achieved compared to regular kerosene with an integrated optimization approach. Future work includes studying the convergence properties as well as increasing the robustness of the architectures.

Thesis Supervisor: Olivier de Weck
Title: Professor of Aeronautics and Astronautics and Engineering Systems

Thesis Supervisor: Michele Carpenter
Title: Draper Advisor

Acknowledgments

What a ride.

I want to first thank my advisors Michele and Oli. Thank you Michele for taking me in, and showing me what a proper graduate student deserves as well as supporting me through everything that was thrown at me, from programs to bureaucracy. Our advising meetings were usually about 25% work and the rest just talking about life, and that honestly was exactly what I needed in this environment. Thank you so much for being a kick-ass advisor. Oli, thank you for taking me in as a graduate student. Your experience, perspectives, and out-of-the-box ideas were eye-opening for me. Thank you for showing me the way. I would also like to thank Prof. Hansman for advising me the first semester and injecting large doses of common sense into my work at varying times during my time here.

I wanted to thank the people around me that helped me so much: in programs, research, physical activities, and in mental health. Adam, Trae, and Allison for ~~sometimes stressful~~/great experiences working on programs at Draper as well as Ravi for great pilot and trajectory optimization conversations. ESL folks (Alex, Matt, Eric, Katie, Skylar, Sydney, Michael) for great times in 33-409. Ned, Chris, and Matt for helping me with GPKit. Cody, Peter, Arthur, Berk, and Jacquie for filling the void of being the only aircraft person in ESL with intense aircraft-nerd-out sessions. George, Cat, Katie, Sam, and Christian for being my Draper fellow buddies. Travis for great business and industry discussions as well as literally-killer-fast dodgeball throws to take our IM team to the playoffs and Leon for weekly intense swim workouts. Finally, a special shout out to friend group comma: Faisal, Regina, Jon, Allen, Lena, Katie, Mark, and Maya. The memories I have her are great because of y'all: the bike rides, Bovas/noodles/rock-climbing/lunch-sailing/donuts!!/ski outings, GBAE, Thought Contagion, and block chain (never again). Thank you all for the great memories. To future test pilots/astronauts Faisal and Jon, my dream is one day to see either one of you fly a plane I was heavily involved in designing. Let's make that happen.

I would not be at MIT nor have finished this without the learning, mentorship, and the push that I received while at Michigan from professors/mentors/classmates alike. Professor Washabaugh: thank you for pushing me out of my comfort zone that is the FXB/Ann Arbor and teaching me all the right ways to think and be an engineer. M-Fly: for teaching me that engineering is a team sport and how to (and not to) lead a team. ARC: for showing me that outreach is an engineer's duty and that there is so much more to aerospace than just building things. My mentors/role models Chris, Jerry, Ben, and Karen: for showing me that to be a great engineer, one has to be a great human being first. Forever and always, Go Blue!

This thesis was primarily motivated by the things I saw while in ADP. I want to thank those in OA and CD as well as the Wolverines and Beavers out there (y'all know who you are). Somehow out of everything that's tried to hold me back, I'm making my way back, COVID-19 won't stop me. I can't wait to join all of you and work on crazy yet important ”stuff” that keeps our nation safe.

Saving the most important for last, I want to thank my closest ones for all their support. My father, for always striving to give me the best and reminding me to make things work regardless of the situation. My mother, for reminding me that I can do it even when I didn't believe it and to embrace the unexpected. My younger sister, for showing me how an older sibling should act. And finally, to Golda, for staying by my side, showing me the lighter/creative side of things, and riding out this roller coaster with me. It seems like we've always been so close yet so far throughout our entire lives (actually though), and I'm so glad that we finally came together at the time we both needed each other.

I can't thank all of you enough.

Funding for this work was generously provided by The Charles Stark Draper Laboratory, Inc. through the Draper Fellowship program.

Contents

List of Figures

List of Tables

Nomenclature

BCD	Block Coordinate Descent
GP	Geometric Program
LH2	Liquid Hydrogen
MCP	Multi-Convex Programs
MDO	Multidisciplinary Design Optimization
QP	Quadratic Program
SCO	Sequential Convex Optimization
SOCP	Second-Order Cone Program
SP	Signomial Program

Chapter 1

Introduction

This chapter introduces the simultaneous vehicle and mission design problem, methodologies used to solve both sets of problems, and the goals and layout of this thesis.

1.1 Motivation

Current vehicle designs point towards the following trends:

- Complex vehicle(s): aerospace vehicles have become more complex. They are not simply just the sum of the classical disciplines of aerodynamics, structures, and propulsion. They include controls, software, complex mechanisms/moving parts, mission systems, etc. Furthermore, vehicles are also asked to interact with each other, and there are usually many vehicles flying simultaneously whether cooperatively or not.

- Complex mission(s): missions are more complex. Rather than simply flying from point A to point B, flight procedures and the squeezing of all possible performance out of the vehicle is making the missions more complex. Furthermore, vehicles are asked to fly multiple missions, sometimes at the same time. This only increases the complexity of the missions.

- Tightly coupled vehicle and mission performance: complexity of vehicle and mission drive each other. The vehicle dictates what mission can be accomplished, and the mission that the operator wants to accomplish drives the requirements of the vehicle. As both become more complex, they become more tightly coupled in that one small change in either will significantly affect the other.

Two modern day examples include Urban Air Mobility (UAM) [18] and hypersonics [12]. In UAM, the trajectory that the vehicle is designed to heavily influences which configuration and battery the vehicle will look like, and Clarke defines the mission envelope first before diving into configuration studies [18]. For hypersonics, Bowcutt includes a trajectory optimization in the MDO loop to ensure that the vehicle closes despite the fact that mission being flown is a relatively simple one: cruise [12]. That inclusion is necessary for the vehicle design to close.

With these trends in mind, it is necessary to look at the design process to ensure that these trends are captured. The typical aerospace vehicle design process can be summarized through the following high level process as shown in Figure1-1 [3] given in the NASA System Engineering Handbook. Starting from the left, a need or want for a mission and associated vehicle is put into the process. The need or want is first defined into a **mission** or **a plan of operation**. This definition usually defines how the need or want is going to be fulfilled as well as the necessary components/vehicles. The end product can be as simple as a mission/control profile or as complex as a Concept of Operations (CONOPS). Specific details about specific operations are often scarce in these definitions, and mostly left until the detailed design phases to be determined. Once the mission is defined, requirements for the vehicle are "derived", and given to a vehicle design team who then produces a design that meets those requirements. The vehicle design is then fed back into the mission to determine whether the actual vehicle performance evaluated through a mission simulation actually satisfies the need and want, If the need/want is satisfied, the process ends and detailed

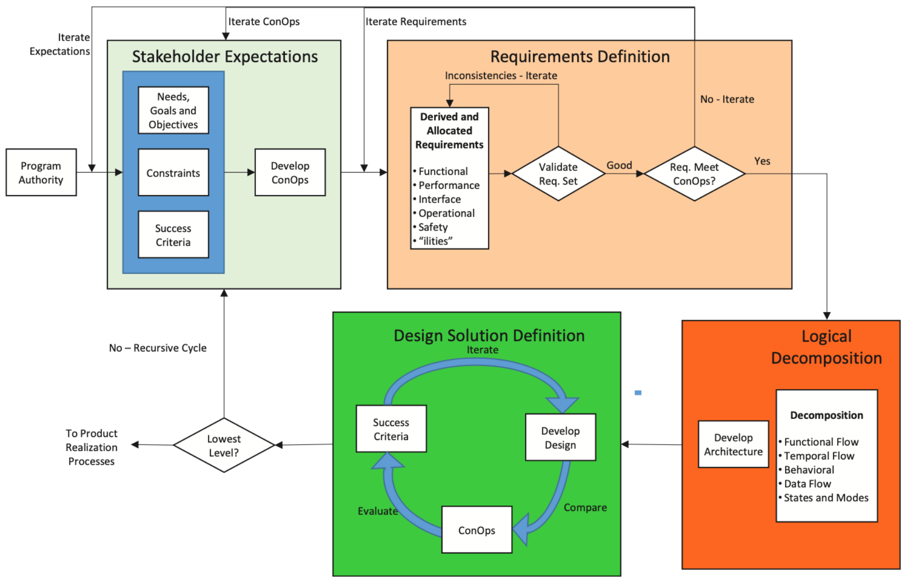


FIGURE 4.0-1 Interrelationships among the System Design Processes

Figure 1-1: NASA Systems Engineering Handbook Design System Design Process [3]

design/manufacturing usually occurs. If it is not, then the process begins again with the mission being redefined, taking into account the lessons learned from the previous cycle. The mission development begins in the "stakeholder expectations" box where the CONOPS is developed. The recursive behavior of the process is apparent in both figures. Further details about this normative process can be found in the NASA Systems Engineering Handbook [3].

Traditionally, mission definition and operations analysis is usually carried out independently of the vehicle design process, and often times not thought of as an integral part of the conceptual design process. Figure 1-1 has no mention of the mission and/or operation design in the design solution definition box: the only time mission considerations are taken into account is in the stakeholder analysis or the requirements definition. This often times results in a sub-optimal vehicle because the stakeholder expectations has no input on feasibility of reaching those expectation as well as other potential inputs from engineering. That may result in the system that is not reaching its full potential or even worse, does not fulfill the needs and wants of the customer. A very challenging or nearly infeasible mission given the state of technology at that time(e.g. heat-resistant materials) can be a major cost driver. Conversely a mission profile that is not aggressive or challenging enough may leave potential of a flight vehicle unexplored.

One way to remedy this separation and resulting sub-optimality is to design and/or optimize the mission and the vehicle at the same time: the goal of this thesis is to find a way such that both design spaces can be explored together, while ideally preserving guarantees of global optimality. Rather than simply defining the mission early, we will allow as much of the detailed mission to change in the vehicle design phase or in other words, allow them to become design variables. For example, for a commercial aircraft, rather than defining apriori that the vehicle must climb to 35000 ft and cruise there, we allow the entire mission profile to be a knob that can be tuned in attempting to find the best vehicle-mission combination. Developing methods and tools that can do this will allow the focus to be spent on maximizing fulfillment of the need/want rather than on unnecessary time spent iterating. This thesis focuses on developing the methods and tools that allow for this type of design optimization.

1.2 Literature Review

Methodologies for designing either the vehicle and/or the mission are introduced here specifically for aerospace vehicles. The sections are split between introducing different vehicle and mission design methodologies and tools with differing levels of fidelity and aggregation. A brief overview of current vehicle and mission design work is also reviewed. The primary takeaways from the prior works found are summarized at the end.

1.2.1 Vehicle Design

Aerospace vehicles are complex systems, and therefore require processes and methodologies that account for that. However, as computational power has increased and become cheaper, the design process has been adapted to account for increased design space exploration. With the increase in computational power, the design process for aerospace vehicles has adapted to incorporate the fact that the many sub-system design trades involve many different disciplines. Multidisciplinary Design Optimization (MDO) is as defined by Martins and Lambe as "a field of engineering that focuses on the use of numerical optimization for the design of systems that involve a number of disciplines or subsystems" [52][7] . MDO has had large successes in finding performance through the synergy of different disciplines such that the overall performance of the vehicle improves. MDO has found some success in the later portion of the conceptual design stage as the focus during the conceptual design stage is on finding "feasible" and preferably non-dominated designs rather than optimizing the performance. Key physical takeaways can also be gleaned from the results of MDO due to the inherent nature of large uncertainties that exist within the conceptual design space exploration.

Regardless, literature does exist for conceptual design-level MDO work. Mull et. al looked at using genetic algorithms to explore different vehicle configurations in one single problem. [54]. Hoburg developed a convex optimization framework that allows for fast optimization of aircraft designs through Geometric Programming [39]; however, because of the simple functional form required of these models severely limits the utility of Geometric Programming.

Preliminary design is where MDO methodologies have shown greatest utility. A great example of MDO improving vehicle design is with the design of the X-43 [12]. Bowcutt was able to use a multidisciplinary design framework to find a feasible and mature hypersonic scramjet configuration that was eventually built and flown. The framework was the state of the art in that it was able to incorporate all major disciplines including a mission optimization in the form of trajectory optimization. However, the analysis was severely limited in both number of variables and the amount of computational resources required was immense [12], not to mention that the trajectory optimization was for a relatively simple mission phase: cruise.

MDO methodologies have improved since then. Martins et. al recognized the importance of gradient based optimization and the importance of accuracy in the calculation of gradients, and has developed many methods such as complex step derivative calculation and adjoint based methods to improve the use of gradient based MDO. OpenMDAO was developed by NASA to incorporate Martin's derivative framework [51] into a software library that can allow the definition and passing of derivatives between the different models [30]. They have had great success in converging on large scale high fidelity aerospace design problems. Mavris et. al at Georgia Tech's Aerospace System Design Lab (ASDL) have used the development of tools that allow seamless coupling of models as well as statistical tools such as JMP to examine very complex systems [41][53].

Convex optimization has made its way into preliminary design though still in the earlier stage of preliminary design. Hoburg et. al extended the work from Geometric Programming to Signomial Programming which trades formulation restrictions with global optimality, and demonstrated the power of Signomial Programming by converting Drela's TASOPT formulation into a convex optimization problem and solving it much faster while producing somewhat similar solutions [42]. However, further work and model development must be done to extend Signomial Programming to other more complex, non-traditional problems. With these convex optimization approaches, the computational resources required to run these kinds of analysis shrinks dramatically at the cost of the fidelity of the models utilized in the analysis. Nevertheless, they represent an interesting approach in examining the vehicle design problem.

In summary, both conceptual design and preliminary design processes incorporate mission design to some extent. MDO has had success in solving the vehicle and mission design problem; however, is still limited in the number of disciplines due to immense computational resources required. Convex optimization is a new technique to solve the aircraft design problem and represents a potential area where new methodologies may arise.

1.2.2 Mission Design

Mission design spans everything from defining a network of routes or behaviors to simply defining the mission parameters that the vehicle would be flying. There are many different methods and processes that go through; the following literature was focused on methodologies for aerospace vehicles.

Taylor et al. demonstrated the benefits of jointly optimizing a cargo aircraft (or a set of multiple such aircraft)and the network in which they operate. A 10 percent improvement in cost was obtained over traditional network flow optimization [68].

Similar to vehicle design, mission design has different levels of fidelity in examining performance of the vehicle. However, mission analysis is often not about the performance of the vehicle, but also based on other metrics such as passengers delivered, mission success rate, etc. Those require a certain degree of aggregation or clustering of behavior to analyze complex systems or even systems of systems. Chapter 2 will go much more into detail about the different levels of fidelity or aggregation and how that relates to how the problem is formulated. Figure 1-2 from AFRL shows the different types of models that are used for differing analyses. For example, if the primary answers that are needed to be answered involve tactics, then mission/few-on-few simulations which don't have high fidelity physics modeling are needed and would be used. If sub-system operations are being simulated, high fidelity physics modeling is needed and any interaction between other systems may be sacrificed to

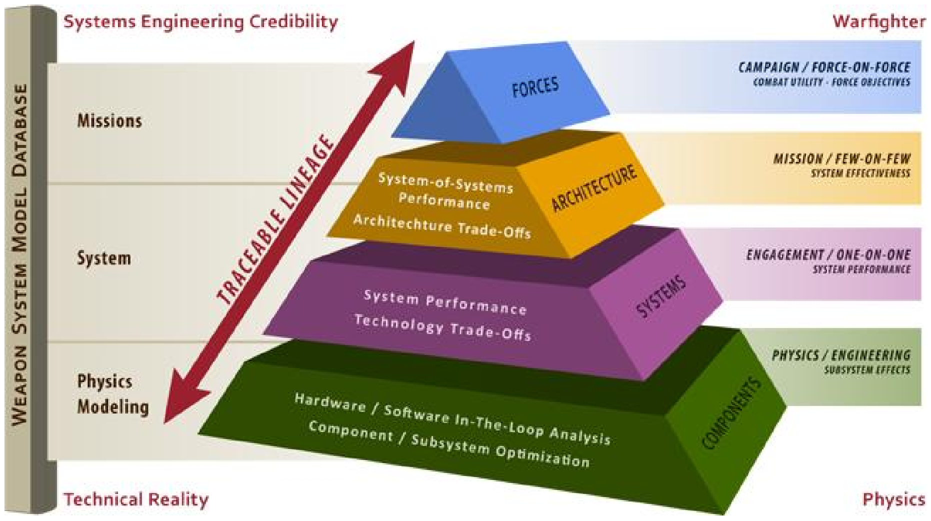


Figure 1-2: AFRL Modeling and Simulation Pyramid [1]

maintain computational tractibility.

There has been much work in developing mission design methodologies, especially from the operations research community. One that is most applicable to aircraft is the development of network optimization algorithms for optimal routes allocation. Network optimization is important in the context of airline fleet design as well as where to place hubs. These decisions also drive which aircraft airlines should purchase from the airframe manufacturers.In this case the aircraft are often assumed to have already been designed and are optimally chosen from a catalogue. These type of methodologies can span from evolutionary network optimization[64] to route design through trajectory optimization[36].

Mission design can also go down to the details such as trajectory design and optimization. One of the daily trajectory optimizations that airlines perform are to minimize fuel burn in the presence of winds at cruise altitude, particularly for long distance flights ¿ 12 hours. Many tools exist that are also used in conjunction with vehicle design processes. These bring in optimal control theory or control system design into the conceptual design process. NASA developed Optimal Trajectories by Implicit Simulations (OTIS)[31] and Program to Optimize Simulated Trajectories

(POST)[4] to do trajectory optimization for different aerospace vehicles ranging from aircraft to reentry vehicles to spacecraft. GPOPS is used for more general optimal control problems [58]. QuickShot developed by SpaceWorks Enterprise takes advantage of parallel computing and multi-threading to improve trajectory optimization speed [5].

Recent developments in convex optimization have allowed it to be utilized in more complex optimal control problems [47] besides the mission profile and multi-mission optimization previously mentioned in [72]. This has allowed convex optimization to become utilized in purposes of conceptual design as it allows designers and analysts to design for trajectories of aerospace vehicles in earlier stages without significant computational resources or time. For example Liu et. al was able to design optimal hypersonic reentry trajectories [46]. This development makes convex optimization potentially attractive for use as a mission design tool.

In summary, mission design tools exist and are vast in fidelity and aggregation. They pull from different corners of engineering and mathematical sciences with optimal control and operations research being the biggest two areas. Military simulations are also used to examine CONOPS and differing concepts that can be used to verify actual vehicle design. New developments in convex optimization solving optimal control problems signal the potential that convex optimization may be a way to solve both the vehicle and mission design problem.

1.2.3 Integrated Vehicle/Mission Design

There are many attempts at solving this problem using MDO frameworks for different mission types and operations. Below are some significant examples and the techniques used to solve the problems.

Trajectory Optimization

Trajectory optimization has been carried out using OpenMDAO through the development of the Dymos module [25]. Jasa et. al used OpenMDAO to look at the design of aero-thermal systems and aircraft trajectory[40]. Hendricks et. al used OpenMDAO to look at [34] propulsion system design and aircraft climb trajectory. However, both of these examples are limited to a couple of subsystems and the mission itself (in this case the trajectory).

Others have attempted to solve this problem using a more custom framework approach outside of OpenMDAO. Gates et. al was able look at propulsion system design with solar aircraft trajectory [27] . Launch vehicle design is another area where this type of problem has been solved [11]; however, the trajectory formulation is limited and the vehicle models used simplistic. Drela developed TASOPT to look at the design of next generation passenger airliners, and does include a mission profile optimization coupled with a full aircraft + engine physics model []. Grant et. al used indirect trajectory optimization methods and analytic hypersonic aerodynamic methods to look at both aerodynamic and reentry trajectory optimization [29] [28]. All of these examples sacrifice either mission fidelity or multi-disciplinarity to achieve some computational tractability.

Network Optimization

Integration of network optimization directly into the aircraft design process has been met with mixed results at best. Taylor et. al was able to devise an embedded optimization algorithm that allows for integrated transportation design; however, the vehicle design fidelity is at the conceptual design stage which retains the larger uncertainties that exist with a conceptual design[68]. Nevertheless at least 10 percent benefit of vehicle and mission co-design were shown, however, without global optimality guarantees. Roy et. al took the opposite approach through a mixed-integer optimization approach in designing next generation passenger aircraft while taking into account economic/market trends [63][62]; however, immense computational resources were undoubtedly required to run such high fidelity analysis.

Military Operations

From the military side of things, there has been a strong push to develop further this type of joint analysis. The most recent program is the EXPEDITE program from AFRL [8][44]. This project utilized MDO principles to incorporate military simulations to develop analysis that would be able to connect the mission effectiveness to engineering design variables. The mission design tool used is AFSIM which is an agent-based mission simulation software; other tools such as the radar cross section model used are described in detail in Harper [33]. The ability to do this will allow the mission trades to be more detailed as designers and analysts can directly point to what part of the mission is influenced by a design decision more finely. However, optimization was not carried out for this problem; a trade-study approach was used to sweep the design space and then filter aposteriori as set of promising solutions.

Control System Design

More generally, MDO has been applied to what can be considered joint plant and control system design. From a bigger picture perspective, the vehicle could be considered as a plant and the mission it is flying as a control system. Allison goes in depth on what is described as Multidisciplinary Design Optimization of Dynamic Engineering Systems [9]. Herber takes this further by demonstrating the benefits gained from such a mind-set of designing both the plant and control at the same time for small subsystems [35]. Control system design during aircraft conceptual design is not new [59]; however, its use has not been widespread. Both Allison and Herber take a controls approach in determining optimally, which can be guaranteed for simple systems such as mechanisms and control systems which could be of use for larger complex systems like aircraft.

In summary, significant progress has been made in integrating both mission design tools with vehicle design tools through MDO, especially in the military realm where missions tend to be very demanding in several dimensions such as range, acceleration, payload amongst others. The change of perspective of simultaneous plant and control system design has lead to new developments and demonstration of the benefits of such mindset. However, typical compromises involve fidelity of analysis in terms of both level of detail and number of sub-systems optimized in both the mission and vehicle definition are typically made as well as the necessity of high performance computing.

1.3 Integrated Vehicle/Mission Design using convex optimization

Only one example of convex optimization being used to solve this joint vehicle-mission design problem exists. York et. al were able to formulate the passenger aircraft optimization problem into a simple mission profile optimization. However, that formulation is very limited as GP/SP programs cannot accept negative variable values, something that could be required for mission design. The use of convex optimization has not been explored extensively, and severe limitations exist.

The primary contribution of this thesis is to develop other methodologies that overcome these limitations to allow a full solution of the simultaneous vehicle and mission design problem. The rise of convex optimization in both vehicle and mission design begs the question: are there ways to take advantage of these developments to gain deeper insight into this problem?

Specifically, this research investigates the following questions:

- How can convex optimization frameworks be leveraged optimally to solve the simultaneous vehicle and mission design problem.

- How can one demonstrate these frameworks on meaningful example problems and determine the benefits and drawbacks of each framework

This thesis limits mission analysis to trajectory optimization as it is the problem that relates to aerospace vehicles and mission analysis most directly.

1.4 Thesis Overview

This thesis develops a novel convex optimization methodology to solve the simultaneous vehicle and mission optimization problem for aerospace vehicles. The intent is to develop a methodology that allows many different disciplines to be considered at the same time while also optimizing the mission overall. Convergence speed, optimality, computational resources required, and any guarantees of optimality will be considered when examining these methodologies.

The objectives of this thesis are to demonstrate the following:

1. Convex optimization is a useful tool in solving the integrated vehicle and mission design problem.

2. Integrated vehicle and mission design is important in designing next generation vehicles

The thesis follows the following format:

Chapter 2 will introduce convex optimization and the different methodologies that involve convex optimization in solving this problem.

Chapter 3 and Chapter 4 will focus on demonstrating these methodologies for the design of a rocket and a hydrogen-powered aircraft respectively. These examples also serve as arguments for incorporating mission design into vehicle design.

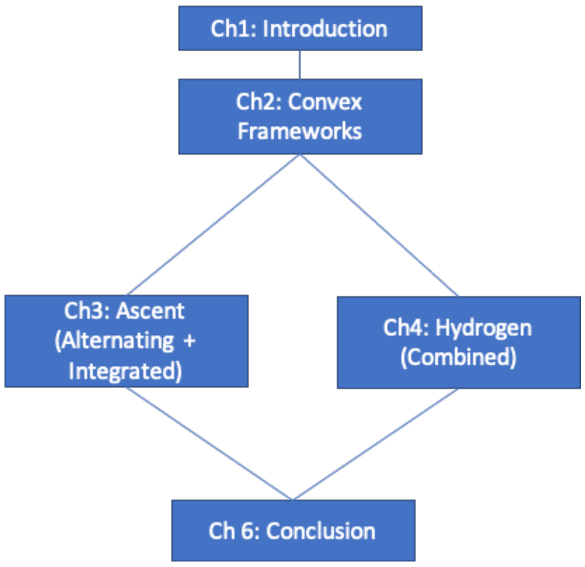


Figure 1-3: Thesis Framework

Chapter 5 will review the primary contributions of the thesis and a discussion about the benefits and drawbacks of this approach. The chapter will also offer future directions of research and lessons learned. Figure 1-3 shows the structure and relations between the chapters of the thesis.

Chapter 2

Convex Optimization Architectures for Integrated Vehicle/Mission Design

In this chapter, convex optimization is formally introduced, and potential methodologies to solve the concurrent vehicle and mission design problem using convex optimization are introduced and developed.

2.1 Convex Optimization

Convex optimization problems are a class of problems that offer many guarantees and solve quickly when the problem is formed correctly and if the problem is feasible. However, the conditions that are necessary to classify an optimization problem as a convex optimization problem are very restrictive. With that being said, convex optimization is still useful and can solve nonconvex optimization problems through many different techniques. In this section, convex optimization is introduced, and various convex optimization formulations and examples that have been used to solve aerospace problems are defined here. Techniques for using convex optimization problems to solve nonconvex optimization problems are also introduced here.

2.1.1 Definition and Convex Programs

A convex optimization problem is defined in Boyd[14] in Problem 2.1.

$$\begin{aligned} \text{minimize} \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq b_i \quad i = 1, \ldots, m \\ & h_i(x) = b_i \quad i = 1, \ldots, n \end{aligned} \tag{2.1}$$

where all functions f are convex, h are affine, and $x \in \mathbb{R}^n$. Convex functions are defined as functions that satisfy the following definition:

$$f(\alpha x + \beta y) \leq \alpha f(x) + \beta f(y) \tag{2.2}$$

for all $x, y \in \mathbb{R}^n$ and all $\alpha, \beta \in \mathbb{R}^n$ such that $\alpha + \beta = 1$, for $\alpha, \beta \geq 0$. In order for this condition to be true over the entire design space the Hessian matrix of $f_0(x)$ has to be semi-positive definite (SPD).

These forms are very restrictive. A non-exhaustive list of example convex functions are listed below[14]:

- Linear functions
- Some quadratic functions
- Affine functions
- Powers x^a for $\mathbb{R}^n_{++}$ when $a \geq 1$ or $a \leq 0$
- Norms $|x|_n$
- Logarithms $log(x)$
- Max function $max(x)$

- Log-sum-exp $log(e^{x_1} + ... + e^{x_n})$

- Exponents e^x

Not listed are some of the most commonly used expressions such as products of two expressions, polynomials, as well as trigonometric functions which are all important for optimization for general aerospace systems. However, if a problem can be put into that form, many guarantees exist for those problems:

- Optimality: By definition, if there exists an optimum and if it is found, it is also the globally optimal solution. The proof for this is given in Section 4.2.2 in Boyd [14]

- Efficient solves: We can utilize state-of-the-art interior point methods to solve the problem quickly. It also scales very nicely and therefore can solve large scale problems.

- Parameter sensitivities: They are by-products of the solving process. Generally, calculating sensitivities requires additional work or utilization of gradient based methods; however, convex optimization produces parameter sensitivities as they are used to solve the problem. These are called dual solutions and can be used for post-processing.

These guarantees are enticing compared to general nonlinear programming where no guarantee of global optimality exists nor any efficient methods to solve without large computational resources. However, many problems are not convex from the outset; some conversion/transformation/tricks are needed. The following are examples of different forms of convex optimization problems that are useful particularly to solving aerospace problems.

Linear Programming

A linear program is an example of a convex optimization where both the objective and the constraints are linear. However, for aerospace problems, they are not as useful besides network optimization. A linear program can be described in the following Problem 2.3 where $x, d, h, b \in \mathbb{R}^n$ and $c, G, A \in \mathbb{R}^{m\times n}$.

$$\begin{aligned} \text{minimize} \quad & c^T x + d \\ \text{subject to} \quad & Gx \leq h \\ & Ax = b \end{aligned} \tag{2.3}$$

Quadratic and Second-Order Cone programs

Linear programs are too restrictive for many design applications, and therefore, other less restrictive convex optimization problems are sought. If the objective function is a quadratic and the constraint functions are affine, the program is called a quadratic program as shown in Problem 2.4 where $x, q, r, h, b \in \mathbb{R}^n$, $G, A \in \mathbb{R}^{m\times n}$, and $P \in \mathbb{S}_+$ Quadratics are convex, and therefore, quadratic programs are convex optimization programs.

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}x^T P x + q^T x + r \\ \text{subject to} \quad & Gx \leq h \\ & Ax = b \end{aligned} \tag{2.4}$$

A well known example of a quadratic program is the least squares regression problem. An extension to quadratic programs and related problem is called the Second-Order Cone Program or (SOCP) as shown in Problem 2.5 where $f, x, b, g, d \in \mathbb{R}^n$ and $A, F \in \mathbb{R}^{m\times n}$.. It is called a second order cone due to the second order norm in the inequality constraint. Although this norm may seem like an arbitrary addition or wrinkle to quadratic programs (especially with a linear objective function), they surprisingly serve as a useful form, especially in solving aerospace-related problems because the norm allows thrust vectors to be modeled as a convex constraint.

$$\begin{aligned} \text{minimize} \quad & f^T x \\ \text{subject to} \quad & ||A_i x + b_i||_2 \leq c_i^T x + d_i \quad i = 1, \ldots, m \\ & Fx = g \end{aligned} \tag{2.5}$$

Geometric Programming

Geometric programs are a special kind of convex optimization problem that is not convex at first glance; however, it can be transformed into a convex optimization problem [13]. A Geometric program can be defined as the following:

$$\begin{aligned} \text{minimize} \quad & p_0(x) \\ \text{subject to} \quad & p_i(x) \leq 1 \quad i = 1, \ldots, m \\ & m_i(x) = 1 \quad i = 1, \ldots, p \end{aligned} \tag{2.6}$$

where p_i are posynomials and m_i are monomials.

Monomials are defined in equation 2.7 where $a_j \in \mathbb{R}$, $c \in \mathbb{R}_{++}$, and $u_j \in \mathbb{R}_{++}$. In other words monomials are products of design variables raised to powers on the real number scale and pre-multiplied by a strictly positive constant coefficient. An example of a monomial is the lift equation: $L = \frac{1}{2}\rho V^2 S C_L$.

$$m(u) = c \prod_{j=1}^{n} u_j^{a_j} \tag{2.7}$$

Posynomials build upon monomials; they are defined as functions that fit the form of equation 2.8 where $a_j \in \mathbb{R}$, $c_k \in \mathbb{R}_{++}$, and $u_j \in \mathbb{R}_{++}$. An example of a posynomial would be a sum of polynomials with positive coefficients: $xy^2 + xz^3$.

$$p(u) = \sum_{k=1}^{K} c_k \prod_{j=1}^{n} u_j^{a_j} \tag{2.8}$$

This formulation, however, is not in convex form. A variable change is required to enforce convexity, and the variable change needed is given in the following equation:

$$x_i = e^{y_i} \tag{2.9}$$

This converts the monomials and posynomials into a log-sum-exp which as noted above is a convex function. Therefore, the transformed geometric program is a convex optimization problem. This allows all convex optimization properties to become true for geometric problems.

There are however some key points about geometric programs that need to be considered:

- **All variable values must be positive.** This is not a problem for many engineering design problems as most parameters are positive or can be converted into positive quantities (e.g. think about material properties such as density, strength etc). This does become a problem when mission design is involved because negative states do exist and are needed. And example would be positive vertical speed to climb and negative vertical speed to descend.

- Posynomials are very restrictive. The constants out front must all be positive, which rules out any polynomials with a negative sign in the constant pre-multiplied factor.

- Constraints must be in those specific forms. Notice on the right hand side that both the inequality and equality constraints are 1, not 0 or some other constant as is usually the case for most optimization problems. This is not as restrictive of a constraint as the other two conditions, but requires some thought especially since for all inequalities the monomial and posynomial constant multipliers in front must be positive.

Specialized modeling languages and solvers exist to solve each of these problems very quickly. MOSEK[10] and CVXOPT[69] are some of the most common solvers that can solve all the problems listed. Modeling languages which are programming languages that allow constraints to simply be formulated rather than forcing the user to form the problem into matrix form exist. Some of these are YALMIP [48], GPKit [17] and CVXPY[20] are the most commonly used though GPKit is only used for Geometric Programming, and all of the modeling languages are integrated with the solvers mentioned before and more specialized solvers for specific problems such as ECOS[22] for SOCPs.

2.1.2 Solving Nonconvex Optimization Problems using Convex Optimization

The previous section focused on methods for solving problems that are convex to begin with or that can be converted directly into a convex optimization problem such as GPs. However, not all problems are convex optimization problems nor can they be directly converted into such. Despite this, convex optimization can still be useful for solving these types of problems. The most common approach is the following: **Approximate the problem or parts of the problem as a convex optimization problem and then solve it. Repeat until the solution stops changing. The solution that results is the final solution**.

This idea is a heuristic; however, because this involves convex optimization, convergence and local optimality can be guaranteed under certain conditions. Unfortunately, only local optimality can be guaranteed as the "approximate the problem as a convex optimization problem" step removes the global optimality: the approximation is only a local one which means that the convex optimization problem is only true for that local area where the approximation is made. The approximation does not cover the entire space, and therefore any optimal point found can only be thought as a locally optimal. Despite this, in most cases, the use of convex optimization in solving non-convex optimization problems is often very useful because the approximate problem exhibits all of the benefits of a natively convex optimization problem such as optimality guarantees, accurate sensitivities, and fast computation time. The question then becomes how accurate is the convex approximation, and there are several techniques to ensure an accurate approximation. This section will introduce expansions of this idea.

Sequential Convex Optimization

Sequential convex optimization is a local optimization method that solves non-convex problems over multiple sequential steps. It is a heuristic and therefore can fail to find an optimal/feasible point and highly depends on the starting point used. Duchi's notes are briefly summarized, and the important parts relating to the problem we are attempting to solve is reproduced here [24].

Suppose you have a non-convex problem in the following form:

$$\begin{aligned} \text{minimize} \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq 0 \quad i = 1, \ldots, m \\ & h_i(x) = 0 \quad i = 1, \ldots, p \end{aligned} \tag{2.10}$$

where f is possibly nonconvex and h is possibly non-affine.

The primary idea is the following:

1. Start with an initial solution $x^{(k)}$ that is feasible and form a convex trust region $\mathcal{T}^{(k)}$ around this solution

2. Form convex $\hat{f}$ and affine $\hat{h}$ approximations for the inequality and equality constraints over the trust region $\mathcal{T}^{(k)}$, respectively

3. Replace the original problem with the convex and affine approximations and solve for $x^{(k+1)}$, the approximate convex problem. The convex approximation problem becomes the following:

$$\begin{aligned} \text{minimize} \quad & \hat{f}_0(x) \\ \text{subject to} \quad & \hat{f}_i(x) \leq 0 \quad i = 1, \ldots, m \\ & \hat{h}_i(x) = 0 \quad\ i = 1, \ldots, p \\ & x \in \mathcal{T}^{(k)} \end{aligned} \tag{2.11}$$

4. Repeat until solution converges

The trust region $\mathcal{T}^{(k)}$ is placed such that the optimizer doesn't go beyond the region of approximation. The trust region can shrink or expand depending on the confidence of the approximation which is judged by how much the solution is changing between each iteration as well as how the approximation is being made. For example, if the original function was already convex, then the trust region can be as large as possible for that function approximation. The trust region size is updated throughout each iteration. A typical trust region can be described as the following where x is the :

$$\mathcal{T}^{(k)} = \{x||x_i - x_i^k|_n \leq \rho_i, i = 1, \ldots, n\} \tag{2.12}$$

where ρ_i is the trust region parameter. The norm of the difference is dependent on the choice of trust region shape.

There are many different ways to produce the convex and affine approximations. A common method is to take the first or second order Taylor expansion of the function. Another method is to take the quadratic trust region in combination with a full second order Taylor expansion. Finally, rather than directly assuming the shape of a function, one can use a particle method to sample the space and fit the data with a convex or affine function. The fitting of the function can sometimes be a convex optimization problem in itself (such as the least squares problem described above), so if that path is chosen, two convex optimization problems are solved sequentially in each iteration. The reader should refer to Duchi's notes for further details on these approximation methods [24]. Further details such as updating the trust region and dealing with infeasibility are covered in Duchi [24].

This formulation forms the basis for many sequential optimization programs such as sequential linear programming and sequential quadratic programming (SQP). Sequential quadratic programming or SQP is the most popular and successful nonlinear optimization method used in many engineering applications. It however takes it a step further by defining the entire problem as a quadratic problem. Nocedal and Wright note that this approximation can also be viewed as an application of Newton's method with enforcement of the KKT optimality conditions, and describes the derivation and practical implementation in Ch. 18 of [57]. Furthermore, local convergence can also be guaranteed in some cases using first order approximations if specific conditions are met [21]. A slightly related sequential convex technique is successive convex approximation. This is a technique used for very large problems and only the objective function is approximated. Readers should refer to Razaviyayn for more details [61].

Signomial Programming

Signomial programming pertains to problems that are similar to Geometric Programs (GPs) but are not GPs. Signomials generalize posynomials in Equation 2.8 such that $c_k \in \mathbb{R}$, allowing even more expressivity. This relaxation of the constants allows different types of constraints to be represented at the expense of global optimality guarantees. SPs are less restrictive compared to GPs in both their objective and constraint functions which allow for local optima to be found reliably. If any of the constraints are signomials, the problem becomes an SP. Although solving an SP is not a convex problem itself, SPs can be represented as a local approximation of the GP. Using these approximations, SPs can be solved through a difference-of-convex optimization problem in the following form:

$$\begin{aligned} &\text{minimize} \quad f_0(x) \\ &\text{subject to} \quad f_i(x) - g_i(x) \leq 0, \quad i = 1, \ldots, m \end{aligned} \tag{2.13}$$

This formulation is advantageous in that no trust regions or parameter tuning is required for the sequential optimization to solve the signomial program. Readers should refer to Kirschen[42] and Boyd[13] for additional details.

Multi-Convex Programming

Another similar idea is the following: What if the overall non-convex problem becomes convex if certain variables were held constant i.e. they became fixed parameters? There is a class of convex optimization problems called multi-convex problems which can potentially apply to multidisciplinary design optimization (MDO)in general. Multi-convex problems are defined as convex optimization problems where if certain variables are fixed (i.e. become constant), the problem becomes a convex optimization problem. An example would be the following simplified problem:

$$\begin{aligned} &\text{minimize} \quad xz + ab \\ &\text{subject to} \quad az^2 < 1 \end{aligned} \tag{2.14}$$

If z and b or the set of $\{z, b\}$ were fixed, (i.e. $z = C, b = K$ where $K, C =$ arbitrary constant), the problem becomes the following:

$$\begin{aligned} &\text{minimize} \quad Cx + Ka \\ &\text{subject to} \quad aC^2 < 1 \end{aligned} \tag{2.15}$$

which is a convex optimization problem. Other sets of variables include $\{a, x\}$ and $\{x, a, b\}$. These problems are interesting in that although a problem may seem non-convex on the surface, convex optimization still can be used to solve the problem by exposing the convex subsets of the overall problem.

Block Coordinate Descent

Generally, mult-convex programs are solved using coordinate descent. The important parts from the primer by Shi et. al on Block Coordinate Descent (BCD) is briefly summarized here [65]. BCD can be used for both convex and non-convex optimization problems. Global optimality and convergence has been mathematically proven for convex and some non-convex optimization problems that have certain mathematical properties, but will not be described here.

The algorithm cycles through the different sets of variables that are fixed. At each iteration , there is a overall solution x^k where k is the iteration number. At each iteration, one set of variables i_k to be fixed is selected. The problem is solved with x_{i_k} fixed (i.e. using the previous/initial solution) while the variables that are not fixed $x_{\neq i_k}$ are used to minimize the objective function f. Once the problem is solved, the overall solution x^k is updated using the following logic: the variables in x^k which were not fixed take on the new solution while the variables that were fixed still retain the previous solution x^{k-1}.

There are many different ways in choosing which variables to fix or how to cycle through them. Shi et. al gives a summary of the pros and cons of different ways to select which variable set is fixed at each iteration [65]. This method has also found success in solving large scale problems and is also used in successive approximation [61].

2.1.3 Applications to Aerospace Problems

Vehicle Design

Hoburg [39] and Kirschen [43] demonstrated the ability of Geometric Programming and Signomial Programming in solving simple aircraft design problems. Kirschen et al. showed that a commercial aircraft design problem could be formed and solved as SPs through the above formulation [42]. This formulation allows a passenger airliner design problem to be solved with similar fidelity in almost an order of magnitude decrease in computational time [72].

Many techniques for formulating design problems into GPs and SPs exist. GP/SP problems require all variables to be positive, which is not a problem for engineering design problems, but can be problematic for mission simulations (ex. non-positive states such as a dive and climb as mentioned earlier). Because both GPs and SPs require posynomial-containing constraints to be expressed as inequalities, some care is needed to determine the proper direction of the inequality sign. The concept of "pressure" is where the designer attempts to predict (through intuition) where the optimizer will send the design variable. Using this concept, equality constraints can be formulated as inequality constraints which is overall a less restrictive constraint. For example, drag can be defined as $C_D = C_{D_0} + C_{D_{wave}} + C_{D_i}$. If the optimization problem being solved is one where C_D is minimized, the drag expression can be relaxed into a compatible posynomial inequality $C_D \geq C_{D_0} + C_{D_{wave}} + C_{D_i}$ because we know the optimizer will continue to push the value of C_D downwards. This formulation forces the optimizer to minimize the components $C_{D_0}, C_{D_{wave}}$, and C_{D_i} because C_D cannot go further lower without the sum of the three components decreasing. In the case of expressions that cannot be expressed as GP/SP-compatible constraints such as trigonometric and logarithmic functions, Taylor approximations and fitting techniques [38] can be utilized over a defined range of design variables.

Mission Design

Optimal control problems where control inputs $u(t)$ must be computed are nonlinear programming problems due to a combination of nonlinear dynamics, objective functions, and constraints. However, recently in a push to develop onboard real-time guidance and control systems, convex optimization has been utilized to reach lower computation times and convergence guarantees that online guidance and navigation systems require. Liu et al. summarizes the development of convex optimization techniques within the context of guidance, navigation, and control [47]. Successive convex optimization (SCO) is used in all cases. SCO works by approximating the trajectory optimization problem into a Second-Order Cone Program (SOCP) which can be solved quickly. The successive part comes in once the approximate SOCP is solved, the original problem is then re-approximated using the previous SOCP solution and solved again. This process repeats until a certain tolerance is met. Szmuk and Açikmeşe used this formulation to solve a 6-DOF planetary landing problem similar to the SpaceX rocket landings [67]. Liu et al. implemented a similar method for hypersonic reentry vehicles, and compared it to standard trajectory optimization tools such as GPOPS which uses the Gauss Pseudospectral method and in these comparisons the SCO methodology was consistently faster[46].

The most common methodology to convert a trajectory optimization problem into a convex problem is through the use of a successive convexification framework. Algorithm 1 describes the generalized framework to approximate the original optimal control problem into a Second Order Cone Program (SOCP), a convex optimization problem represented as Ψ, through linearization and discretization. The SOCP approximate problem is then solved, and the solution is used to linearize and discretize the problem again to create a closer approximation to the original problem. This iteration process is repeated until the SOCP problem solution reaches a user-specified tolerance that is determined through comparison to its previous solution. The superscript given to the approximate problem Ψ and its solution z in Algorithm 1 represents the iteration number. This process is advantageous in that nonlinear dynamics as well as nonlinear and non-constant states are made into convex relations or "convexified".

Algorithm 1 Successive Convexification General Process

```
Result: Optimal Trajectory and Control
Linearize and discretize problem
 Generate initial trajectory $z^{(0)}$
 Form SOCP problem $\Psi^{(0)}$ using $z^{(0)}$
 $k = 1$
 Solve $\Psi^{(0)}$ and produce $z^{(k)}$
 while $max\ |z^{(k)} - z^{k-1}| \geq \epsilon$ do
     Form SOCP problem $\Psi^{(k)}$ using $z^{(k-1)}$
      Solve $\Psi^{(k)}$ and produce $z^{(k)}$
end
```

Furthermore, controls and non-convex constraints can be approximated utilizing a similar concept such as "pressure" that was used in vehicle design. In many cases, proofs can be generated that certain equality constraints for controls can be relaxed into inequality constraints as well as clever trigonometric relations relating to the control term can be made into convex constraints. Liu et. al provides a comprehensive overview of some of those techniques in [47].

Integrating the two problems

Despite there being convex optimization formulations of the vehicle and mission design problems separately, there is no obvious way of combining them simply under convex optimization. Fundamentally, GP/SPs and SOCPs look different in different spaces. Sequential optimization is not possible since GP/SP and SOCPs cannot be combined under a single convex optimization problem. Converting a mission optimization problem into a GP/SP is possible, but many assumptions have to be baked in and the fact that state information (such as velocity or acceleration) cannot go negative severely limits the problems that can be solved. The idea of setting certain variables constant to find a convex optimization problem may be potentially useful if we consider the entire integrated problem as a single problem with the blocks being the sub-problems; however multi-convex programming cannot be used directly since the optimal control problem is not a convex problem itself. In the same vein, Block Coordinate Descent cannot be used because the problem is not multi-convex. Therefore, a new approach in combining these convex optimization problems is needed. The ideas introduced in multi convex programming and sequential optimization can be used to integrate the problems.

2.2 Convex Optimization Approaches for Integrated Vehicle/Mission Design

Non-convex optimization problems can be solved using convex optimization; however, our goal is to use some of the convex optimization formulations that were used to solve the vehicle and mission design problems individually and somehow connect them together. However, it is clear that multi-convex programming and sequential convex optimization are not flexible enough to do so. Therefore, new methodologies are developed to allow the connection of different sequential convex optimization problems through the MDO architecture and block coordinate descent mindset.

2.2.1 Sequential Convex Optimization Architectures

The previous section demonstrates that sub problems for the simultaneous vehicle and mission problem can be solved using convex optimization. However, no one except for York has looked at solving the simultaneous problem together using convex optimization, and York's implementation is very limited for simplified mission profiles. Therefore, the key question here becomes: can we utilize the above introduced material to solve a more complex joint vehicle and mission problem? The problem is unique in several ways:

- Each problem may use different convex forms. For example, the vehicle design uses GP/SP, mission design may use SOCP.
- Each problem may use a different solver
- Direct coupling (i.e. using the same variables) is not always possible without breaking convex optimization restrictions on constraints and objective functions
- Different subspaces are being looked at. For example, in combining an GP/SP with an SOCP can be troublesome because an SOCP is $\mathbb{R}^n$, while GP/SP is $\mathbb{R}^n_{++}$

Therefore, we seek a problem solving methodology that can overcome the shortcomings of sequential convex optimization and MCPs. We want to use the concepts and ideas developed in the previous section and mix-and-match as we please to fit our problem.

For clarity, the overall notation to describe the problem we are trying to solve is introduced in Table 2.1. The superscripts in parentheses indicate which sub-problem the variable, parameter, and/or function belong to.

Using the defined notation, the problem we are attempting to solved can be formulated into the following:

Notation	Meaning
$x^{(j)}$	variables for sub-problem j
$f^{(j)}$	objective function for sub-problem j
$g^{(j)}$	inequality constraints for sub-problem j
$h^{(j)}$	equality constraints for sub-problem j
m	Total number of inequality constraints for the overall problem
n	Number of equality constraints in the overall problem
k	Total number of sub-problems
$m^{(j)}$	Total number of inequality constraints for sub-problem j
$n^{(j)}$	Number of equality constraints for sub-problem j
J	objective function for the overall problem

Table 2.1: Notation for convex coupled architectures

$$\begin{aligned} \text{minimize} \quad & J(f^{(j)}(x)) & & j = 1, \dots, k \\ \text{subject to} \quad & g_i^{(j)}(x) \leq 0 & i = 1, \dots, m^{(j)} \; & j = 1, \dots, k \\ & h_i^{(j)}(x) = 0 & i = 1, \dots, n^{(j)} \; & j = 1, \dots, k \end{aligned} \tag{2.16}$$

To solve this problem, the concept of convex architectures is proposed as a method to integrate the methodologies and solve the overall problem. Convex architectures are the decomposition of the overall non-convex problem into smaller problems that are either solvable through convex optimization or through approximations such as sequential convex optimization or multi-convex programming. We refer to them as architectures to highlight the very additive/modular nature of our attempts at using convex optimization to solve this problem. This is bringing the MDO architecture mindset of optimization into convex optimization. This is an original contribution of this thesis.

These architectures are necessary for us to utilize convex optimization for these types of problems because neither sequential convex optimization nor multi-convex programming can be used to solve these kinds of problems. For example, if we were to combine SP and SOCP problems, how would we do that? Sequential convex optimization has only been used to solve single convex optimization type problems.

Multi-convex programming is not general enough to analyze sequential convex optimization problems. These mix-and-match architectures are hybrids of these two different ideas and allow us to blend these concepts from both.

We propose three different competing architectures that allow us to combine those different ideas. We refer to them as alternating, integrated, and combined architectures. Each architecture has different levels of integration between the sub-problems. The following subsections introduce the three architectures in detail.

2.2.2 Alternating

The alternating architecture is the simplest of them all: we solve each sub-problem fully and then move to the other sub-problems. Algorithm 2 describes the architecture generally where Δ is the sub-problem tolerance, δ is the integrated problem tolerance, and $\epsilon^{(j)}$ is the required tolerance of sub-problem j. Figure 2-1 shows a two sub-problem implementation of the alternating architecture.

Algorithm 2 Alternating Architecture

```
Set convergence criteria $\epsilon$
Initialize $J, J_i, \delta$
**while** $\delta \geq \epsilon$ **do**
    **for** $j = 0, j \leq k$ **do**
        Initialize $\Delta, f_{prev}^{(j)}$
        **while** $\Delta^{(j)} \geq \epsilon^{(j)}$ **do**
            **Transfer** solution from other sub-problems into current sub-problem
            **Approximate** sub-problem $j$ as a convex optimization problem.
            **Solve** sub-problem $j$ for $f^{(j)}$
            $\Delta = f^{(j)} - f_{prev}^{(j)}$
            $f_{prev}^{(j)} = f^{(j)}$
        **end**
    **end**
    Calculate $J$
    $\delta = J$ - $J_i$
    $J_i = J$
**end**
```

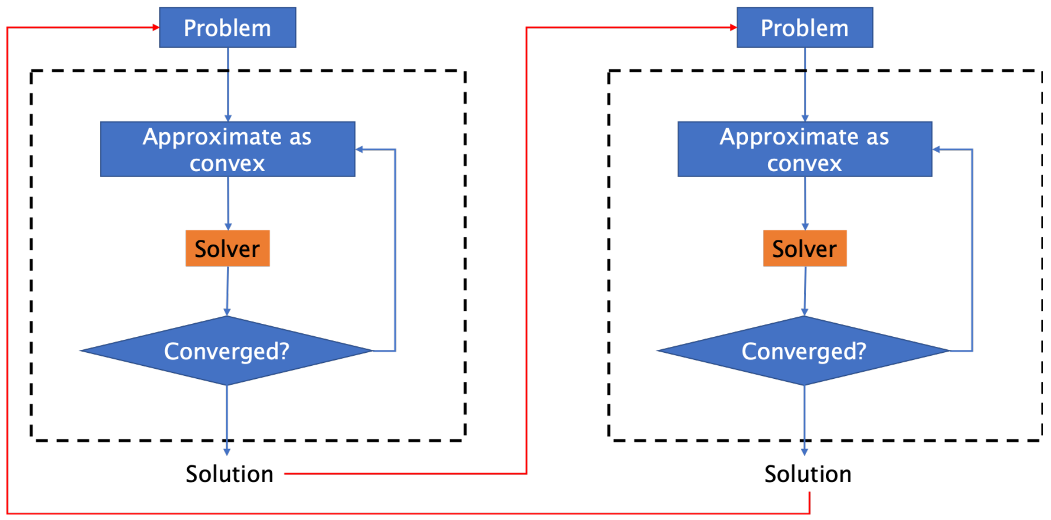


Figure 2-1: Two sub-problem example of alternating architecture schematic

This trivial architecture however, does not take advantage of convex optimization. In fact, one can replace each of the approximation-solve-convergence cycles with any analysis/optimization routine. Despite this, this is still useful for problems where an approximation is not necessary i.e. the sub-problems is a convex optimization. For example, you can have a GP and a LP in the two sub-problem architectures, and it is possible to get fast convergence and iterations because well-posed GPs and LPs can be solved easily. This can also be viewed as a more general form of Block Coordinate Descent, except it takes into account cases where a direct transformation to a convex optimization problem is not possible.

2.2.3 Combined

The combined architecture attempts to model the entire overall problem into one single convex optimization problem. At the core, each sub-problem is a set of constraints and objective functions, and combining them simply results in a larger set of constraints. In this case, we are accounting for the case where the sub-problems can be combined into one problem. This means for example that the overall design vector is stacked between the design and mission parts of the problem (with or without overlap). There are two conditions for when this can happen:

- Both use the same type of convex optimization problem and can be combined without breaking the convex optimization rules
- The least restrictive convex optimization problem used to solve a sub-problem can take the most restrictive convex optimization case.

The first condition is trivial; however, the second condition requires some thought. This requires the user to think carefully about problem formulation: each problem has specific conditions in the objective and constraints that can be taken. In addition, variable subspaces should also be taken into account. For example, if we were to attempt to combine a GP with an LP, we have to ensure that the LP is only looking the $\mathbb{R}_{++}$ space, not just the R space. Algorithm 3 describes the general idea. Figure 2-2 shows a two sub-problem implementation of the combined architecture.

Algorithm 3 Combined Architecture

```
Set convergence criteria $\epsilon$
Initialize $J, J_{prev}, \delta$
while $\delta \geq \epsilon$ do
    Approximate sub-problem $j$ as a convex optimization problem.
    Combine All sub-problems into one single problem.
    Solve the overall problem for $J$
    $\delta = J$ - $J_{prev}$
    $J_{prev} = J$
end
```

This architecture is the hardest to formulate because of those two restrictions. However, if possible, this formulation is the simplest as Algorithm 3 is really just Sequential Convex Optimization with the additional step of combining the different sub-problems together. Also, the combining and approximation steps can be interchanged as necessary. It is very similar to the All-At-Once (AAO) architecture as described in Martins [51]; however, applied to convex optimization.

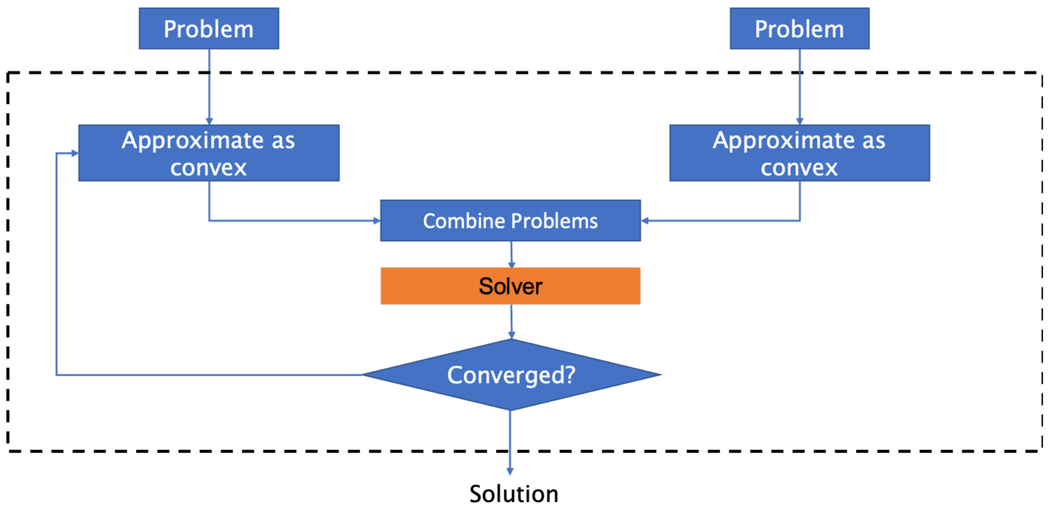


Figure 2-2: Two sub-problem example of combined architecture schematic

2.2.4 Integrated

The previous two architectures are on opposite ends of the spectrum in their ability to combine two or more sub-problems to be solved using some form of convex optimization-based methodology. Those two architectures are trivial and obvious especially looking at the problem from an MDO architecture perspective; however, we want to incorporate some coupling between the sub-problems that use convex optimization. The Combined architecture is not always feasible, and the alternating architecture does not offer much if any improvement.

The integrated architecture is a novel combination of sequential convex optimization and multi-convex programming. The basic idea of the integrated architecture is directly applying the sequential convex optimization problem to the entire problem 2.16 while incorporating the ”block” idea of multi-convex programming and block coordinate descent. We take advantage of the fact that we know how to solve each sub problem individually. Rather than wasting iterations on solving for information we know will change, we use the approximate solution of one sub-problem to solve the other approximate sub-problem and so on. We repeat this process until the convergence criteria for each sub-problem are met. This is a different requirement

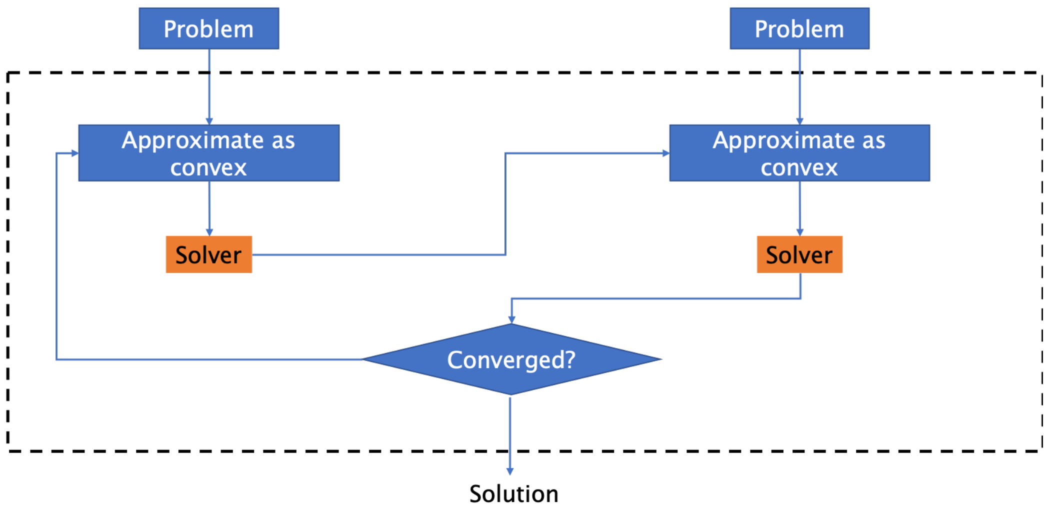


Figure 2-3: Two sub-problem example of integrated architecture schematic

compared to the alternating architecture where the convergence criteria involved the overall problem objective function. Here, we are assuming that **once all the sub-problems have converged, the solution for the overall problem is found**. In other words, we simultaneously (sequentially or in a specific order) solve each sub-problem while solving the overall problem. Algorithm 6 gives the outline of the framework. Figure 2-3 shows a two sub-problem example implementation of the integrated architecture.

Algorithm 4 Integrated Architecture

```
Set convergence criteria $\epsilon^{(j)}$ for $j = 1, ..., k$
Initialize $\delta^{(j)}$, $f^{(j)}_{prev}$
**while** *Any* $\delta^{(j)} \geq \epsilon^{(j)}$ **do**
    **for** $j = 0, j \leq k$ **do**
        **Transfer** solution from other sub-problems into current sub-problem
        **Approximate** subproblem $j$ as a convex optimization problem.
        **Solve** subproblem $j$ for $f^{(j)}$
        $\delta^{(j)} = f^{(j)} - f^{(j)}_{prev}$
        $f^{(j)}_{prev} = f^{(j)}$
    **end**
**end**
```

The intention here is to save time and allow each sub-problem to get the most accurate approximation it can possibly get from the other sub-problems and not waste time on information that can change. Although that introduces the possibility of instability due to the constantly changing other sub-problem solutions, the algorithm is not wasting time attempting to converge using approximate information. This will allow the algorithm to converge quickly while ensuring that all the sub-problems are being solved.

2.3 Discussion

2.3.1 Similar Work

These architectures (except for the alternating architecture) are different than standard MDO architectures as those architectures assume that each model/sub-problem is a black box the approach developed in this thesis does not and allows the optimization process to be integrated. However, with that being said, any MDO architecture that involves sub-problems to be optimized could replace those with convex optimization or sequential optimization such as Collaborative Optimization [15] or BLISS [66]. What separates these architectures from other MDO architectures is the fact that these are specifically designed to allow for further integration of the convex optimization solve processes. The goal is to ideally solve this problem all-at-once; however, convex optimization formulations prevent that. Even partial integration will allow the entire process to converge faster and not waste any time on converging using wrong approximate solutions.

2.3.2 Intuition

The core of this idea is that block coordinate descent is combined with sequential optimization. Rather than each "block" being a convex optimization problem, each block is a non-convex optimization that can be solved using convex optimization.

Each block is therefore a sequential optimization problem that at best converges to a local optimum.

Another way to think of this is the fact that there are two types of approximations that can be utilized: approximation of the problem through the choosing of the blocks and the approximation of the problem through the sequential optimization solving. For the alternating and integrated architecture, both the multi-convex approximation and sequential optimization approximation are used: they differ in how mixed the approximations are. The combined architecture is only using the sequential optimization approximation. In our specific vehicle and mission case, our two blocks are the vehicle(s) and mission(s) that are being optimized. There may be other ways to determine whether other multi-convex blocks exist, but in this thesis, it is assumed that only the vehicle and mission blocks are used.

A measurement of how close an approximation of the convex optimization problem is to the area of the design space being explored is the outer loop tolerance. For the integrated architecture, it is the sub-problem tolerances. For the combined architecture, it is the overall tolerance between the different iteration solutions. For the alternating architecture, it is the tolerance of the overall system objective function. These tolerances are not measures of optimality, rather measures of how closely the approximated problem is to the original problem.

2.3.3 Optimality

At best, only local optimality can be expected. This thesis does not propose any mathematical proofs of optimality for the overall architecture; however, the fact that we are solving a convex optimization problem means all is not lost as local optimality can also be expected. Despite the fact that all convex optimization problems only have one optimum if the problem is feasible, the convex optimization problem that is being solved is a local approximation of the entire problem, so therefore, only local optimality can be expected. One could attempt to check the Karush-Kuhn-Tucker (KKT) conditions; however, by design, all convex optimization solvers solve the KKT conditions in some way or do not converge until the KKT conditions are met to some set tolerance. Therefore, the KKT conditions will be automatically met for all sub-problems individually.

2.3.4 Properties

Each architecture has strengths and weaknesses, as such problem formulation from the convex optimization perspective should be done with care. Each architecture can also theoretically take as many sub-problems as necessary.

Some key characteristics of each architecture are as follows:

- Convex Mixing: No assumption for what type of convex optimization problem each sub-problem represents is made in any of the architectures. This means that as long as the sub-problem can be approximated by a convex optimization, these architectures could be used. Any convex optimization problem can be added and used in the architecture.

- Transition: No assumption is made on the way the approximation is made. Therefore, these architectures allow for some data reformatting/transition to allow the sub-problem to be approximated in the most accurate way. This is very advantageous, especially in the vehicle and mission design problem where both the vehicle and mission optimizations are asking for different parameters that each side can compute but did not necessarily do so during the optimization.

- Use of Trust Regions: For the sequential convex optimization portion, the use of the trust region is optional and dependent on the convex optimization problem being solved. The architecture is not reliant on any trust region being used and in all architectures, the trust region methodologies or lack of them is only dependent on the sub-problem begin solved

- Flexibility in the architectures: These three architectures cover a wide spectrum of different combinations of convex optimization programs that can be combined. Depending on the level of integration possible, these architectures offer some flexibility.

2.3.5 Sensitivities

Parameter sensitivities are also very easy to calculate since each sub-problem produces the sensitivities as a by-product of the solve process. Some additional post processing is needed as each sub-problem's objective function is only a part of the overall objective function; however, the bonus of having sensitivities in the form of the dual solution is important.

However, variable sensitivities will require additional work to determine. Variable sensitivities of the form df/dx are important especially in engineering design where a designer may be interested in determining how much the objective function will change if a design variable is slightly perturbed. Unfortunately, convex optimization solvers do not produce that information as part of the solve processes, and unlike traditional multidisciplinary models, constraints cannot be "run through" to carry out gradient calculation. Therefore, static multidisciplinary models must be created to carry out gradient calculations.

2.3.6 Limitations

This methodology only works if both analysis and optimization processes can be formulated as sequential convex optimization problems or convex optimization problems. However, not all non-convex optimization problems can be approximated as such. It may be possible to develop other aerospace related analysis codes into convex optimization, and if that occurs, sequential convex optimization similar to the process developed in this thesis could be used to combine the different analyses. Combining the analysis whether through the different architectures may allow for a faster convergence to a similar or more optimal design point. Other architectures may exist depending on how the individual analysis/optimization convex formulations are created.

In addition, these architectures are very basic in that there is no mechanism for checking whether the solution of a sub-problem will result in another a sub-problem becoming infeasible. There are also no robustness measures or protections to prevent that in case that does occur. However, these can potentially be mitigated by measures such as adding slack terms to the sub-problems, similar to how Block Coordinate Descent modifies the objective function as explained in the Future Work section.

2.3.7 Choosing the appropriate architecture

The choice of which architecture is best for the problem at hand primarily depends on what the sub-problems forms are. Figure 2-4 shows qualitatively the level of integration between the different architectures. With the development of the integrated architecture, most if not all convex optimization problems can use the integrated architecture. If the problem can be combined into one single optimization problem, then the combined architecture should be used.

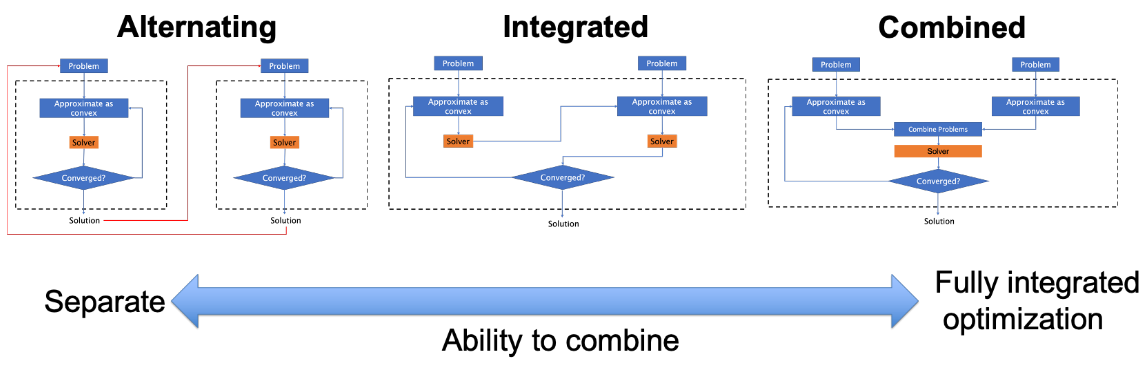


Figure 2-4: Level of integration

One thing to note is that the separation of the two problems must be thought out carefully. By dividing the original problem into two or more sub-problems, the objective function for each sub-problem must be aligned so that the sub-problem optimizations are pushing the solution towards the same direction. Another potential issue is that each sub-problem should ensure that the variables and parameters are properly split; in other words, a variable in a sub-problem should not be a variable in another sub-problem. This will ensure that there is consistency in the design variable iterations when the outer loop iterations are occurring. As with all MDO problems, problem formulation is key to having a successful MDO optimization, and it is no different with convex optimization architectures.

2.4 Example Problem

An example problem that we can try to solve by hand to demonstrate the architectures in a more concrete way is the following:

$$\begin{aligned} \text{minimize} \quad & f = 100 - x_1^2 x_2^2 + 10x_1 - 15x_2 \\ \text{subject to} \quad & x_1 \geq 1 \\ & x_2 \geq 1 \\ & x_1 + x_2 \leq 5 \end{aligned} \tag{2.17}$$

This is a non-convex problem. The product between the squares of x_1 and x_2 as well as the negative coefficient in front make this not a quadratic program. Therefore, it is not possible to solve this problem directly using convex optimization. By inspection, it is obvious that the solution is (1,4) because the x_2 term must be maximized as much as possible due to the coefficients of the terms containing x_2 both being negative. Figure 2-5 shows the visualization of Problem 2.4.3.

This problem can be split into two sub-problems: Problem 2.18 and 2.19. Problem 2.18 is formed with $x_2 = c_2$, and Problem 2.19 is formed with $x_1 = c_1$: both c_1, c_2 are constants. However, even with this approximation, the problem is not convex due to the negative coefficient in the squared term of the objective function.

$$\begin{aligned} \text{minimize} \quad & f_1(x_1, c_2) = 100 - 15c_2 - c_2^2 x_1^2 + 10x_1 \\ \text{subject to} \quad & x_1 \geq 1 \\ & x_1 + c_2 \leq 5 \end{aligned} \tag{2.18}$$

$$\begin{aligned} \text{minimize} \quad & f_2(x_2, c_1) = 100 + 10c_1 - c_1^2 x_2^2 - 15x_2 \\ \text{subject to} \quad & x_2 \geq 1 \\ & c_1 + x_2 \leq 5 \end{aligned} \tag{2.19}$$

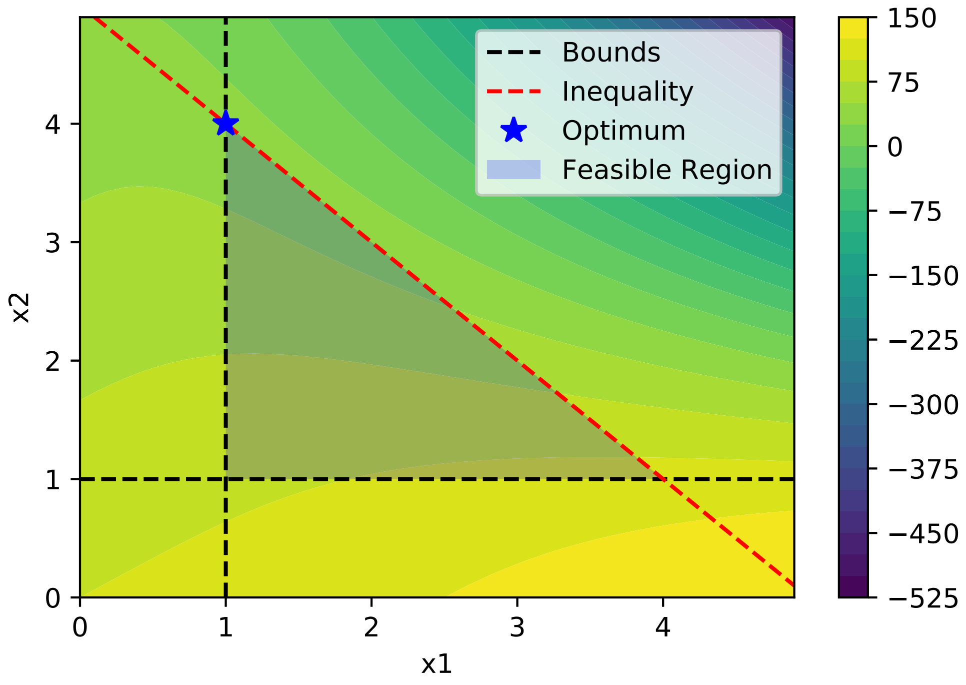


Figure 2-5: Example Problem Contour

Therefore, this problem is solved with the introduced architectures:

2.4.1 Alternating

For the alternating architecture, Problem 2.18 and 2.19 are each solved using sequential convex optimization through linearization. Once one sub-problem is solved, the solution is passed to the other sub-problem ($x_1^{(k)} = c_1, x_2^{(k)} = c_2$ in each problem where (k) denotes the previous iteration). Once each sub-problem is solved fully once, the original problem 2.4.3 objective function f is calculated, and compared with the previous iteration objective value $f^{(k)}$. This process repeats itself until the original objective function has converged for some tolerance ϵ, in other words until $|f - f^{(k)}| \leq \epsilon$. Figure 2-6 shows the data and process flow.

The problem is linearized by approximating the non-convex objective function with the convex part of the second order Taylor expansion or Equation 2.20.

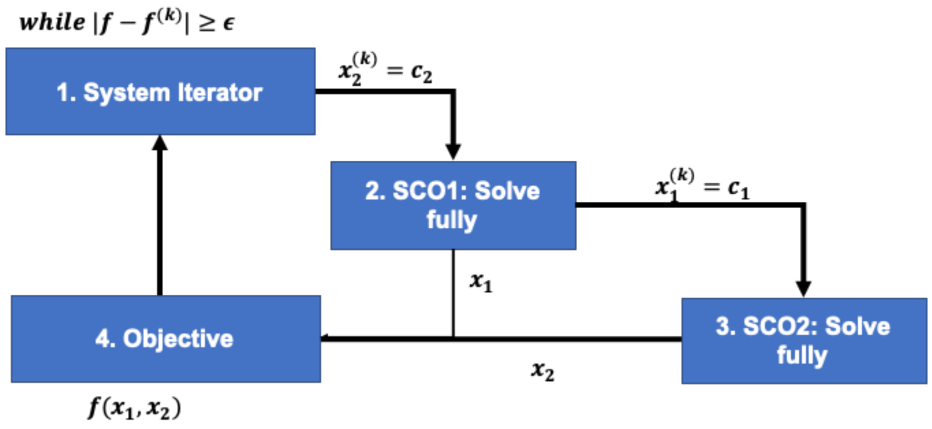


Figure 2-6: Example Problem: Alternating Architecture Information and Process Flow

$$\hat{f}(x) \approx f(x^{(k)}) + (\nabla f(x^{(k)}))^T (x - x^{(k)})^T + \frac{1}{2}(x - x^{(k)})^T [\nabla^2 f(x^{(k)})]_+ (x - x^{(k)}) \quad (2.20)$$

Therefore, the objective function becomes the following Equation 2.21.

$$\hat{f}(x) \approx f(x^{(k)}) + (Px^{(k)} + q)^T (x - x^{(k)})^T + \frac{1}{2}(x - x^{(k)})^T P_+ (x - x^{(k)}) \quad (2.21)$$

where P is the coefficient in front of the quadratic and q is the coefficient in front of the variable. P_+ the semi-definite part of P, and in this case, $P_+ = 0$ since this is a single variable problem. In multi-variable sub-problems, the spectral decomposition of $\nabla^2 f(x^{(k)}) = U\Lambda U^T$ will need to be carried out, and $P_+ = U[\Lambda]_+ U^T$ where the $[\Lambda]_+$ denotes that all negative eigenvalues are set to zero.

A trust region is also needed because this approximation is only valid locally. Therefore, a simple trust region constraint of Equation 2.22 is added.

$$x - x^{(k)} \leq \rho = 0.2 \quad (2.22)$$

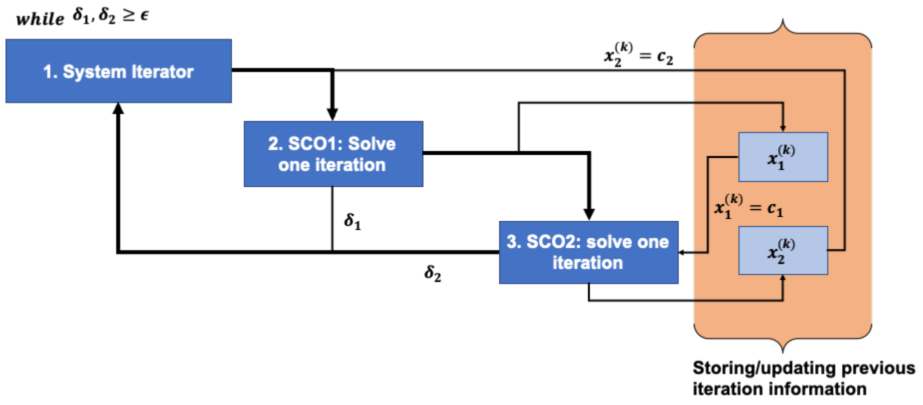
With the added transformations, the sub-problems 2.18 and 2.19 become the following Equation 2.23 and 2.24

$$\begin{aligned} \text{minimize} \quad & \hat{f}_1(x_1, c_2^{(k)}) = f_1^{(k)}(x_1^{(k)}, c_2^{(k)}) + (-(c_2^{(k)})^2 x_1^{(k)} + 10)^T (x_1 - x_1^{(k)})^T \\ \text{subject to} \quad & x_1 \geq 1 \\ & x_1 + c_2^{(k)} \leq 5 \\ & x_1 - x_1^{(k)} \leq \rho \end{aligned} \tag{2.23}$$

$$\begin{aligned} \text{minimize} \quad & \hat{f}_2(x_2, c_1^{(k)}) = f_2(x_2^{(k)}, c_1^{(k)}) + (-(c_1^{(k)})^2 x_2^{(k)} - 15)^T (x_2 - x_2^{(k)})^T \\ \text{subject to} \quad & x_2 \geq 1 \\ & c_1^{(k)} + x_2 \leq 5 \\ & x_2 - x_2^{(k)} \leq \rho \end{aligned} \tag{2.24}$$

2.4.2 Integrated

The integrated architecture follows essentially the exact same process as the alternating architecture with the exception that only one iteration of the sequential convex optimization is carried out to approximately solve the problem. The approximate problem is then passed to the other sub-problem: ($x_1^{(k)} = c_1, x_2^{(k)} = c_2$ in each problem where m denotes the overall iteration). The iteration ends when the tolerance for both sub-problems is met. Figure 2-7 shows the data flow and a more problem specific process flow. The integrated architecture requires information storage from previous iterations to continue the SCO solve.



while $\delta_1, \delta_2 \geq \epsilon$
1. System Iterator
2. SCO1: Solve one iteration
3. SCO2: solve one iteration
$x_2^{(k)} = c_2$
$x_1^{(k)}$
$x_1^{(k)} = c_1$
$x_2^{(k)}$
δ_1
δ_2
Storing/updating previous iteration information

Figure 2-7: Example Problem: Integrated Architecture Information and Process Flow

2.4.3 Combined

The combined architecture in this case is Problem being solved. This is possible with Signomial Programming: we can reformulate the original problem into a Signomial Program by creating a dummy variable z that pushes the Signomial-compatible objective function into the constraint as shown in Problem 2.25.

$$\begin{aligned} \text{minimize} \quad & z \\ \text{subject to} \quad & x_1 \geq 1 \\ & x_2 \geq 1 \\ & x_1 + x_2 \leq 5 \\ & 100 - x_1^2 x_2^2 + 10x_1 - 15x_2 \leq z \end{aligned} \tag{2.25}$$

2.4.4 Numerical Example Solve and Comparison

The example problem is implemented in Python using CVXPY [20] as the quadratic solver with the default OSQP quadratic solver for the alternating and integrated architectures. GPKit [17] was used to solve the combined architecture implementation. All 3 architectures converged; however, only the combined architecture converges to the minimum with the other architectures converging close to the optimum but not quite. Table 2.2 shows the solve statistics for the architectures. The combined architecture was the quickest, with the integrated architecture coming in next. Interestingly, the integrated architecture had the most number of iterations; however, it was still faster than the alternating architecture. This is due to the fact that each SCO solve is not fully solving the problem, and therefore resulting in additional iterations.

Parameter	Alternating	Integrated	Combined
Converged?	Yes	Yes	Yes
Iterations	5	14	7
Final Solution	(1.4, 3.6)	(1.4, 3.6)	(1,4)
Solve Time (s)	0.536	0.447	0.339
Objective Function	34.59	34.59	34
Required tolerance	1e-8	1e-8 (both)	1e-8

Table 2.2: Architecture Solve Statistics

Figure 2-9 shows the optimization path taken by each architecture. Both the alternating and integrated architecture take the same paths in Figure 2-9a and 2-9b. Both reach the correct x_1 value before diverging to the incorrect location. This is primarily due to the fact that our linearization heavily relies on the gradient (f), and at that point, the gradient slightly points toward the inequality bound rather than the actual optimum as shown in Figure 2-8. This leads to the approximation to wrongly assume that the minimum is in that direction rather than towards the actual optimum. This is expected as sequential convex optimization is after all a heuristic; and to avoid this, other convex approximations are needed. For the combined architecture however, it may seem that the combined architecture took no path at all as seen in Figure 2-9c; this is not true. The GP approximation of the objective was so accurate that the optimizer immediately went to a point extremely close to the actual optimum.

Figure 2-10 shows the specific values of the variables in each overall iteration for each architecture. For the combined architecture, the architecture immediately jumps very close to the optimum in Figure 2-10c, and the later iterations are only needed to meet the tolerance. Both the alternating (Figure 2-10a) and integrated (Figure 2-10b) architecture head directly towards the expected optimal variable values until

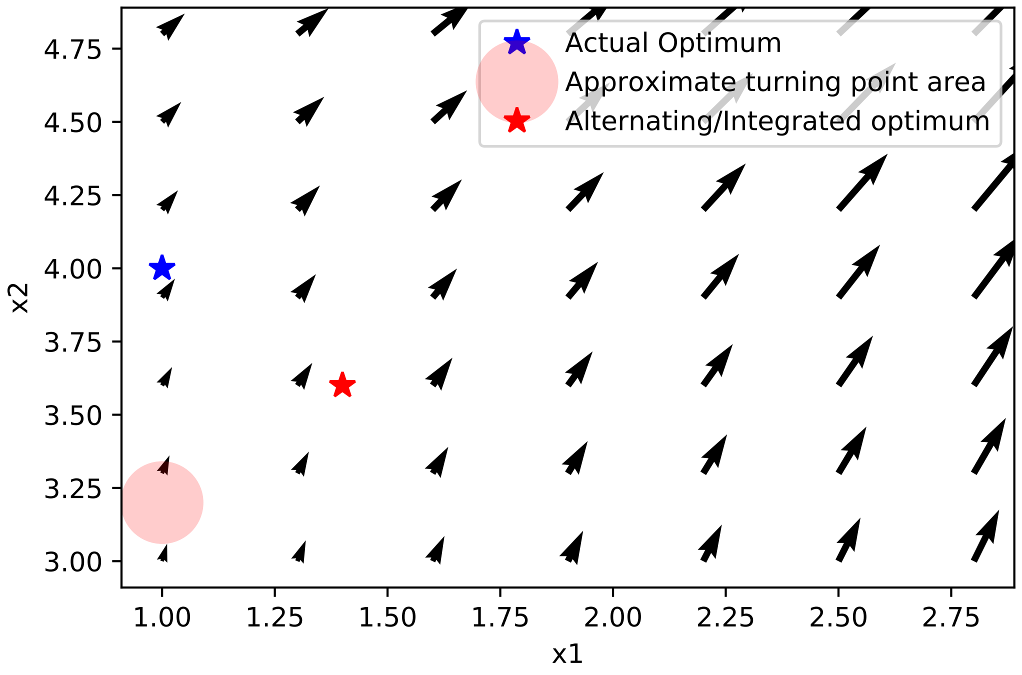


Figure 2-8: $-\nabla f$ Field near optimum

reaching the region where the x_2 component of the gradient begins to increase to the point that causes the linearization to change direction away from the optimum.

Figure 2-11 shows the original problem's objective function evolution for each architecture, and Figure 2-12 shows the tolerance for the system level iterations as the optimization progressed. Interestingly for both the alternating (Figure 2-11a and 2-12a) and the integrated architecture (Figure 2-11b and 2-12b), the convergence rate varies until the optimizer is close to the actual optimum; after it reaches what it thinks is the optimum, the tolerance decreases significantly and the optimization stops The same does not occur for the combined architecture and in this case, the SP. This is most likely due to the fact that the GP approximation is most likely still evolving, as the objective function cost are orders of magnitude higher despite the fact that the variables are already close to the optimum values.

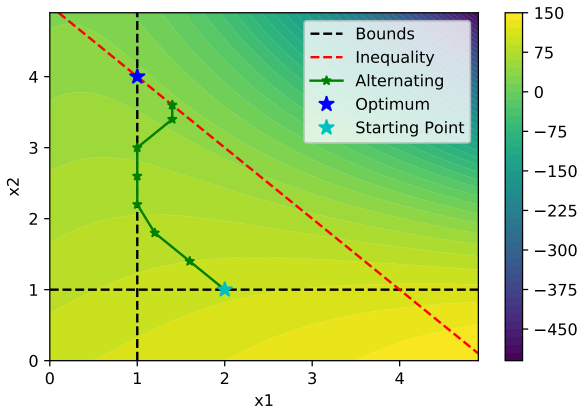
One key takeaway from this example problem is that the accuracy of the convex approximation matters significantly, not just in the accuracy of the numerical values but also the mathematical structure. This is highly apparent in the discrepancy between the objective function values during the optimization for each architecture. Despite the objective function cost being significantly higher, the GP approximation in the combined architecture is much better because it immediately leads to the actual optimum of the problem whereas the integrated and alternating, despite limited by the trust region, heavily relies on the gradient and converges to possibly a point that is close but not correct. The linearization was in this case not a great method to use to approximate the objective function as the gradient signifcanlty influenced the approximationand ultimately lead the optimizer astray. The trust region ρ in this case can be expanded or shrunk; however, this risks in the optimizer "wandering" into regions where the approximation is not valid. SPs do not use trust regions because the underlying structure is still maintained for this problem; and therefore, the optimizer is not limited to only exploring a certain region. Using other convex approximations methods to approximate non-convex parts of the problem may lead to further improvement of the optimization.

From this example, it is clear that the combined architecture is best in terms of optimality and speed. However, not all problems can be formulated into a combined architecture problem; therefore, the integrated and alternating architectures are needed. Despite not reaching the expected and correct optimum, there are different methods such as other approximation strategies such as particle methods that may allow for these architectures to improve and reach a solution close to the optimum.

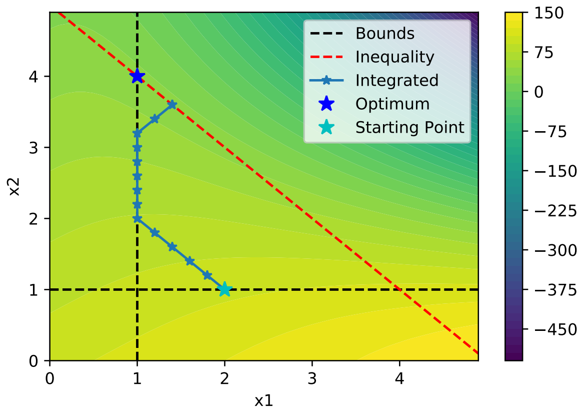
2.5 Chapter Summary

Convex optimization and methodologies to solve non-convex problems using convex optimization is introduced in this chapter. Sequential convex optimization architectures are introduced in the second section, and this is the primary contribution of the thesis. Some preliminary discussion on expected behavior, benefits, and limitations are discussed in the final section. Finally, an example problem is solved using each architecture, and the numerical behavior and results are discussed. The benefits and drawbacks of the different approaches are demonstrated.

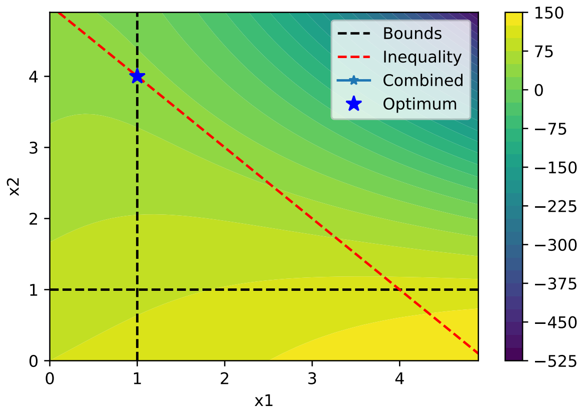
This approach of thinking about each sub-problem and combining it together through convex optimization is demonstrated with in the next two chapters through the design of a rocket and the design of a hydrogen-powered aircraft.



(a) Alternating

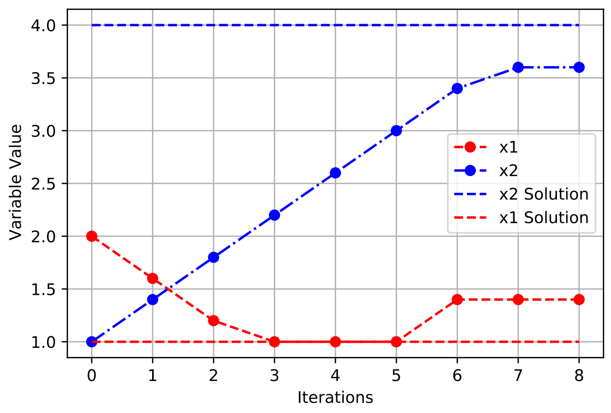


(b) Integrated

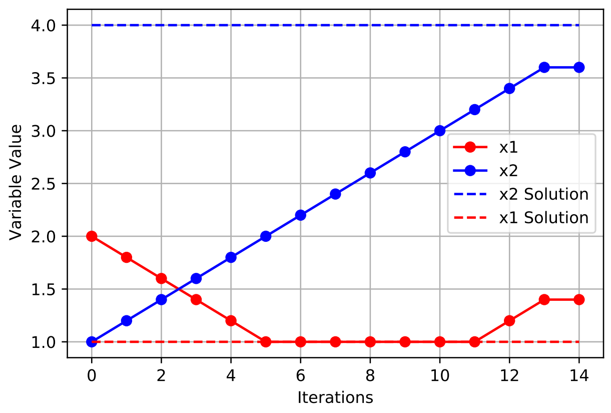


(c) Combined

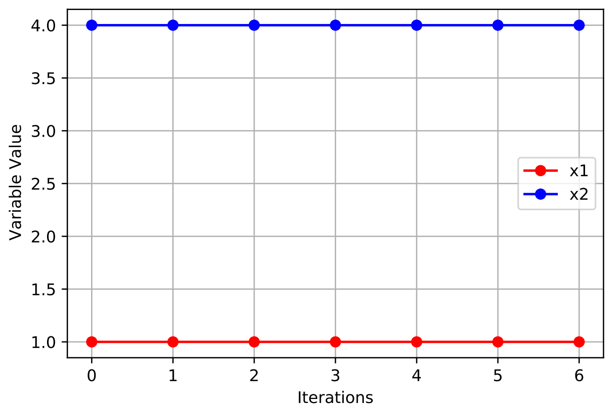
Figure 2-9: Example Problem Optimization Travel



(a) Alternating

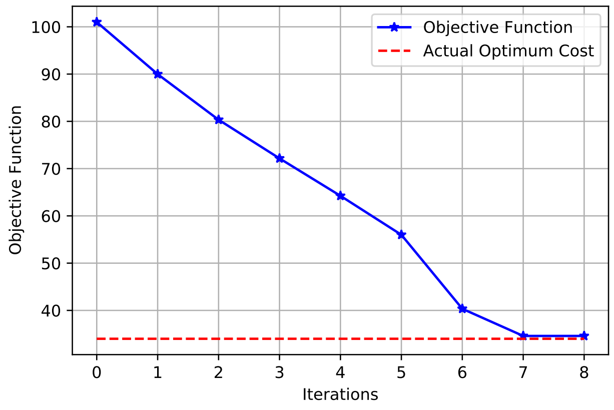


(b) Integrated

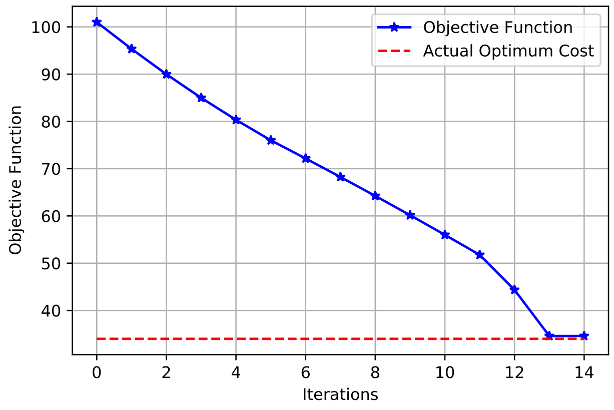


(c) Combined

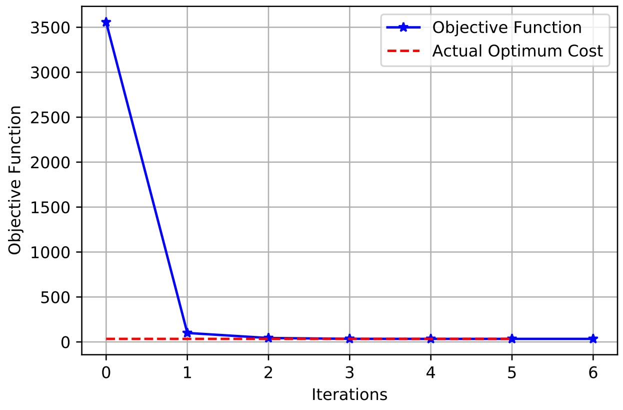
Figure 2-10: Example Problem Variable Change



(a) Alternating

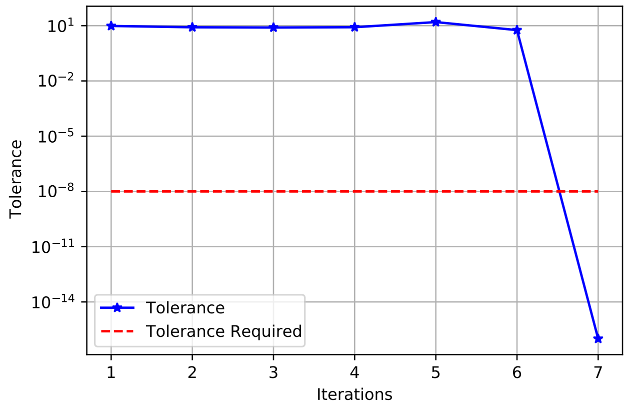


(b) Integrated

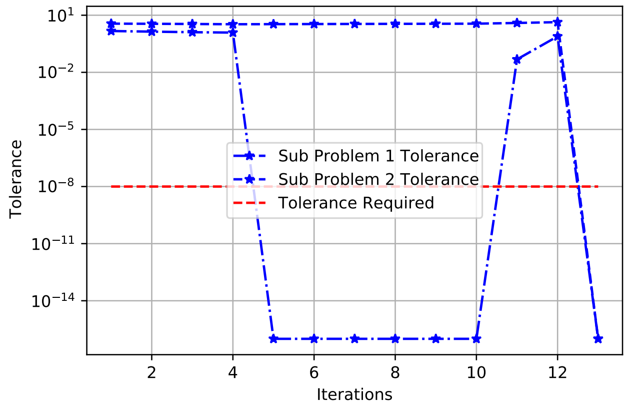


(c) Combined

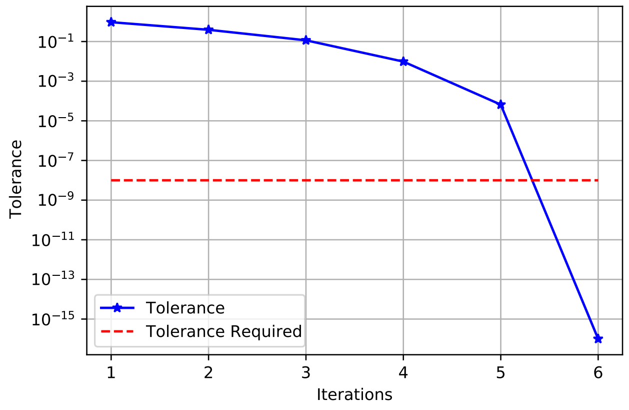
Figure 2-11: Example Problem Objective Function through Iterations



(a) Alternating



(b) Integrated



(c) Combined

Figure 2-12: Example Problem Tolerance

Chapter 3

Rocket Design Case

3.1 Problem Formulation

3.1.1 Case Study Context

The example scenario analyzed in this chapter is the design of a solid-fueled sounding rocket. The sounding rocket must reach a specified state within a specified time. A simplified visualization is shown in Figure 3-1. The objective of the sounding rocket problem is to design a vehicle that has the minimum initial mass to achieve its flight mission in terms of specified end condition. The trajectory itself is also subject to optimization. This design will require a compromise between the vehicle geometry, propulsion, and the trajectory that is flown.

The problem is formulated as a 1D trajectory problem combined with a vehicle design problem. This formulation is chosen due to the fact that both problems are well studied and understood and previous work can be used to verify the solution. A mathematical formulation of the problem is given in Problem (3.1). Geometry limits are specified to ensure the feasibility and manufacturability of the rocket design. The problem is formulated such that the vehicle design has influence only on W_{dry} and W_{fuel}, the vehicle dry mass and fuel mass, respectively. The separate problems

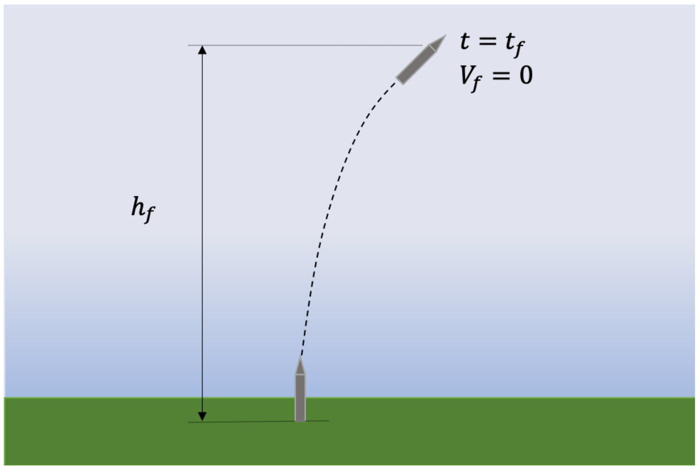


Figure 3-1: Sounding rocket design problem with prescribed final state at $t = t_f$

which are given in Problem (3.3) and (3.19) will focus on minimizing their individual contributions to W_{total}.

$$\begin{aligned} &\textbf{minimize} && W_{total} = W_{dry} + W_{fuel} \\ &\textbf{w.r.t.} && \text{Vehicle Design} \\ &&& \text{Trajectory Design} \\ &&& \text{Control History} \\ &\textbf{subject to} && \text{Mission: Beginning and end states, time of flight} \\ &&& \text{Geometry limits} \end{aligned} \tag{3.1}$$

For the vehicle design optimization, GPKit[17] is used to form the problem while MOSEK is used as the solver. GPKit is an MIT-developed Python library that allows GPs and SPs to be modeled. For the trajectory optimization, CVXPY[20] is used to form the problem while ECOS [23] is used as the solver. The code [56] used for trajectory optimization is an implementation of the SCvx algorithm [50]. This problem is a good example to test the different possible convex optimization architectures as the mission part is non-intuitive in that there are different phases of flight inherently built into the optimal solution. Furthermore and most importantly, solutions for both are well know and can be used to check the solution for each sub-problem.

3.1.2 Convex Problem Architecture

We can formulate the problem into the mathematical notation denoted in Chapter 2 in Problem 3.2 using the formulation in Equation 3.1. The subsequent notation is described in Table 3.1. With this problem formulation, the convex optimization architectures can be used to solve the problem.

$$\begin{array}{lll} \text{minimize} & J(f) = f^{(1)}(x^{(1)}) + f^{(2)}(x^{(2)}) & \\ \text{subject to} & g_i^{(j)}(x) \leq 0 & i = 1, \ldots, m^{(j)} \ j = 1, 2 \\ & h_i^{(j)}(x) = 0 & i = 1, \ldots, n^{(j)} \ j = 1, 2 \end{array} \tag{3.2}$$

Notation	Problem Variable/Parameter
$x^{(1)}$	vehicle design problem variables
$f^{(1)}$	W_{dry}
$g^{(1)}$	inequality constraints for vehicle design problem
$h^{(1)}$	equality constraints for vehicle design problem
$x^{(2)}$	trajectory design problem variables
$f^{(2)}$	W_{fuel}
$g^{(2)}$	inequality constraints for trajectory design problem
$h^{(2)}$	equality constraints for trajectory design problem
k	$= 2$
J	W_{total}

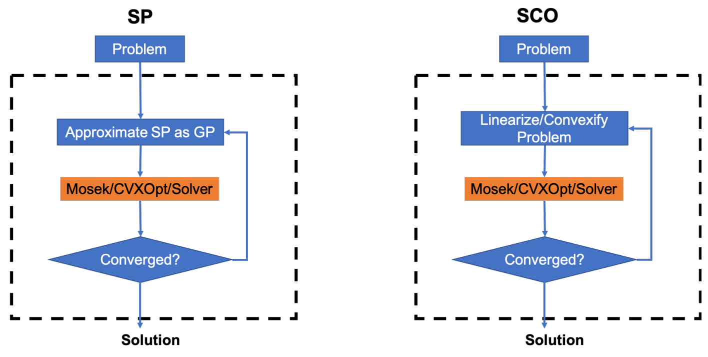
Table 3.1: Notation for convex architecture and the rocket design problem

The convex solve processes for vehicle design and trajectory optimization are very similar in structure. Figure 3-2 shows the general process outline for the vehicle design (GP/SP) and the trajectory optimization (SCO) processes. Both processes assume that the original problem is approximated as a convex design problem. Both processes use an interior point solver and converge to a solution when the approximate solution stays within a specified tolerance. There are differences between the two processes such as the fact that the trajectory optimization may require the use of trust regions as well as the addition of slack variables and constraint transformations [47]; however, those changes are limited to the forming of the approximation itself, not the general optimization process. Using these similarities, the two processes may be combined in a more coupled manner utilizing the architectures mentioned in the previous chapter rather than considering the two processes to be black boxes. Each architecture implementation is described briefly in the following and shown in Figure 3-2.

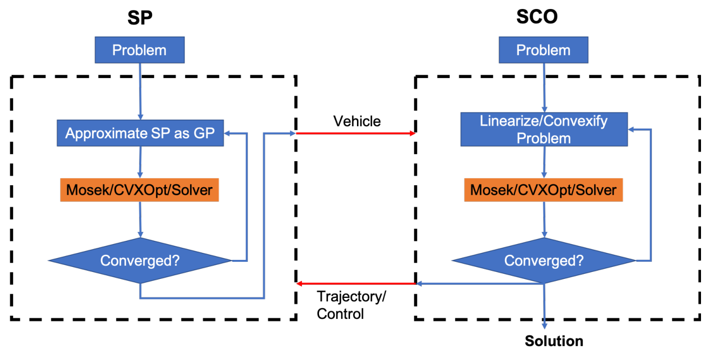
Alternating Architecture

The alternating architecture is the implementation of the current approach of designing a system when mission design (in this case, trajectory optimization) is needed as shown in Figure 3-2b. Either the trajectory optimization or vehicle design starts, and once one process finishes, it passes that solution to the other process and alternates between the two processes. The alternating optimization process means that the respective design variable (vehicle and trajectory) should stay constant during the complementary optimization processes; in other words, the vehicle design process uses a constant trajectory and vice versa. This means that all vehicle models and trajectories will need to be passed between the two processes, and both processes should be formulated such that each side can receive its input from the other side.

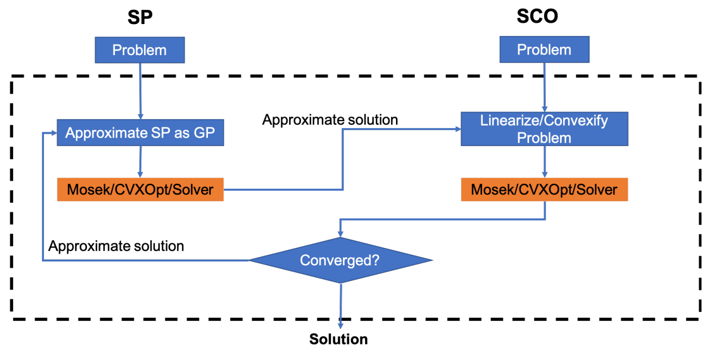
For this problem, this architecture is the simplest because it does not involve



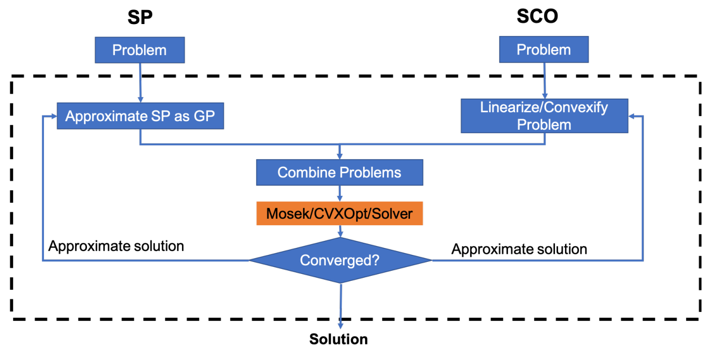
(a) Separated



(b) Alternating



(c) Integrated



(d) Combined

Figure 3-2: Conceptual summary of each architecture

changing the respective processes. With the use of convex optimization, this combined process could be much faster than past practices due to each individual optimization processes being faster. This architecture is equivalent to sequential optimization, and may not reach the true optimal solution due to the separated optimization process. The alternating architecture will be used as a reference for the other two architectures.

Integrated Architecture

The integrated architecture takes advantage of the similarities between the two optimization processes by combining the two convergence loops into one large loop as shown in Figure 3-2c. Rather than continue iterating within each optimization, the integrated architecture uses the approximate solution produced by the solver as input to the other portion of the problem. The overall optimization ends after both the vehicle and trajectory solutions reach a certain tolerance between iterations.

By combining the convergence loops, the intent is to allow each optimization process to have access to the most up-to-date information from the other process. The primary hypothesis of the integrated architecture is that combining the separate convergence loops into a single iteration loop will generate both an optimized vehicle and trajectory. This approximation-based technique to couple the two processes is similar to using the gradients to guide the optimizer towards the optimal solution. The integrated architecture allows each side to take into account the other processes direction of improvement as optimization unfolds.

Combined Architecture

The combined architecture brings the entire problem under a single successive convexification problem as shown in Figure 3-2d. The architecture works by setting up both problems under one unified convex optimization problem with disjointed variables (i.e. separate variables for each optimization, but executed under a single optimization run). The objective function would be the only part where both optimization processes interact outside of the approximation process as direct coupling (i.e. the declaration of a parameter as a variable in both processes) is not possible within the convex problem. Because both utilize the same structure of convergence, the idea is to save the number of times that the solver is called while using the approximate solution of either process to reach an optimal solution. Like the integrated architecture, the convergence is achieved when both the vehicle and trajectory solutions reach a certain tolerance between iterations. This architecture is a more computationally efficient version of the integrated architecture.

The key disadvantage of the combined architecture is that a direct coupling of the two problems into a single convex problem is challenging due to rules that the problem formulation must follow to maintain the characteristics of a convex optimization problem. The vehicle design problem is formulated in the log-space through the change of variables and the trajectory optimization is not. Since the products of log functions and other variables are not convex, the optimization processes cannot be connected. Furthermore, the trajectory optimization cannot be formed into a geometric problem since the state variables in the trajectory optimization problem can be negative and trigonometric functions/expressions are not GP-compatible. Therefore, this architecture is not implemented because it is not possible to formulate a convex approximation with current mathematical techniques.

3.1.3 Vehicle Design Formulation: Multidisciplinary Design of Body and Nose Configuration

The vehicle design sub-problem is modeled as a multi-point analysis problem. The optimization is set up to ensure that the vehicle can fly through the trajectory given by the trajectory optimizer. The trajectory is discretized at a number of flight points, which are called flight states. Each flight state contains flight information of the vehicle at a specific time such as velocity, altitudes, position, etc. The vehicle must be able to fly through all flight states as well as survive the loads encountered at each flight state. The vehicle optimization sub-problem is described in (3.3), where W_{dry} is the dry mass of the vehicle.

$$
\begin{aligned}
&\textbf{minimize} && f^{(1)} = W_{dry} \\
&\textbf{w.r.t.} && \text{Geometry} \\
& && \text{Structural design} \\
& && \text{Motor design} \\
&\textbf{subject to} && \text{Trajectory} \\
& && \text{Aerodynamics} \\
& && \text{Structures} \\
& && \text{Thermal} \\
& && \text{Propulsion} \\
& && \text{Weights} \\
& && \text{Geometry limits}
\end{aligned}
\tag{3.3}
$$

There are two types of variables that are set up in the GPkit model of the rocket: static and dynamic. Static variables are used to describe the vehicle properties that remain constant throughout the vehicle's trajectory, such as geometry and material thickness. Dynamic variables describe vehicle properties that vary during flight, such as the mass of fuel onboard. Furthermore, both subsonic and supersonic characteristics are being considered; therefore, the two aerodynamic models exist to account for both flight regimes. Constraints connect the different components within the problem as well as provide material and geometry limits across the different disciplines.

A conceptual vehicle configuration is shown in Figure 3-3. The primary body contains all components such as the payload and the solid rocket motor. Some of the variables that are optimized such as body geometry and propellant sizing are shown in Figure 3-3. In this very simple single stage model, the payload is located in the front and modeled as a point mass.

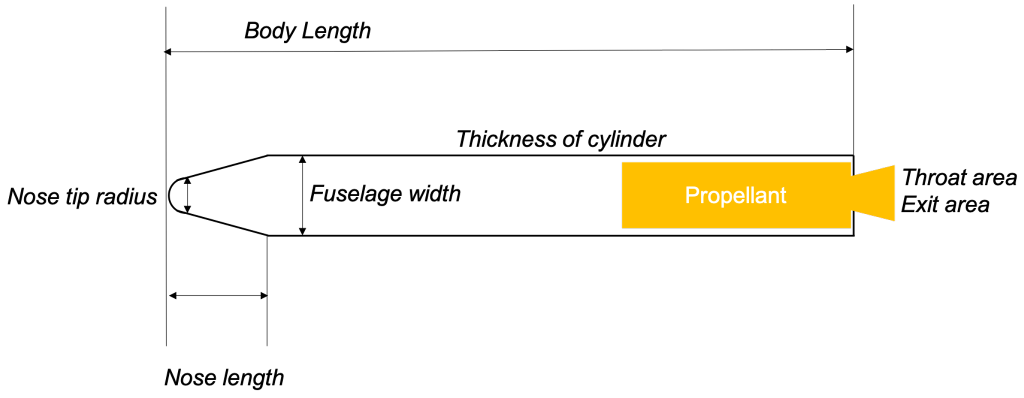


Figure 3-3: Rocket vehicle configuration with major variables.

A summary of the disciplines considered in the vehicle optimization problem is described in the following section. In the interest of compactness, not all details of the GPkit model are given here. The model will be published online [6] for anyone to use. Important details of the implementation of the different disciplines and conversion to GP-compatible constraints are described here:

Aerodynamics:

For subsonic flow, slender body theory is considered for the body aerodynamics. For supersonic flow, slender body and analytical Newtonian flow relations are used for the body. Wave drag is accounted for through analytical relations found in Fleeman [26]. The drag coefficients can be described through the following equation 3.4:

$$C_D = C_{D_0} = C_{D_{0_f}} + C_{D_{0_W}} + C_{D_{0_B}} \tag{3.4}$$

where $C_{D_{0_f}}$ is the friction drag, $C_{D_{0_B}}$ is the base drag, and $C_{D_{0_w}}$ is the wave drag which is the sum of the nose and body section wave drag. Since this is only a 1D problem, only those drag terms need to be taken into account (no yaw and pitch angle impact on drag). To convert this expression into a GP-compatible constraint, the equality sign is converted into a $\geq$ sign. The friction drag term can be calculated using the following expression in Equation 3.5 given in Fleeman and converted into the GP-compatible constraints by simply keeping the equality operator. $\frac{l}{d}$ is the finesse ratio of the body, l is the length of the body, q is the dynamic pressure and M is the Mach number. Equation 3.5 is a monomial expression only one term exists and are all powers.

$$C_{D_{0_f}} = 0.053\left(\frac{l}{d}\right)\left(\frac{M}{ql}\right)^{0.2} \tag{3.5}$$

For subsonic flow, the wave drag is set to a very small number (approximately 1e-8) due to its negligible effects. For supersonic flow, the wave drag consists of the sum of two portions: one from the body and another from the nose shape which itself is further decomposed into the sharp portion and the hemispherical portion. The body wave is calculated through the following empirical equation from Fleeman [26]:

$$C_{D_{0_W}} = \left(1.59 + \frac{1.83}{M^2}\right)\left(tan^{-1}\left(\frac{0.5}{\frac{l_N}{d}}\right)\right)^{1.69} \tag{3.6}$$

where l_N/d is the finesse ratio of the nose. The inverse-tangent function is approximated using the GPfit library [37] which fits data to GP-compatible functions. The approximation is substituted in to convert Eq. 3.6 to a GP-compatible expression. The resulting GP-compatible constraint is the following:

$$C_{D_{0_W}} = \left(1.59 + \frac{1.83}{M^2}\right)\left(0.458064\left(\frac{l_N}{d}\right)^{-0.97058}\right)^{1.69} \tag{3.7}$$

The nose portion of the wave drag is calculated by determining the wetted area of the hemispherical/"tip" portion of the nose and the sharp/non-tip portion of the nose. The sharp portion uses the same equation as Equation 3.6, however, the hemispherical portion uses the following expression (3.8):

$$C_{D0_{W_{hemi}}} = 0.665\left(1.59 + \frac{1.83}{M^2}\right) \tag{3.8}$$

Both portions are combined in a weighted fashion through the following expression that relates the area of the nose tip with the S_{ref} in Eq. (3.9).

$$C_{D0_{W_{nose}}} = C_{D0_{W_{sharp}}}\left(\frac{S_{ref} - S_{nose}}{S_{ref}}\right) + C_{D0_{W_{hemi}}}\left(\frac{S_{nose}}{S_{ref}}\right) \tag{3.9}$$

Finally, the base drag is calculated differently for supersonic and subsonic flow. For subsonic flow, the base drag is calculated using Eq. (3.10) while for supersonic flow, the base drag is calculated using Eq. (3.11).

$$C_{D_{0_B}} = \left(0.12 + 0.13M^2\right)\left(1 - \frac{A_e}{S_{ref}}\right) \tag{3.10}$$

$$C_{D_{0_B}} = \left(\frac{0.25}{M}\right)\left(1 - \frac{A_e}{S_{ref}}\right) \tag{3.11}$$

For Eq. 3.6 through 3.11, all of the expressions are either posynomial or signomial expressions, so inequalities are required. The pressure is for decreasing the drag, so all equal signs are converted to $\leq$ signs.

Structures:

The body structure is modeled as a thin-walled pressure vessel. The primary load constraint is the axial load from the thrust of the motor. Fleeman [26] provides many conditions that the thickness of the cylindrical vessel must meet. The following conditions are checked to ensure that the body thickness (t) chosen maintains integrity during the flight: internal pressure from motor (p_{int}), thrust force (T_{max}), localized buckling from axial pressure, localized buckling condition from bending, and minimum gauge for manufacturability. σ is the maximum stress allowed for the material chosen, E is the elastic modulus of the material chosen, r is the radius of the cylinder. Equation 3.12 outlines the inequalities that must be true for the design. For each flight state, these conditions must be met.

$$\begin{aligned} t &\geq \frac{T_{max}}{2\pi\sigma r} \\ t &\geq 0.06 \text{ inches} \\ t &\geq 4.9r\frac{\sigma}{E} \\ t &\geq 2.9r\frac{\sigma}{E} \\ t &\geq \frac{p_{int}r}{\sigma} \end{aligned} \tag{3.12}$$

Thermal:

The nose and leading edge sections are the primary thermal constraints as they will see the largest heating rates in the vehicle during ascent. The heating rates are calculated using equations in Nicolai [55] that are a function of speed (V). Eqs. (3.13) and (3.14) provide estimates of the equilibrium wall temperatures (θ_{wall}) and heat flux ($\dot{q}$) [55]. ϵ is the emissivity of the surface (set at 0.8), v is the Boltzmann's constant, ρ is the air density, and r_{nose} is the radius of the nose.

$$\theta_{wall} = \left(\frac{\dot{q}}{\epsilon v}\right)^{0.25} \tag{3.13}$$

$$\dot{q} = 15\left(\frac{\rho}{r_{nose}}\right)^{0.5}\left(\frac{V}{1000}\right)^{3} \tag{3.14}$$

Since both equations are in monomial form, the only conversion necessary to GP-compatible constraints is changing the equal sign to an equality sign.

Propulsion:

To size the motor properly and ensure that the design is also feasible, a modified end-burning solid rocket model found in the GPKit library of models, gplibrary [6], is used.

Weights:

Linear regressions and density calculations are used to calculate the total weight of the vehicle. Although calculating the weight of the body and the motor grain is pretty straightforward, the weight of the nose is slightly more complicated. The material of the nose is highly dependent on the temperature that is observed on the nose. Therefore, a linear correlation is used to model the relationship between density of the thermal protection system (TPS) required and the temperature that the TPS materials can take. A list of TPS materials and their allowable temperatures are given in [55]. Thickness of the TPS is set to be 0.01 m for simplicity of optimization. Therefore, the density of the nose is modeled using the following linear regression equation:

$$\rho_{nose} = (6.0768\theta_{wall} + 393.72) \tag{3.15}$$

Atmosphere:

The 1976 standard atmosphere model is used. Atmospheric density, pressure, speed of sound, and temperature are calculated outside of GPKit and are inputs to the flight condition as will be discussed in Section 3.2.1.

Integration of Vehicle Design

To ensure that the vehicle can fly the trajectory given by the trajectory optimization and that the motor is sized properly, a constraint is added such that the vehicle must be able to produce the equivalent or more acceleration at each flight state. The set acceleration is set as an signomial inequality constraint is given by:

$$a_z \leq \frac{T - D}{W_{wet}} - g \tag{3.16}$$

where a_z is the vertical acceleration at each flight state. This inequality allows the thrust acceleration to be larger than what the trajectory optimization requests.

To track the total mass W_{wet} since GPKit is allowed to decrease or increase the W_{dry} as it pleases, the following constraints are added to each flight state:

$$W_{wet_i} >= W_{wet_{i-1}} \text{ where } W_{wet} = W_{dry} + W_{fuel} \tag{3.17}$$

where W_{fuel} is given by the trajectory optimization at each flight state. Additional constraints that connect the models together to ensure they do not interfere with each other (ex. motor length exceeding the body length) are added, but not outlined here. Furthermore, to ensure that the drag does not increase significantly, the coefficient of drag term for each flight state is added into the objective function such that the drag is bounded with a small multiplier in front to make sure that the vehicle weight W_dry is still prioritized by the optimizer.

Approximations

The wave drag expression contains an inverse-tangent term that is not GP-compatible, see above. However, when we plot this function, the curvature of this function is GP-compatible. Therefore, the expression is fitted using GPfit.

$$tan^{-1}\left(\frac{0.5}{\frac{l_N}{d}}\right) \approx 0.485775\left(\frac{l_N}{d}\right)^{-0.988596} \tag{3.18}$$

The RMS of this fit is 0.0071538. Figure 3-4 shows the approximation. For the thermal protection material weight estimation, Figure 3-5 shows the data and approximately fitted line. The quality of the fit is not great; however, it is good enough for surrogate for estimating the TPS material required for the vehicle to fly the trajectory.

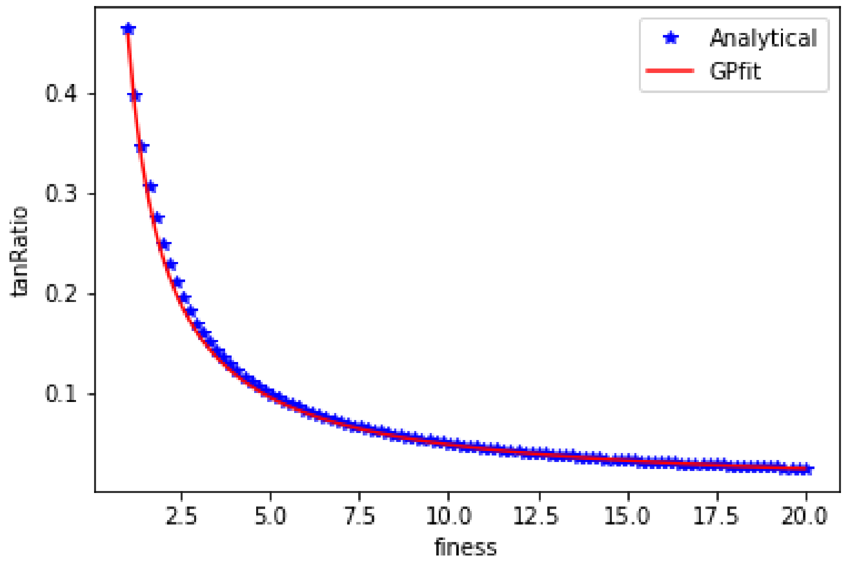


Figure 3-4: Wave drag expression fit

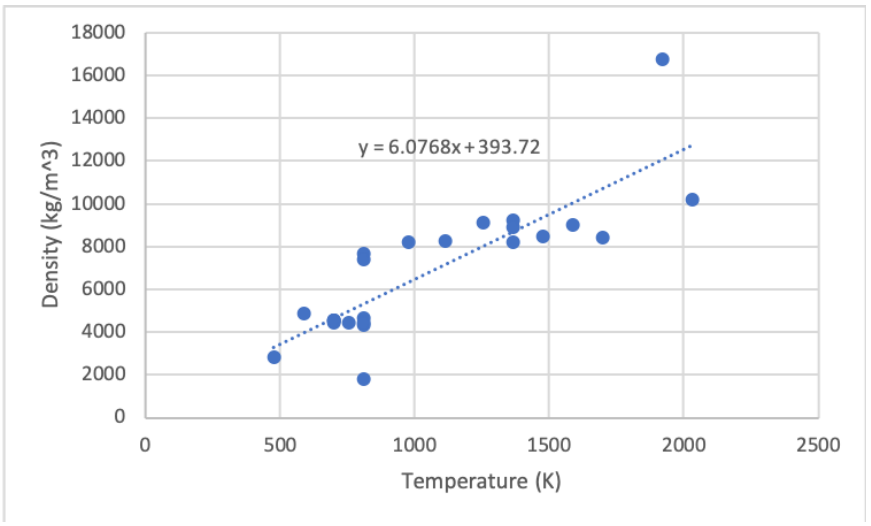


Figure 3-5: Thermal protection data and fit

3.1.4 Trajectory: Modified Goddard-ascent problem

For the full problem, the trajectory optimization problem is modeled similar to the Goddard 1-D maximum ascent problem except the objective function is modified from final altitude to be the mass of fuel used to reach the prescribed final state and the time of flight is constrained. The control problem is posed as the following:

$$\begin{aligned} \text{minimize} \quad & f^{(2)} = W_{fuel} \\ \text{subject to} \quad & \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) \\ & T_{min} \leq \mathbf{u}(t) \leq T_{max} \\ & \mathbf{x}_{min} \leq \mathbf{x}(t) \leq \mathbf{x}_{max} \\ & \mathbf{x}(t_0) = \mathbf{x}_0 \\ & \mathbf{x}(t_f) = \mathbf{x}_f \\ & t_f = t_{flight} \end{aligned} \tag{3.19}$$

$\mathbf{x}$ is the state, $\mathbf{u}(t)$ is the control action (thrust level), T_{min} and T_{max} are the minimum and maximum thrust, and t_{flight} is the time of flight.

The control can be described as:

$$\mathbf{u}(t) = T(t) \tag{3.20}$$

where $T(t)$ is the thrust of the vehicle in the vertical z-direction at time t.

The dynamics are described as:

$$\dot{m} = -\frac{T}{I_{sp} g} \tag{3.21}$$

$$\dot{h} = V \tag{3.22}$$

$$\dot{V} = \frac{T - D}{m} - g \tag{3.23}$$

where I_{sp} is the specific impulse of the vehicle's propulsion system, m is the mass of the vehicle, g is the gravitational acceleration, h is the height above sea level, V is the velocity, and D is the drag.

Drag is calculated by using a simplified drag equation:

$$D = \frac{1}{2}\rho V^2 S C_D \tag{3.24}$$

The SCvx algorithm was chosen due to its ability in solving these fixed-time trajectory optimization problems [50]. The SCvx algorithm follows the general process for successive convex optimization; however, it adds trust regions as well as a virtual control to prevent artificial infeasibility, which is when linearization determines a specific trajectory as not feasible despite it being feasible. The implementation is adapted from the open-source Python implementation of Szmuk and Açıkmeşe [56]. For the linearization process, analytical derivatives were used to calculate derivatives. To account for the different drag coefficients at different Mach numbers, the drag coefficients were not explicitly interpolated; rather, the drag coefficients were treated as parameters despite their dependency on the variables (V). This strategy proved effective for modeling the nonlinear nature of the drag rise.

This is the first known instance that the convex optimization method is used to solve the Goddard problem, and GPOPS [60] is used to validate the convex formulation of the Goddard problem. The figures show agreement between the two methodologies. Appendix A has a thorough investigation of using SCO/SCvx to solve the Goddard problem.

3.2 Single Mission Design

3.2.1 Integration and Implementation

The combined problem focuses on two objectives: the vehicle design problem minimizes the overall weight by minimizing W_{dry}, and the trajectory optimization minimizes overall weight by minimizing W_{fuel}. The connections between each process must be examined closely such that each individual optimization takes the other side's input into account.

Figure 3-6 shows a high level overview of the integrated data structure. The vehicle design optimization passes the geometry and vehicle parameters to an analysis program that contains the same models used for the vehicle design optimization. The program is used to produce full performance tables, which in this case is the array containing C_L, C_D, Mach number M, as well as any weights and motor performance numbers that are input to the trajectory optimization. Providing the full performance tables allows the trajectory optimization to use the full range of the vehicle performance data if it needs to to come up with an optimal trajectory for the vehicle. It will not be limited just by the flight conditions used to optimize the vehicle.

The trajectory optimization passes the discretized flight states to the vehicle design side so that the vehicle design is optimized subject to the relevant flight states (FS). These flight states are the bounding flight conditions that the vehicle must be able to fly through. The flight states are incorporated into the vehicle design as constraints, providing flexibility to the vehicle requirements as the trajectory changes. The 1976 atmosphere model parameters are also calculated at each FS and passed along here.

Initialization for the trajectory optimization begins with a linear trajectory that starts from the initial to the final condition at equally spaced points. That same initial trajectory is used initially to start the vehicle optimization, with an additional constraint that the initial acceleration after ignition be 20 m/s^2 as an initial guess

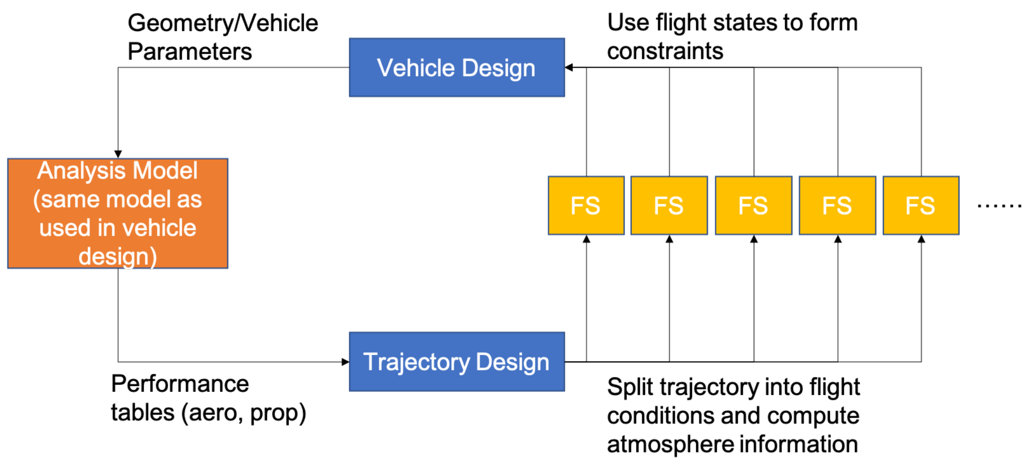

Geometry/Vehicle Parameters
Vehicle Design
Use flight states to form constraints
Analysis Model (same model as used in vehicle design)
FS
FS
FS
FS
FS
......
Trajectory Design
Performance tables (aero, prop)
Split trajectory into flight conditions and compute atmosphere information

Figure 3-6: Data flow between the two optimization processes.

(approximately 2g's). To ensure the feasibility of the vehicle to fly the mission, an additional constraint of thrust to weight ratio of 1.5 or higher was added. This ensures that the vehicle has enough thrust to accelerate and lift off. Once the optimization begins, the acceleration parameter is calculated during the transfer between the sub-problems.

The algorithm flow of both the alternating and integrated architectures are given in the Algorithm 5 and 6. The primary differences are the fact that each problem is only solved with one iteration for Algorithm 6 and that intermediate solution is then transferred to the other optimization problem. Because of the SCvx algorithm, each trajectory optimization iteration in Algorithm 6 solves the problem twice to be able to update the trust radius parameter.

3.2.2 Computational Results

The representative mission is a 1D ascent to an apogee of 3000 m with a set flight time of 50 sec. The vehicle carries a 20 kg mass-invariant payload, which is modeled as a point mass at the front of the vehicle. The mission parameters are shown in Table 3.2. A maximum of 30 iterations are allowed. All computations were run on a 2015 13-inch MacBook Pro with a 2.7 GHz Intel Core i5 and 8 GB of RAM.

Algorithm 5 Alternating Architecture for Single Mission

```
Set convergence criteria $\epsilon$
Initialize $J, J_i, \delta$:
Initialize vehicle problem
Initialize trajectory problem
Generate initial trajectory
while $\delta \geq \epsilon$ do
    Initialize $\Delta$

    // Vehicle Problem Solve
    while $\Delta^{(1)} \geq \epsilon^{(1)}$ do
        Transfer solution from other sub-problems into current sub-problem: Receive
          flight states from trajectory and incorporate into problem as constraints
        Approximate sub-problem 1 as a convex optimization problem.
        Solve sub-problem 1 for $f^{(1)}$
        $\Delta = f^{(1)} - f^{(1)}_{prev}$
        $f^{(1)}_{prev} = f^{(1)}$
    end
    // Trajectory Problem Solve
    while $\Delta^{(2)} \geq \epsilon^{(2)}$ do
        Transfer Take in vehicle model
        Approximate sub-problem 2 as a convex optimization problem.
        Solve sub-problem 2 for $f^{(2)}$
        $\Delta = f^{(2)} - f^{(2)}_{prev}$
        $f^{(2)}_{prev} = f^{(2)}$
    end
    Calculate $J = f^{(1)} + f^{(2)}$
    $\delta = J$ - $J_i$; $J_i = J$
end
```

Algorithm 6 Integrated Architecture for Single Mission

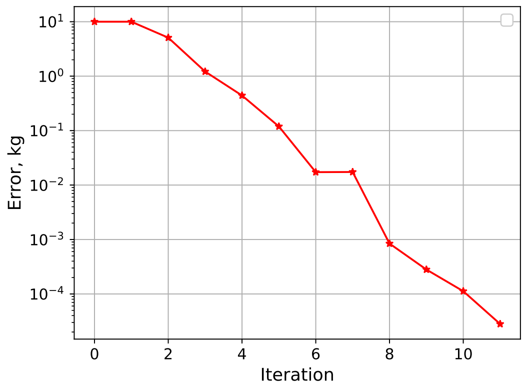
```
Set convergence criteria $\epsilon^{(j)}$ for $j = 1, 2$
Initialize $\delta^{(1)}, \delta^{(1)}, f^{(2)}_{prev}, f^{(2)}_{prev}$
while $\delta^{(1)} \geq \epsilon^{(1)}$ and $\delta^{(2)} \geq \epsilon^{(2)}$ do
    // Vehicle Problem Solve
    Transfer solution from other sub-problems into current sub-problem: Receive
      flight states from trajectory and incorporate into problem as constraints
      Approximate sub-problem 1 as a convex optimization problem.
      Solve sub-problem 1 for $f^{(1)}$
      $\delta^{(1)} = f^{(1)} - f^{(1)}_{prev}$
      $f^{(1)}_{prev} = f^{(1)}$

    // Trajectory Problem Solve
    Transfer Take in vehicle model
      Approximate sub-problem 2 as a convex optimization problem.
      Solve sub-problem 2 for $f^{(2)}$
      $\delta^{(2)} = f^{(2)} - f^{(2)}_{prev}$
      $f^{(2)}_{prev} = f^{(2)}$

end
$J = f^{(1)} + f^{(2)}$
```

The tolerance for the Alternating architecture is set at 1e-4 due to this model being a conceptual design model which means that high detailed convergence is not necessary. For the integrated architecture, the tolerance required is set by the specific SP/SCO optimizations itself, which in this case are both 1e-5.

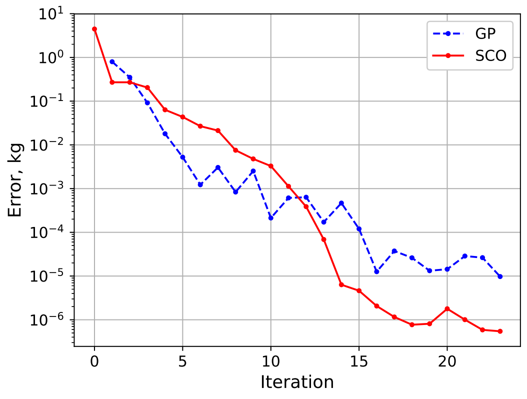
Parameter	Value
h_i	100 m
v_i	0 m/s
h_f	3000 m
v_f	0 m/s
h_{min}	100 m
h_{max}	3000 m
v_{min}	0 m/s
v_{max}	1000 m/s
t_{flight}	50 sec.
Payload mass	20 kg
Number of Flight States	50
Max iterations	30
Tolerance (Alternating)	1e-4
Tolerance (Integrated: SP)	1e-5
Tolerance (Integrated: SCO)	1e-5

Table 3.2: Mission and optimization parameters for the 1D rocket ascent problem

Both optimization processes converge and Fig. 3-7a and Fig. 3-7b show the convergence plots, where change in objective function (kg) is shown as a function of iteration number for the alternating and integrated architectures respectively. From here on, SP will be referenced to indicate the vehicle design portion of the process and will be used to indicate the trajectory optimization portion of the process. In the integrated architecture, both SCO and SP reach their needed tolerances at approximately the same time. The alternating architecture reached the required tolerance in fewer iterations compared to the integrated architecture. Although the SP optimization in the integrated architecture has a general downward trend, there are times where the tolerances are not decreasing monotonically. A feature of this architecture is that even though one optimization process may have converged, it will continue optimizing until the other process has converged. This process does not have any adverse effects on the final solution in this case, though it is something to explore further in Table 3.3.



(a) Alternating



(b) Integrated

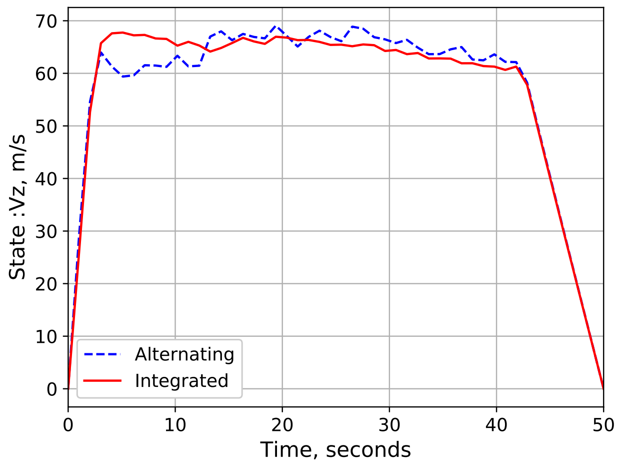
Figure 3-7: Iteration convergence history for 1D rocket problem

The expected trajectory and control is bang-singular-bang: where the control history starts with a full thrust, then a mid-level thrust when the singular arc occurs, and finally a cut-off (in this case lowest thrust possible). This is similar to the trajectory and control of the Goddard problem [60]. A singular arc occurs when Pontryagin's minimum principle cannot be applied to find the optimal control problem, and this occurs when the Hamiltonian does not depend on the control[32]. This means that the singular arc portion of the control requires additional work (such as another optimality condition) to find, and can be hard to find for optimization algorithms because the original optimality conditions do not help. This is the first known attempt at using convex optimization to solve an optimal control problem with a singular arc. Both the architectures should converge to the bang-singular-bang solution.

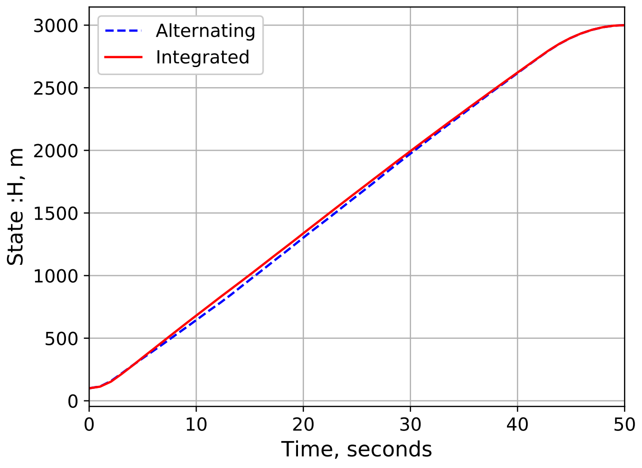
The final vehicle and trajectory solutions produced by the two convex optimization architectures are slightly different yet follow the same general flight path. Figures 3-8-3-9 show the final trajectories and controls chosen by the different architectures. The values shown are all in the Earth-Centered, Earth Fixed (ECEF) frame of reference. Both architectures show similar trajectories according to Figure 3-9, with the optimizer struggling to find the appropriate control to maintain that singular arc as seen in Fig. 3-9b. Both architectures converge to a similar altitude versus time profile. Figure 3-8b indicates that the integrated altitude profile climbs slightly faster by approximately 5 m/s before slowing down.

The vehicle design solutions from each architecture have slight differences. The optimized vehicle design results from each architecture are indicated in Table 3.3. The single iteration run where each optimization process is run once is also shown for comparison. Both architectures decrease the needed mass by half of what is actually needed as determiend by the trajectory optimization compared to the single iteration design, with large decreases coming from the higher fidelity fuel estimation from the trajectory optimization. From this comparison, it is clear that the vehicle weight is decreased by almost 50% due to the more accurate fuel weight estimation from the trajectory optimization process, which results in a lighter total vehicle. The added body finesse ratio constraint that keeps the body fineness ratio above 5 is the only limiting factor that prevents the vehicle from further reducing weight by only letting the length of the vehicle be enough to cover both the payload and the motor grain.

One interesting note from the results in Table 3.3 is the fact that the optimization processes did not improve the aerodynamic performance of the vehicle much if at all for both architectures. For the integrated architecture, the aerodynamic performance



(a) Velocity

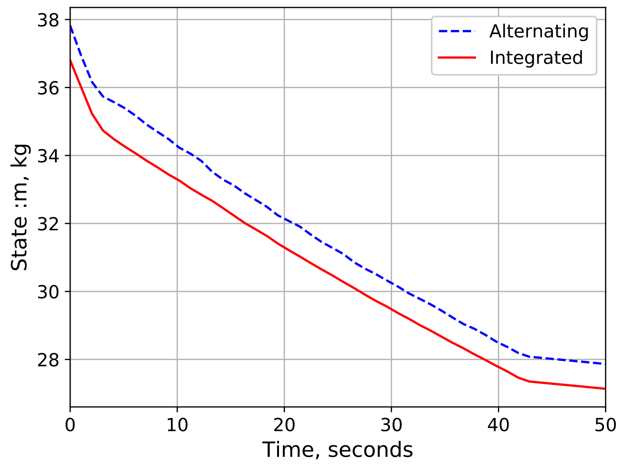


(b) Altitude

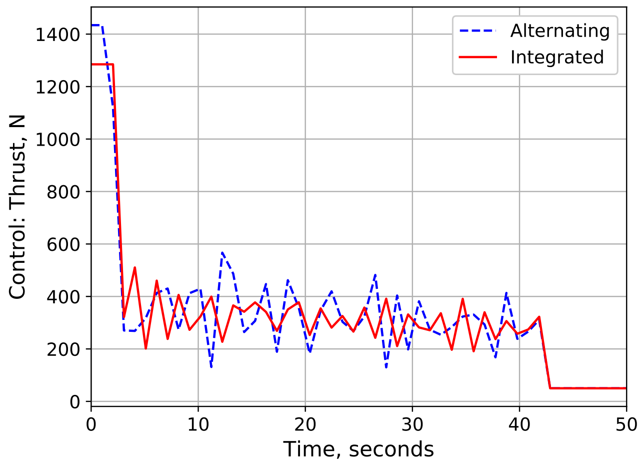
Figure 3-8: Final trajectories for each architecture

was actually worse by examining the subsonic C_D; yet, the motor sizing was decreased a bit from the single iteration case. This result is somewhat expected since the problem formulation makes drag only a second or even third order effect on the overall design; however, it indicates that motor thrust/sizing and an accurate fuel burn estimation will have a bigger impact on the design than improved aerodynamics for this problem.

One of the primary goals of this work is to create a methodology that allows design-



(a) Vehicle Mass



(b) Vehicle Thrust

Figure 3-9: Final control for each architecture.

ers to look at both the vehicle and trajectory problems simultaneously and quickly. Table 3.4 shows the computational resources and times used by each architecture for a single iteration, as well as for the overall process. As one can see, the total computation time for the different architectures varies quite significantly with the integrated approach taking approximately half of the time to converge, but roughly double the number of iterations. The alternating architecture spends significant time ensuring convergence for the separate processes while the integrated architecture en-

Vehicle Design Parameter	Single Iteration	Alternating	Integrated
W_{total} (kg)	73.1	39.27	38.03
W_{dry} (kg)	30.1	28.9	28.02
W_{fuel} (kg)	43.0	10.37	10.01
Body weight (kg)	9.62	8.78	7.91
Body thickness (m)	0.003	0.003	0.003
Body length (m)	1.12	1.12	1.06
S_{ref} (m^2)	0.039	0.039	0.036
Body nose radius (m)	0.012	0.0058	0.0054
Motor thrust (N)	1430	1430	1290
Motor length (m)	0.64	0.15	0.16
C_D at M = 0	0.141	0.136	0.154

Table 3.3: Summary of optimized major variables.

sures convergence for both processes simultaneously and more smoothly. Iterations are not wasted on converging to the respective appropriate tolerances while using approximate information from the respective sides.

Computation Parameters	Single Iteration	Alternating	Integrated
Total computation time (seconds)	27.8	385.9	168.4
# of iterations	1	12	23

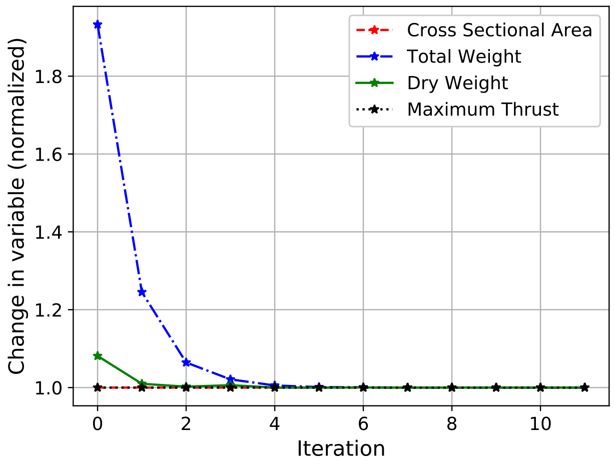
Table 3.4: Results Computation Summary for 1D Rocket Problem

Due to the singular arc, the SCO methodology for this process takes much longer than pseudo-spectral methods and other SCO problem solves. SCO struggles to find the accurate control by introducing oscillatory control behavior that averages out into the correct singular arc. Typical SCO problem total solve times are in the seconds [45] while for this implementation of the Goddard problem, the SCO takes at least 15+ iterations with each iteration being 1-3 seconds each for approximately minute scale solve times. However, this is still almost twice as fast as GPOPS solve times (see Appendix A for details). Therefore, a further speed-up in the SCO problem solve (whether through faster solves or less iterations) will improve the time performance significantly for these architectures.

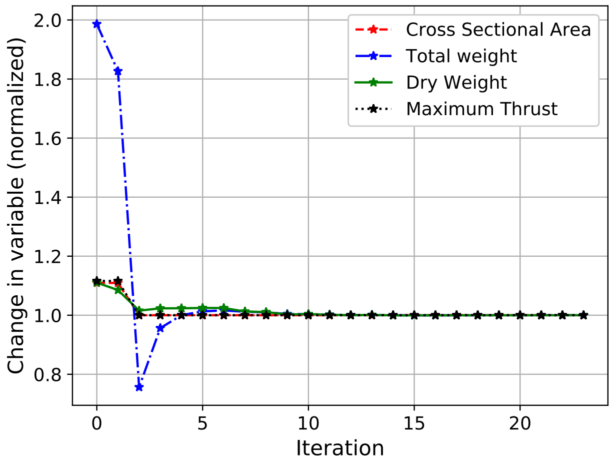
This section examines the iteration behavior between the successive system-level iterations. We attempt to show that the hypothesis in section 2.2.4 is true. The previous section showed that the combined loops in the Integrated Architecture resulted in a design that is lighter compared to the standard alternating loop; however, we want to gain additional insight into the iteration process for further potential algorithm improvement.

Figure 3-10 shows how several vehicle design variables evolve with system-level iterations. All values are normalized by the final solution given by the optimizer. As expected, the vehicle size (indicated by cross-sectional area) shows little change as the optimization process continues. For both architectures, most of the effort spent by the optimizer is on decreasing the total weight of the system. The initial guess for W_{fuel} by GPKit over-designs for the mission, and the trajectory optimization gives a more accurate estimation on how much fuel is needed.

As expected, the body geometry and related parameters such as the cross-sectional area remain unchanged for both architectures. This result is due more to the problem set-up rather than the optimization process itself, since there is no direct feedback mechanism for the trajectory optimization to return the effect of the aerodynamics to the vehicle design. For the alternating architecture, passive dynamics such as aerodynamics are not captured in the design cycle. In this problem set-up, the trajectory optimization is able to directly influence W_{total} used by GPKit to size the vehicle because it is one of the primary outputs of both processes. There is a constant exchange at each iteration of weight information. However, for the aerodynamics portion of the vehicle, it is hard if not impossible for the trajectory optimization to "signal" to the vehicle optimization side to change because there is no variable/indicator that tells the vehicle optimization side to either go more aerodynamic or less aerodynamic despite aerodynamics having a non-negligible effect on fuel weight. Despite this, the integrated architecture allows for some flexibility in change in the early iterations as seen in Figure 3-10b. The strategy for the integrated architecture is thus slightly changed since the trajectory information influences the GPKit optimization process. The optimization goes for a vehicle that is lighter while having a higher drag coefficient. This strategy does result in a vehicle that is 3% lighter than the alternating architecture design and 52% lighter than the feasible single iteration design.



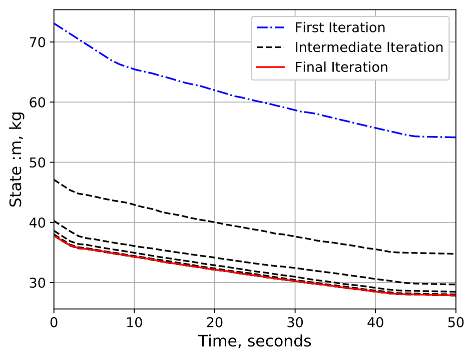
(a) Alternating architecture



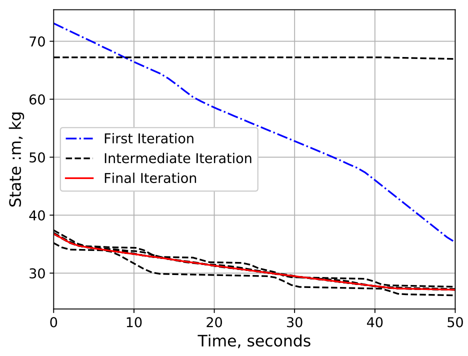
(b) Integrated architecture

Figure 3-10: Vehicle parameter change for each architecture, normalized by final value.

Figure 3-11 shows how the mass history evolves with system-level iterations and Figure 3-12 shows the different control histories evaluated by the optimizer. The mass history varies quite significantly for both architectures due to the problem set-up. For the integrated architecture, only the trajectories for every third iteration is shown to allow for less clutter. The optimizer has flexibility in deciding which control strategy is best for the vehicle design at each iteration. The varying degrees of freedom are visible by the comparison of a variety of control histories between the two architectures as shown in Figure 3-12a and 3-12b. The integrated architecture shows different control histories in Figure 3-12b and has many different switching times as well as different strategies as it simultaneously attempts to find an optimal trajectory for the vehicle as well as the overall mission whereas for the alternating architecture in Fig. 3-12a, only the length and magnitude of the singular arc section in the middle is varied since it is focused on optimizing the trajectory for the vehicle given. This is also apparent in the mass history as Fig. 3-11b shows that the integrated architecture evaluated a single trajectory where the mass did not change throughout the flight. This erroneous mass time history shows that the trajectory optimization may not have been working correctly as intended throughout the entire process, especially if a single trajectory was shown as "feasible" despite no mass decreasing. This is most likely due to the fact that the optimizer found another way to satisfy the constraints. In the future, another constraint such as ensuring the mass history should differ by some set amount should be added to ensure that the optimization is robust. Another possibility is the fact that the mission is infeasible for the vehicle. Regardless, the optimizer brought the trajectory optimization back on track without manual intervention.

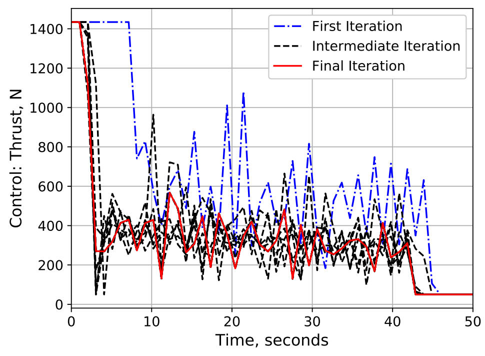


(a) Alternating architecture

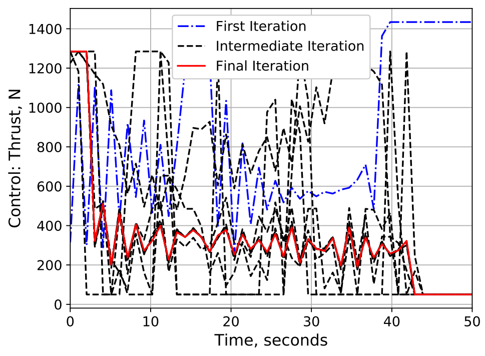


(b) Integrated architecture

Figure 3-11: Vehicle mass evolution during optimization process.



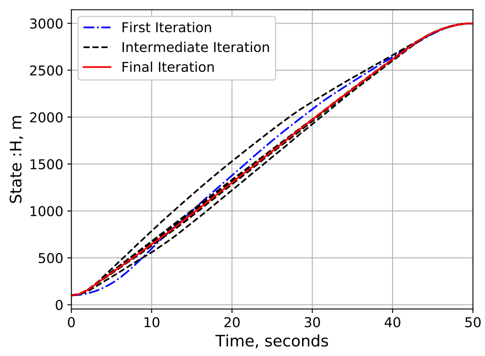
(a) Alternating architecture



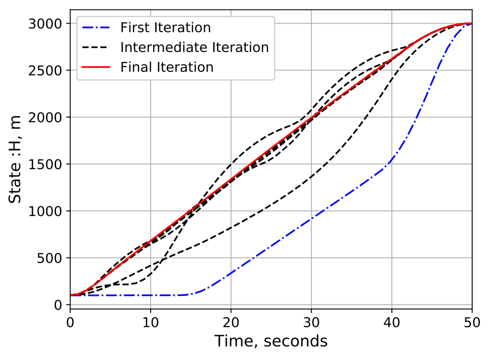
(b) Integrated architecture

Figure 3-12: Control evolution during optimization process.

Figures 3-13-3-14 show the evolution of height and velocity with system-level iterations. Similar to the mass and control histories, the alternating architecture shows a more predictable progress towards convergence since most of the iteration results have very similar features in all state histories. The integrated architecture shows many different trajectory structures such as multiple velocity peaks and dives as well as "waits" where the vehicle attempts to hold a constant velocity or location for a length of time. However, the SCvx algorithm gradually decreases the trust radius, forcing the trajectory to be become more compliant to the problem's constraints.

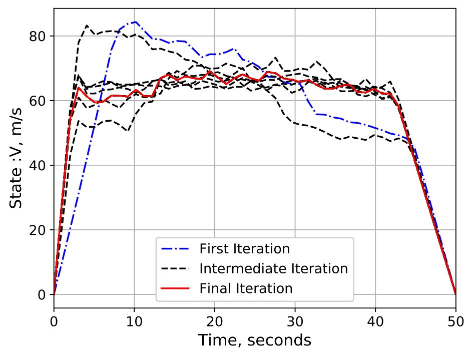


(a) Alternating architecture

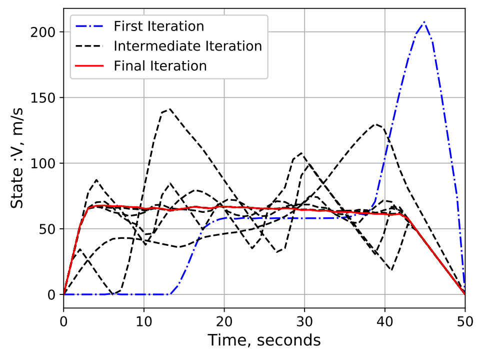


(b) Integrated architecture

Figure 3-13: Height evolution during optimization process.



(a) Alternating architecture



(b) Integrated architecture

Figure 3-14: Velocity evolution during optimization process.

3.3 Multi Mission Design

3.3.1 Problem Formulation

We can expand this framework by looking at another problem of multi-mission design. Often times, vehicles are not designed for a single mission; rather, they are designed for multiple alternative or sequential missions. The convex optimization architectures are flexible enough to accommodate this type of problem. We test this statement by implementing a two mission design problem.

The original problem is modified to the following Problem 3.25 with the notation given in Table 3.5. The primary modification is the change of the overall system

Notation	Problem Variable/Parameter
$x^{(1)}$	vehicle design problem variables
$f^{(1)}$	W_{dry}
$g^{(1)}$	inequality constraints for vehicle design problem
$h^{(1)}$	equality constraints for vehicle design problem
$x^{(2)}$	trajectory design problem 1 variables
$f^{(2)}$	W_{fuel_1}
$g^{(2)}$	inequality constraints for trajectory design problem
$h^{(2)}$	equality constraints for trajectory design problem
$x^{(3)}$	trajectory design problem 2 variables
$f^{(3)}$	W_{fuel_2}
$g^{(3)}$	inequality constraints for trajectory design problem
$h^{(3)}$	equality constraints for trajectory design problem
k	$= 3$
J	W_{total}

Table 3.5: Notation for convex architectures

objective function J. Rather than just taking the fuel burned in a single mission, we take the maximum of both missions since that is how much fuel that the rocket must be able to carry. The two-mission design problem adds a third sub-problem to accommodate the additional mission that will need to be analyzed.

$$\begin{aligned} &\text{minimize} && J(f) = f^{(1)}(x^{(1)}) + max(f^{(2)}(x^{(2)}), f^{(3)}(x^{(3)})) && \\ &\text{subject to} && g_i^{(j)}(x) \leq 0 && i = 1, \ldots, m^{(j)} \; j = 1, 2, 3 \\ & && h_i^{(j)}(x) = 0 && i = 1, \ldots, n^{(j)} \; j = 1, 2, 3 \end{aligned} \tag{3.25}$$

3.3.2 Implementation and Integration

The implementation and integration to accommodate the second mission is very straightforward. To test this, we choose the following mission parameters shown in Table 3.6. The problem is simplified by adding another mission that is similar to the original mission: the vehicle must also be able to fly longer (80 sec) and higher than the previous mission.

Parameter	Mission 1 Value	Mission 2 Value
h_i	100 m	100 m
v_i	0 m/s	0 m/s
h_f	3500 m	5500 m
v_f	0 m/s	0 m/s
h_{min}	100 m	100 m
h_{max}	3500 m	5500 m
v_{min}	0 m/s	0 m/s
v_{max}	1000 m/s	1000 m/s
t_{flight}	50 sec.	80 sec.
Payload mass	20 kg	20 kg
Number of Flight States	50	50
Max iterations	100	
Tolerance (Alternating)	1e-4	
Tolerance (Integrated: SP)	1e-5	
Tolerance (Integrated: SCO)	1e-5	

Table 3.6: Mission and optimization parameters for Multi-Mission Design

The algorithms for the multi-mission problem are shown in Algorithm 7 and Algorithm 8 which describe the alternating and integrated mission respectively. The only change that was added into the algorithm is the modified objective function calculation as well as the additional trajectory optimization problem solve process. Otherwise, the algorithms are unchanged.

The data flow is also very similar as shown in Figure 3-15. The only modification is from the single mission case is the addition of the second trajectory optimization process as well as the additional flight states that result from that. Because the vehicle design process is formulated as a multi-point design problem, the additional flights states does not require modifications to the problem.

Algorithm 7 Alternating Architecture for Multi Mission

```
Set convergence criteria $\epsilon$
Initialize $J, J_i, \delta$:
Initialize vehicle problem
Initialize trajectory problem
Generate initial trajectory
while $\delta \geq \epsilon$ do
    Initialize $\Delta$

    // Vehicle Problem Solve
    while $\Delta^{(1)} \geq \epsilon^{(1)}$ do
        Transfer solution from other sub-problems into current sub-problem: Receive
          flight states from trajectory and incorporate into problem as constraints
        Approximate sub-problem 1 as a convex optimization problem.
         Solve sub-problem 1 for $f^{(1)}$
         $\Delta = f^{(1)} - f^{(1)}_{prev}$
         $f^{(1)}_{prev} = f^{(1)}$

    end
    // Trajectory Problem Solve 1
    while $\Delta^{(2)} \geq \epsilon^{(2)}$ do
        Transfer Take in vehicle model
        Approximate sub-problem 2 as a convex optimization problem.
         Solve sub-problem 2 for $f^{(2)}$
         $\Delta = f^{(2)} - f^{(2)}_{prev}$
         $f^{(2)}_{prev} = f^{(2)}$

    end
    // Trajectory Problem Solve 2
    while $\Delta^{(3)} \geq \epsilon^{(3)}$ do
        Transfer Take in vehicle model
        Approximate sub-problem 3 as a convex optimization problem.
         Solve sub-problem 3 for $f^{(3)}$
         $\Delta = f^{(3)} - f^{(3)}_{prev}$
         $f^{(3)}_{prev} = f^{(3)}$

    end
    Calculate $J = f^{(1)} + max(f^{(2)}, f^{(3)})$
     $\delta = J$ - $J_i$; $J_i = J$

end
```

Algorithm 8 Integrated Architecture for Multi Mission

```
Set convergence criteria $\epsilon^{(j)}$ for $j = 1, 2, 3$
Initialize $\delta^{(1)}, \delta^{(2)}, \delta^{(3)}, f^{(1)}_{prev}, f^{(2)}_{prev}, f^{(3)}_{prev}$
while $\delta^{(1)} \geq \epsilon^{(1)}$, $\delta^{(2)} \geq \epsilon^{(2)}$ and $\delta^{(3)} \geq \epsilon^{(3)}$ do
    // Vehicle Problem Solve
    Transfer solution from other sub-problems into current sub-problem: Receive
      flight states from trajectory and incorporate into problem as constraints
      Approximate sub-problem 1 as a convex optimization problem.
      Solve sub-problem 1 for $f^{(1)}$
      $\delta^{(1)} = f^{(1)} - f^{(1)}_{prev}$
      $f^{(1)}_{prev} = f^{(1)}$

    // Trajectory Problem Solve 1
    Transfer Take in vehicle model
      Approximate sub-problem 2 as a convex optimization problem.
      Solve sub-problem 2 for $f^{(2)}$
      $\delta^{(2)} = f^{(2)} - f^{(2)}_{prev}$
      $f^{(2)}_{prev} = f^{(2)}$

    // Trajectory Problem Solve 2
    Transfer Take in vehicle model
      Approximate sub-problem 3 as a convex optimization problem.
      Solve sub-problem 3 for $f^{(3)}$
      $\delta^{(3)} = f^{(3)} - f^{(3)}_{prev}$
      $f^{(3)}_{prev} = f^{(3)}$
end
$J = f^{(1)} + max(f^{(2)}, f^{(3)})$
```

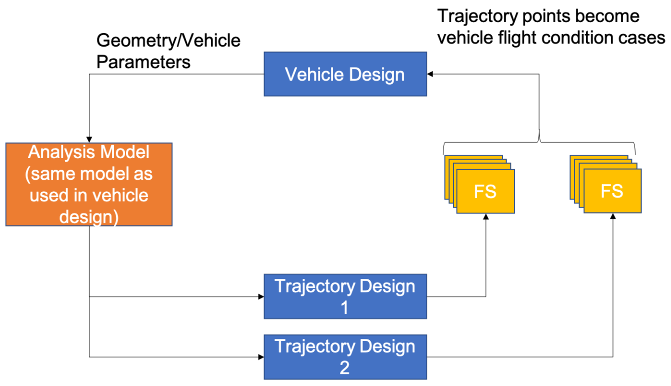

Trajectory points become vehicle flight condition cases
Geometry/Vehicle Parameters
Vehicle Design
Analysis Model (same model as used in vehicle design)
FS
FS
Trajectory Design 1
Trajectory Design 2

Figure 3-15: Integration Picture

A note on this formulation and implementation is the potential for parallelization of the problem. The multi-mission problem can be easily parallelized across different processors because once it receives the vehicle model, it does not need anything from the other trajectory problem. This means that the multi-mission problem optimization time may be as fast as the the single mission design case if there is a multiprocessor core or multiple processors available for each trajectory optimization process. However, for this thesis, the implementation does not include any parallel processing, and each sub-problem was done sequentially and is left for future work.

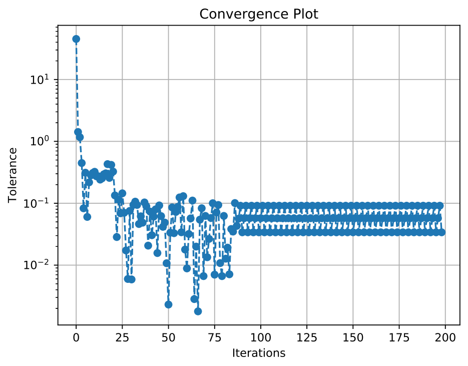
3.3.3 Computational Results

The multi-mission problem was implemented using the same equipment than on the single mission problem. Table 3.7 shows the computational results from each architecture. The alternating architecture did not converge to the specified tolerance; it settled into a cyclic behavior that is very close to the solution that the integrated architecture settled on. The integrated architecture converged in approximately 10 min and within 31 iterations with both the vehicle and two trajectory optimization problems reaching their needed tolerances together. The integrated architecture is more than three times faster in each iteration; however, the alternating architecture was not parallelized in this implementation and this would result in a speed-up that would bring the per-iteration speed to be on par with the integrated architecture.

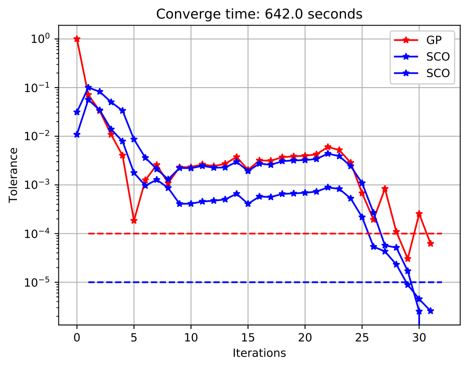
Computation Parameters	Alternating	Integrated
Total computation time	4 hours 6 min	10 min 42 sec
# of iterations	200	33
Time per iterations	73.8 secs	20.7 secs
Converged?	No	Yes

Table 3.7: Results Computation Summary

Figure 3-16 shows the convergence plots for both architectures. The alternating architecture in Figure 3-16a shows the cyclic behavior that occurs after approximately iteration 80. This is similar to a limit-cycle where the solution switches back and forth between two designs without converging. This behavior means that the solve process is stuck and therefore unable to get out of the alternating pattern. The same problem does not exist for the integrated architecture in Figure 3-16b though there is a period during the convergence where the tolerance does not change and actually increases for all three sub-problems. However, the integrated architecture reaches a point where all three processes quickly begin to decrease in tolerance towards the needed convergence tolerance. The reasoning behind this is explained through the trajectory characteristics in the latter portion of the chapter.



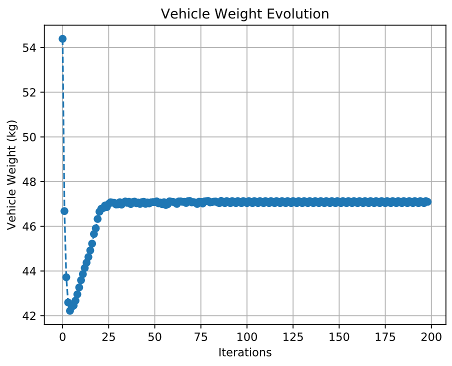
(a) Alternating architecture



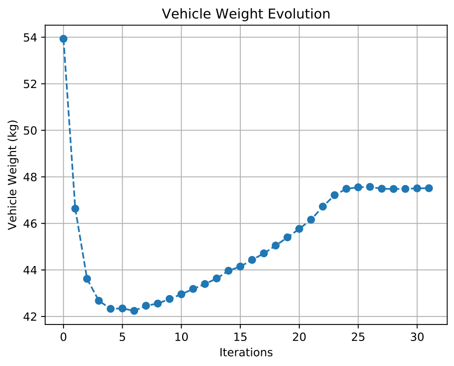
(b) Integrated architecture

Figure 3-16: Overall convergence history

The specific dry weight and fuel weight evolution as iterations progress can be found in Figure 3-17 and Figure 3-18. Both architectures dip down in terms of total vehicle weight before correcting upwards towards the final vehicle weight around iteration 25. It is interesting to see that both architectures reach approximately the same settling point around the same number of iterations. This may be a possible indication that the approximate solution is adequate for iterations in the vehicle weight evolution in Figures 3-17a and 3-17b. The same thing also occurs in the trajectory optimization in Figure 3-18a and 3-18b. The trajectory optimization initially undershoots for the less intensive mission (shown in blue); however, corrects in both vehicle weight and fuel weight before settling for a higher fuel amount. This is most likely due to the trajectory optimization and vehicle optimization accounting for the extra fuel needed to be carried for the longer mission: it continuously iterates as both weights increase before a settling point is reached. It should be clear here that in the case of the lower mission to an altitude of only 3,000 meters that the sounding rocket will still have unspent fuel onboard once the terminal condition is reached.



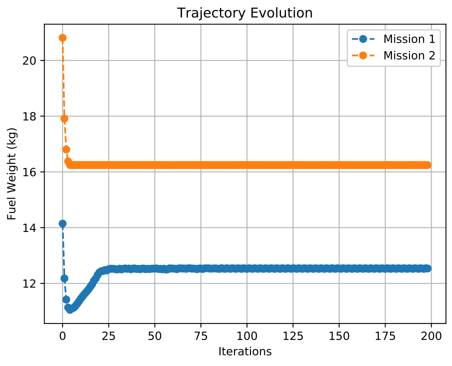
(a) Alternating architecture



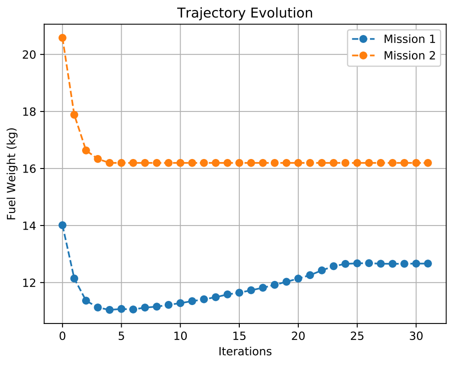
(b) Integrated architecture

Figure 3-17: Vehicle Mass Evolution History

The final vehicle properties for each architecture are shown in Table 3.7. The final weight parameters are very similar with each architecture off by a fraction of kilogram; however, the control strategy is completely different. The motor thrust, length/duration of firing, and the drag of the vehicle is completely different. The



(a) Alternating architecture



(b) Integrated architecture

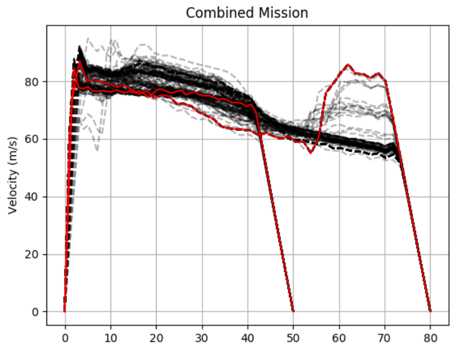
Figure 3-18: Trajectory/Fuel Weight Final Evolution

integrated optimization chose a high drag design with higher thrust with a shorter motor. Both architectures resulted in a similar fuel weight, which means that the trajectory or control strategy must have been different in order for the result from each architecture to be feasible.

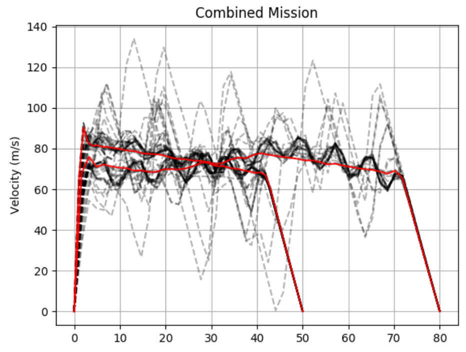
Vehicle Design Parameter	Alternating	Integrated
W_{total} (kg)	47.1	47.5
W_{dry} (kg)	30.8	31.3
W_{fuel} (kg)	16.2	16.2
Body weight (kg)	10.6	11.2
Body thickness (m)	0.003	0.003
Body length (m)	1.00	1.02
S_{ref} (m^2)	0.048	0.051
Body nose radius (m)	0.0098	0.0100
Motor thrust (N)	3388.4	3567.8
Motor length (m)	0.191	0.111
C_D at M = 0	0.134	0.242

Table 3.8: Summary of optimized major variables for multi-mission design

The velocity and height profiles and the evolution of each profile is shown in Figure 3-19 and 3-20. As expected, both architectures settled on different trajectories. The alternating architecture settled on a short-mission trajectory that is faster than the long-mission trajectory whereas the opposite is true for the integrated architecture. Interestingly enough, the alternating architecture also settled for different trajectories



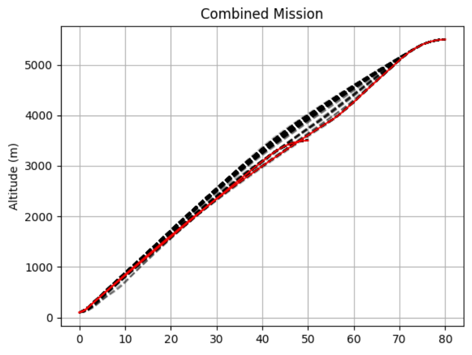
(a) Alternating architecture



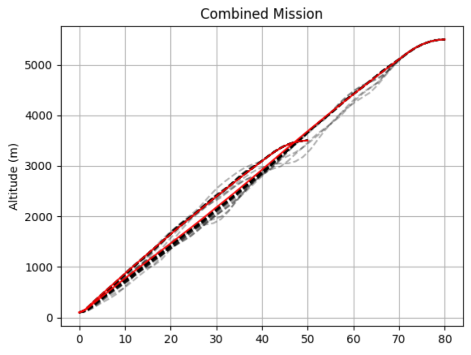
(b) Integrated architecture

Figure 3-19: Trajectory Velocity Profile Final Evolution

between the two mission as shown in Figure 3-20a and 3-19a: the short-mission trajectory has a simple cruise in the middle while the long mission speeds up in the end to meet the prescribed end time at 5,500 meters. This was not an expected result and likely is due to the smaller motor thrust that resulted from the vehicle design. The integrated architecture settled for the same strategy though with different cruise speeds as shown in Figure 3-20b and 3-19b. The evolution also varied significantly more for the integrated architecture as expected, similar to the behavior already seen in the single mission case.



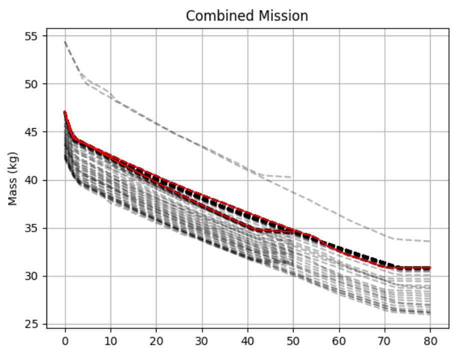
(a) Alternating architecture



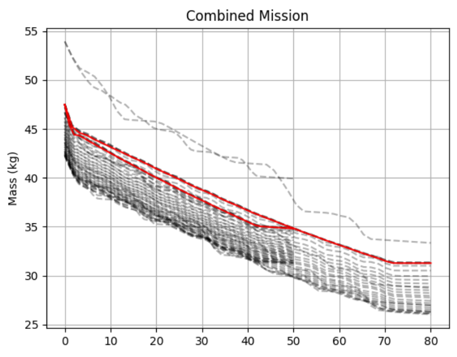
(b) Integrated architecture

Figure 3-20: Trajectory Height Profile Final Evolution

The control and mass profiles tell a similar story as well in Figure 3-22 and 3-21. Interestingly, the integrated architecture shows an oscillatory behavior that is



(a) Alternating architecture

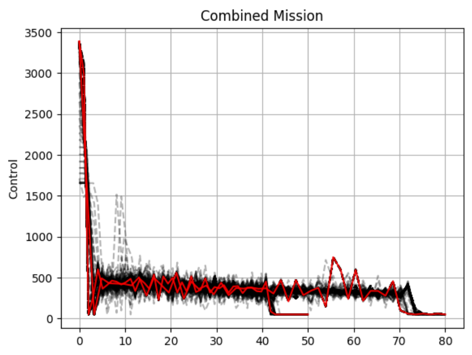


(b) Integrated architecture

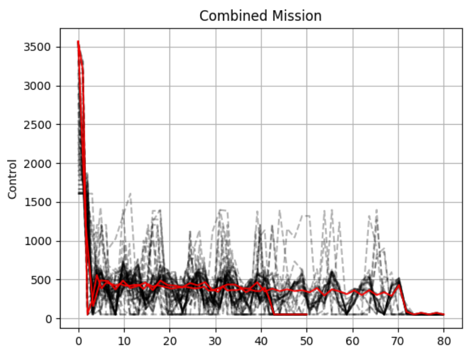
Figure 3-21: Trajectory Mass Profile Final Evolution

almost sinusoidal in Figure 3-22b and 3-21b that is not apparent in the alternating architecture solution evolution in Figure 3-22a and 3-21a. This behavior is due to a singular arc solution existing within the optimal trajectory. This is most likely due to the fact that the first solution from the trajectory optimization was highly oscillatory when the trust radius for the SCvx algorithm was so large that it still accepted possibly infeasible solutions, and due to the architecture utilizing the solution from a previous iteration, the oscillatory solution remained. However, as the tolerance decreased and the trust region decreased, the oscillations decreased in amplitude. This behavior can be mitigated through careful tuning of the trust radius at the start of the optimization. Appendix A has a small investigation into the behaviors and factors that affect the singular arcs.

Finally, the maximum velocity and maximum thrust for each iteration is observed for each iteration in Figure 3-23 and 3-24. The max initial thrust increases steadily for both architectures in Figure 3-24a and 3-24b. This can most likely be attributed to the fact the vehicle design was making the vehicle's initial boost phase more intense such that the cruise period does not have to be at a higher velocity. This explanation holds for the integrated architecture in Figure 3-23b but does not necessarily hold for the alternating architecture in Figure 3-23a because the max velocity holds constant

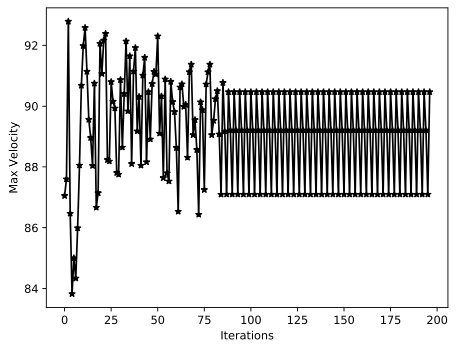


(a) Alternating architecture

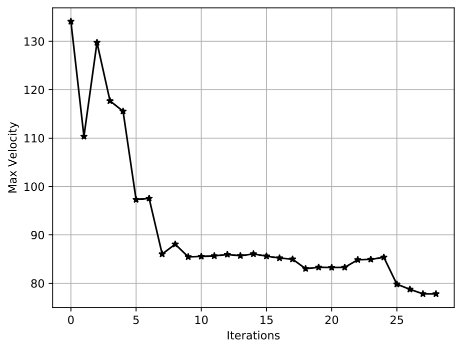


(b) Integrated architecture

Figure 3-22: Trajectory Control Profile Final Evolution



(a) Alternating architecture



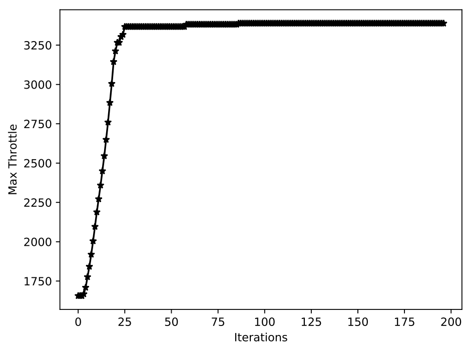
(b) Integrated architecture

Figure 3-23: Max Velocity Evolution

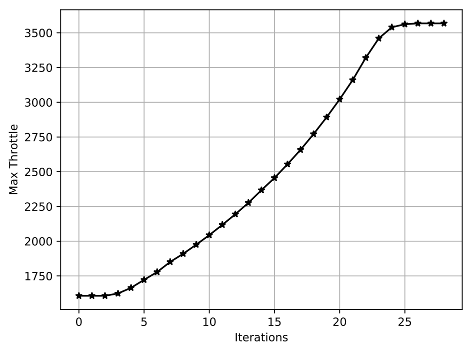
as the optimization progresses. Another possible explanation is that the max thrust is increased to maintain the acceleration. As the vehicle weight increases to allow for the increased thrust (due to structural constraints), the vehicle thrust must also increase, which increases the weight and so on. This explanation holds when examining the vehicle weight evolution in Figure 3-17.

3.4 Chapter Summary

A simple rocket design problem is implemented using the convex optimization architecture for both a single and multi-mission design problem. Both the alternating



(a) Alternating architecture



(b) Integrated architecture

Figure 3-24: Max Thrust Evolution

and integrated architectures successfully converge to a solution for the single-mission; however, the alternating architecture reaches cyclical behavior despite the integrated architecture reaching a solution. Future additional improvements could potentially be gained through the parallelization of the process which in some cases could allow the multi-mission design case to have similar run-times as the single-mission design case. Scaling this approach to a more complex rocket design problem one would consider the full 2D and 3D trajectory as well as additional missions, each representing the corner points of the entire flight envelope.

This chapter demonstrated the utility of convex optimization architectures on an integrated vehicle and mission design problem. The next chapter will apply this mindset to a more realistic and complex problem of designing a hydrogen-powered aircraft.

Chapter 4

Hydrogen Aircraft Design Case

4.1 Case Study Context

4.1.1 Hydrogen in Civil Aviation

Hydrogen is an alternative fuel that could be used to replace kerosene for flight operations. Specifically, the use of liquid hydrogen is explored in this chapter. Although fuel cells and gaseous hydrogen are promising for aviation use, they are more suited for shorter range missions. The benefits and viability of liquid hydrogen as a fuel can be summarized in the following table [71]. Liquid hydrogen or LH2 has three times more energy than kerosene; however, liquid hydrogen has only 1/10th of the density of kerosene and also needs to be stored at very cold temperatures unlike kerosene which can be stored in most atmospheric states.

Property	Liquid Hydrogen	Kerosene
Heat of combustion (MJ/kg)	120	43
Volumetric Density (kg/m^3)	70.9	810.53
Boiling Temperature (K)	20.369	439.8

Table 4.1: Hydrogen Properties compared to Kerosene

Other critical design challenges that arise are briefly summarized below:

Cryogenics

Due to the low temperature of the fuel, cryogenics are needed to store the fuel. Unlike kerosene, LH2 cannot be stored anywhere there is space. Cryogenics are needed to 1. maintain the low temperature and 2. maintain the liquid state of hydrogen. With liquid hydrogen, boil-off occurs where the liquid changes state to a less-useful state of gas; in other words, during operations, fuel is lost without useful work gained out of it. The amount of boiloff losses depends on several factors such as the initial total amount of fuel, the environmental temperature, the geometry of the LH2 tank(s) as well as thermal insulation. The amount of boiloff can be reduced or eliminated by so called zero-boiloff (ZBO) technologies such as cryo-coolers, but this comes at the cost of additional mass and complexity.

Additional equipment and therefore weight is needed to contain liquid hydrogen to minimize the boil-off. Furthermore, the tanks must ideally be a surface area minimizing shapes (i.e. spherical). Therefore, any shape conforming tanks will cause major losses in fuel; and any ideal tank shapes may have large aerodynamic penalties. The ideal design is somewhere in the middle. Colozza has previously devised a simple sizing methodology [19], and Winnefeld [71] created a cryogenic tank design methodology based off of Verstraete's work in designing hydrogen aircraft [70].

Wing Weight

In passenger aircraft, most of the fuel is stored in the wings. This helps decrease structural weight needed for the wing due to the fact that the fuel also provides load alleviation during flight due to gravity which counteracts the lift and reduces the wing root bending moment. However, cryogenics require the tanks to be stored in the fuselage to minimize drag. Therefore, the wing structure will likely need to be reinforced to compensate for the lack of load alleviation.

Cryogenic fuel effect on aircraft performance

Cryogenic fuel provides many benefits to engine performance, mostly due to reduced thrust specific fuel consumption or TSFC. This is driven in large part by the higher energy content of the fuel per unit mass. Also due to the cryogenic fuel, cooling can be provided to the engine which results in higher performance. Furthermore, due to the absence of carbon in the fuels, the by-products of the combustion are simply water vapor and NOx.[16] Although this is definitely better than the exhaust products of the combustion from hydrocarbons such as CO, CO_2, water vapor is also a green-house gas; and work to minimize NOx emissions and water vapor production is focused on the combustor design [73].

Finally, aircraft integration also remain as a challenge. Due to the fuel not being stored within the wings and cryogenic tanks needing to be spherical shaped or near-spherical, the L/D of the aircraft must also increase. Any storage outside will further increase the drag due to increased wetted area. Another consideration is the placement of the tank as loss from fuel-lines needs to be minimized as much as possible through close placement of the tanks. This limits the possible configurations that can be used for hydrogen aircraft.

4.1.2 Operations/Missions Perspective

In past work, most hydrogen aircraft studies have focused on the aircraft design itself fitted into current operations of a single flight mission profile. Brewer summarizes the challenges and past work done in hydrogen aircraft design in the past [16]. In recent work, Mancini explored 747-sized aircraft as a hydrogen aircraft, attempting to fit the cryogenic tanks and determining the performance [49]. Verstraete created an entire set of tools examining such configurations from cryogenic fuel tank design to aircraft

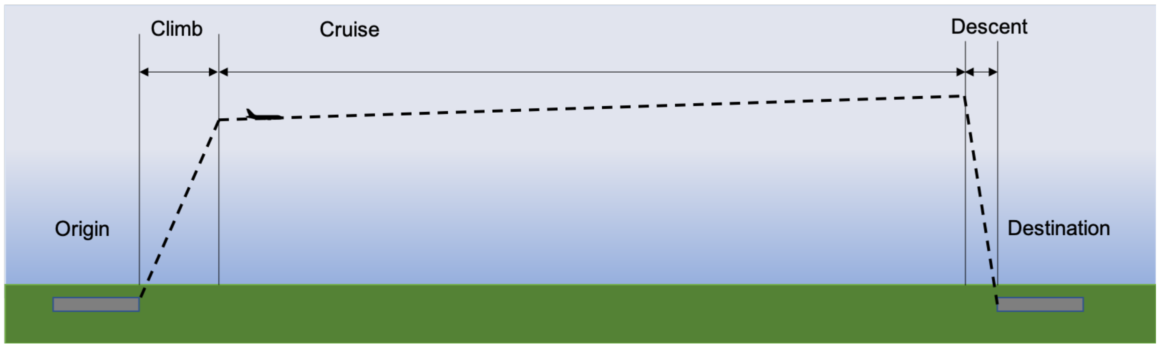


Figure 4-1: Single Aircraft CONOPS for a Hydrogen-powered aircraft

design to hydrogen engine design. Cryoplane was a project by Airbus to explore the feasibility of a liquid hydrogen fueled aircraft [2]. All of this work assumes a single aircraft with the simple mission profile as shown in Figure 4-1, and rather than optimization this work consisted of exploratory design studies that examined certain scenarios, without claims of optimality. The mission profile adopted is a simple climb and cruise with a cruise-climb incorporated to accommodate the change in fuel weight as the mission goes on. However, the cruise climb is not an imposed constraint, rather something that is potentially expected and shown in Figure 4-1.

However, here we adopt the flight profile optimization perspective: what if we changed the trajectory or even the operations paradigm to accommodate the properties of cryogenic LH2? Cryogenics requires a different operations mindset, with the trajectory the aircraft is flying affecting the boil-off rate of the aircraft. This requires a concurrent vehicle and mission design perspective. However, a total optimization has been computationally intractable in the past due to the complex modeling needed to capture these coupled effects. This chapter tackles the joint vehicle and trajectory design problem for a hydrogen aircraft with convex optimization.

4.2 Problem Formulation

For this mission, there are two parts that need to be modeled: the aircraft and the operations. The operations/mission profile is important because how the aircraft flies determines the boil-off rate at each point and how much hydrogen fuel is consumed. The environmental temperature has a big impact and depends on flight altitude. Furthermore, decreasing the boil-off rate may result in a tank that is too large and too heavy, negating any benefit from the higher energy density that is gained from using hydrogen. Therefore, our model needs to be able to model and capture those physical effects. We build off existing work to model these effects using convex optimization, specifically Signomial programming.

The overall problem is the following as described in Problem 4.1. We want to minimize the fuel burn of the aircraft such that the aircraft can still fulfill the missions that it is asked to complete. The trajectory analyzed is a simple climb and cruise segment with no descent profile modeled. The cumulative fuel burn is chosen as the objective of the optimization problem because the primary rationale of converting to liquid hydrogen as a fuel is because of the anticipated lower fuel burn required to complete the mission. In choosing this as an objective we are implicitly accepting a potential higher aircraft dry mass and gross takeoff weight (GTOW).

$$\begin{array}{lr} \textbf{minimize} & W_{burn} \\ \textbf{w.r.t.} & \text{Vehicle Design} \\ & \text{Trajectory Design} \\ \textbf{subject to} & \text{Range required} \\ & \text{Payload} \end{array} \tag{4.1}$$

The configuration chosen to analyze is a simple tube-and-wing configuration with engines below the wing. A single cryogenic tank is assumed to be at the wing longitudinal location of the aircraft to minimize the amount of tubing needed to get the liquid hydrogen to the wing to minimize fuel losses. This configuration however does split the passenger cabin into two: a forward cabin and an aft cabin; however, solving that problem is beyond the scope of this thesis.

4.2.1 Vehicle Design Formulation: Aircraft and Engine and Cryogenics

The aircraft model utilized is a modified version of SPaircraft which is a Signomial Programming model of TASOPT [72]. TASOPT utilizes low order physics model to analyze the aircraft design model from the aerodynamic, structural, thermal, and propulsion perspectives. This prior model is augmented with a cryogenics model that was developed as part of this thesis. The engine model used is the turbofan model found in the GPlibrary [6]. In this chapter we focus on direct combustion of hydrogen in a turbofan engine, and not fuel cells.

Cryogenic Model

The cryogenic Signomial model developed in this thesis is based off a NASA fuel tank sizing method outlined by Colozza [19]. Table 4.2 lists the parameters and variables used in this model. This model ensures basic first order physical principles including thermodynamics, states of gases, and mechanical properties are satisfied based off the flight state of the tank. Each parameter in Table 4.2 is either a variable, parameter, or a mission variable. A mission variable is a variable that is dependent on the mission/trajectory chosen by the optimizer. Figure 4-2 shows a basic schematic of the LH2 tank.

Symbol	Parameter	Parameter type

R	tank radius	variable
L	tank length	variable
A	tank surface area	variable
t_w	tank wall thickness	variable
t_{ins}	insulation wall thickness	variable
V_t	tank volume	variable
m_{ins}	mass of insulation	variable
m_t	mass of tank structure	variable
m_{tank}	mass of cryogenic tank	variable
$M_{boil-off}$	boil-off rate	variable
h_{ext}	external convective heat transfer coefficient	variable
Nu_L	Nusselt number	variable
Pr	Prandtl number	variable
Re	Reynold's number	variable
T_s	tank surface temperature	variable
Q_{in}	heat flux in	variable
Q_{out}	heat flux out	variable
ρ_{CC}	tank wall density	parameter
ρ_{ins}	insulation density	parameter
ρ_{LH2}	LH2 density	parameter
P_{LH2}	LH2 pressure	parameter
T_{LH2}	LH2 temperautre	parameter
V_i	extra volume for gaseous hydrogen as percentage	parameter
FoS	Factor of safety	parameter
σ_y	Wall material stress limit	parameter
K_{ins}	thermal conductivity of insulation material	parameter

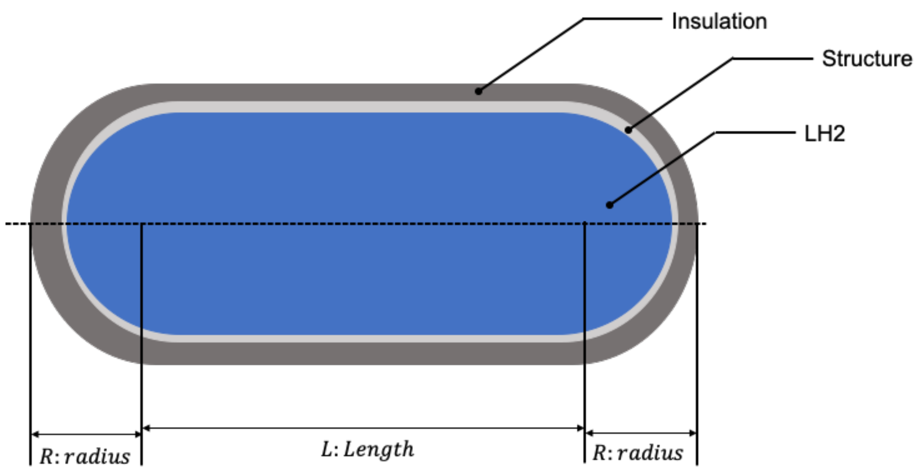


Figure 4-2: Schematic of Cryogenic Tank

K_{air}	thermal conductivity of air	parameter
ϵ_{air}	emissivity of air	parameter
h_{fg}	heat of vaporization of LH2	parameter
σ	Boltzmann's Constant	parameter
C_p	specific heat at constant pressure	parameter
μ_{air}	dynamic viscosity of air	mission variable
T_∞	ambient temperature	mission variable
V_∞	flight velocity	mission variable
ρ_∞	flight velocity	mission variable
W_{LH2}	weight of LH2 contained	mission variable

Table 4.2: Cryo Tank Model Variables and Parameters

A primary constraint for the cryogenic tank is to be able to contain the fuel it contains. However, the tank also needs to be filled with gaseous hydrogen as temperature fluctuations and pressure fluctuations will cause the hydrogen to fluctuate in state as well. Therefore, to ensure that some hydrogen can be vented off in case pressures and temperatures rise, some gaseous hydrogen is maintained within the tank. To allow for that, a monomial constraint is defined to allow for that with V_i providing that safety margin in equation 4.2 where g is the graviational acceleration. To ensure the geometric dimensions are also constrained properly, the Signomial constraint in equation 4.3 is added to make sure the tank volume does not exceed the dimensions specified by the optimizer.

$$V_t = \frac{W_{LH2}V_i}{\rho_{LH2}g} \tag{4.2}$$

$$V_t \leq \frac{4\pi R^3}{3} + \pi R^2 L \tag{4.3}$$

The surface area of the cylindrical tank with hemispherical end caps can be converted into a posynomial constraint as shown in equation 4.4:

$$A \geq 4R^2\pi + 2R\pi L \tag{4.4}$$

To ensure that the tank walls are sized to be thick enough to be handle the tank pressures, a hoop stress constraint is incorporated in equation 4.5.

$$t_w \geq \frac{PRFoS}{2\sigma_y} \tag{4.5}$$

The mass of the entire cryogenic system can be calculated using material densities. Although the more accurate estimate would be to calculate the actual material thickness at each location including the curvature, to keep as many of the constraints GP-compatible, surface areas are used to simplify the calculation as shown in equation 4.6, 4.7, 4.8. These can be made into posynomial constraints because we can expect the optimizer to add additional pressure to the constraints.

$$m_t \geq \rho_{CC}At_w \tag{4.6}$$

$$m_{ins} \geq (2\pi LR + 4\pi R^2)t_{ins}\rho_{ins} \tag{4.7}$$

$$m_{tank} \geq m_{ins} + m_t \tag{4.8}$$

Now that the geometric, structural and weight property constraints have been defined, the thermodynamic properties of the cryogenic tank need to be modeled. At the core, the basic thermodynamic principle followed here is that the heat flux in has to equal the heat flux going out while maintaining the cryogenic temperature inside the tank:

$$Q_{in} = Q_{out} \tag{4.9}$$

The heat fluxes can be calculated through accounting for convection, conduction and radiation as seen in equation 4.10 and 4.11. This is a signomial equality constraint, and made such to ensure that proper thermodynamics are accounted for.

$$Q_{in} + h_{ext}T_s + \sigma\epsilon T_s^4 = h_{ext}T_\infty + \sigma\epsilon T_{\mathrm{inf}^4} \tag{4.10}$$

$$Q_{out} + \frac{T_{LH2}K_{ins}}{t_{ins}} = \frac{T_sK_{ins}}{t_{ins}} \tag{4.11}$$

The h_{ext} can be calculated by determining the Nusselt number (Nu_L):

$$Nu_L = \frac{h_{ext}R}{k_{air}} \tag{4.12}$$

The Nusselt number can be estimated using correlation between the Prandtl number and the Reynolds number given in Verstraete [70] . This is converted into a equality constraint:

$$Nu_L = 0.03625(Pr^{0.43})(Re^{0.8}) \tag{4.13}$$

where the Prandtl number Pr and Reynolds number Re can be calculated in using the flight state information:

$$Re = \frac{\rho_{\text{inf}} V R}{\mu_{air}} \tag{4.14}$$

$$Pr = \frac{\mu_{air} C_p}{k_{air}} \tag{4.15}$$

Finally, the hydrogen boil-off rate can be calculated using the following thermodynamic relation and Signomial inequality constraint. The optimizer will attempt to decrease the boil-off rate, so this we can use optimizer pressure to push the inequality such that is an active constraint. An interesting question will be to see how much boiloff - if any - the optimizer will allow. While in theory zero boiloff is possible the added mass of equipment to achieve it may not be worth it.

$$M_{boil} + \frac{K_{ins} A T_{LH2}}{t_{ins} h_{fg}} \geq \frac{K_{ins} A T_s}{t_{ins} h_{fg}} \tag{4.16}$$

For this model, the default structural material for the tank wall that was chosen is carbon composite. The default insulation material chosen is rigid closed cell polymethacrylimide. The specific material properties are given in Colozza [19].

4.2.2 Mission: Flight Profile

The mission model utilized is the mission profile used in SPaircraft. However, to accomodate the cryogenics, some modifications were required. The cryogenic tank has a boil-off rate (M_{boil}). Therefore, in the fuel-burn calculation, the boil-off rate must be taken into account in addition to the actual fuel burn in the engines. Following the

Property	Value
Heat of Combustion of Jet Fuel h_f	120 MJ/kg
Cp Value for Fuel/Air Mix in Combustor C_{p_c}	1506 J/K/kg
Specific Heat Capacity of the fuel $C_{p_{fuel}}$	9780 J/K/kg

Table 4.3: LH2 Fuel Properties

convention used in the SPaircraft documentation, the constraint that was modified: this was calculated in the fuel-burn calculation portion of the model as shown in Equation 4.17.

$$W_{burn} \geq n_{eng}(TSFC)(F)(t) + t * M_{boil_{off}} g \tag{4.17}$$

4.2.3 Model Integration

The only other modifications made to the SPaircraft model was to accommodate the geometry and ensure that it is integrated into the aircraft model. This means for example that the cryogenic tank radius does not exceed the fuselage constraints, etc. Furthermore, the only engine modifications made were to the fuel properties. This was chosen because this allows determining what if liquid hydrogen were used in current engines and currently no public data exists for hydrogen-specific turbofans. The fuel properties used are listed in Table 4.3.

4.3 Vehicle and Flight Profile Design Optimization

The case of interest examined is to determine what the performance would be if we were to simply retrofit a 737/777 style tube and wing aircraft into a hydrogen aircraft. We also allow the mission profile to be optimized to allow for some operations change to accommodate the properties of LH2; however, as a validation case, for these results, we limit the operating bounds to be the same as the current trajectories flown by current passenger aircraft.

Two separate design missions are examined: a 737-type mission designated as SR (short range) and a 777-type mission designated as LR (long range). Table 4.4 shows the mission optimization parameters. The default optimization parameters for relative tolerance found in SPaircraft are used.

Parameter	LR	SR
Range [nmi]	6000	3000
Passengers	450	180
Cruise Segments	4	4
Climb Segments	4	4
M_{min}	0.84	0.8
h_{min} [ft]	32000	32000
Tolerance	0.01	0.01

Table 4.4: Optimization and Mission Parameters

4.3.1 Convex Architecture

The convex architecture chosen here is the combined architecture. Both the vehicle and the mission in this case have been formulated as a SP problem and implemented as a combined architecture in SPaircraft [42]. Because the mission and the cryogenics are simple and predictable, we can model the problem as a SP. The alternating and integrated architectures could easily be implemented here. However, there is no need to do so as Signomial programs are flexible enough to accommodate both the vehicle and mission together.

4.3.2 Short Range Mission

Both optimizations converged within 7 GP iterations and approximately 30 seconds. Each mission profile is divided into two phases: climb and cruise with 4 segments for climb and 4 segments for cruise. The results for the 737-style mission are presented here.

Vehicle Design Results

The weight build up for both the 737 model and the LH2-SR aircraft are shown in Figure 4-3 and 4-4. As one can see, the fuel fraction dramatically decreased with the conversion to LH2 as a fuel though the mass penalty of the cryogenics did cut into gains just slightly, accounting for approximately 15% of the fuel + tank weight. Surprisingly, the wing weight did not increase as much as expected due to the lack of load alleviation from the fuel during flight. Overall the hydrogen aircraft is about 18,000 lbs lighter at takeoff even though the fuselage is about 3 meters longer and the wing reference area slightly larger. This is consistent with conventional wisdom which says that hydrogen aircraft, for the same payload and range, will be somewhat larger compared to kerosene-powered aircraft.

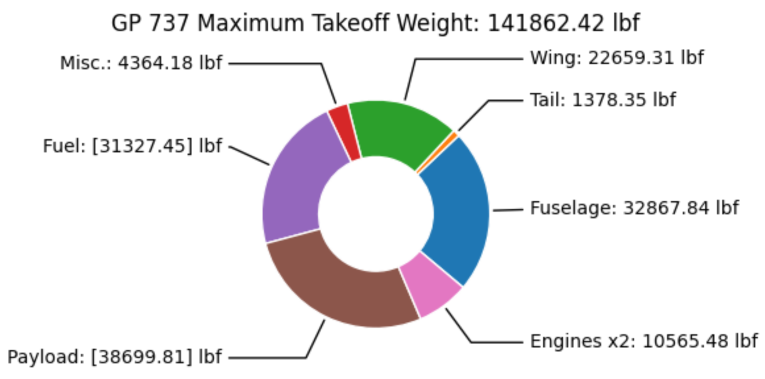


Figure 4-3: 737-Model Weight Build-up: 141,862 lbs

Parameters	737-model	LH2 SR
b [m]	35.8	37.92
AR	11.9	12.45
c [m]	5.21	5.30
S_{ref} [m^2]	107.3	115.5
L_{fuse} [m]	36.57	39.38
R_{fuse} [m]	1.85	1.85

Table 4.5: SR Aircraft Geometry

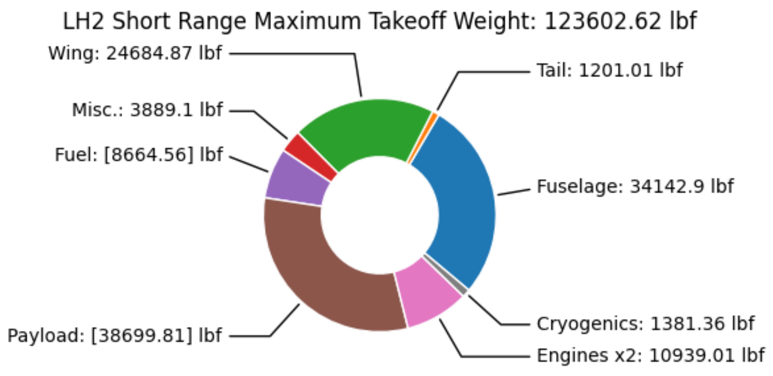


Figure 4-4: LH2-SR Weight Build-up: 123,603 lbs

Table 4.5 presents the geometric parameters of the LH2 short range aircraft compared to the conventional 737-model. The geometry did not increase much; however, the S_{ref} surprisingly increased slightly. Interestingly, the optimizer decided to fit the cryogenic tank within the 737 fuselage rather than expand the fuselage, a surprising design choice given that boil-off is an important consideration. As expected, the fuselage length increased to accommodate the cryogenic tank in the center while also accommodating the passengers at the front and back.

Table 4.6 presents the engine performance for the design that the optimizer settled on. As expected, the TSFC for the LH2 aircraft was approximately 3 4 times lower than the kerosene based aircraft. The max thrust for the LH2 engine also decreased by almost 25 % which resulted in the inlet area able to decrease as well as the reference area for the nacelle to decrease.

Finally for the vehicle design, Table 4.7 shows the cryogenic tank design. The

Parameters	737-model	LH2 SR
TSFC (climb, avg.) [1/hr]	.767	.225
TSFC (cruise, avg.) [1/hr]	.607	.182
BPR (max)	4.86	4.13
F (max) [N]	46314	36831
OPR (max)	35	35
A_{inlet} [m^2]	3.833	3.79
$S_{nacelle}$ [m^2]	9.58	9.47

Table 4.6: SR Engine Performance

Parameters	LH2 SR
t_w (m)	.0011
t_{ins} (m)	.188
m_w (kg)	123.2
m_{ins} (kg)	503.1
m_{sys} (kg)	627.0
L (m)	2.8
R (m)	1.85
V_t (m^3)	56.97
A (m^2)	75.86
W_{LH2} (lbf)	8655
$M_{boiloff}$ (climb) (kg/s)	.0020
$M_{boiloff}$ (cruise) (kg/s)	.00158

Table 4.7: SR Cryogenic Tank Properties

boil-off rate is pretty high as the aircraft loses approximately 1 kg of fuel per 10 min. However, this is an acceptable loss for the optimizer, especially for this short of a flight. The cryogenic weight in the end is dominated by the insulation weight which is almost 20 cm thick.

Trajectory Design

The mission design results are interesting to examine. Figure 4-5a shows the altitude profile for the entire mission. The hydrogen aircraft chooses a higher initial altitude compared to the traditional B737. This makes physical sense as the higher the aircraft goes, the colder it gets which minimizes the boil-off. The aircraft does not go higher than 45000 ft as once it gets higher, the wing area must also increase which may result in a larger fuel-burn and diminishing returns.

The velocity profile and mass profile are presented in Figure 4-5b and 4-6. There is nothing particularly different or unexpected here, as both of these plots make sense. The LH2-SR chooses to go as slow as it can to decrease the Reynolds number which increases the Nusselt number which increases the external convective heat transfer coefficient which in turn increase the boil-off rate. Therefore, in our model, the slower the vehicle goes, the less boil-off there is. The mass profile also makes sense since the TSFC and the thrust of the aircraft is also much lower compared to the kerosene counter part.

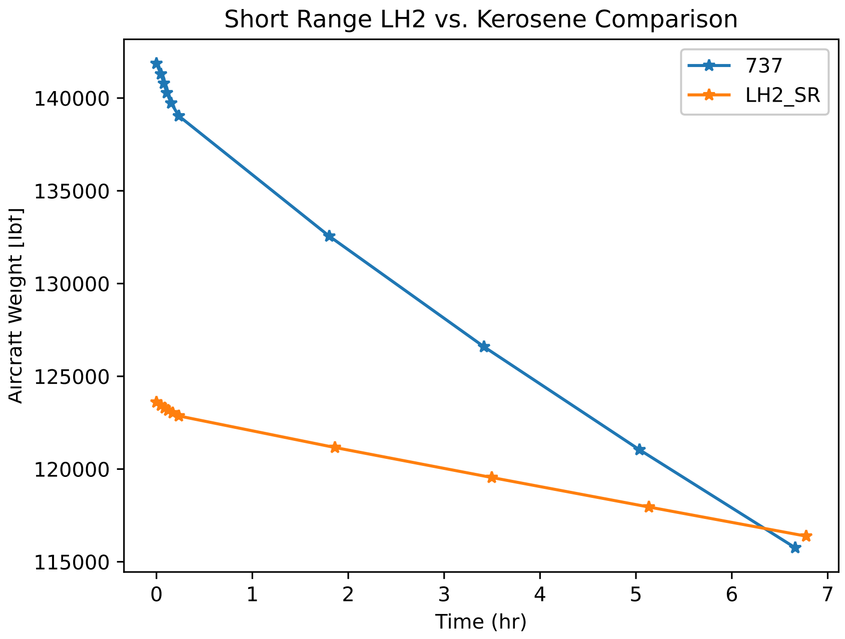
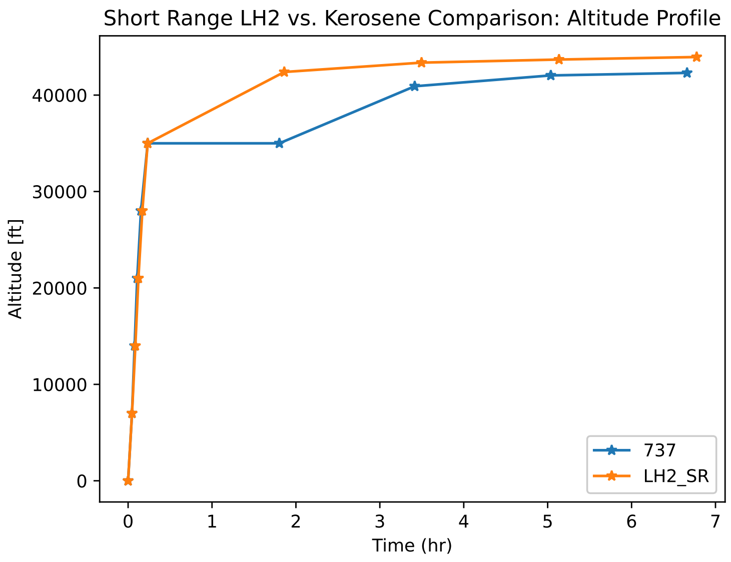
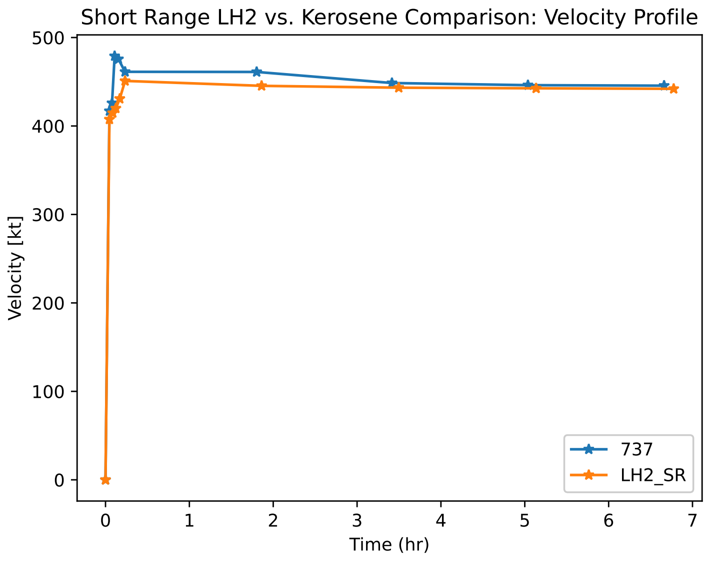


Figure 4-6: SR Mass Profile

We can look at the some of the major sensitivities dW_{fuel}/dx related to the mission and the cryogenic tank for the short range mission. Some notable parameter sensitivities are listed in Table 4.8. As expected, the minimum cruise Mach number is the most sensitive variable in terms of the mission, and increases in sensitivity when hydrogen fuel is substituted for kerosene. For the cryogenics itself, the volume plays a



(a) Altitude Profile



(b) Velocity Profile

Figure 4-5: Altitude and Velocity Profile for LR Mission

large role in the cryogenic tank sizing, and therefore is most sensitive. The insulation material is the next most sensitive parameter as expected as it drives the weight of the tank itself.

Parameter	737	LH2	Description
M_{min}	+0.58	+0.7	Minimum cruise Mach
MinCruiseAlt	+0.0059	+0.0052	Minimum Cruise altitude
MaxClimbTime	-0.093	-0.017	Total time in Climb
V_i	N/A	+0.048	Extra volume fraction for GH2
ρ_{LH2}	N/A	-0.048	Density of liquid hydrogen
h_{fg}	N/A	-0.013	Heat of vaporization
K_{ins}	N/A	+0.013	Thermal conductivity of insulation
ρ_{ins}	N/A	+0.013	Density of insulation
FoS	N/A	+0.0033	Factor of safety
P_{LH2}	N/A	+0.0033	Pressure hydrogen is stored at
ρ_{CC}	N/A	+0.0033	Density of carbon composite
σ_y	N/A	-0.0033	Carbon composite stress limit
T_{LH2}	N/A	-0.0014	Temperature of LH2

Table 4.8: Notable Sensitivities for SR Mission

4.3.3 Long Range Mission

Both optimizations converged within 7 GP iterations and approximately 30 seconds. Each mission profile is divided into two phases: climb and cruise with 4 segments for climb and 4 segments for cruise. The results for the 777-style mission is presented here.

Vehicle Design Results

The weight built-up of both the 777 model and the LH2-LR is shown in Figure 4-7 and 4-8. Similar to the 737-style mission, fuel weight decrease for the hydrogen aircraft is dramatic: the fuel weight decreases by almost 75% from 141,000 lbs to only 40,500 lbs. The cryogenic equipment does cut into that gain with approximately 4,000 lbs of added weight. On balance, however, this appears to be a worthwhile tradeoff. However despite this, the MTOW of the LH2 variant decreases by approximately 25%.

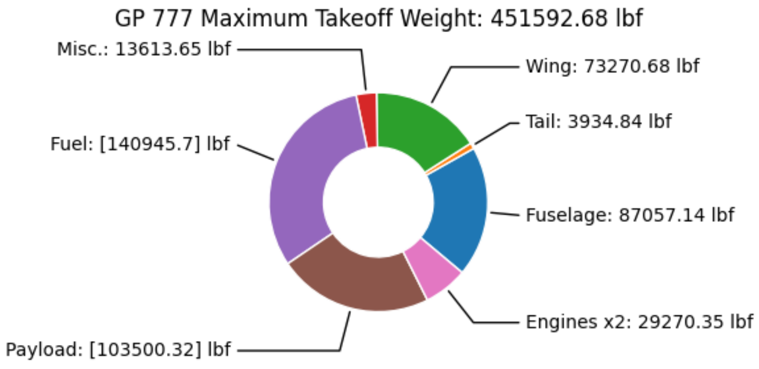


Figure 4-7: 777 model Weight Build-up

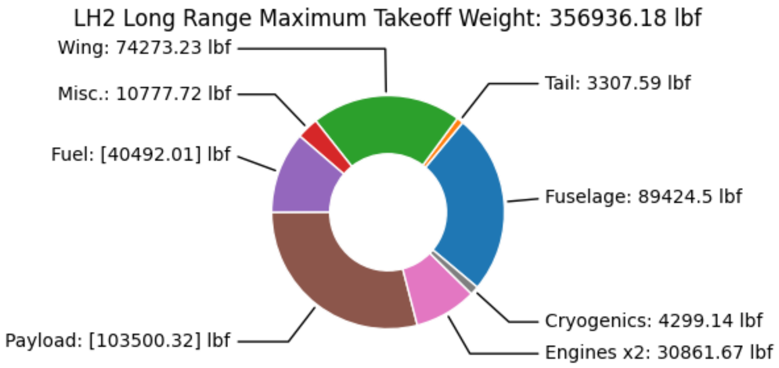


Figure 4-8: LH2-LR Weight Build-up

Table 4.9 shows the geometric parameters of the modified aircraft. The wing design for the LH2 variant is surprisingly similar to the kerosene counterpart except for the wing span that is 1 m longer. Similar characteristics from the short range variant are also apparent in the long range variant where the optimizer chose not to increase the radius of the fuselage to accommodate a wider radius cryogenic tank.

Parameters	777-model	LH2 LR
b (m)	55.7	58.3
AR	10.5	10.76
c (m)	9.2	9.4
S_{ref} (m^2)	295.8	315.3
L_{fuse} (m)	55.96	60.58
R_{fuse} (m)	3.1	3.1

Table 4.9: LR Aircraft Geometry

Table 4.10 shows the engine performance characteristics. The TSFC shows similar decreases as the short range variant; interestingly, the BPR shows a dramatic decrease despite similar thrust requirements. This may indicate that the optimizer decided to go with a less efficient engine to ensure that the LH2 variant can continue to produce enough thrust at lower than optimal engine efficiencies.

Parameters	777-model	LH2 LR
TSFC (climb, avg.) (1/hr)	.771	0.25
TSFC (cruise, avg.)(1/hr)	.597	0.193
BPR (max)	5.6	4.16
F (max) (N)	121017	110000
OPR (max)	42	42
A_{inlet} (m^2)	7.8	7.36
$S_{nacelle}$ (m^2)	19.5	18.27

Table 4.10: LR Engine Performance: Kerosene versus LH2

Parameters	LH2 LR
t_w (m)	0.00178
t_{ins} (m)	0.184
m_w (kg)	576.7
m_{ins} (kg)	1373
m_{sys} (kg)	1951.2
L (m)	4.62
R (m)	3.11
V_t (m^3)	266.2
A (m^2)	211.7
W_{LH2} (lbf)	40490
$M_{boiloff}$ (climb) (kg/s)	.0053
$M_{boiloff}$ (cruise) (kg/s)	.0045

Table 4.11: LR Cryogenic Tank Properties

Finally for vehicle design, Table 4.11 shows the cryogenic tank dimensions. The LR variant and the SR variant actually end up having similar dimensions in terms of material thicknesses, with about 18 cm thick insulation. However, the LR tank is significantly larger in terms of length and radius, which was expected. With this comes a penalty, the boil-off rate does increase since the surface area increased; therefore, a three-fold increase in boil-off rate is observed. However, it appears that because the fuel weight is decreased so much compared to the kerosene baseline that this amount of LH2 boiloff does not seem to matter much. The optimizer also chooses to keep the cryogenic tank within the fuselage dimensions of the aircraft, and does not increase the radius of the tank for a lower boil-off rate even with carrying the same amount of payload and cargo. This is a curious decision as the LR variant has a much longer flight; however, the optimizer may have decided that it was worth it to accept this higher boil-off rate rather than carry a much larger cryogenic tank with more insulation.

Trajectory Design

The same trends can be seen for the mission profiles for the LR variant as we saw in the SR variant. Figure 4-9a shows the altitude profile for both the 777-model and the LH2-LR, and as expected, the LH2-LR goes for a much higher altitude earlier and stays there. This makes sense as the higher altitude, the less boil-off the cryogenic tank will have due to the colder ambient temperature.

Figure 4-9b and Figure 4-10 shows the velocity and the mass profiles respectively. The velocity profile does not change much though the LH2-LR hugs the lower bound of the velocity of the optimization while the 777 goes a bit faster initially. The weight profile also shows the dramatic decrease in weight of the LH2-LR as expected due to the lower TSFC and the lower thrust required.It is interesting to note that while the LH2-LR aircraft is lighter than the kerosene-powered 777 for most of the mission, at the very end of the flight the empty weight of the 777 is slightly less than the empty LH2-LR. This is due to the fact that the LH2-LR is slightly larger structurally due to a larger wing and longer fuselage and the extra mass of the cryogenics equipment.

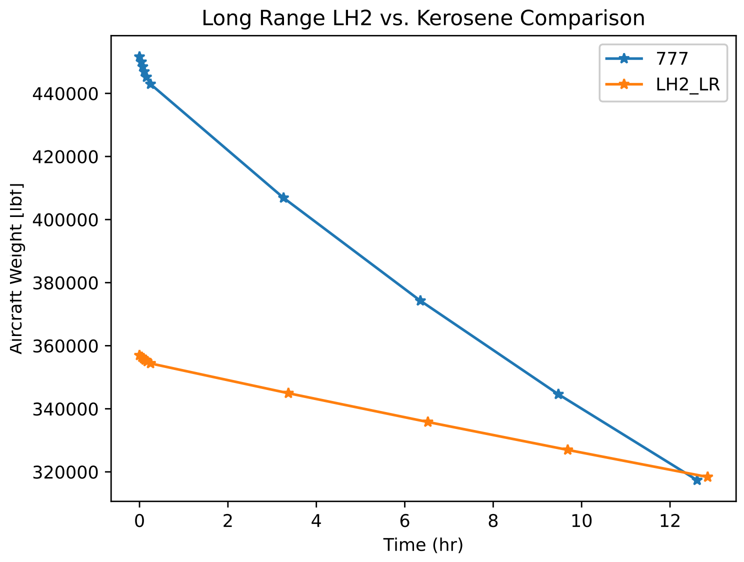
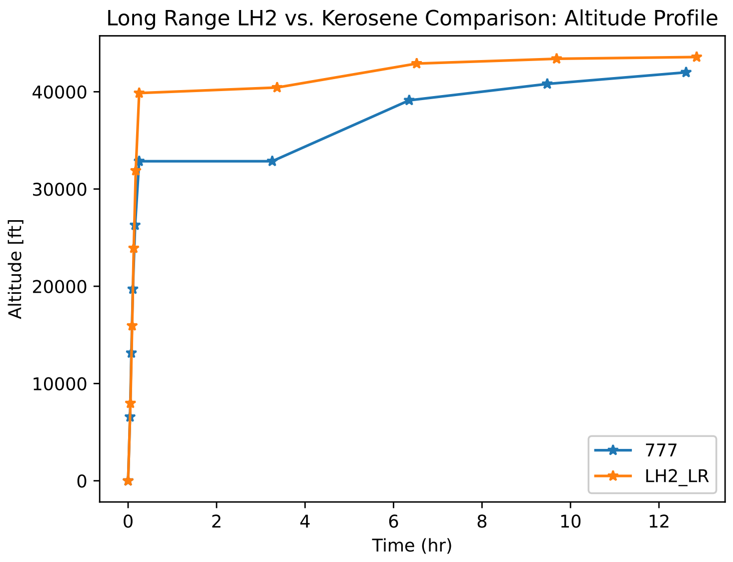
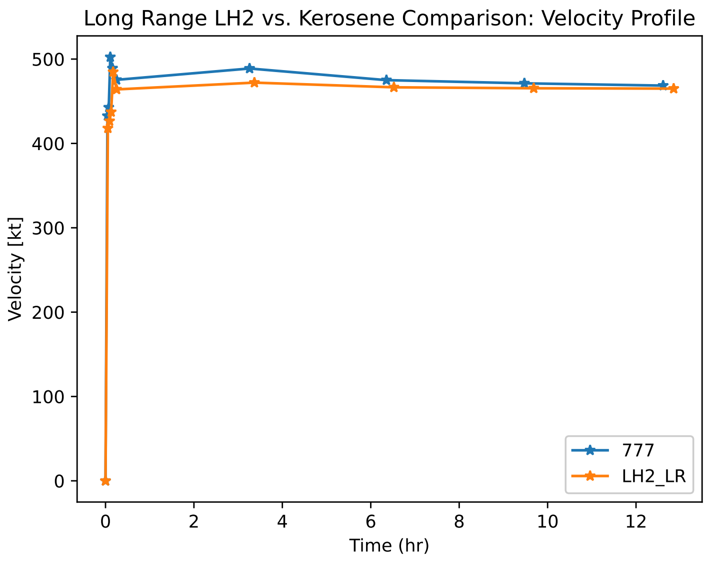


Figure 4-10: LR Mission Mass Profile



(a) Altitude Profile



(b) Velocity Profile

Figure 4-9: Altitude and Velocity Profile for LR Mission

Parameter	777	LH2 LR	Description
M_{min}	+0.45	+0.53	Minimum cruise Mach
MinCruiseAlt	+0.0	0.0	Minimum Cruise altitude
MaxClimbTime	-0.089	-0.043	Total time in Climb
V_i	N/A	+0.09	Extra volume fraction for GH2
ρ_{LH2}	N/A	-0.09	Density of liquid hydrogen
h_{fg}	N/A	-0.015	Heat of vaporization
K_{ins}	N/A	+0.015	Thermal conductivity of insulation
ρ_{ins}	N/A	+0.015	Density of insulation
FoS	N/A	+0.0061	Factor of safety
P_{LH2}	N/A	+0.0061	Pressure hydrogen is stored at
ρ_{CC}	N/A	+0.0061	Density of carbon composite
σ_y	N/A	-0.0061	Carbon composite stress limit
T_{LH2}	N/A	-0.0016	Temperature of LH2

Table 4.12: Notable Sensitivities for LR Mission

The notable parameter sensitivities comparison are shown in Table 4.12. The most sensitive parameter within the mission space for the 777 is the minimum Mach number, and that parameter also increases sensitivity as the LH2 is incorporated. Surprisingly, the climb time sensitivity decreases when the hydrogen is incorporated, because one could expect the vehicle to climb quickly to spend as much time in colder atmospheric temperatures. For the cryogenics itself, the most sensitive parameters as expected are related to the volume of the hydrogen tank. Furthermore, the material selection of the insulation is the second most sensitive for the cryogenic model.

4.4 Cryogenic Technology Case Study

The previous section showed that a direct conversion to liquid hydrgen as a fuel has enabled a significant decrease of TSFC which allows for a decrease of total fuel burn by approximately 60%. Now that we have a Signomial model working for a hydrogen aircraft, we can begin looking at ways to obtain even further reductions in fuel burn while keeping the mission range and payload fixed. Because our model allows the optimizer to change both the trajectory and the aircraft design at the same time. In this section, two changes are investigated: mission constraints and cryogenic material. These two changes are selected due to the sensitivity analysis which showed that these two have large effects on the overall design.

4.4.1 Modified Mission Limits

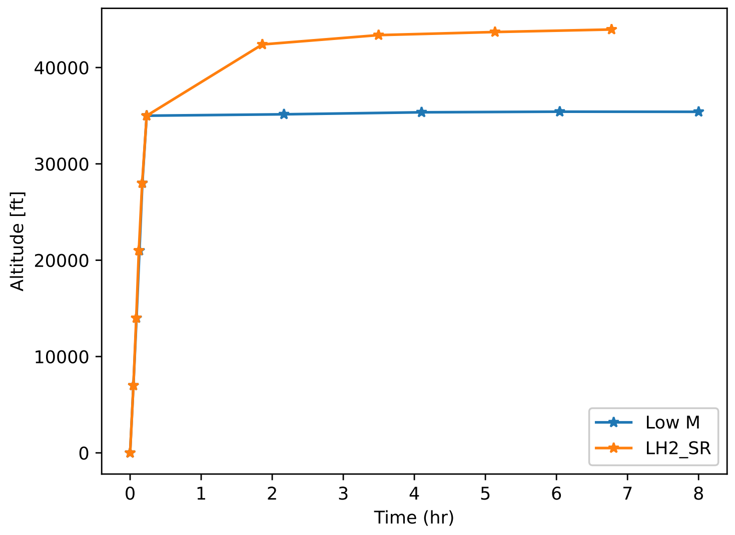
In the previous section, the current passenger aircraft mission profile is used for the hydrogen aircraft. The general strategy for current passenger aircraft is to fly at transonic speeds and as fast as possible without incurring a large wave drag penalty (typically at about Mach 0.84). However, with the cryogenic fuel tank requiring a minimum surface area, the cryogenic fuel tank design may push the fuel tank to minimize surface area by increasing the radius of the tank, approximating as closely as possible a spherical tank geometry. To determine that effect, we modify the mission limits in the following in Table 4.13. The only modification is the cruise Mach number which is set to 0.6, significantly slower than 0.84. The optimization was run again for the LR and SR missions, and the results are shown in this section.

Parameter	LR	SR
Range [nmi]	6000	3000
Passengers	450	180
Cruise Segments	4	4
Climb Segments	4	4
M_{min}	**0.6**	**0.6**
h_{min} [ft]	32000	32000
Tolerance	0.01	0.01

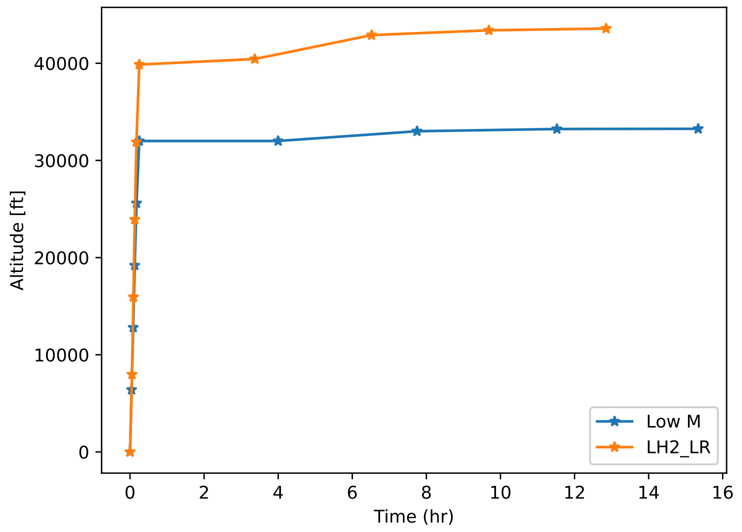
Table 4.13: Modified Mission Optimization and Mission Parameters

The mission changes that occurred are shown in Figure 4-11 to 4-14. Figure 4-11 shows the changed altitude profile that the optimization settled on. For the lower

Mach number, the optimizer decided to fly at a lower altitude for both the SR mission and the LR mission as shown in Figure 4-11a and Figure 4-11b. This is most likely due to the optimizer choosing a lower speed, as evidenced by the time of flight difference.



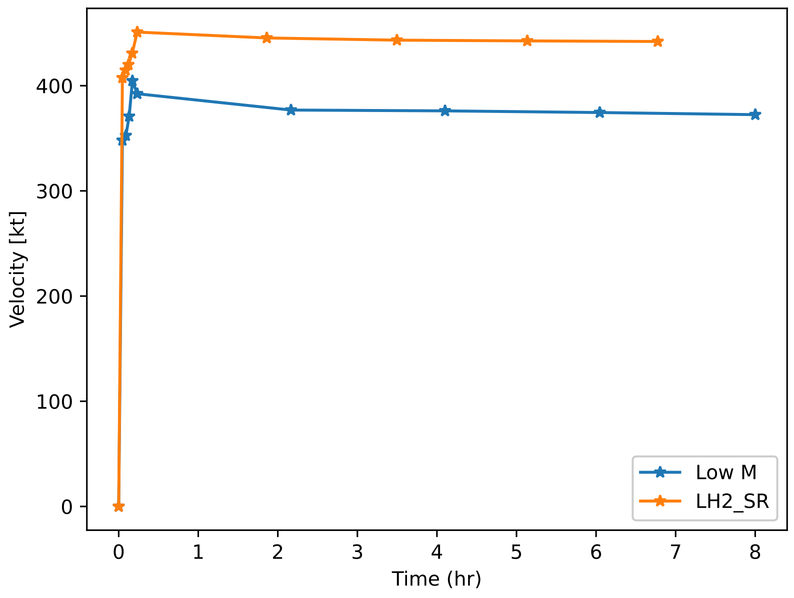
(a) SR Mission



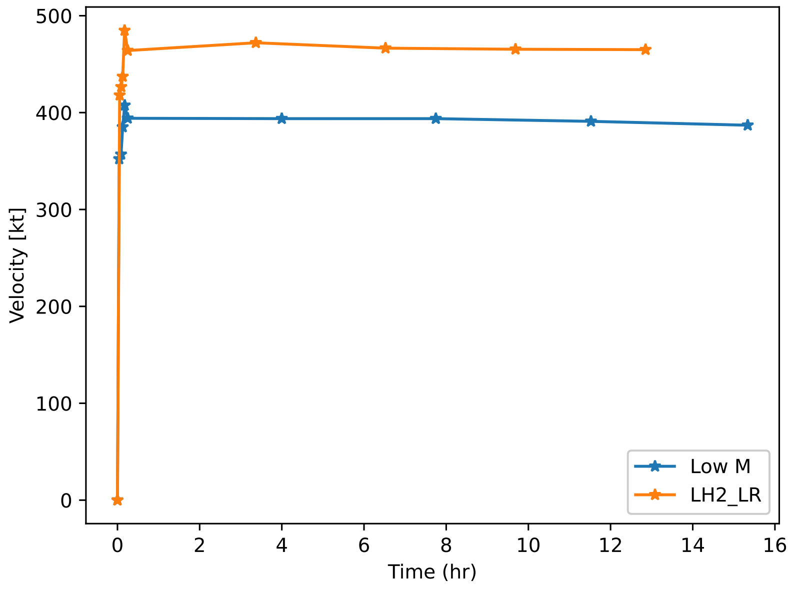
(b) LR Mission

Figure 4-11: Mission Altitude Profile for Modified Mission Limits

Figure 4-12 shows the actual velocity profiles. As expected the both the LR mission and the SR missions did immediately go to a lower Mach number as shown in Figure 4-12b and 4-12a. However, the effects of the modified mission profile on the vehicle need to be explained.



(a) SR Mission

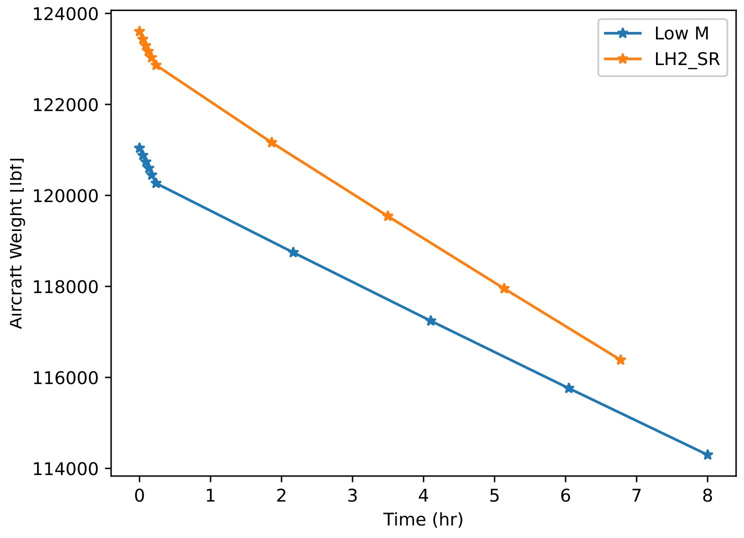


(b) LR Mission

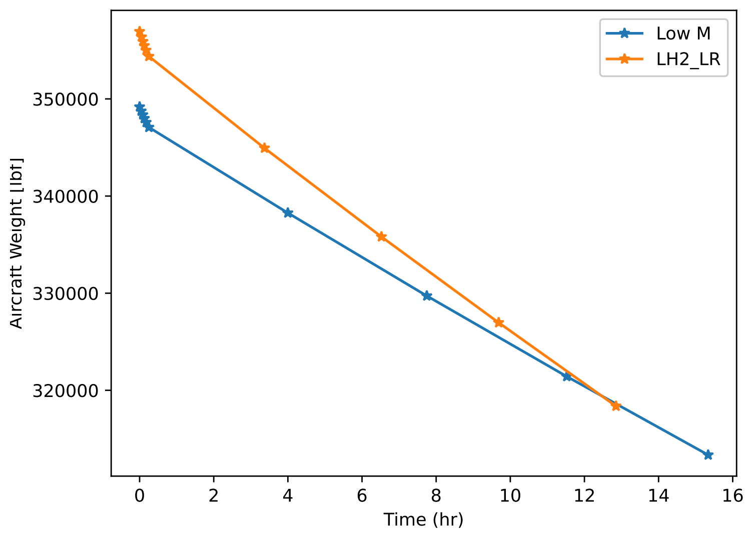
Figure 4-12: Mission Velocity Profile for Modified Mission Limits

The weight change profile for the both the LR and SR mission is shown in Figure

4-13. Both missions show that the vehicle starts at a lighter vehicle weight and ends at a lower vehicle weight. Furthermore, the slope of the cruise phase is less steep for both the SR and LR missions, indicating that reduced fuel burn was achieved for both variants.



(a) SR Mission



(b) LR Mission

Figure 4-13: Mission Weight Profile for Modified Mission Limits

An examination into just the boil-off rate is shown in Figure 4-14. Interestingly, the lower Mach mission profile does decrease the total amount boil-off; however, with a lower boil-off rate than the higher Mach mission counterpart. This is primarily due to the longer mission time; if the lower Mach mission time was shorter, it would still have a lower boil-off mass than the original mission profile.

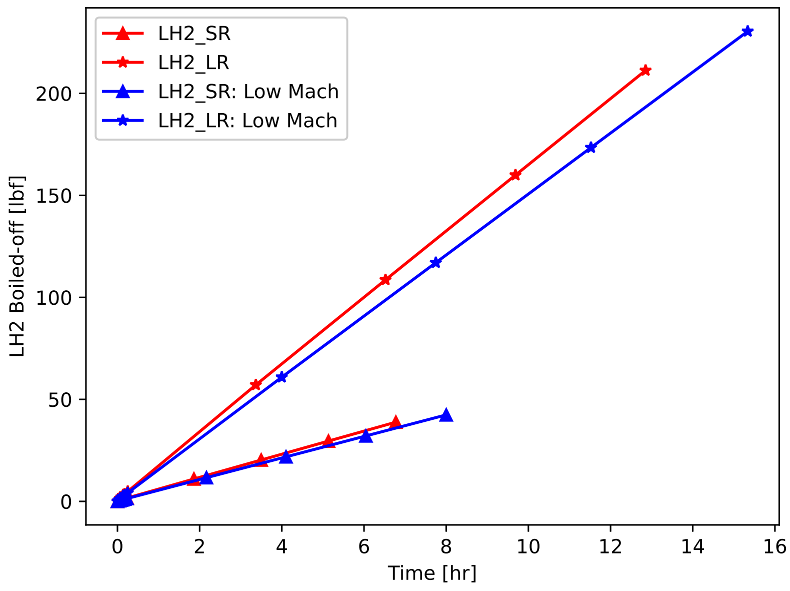
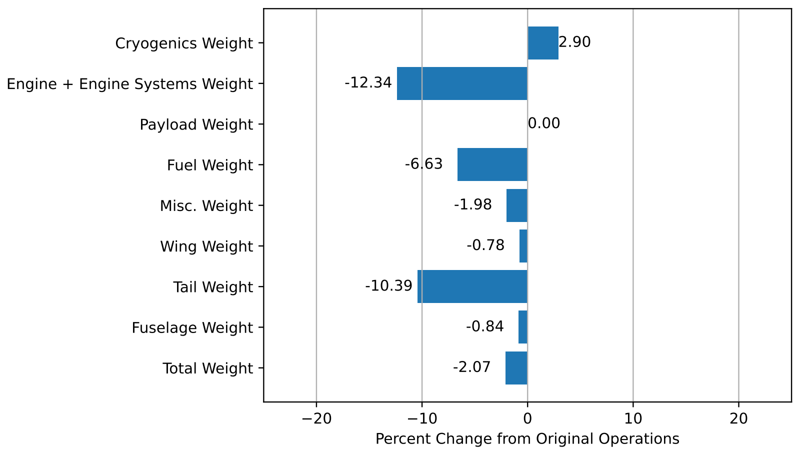
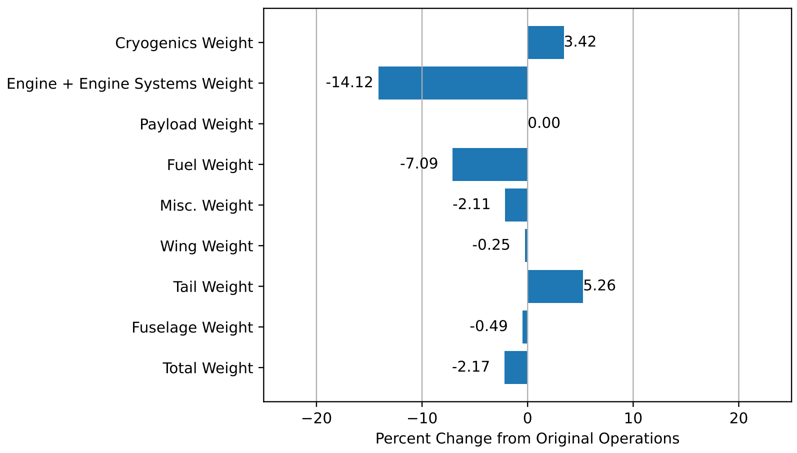


Figure 4-14: Boil-off Profile

Now that the mission changes have been explored, the resultant changes to the aircraft design can be examined. The change in weight build-up is shown in Figure 4-15. As expected, the the total weight and the fuel weight decreased compared to the higher cruise Mach number minimum. The fuel weight decrease is expected because flying slower usually means less thrust is required (thrust goes with velocity squared, power goes with velocity cubed). The engine weight also decreased significantly for both missions, and this makes sense as a smaller engine is most likely chosen. Both the LR mission and the SR mission also had increases in cryogenic weight, most likely to keep the boil-off weight low given the longer mission duration.



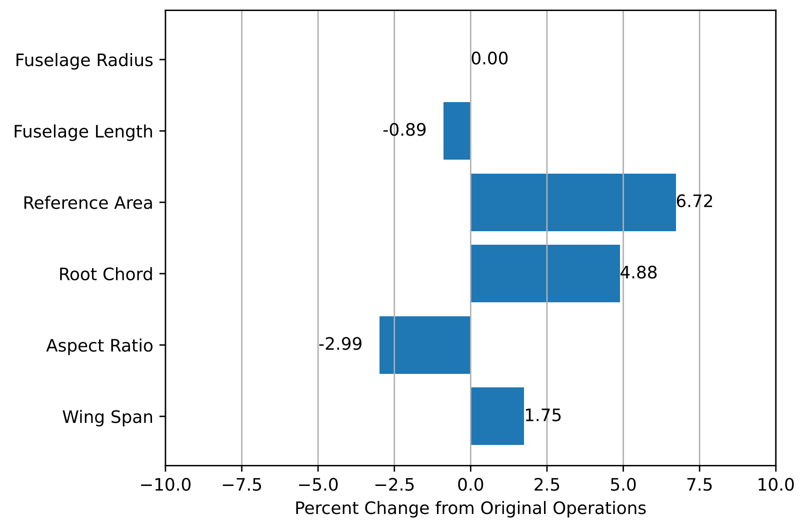
(a) SR Mission



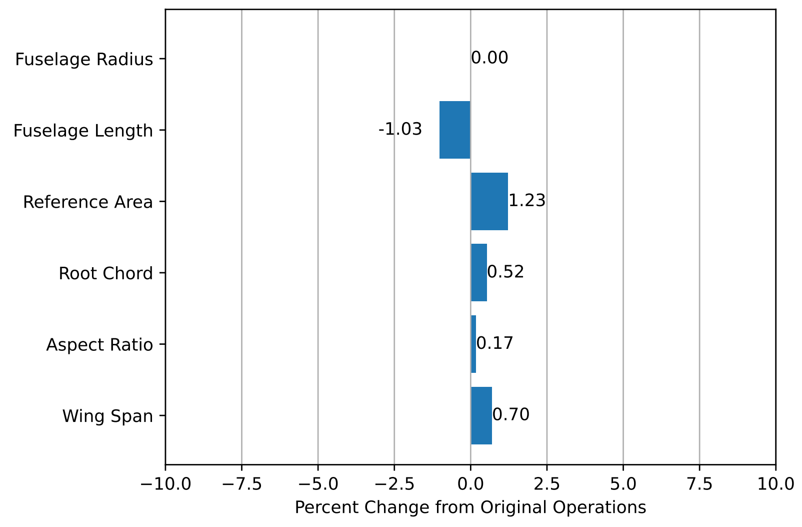
(b) LR Mission

Figure 4-15: Weights Design Point for Modified Mission Limits

The aircraft geometry change from the modified mission shown in Figure 4-16. The aircraft geometry has an increased wing area to compensate for the slower velocity. Interestingly, even for the slower cruise altitude, both missions converged to no change in fuselage radius. This means that the optimizer decided not to change the tank radial geometry. In fact, the optimizer was able to take advantage of the decrease fuel weight to decrease the length of the tank.



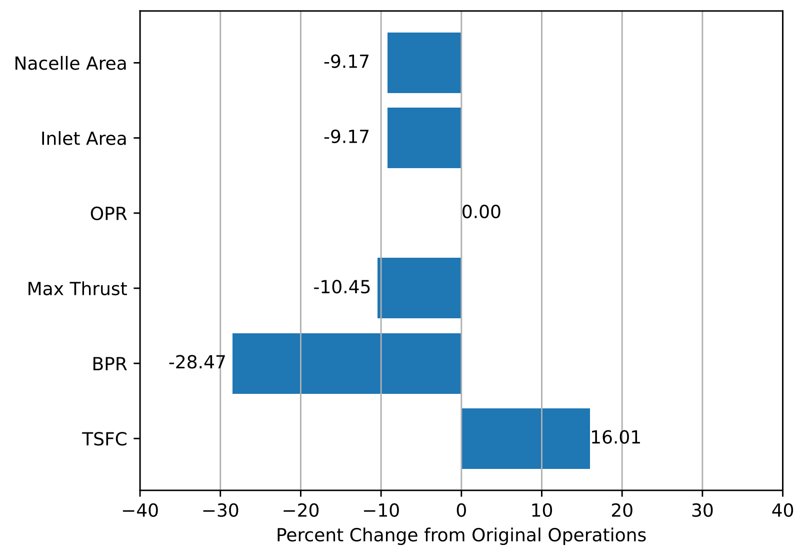
(a) SR Mission



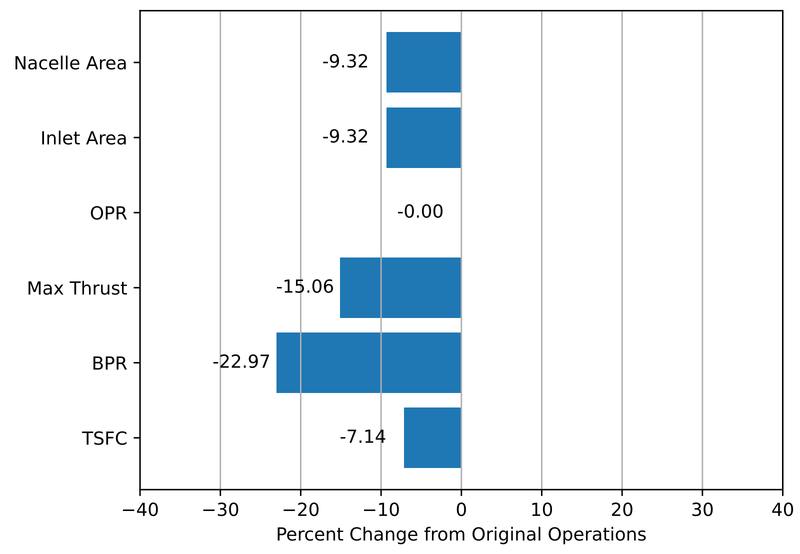
(b) LR Mission

Figure 4-16: Geometry Change for Modified Mission Limits

Figure 4-17 shows the engine design changes. Notably, the SR mission showed an increase in TSFC in Figure 4-17a. This is most likely due to the decrease in the bypass ratio, and is acceptable since the maximum thrust is decreased. Interestingly, the LR mission shows the opposite trend where TSFC decreases even with the smaller BPR. This is most likely due to the lower thrust the engine is producing.



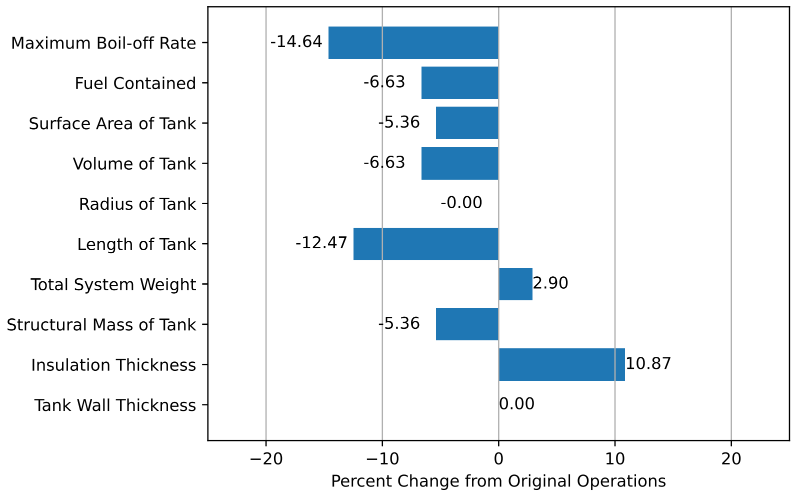
(a) SR Mission



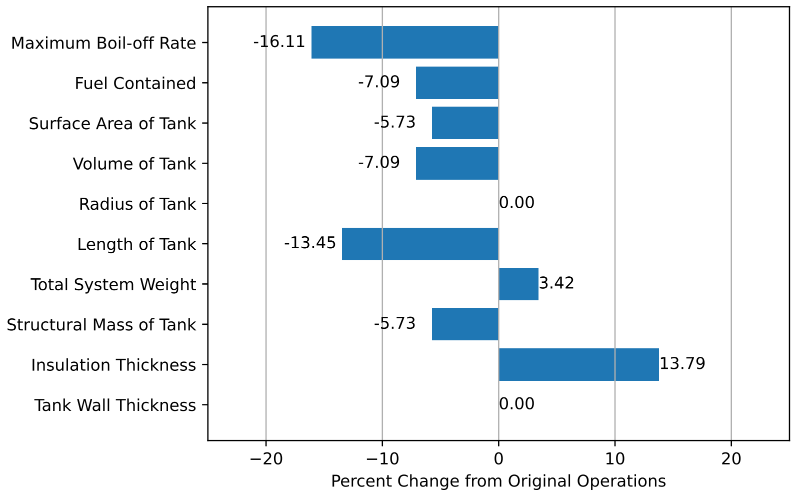
(b) LR Mission

Figure 4-17: Engine Design Point for Modified Mission Limits

A closer examination of the cryogenic design point shows trends that are expected in Figure 4-18. Because the insulation thickness was increased by 10+%, the boil-off rate significantly changed. The structural mass also decreased due to the length and the volume of the tank decreasing. Flying at a lower altitude and slower does decrease the cryogenic equipment weight.



(a) SR Mission



(b) LR Mission

Figure 4-18: Cryogenic Design Point for Modified Mission Limits

Overall, decreasing the flight velocity does result in a lighter aircraft as well as a decreased fuel burn. This is expected as flying slower does mean a decrease in fuel burn. This intuition holds true even for liquid hydrogen fueled aircraft. However, the change to hydrogen fuel is already significant enough that the mission parameter relaxation does not result in significant changes to cryogenic tank designs such as increasing the radius of the tank for a significantly lower tank surface area. Nevertheless, changing the mission constraints does result in a lower fuel burn in exchange for a somewhat longer flight time.

Material	Abbreviation	Thermal Conductivity (W/m/K)	Density (kg/m^3)
Rigid closed cell polymethacrylimide	R-CC-PMCL	0.0096	25.3
Rigid open cell polyurethane	R-OC-PU	0.0112	32.1
Rigid closed cell polyvinalchloride	R-CC-PVC	0.0046	49.8
Rigid closed cell polyurethane and chopped glass fiber	R-CC-P+CGF	0.0064	64.2
Evacuated aluminum foil separated with fluffy gass mats	E-AF-FGM	0.00016	40
Evacuated aluminum foil and glass paper laminate	E-AF-GPL	0.000017	120
Evacuated silica powder	E-SP	0.00017	160

Table 4.14: Insulation Material Properties [19]

4.4.2 Different Cryogenic Tank Materials

Another consideration that can be examined using this model is the choice of cryogenic tank materials. From the sensitivity analysis carried out in the Table 4.8 and 4.12, the insulation material has the second highest parameter sensitivity for the cryogenic tank design model that the designer can tune. Furthermore, a change in the tank structural material is explored in this section as well.

The potential alternative insulation material and tank structural weight properties are shown in Table 4.14 and 4.15 and are taken from Colozza [19]. It is not immediately clear which insulation material is best used. Figure 4-19 plots the different materials listed with the properties. The ideal material is both light and has low thermal conductivity (lower left corner); however, there is a tradeoff in terms of the materials available. As for the structural tank material, the best material is the one that has the highest strength per density, and it is clear that carbon composite is the best (and potentially also the most expensive). However, because the optimization is so fast, we can afford to optimize for each permutation of insulation and structural material. The original mission limits are kept. For this case, only the LR mission is examined.

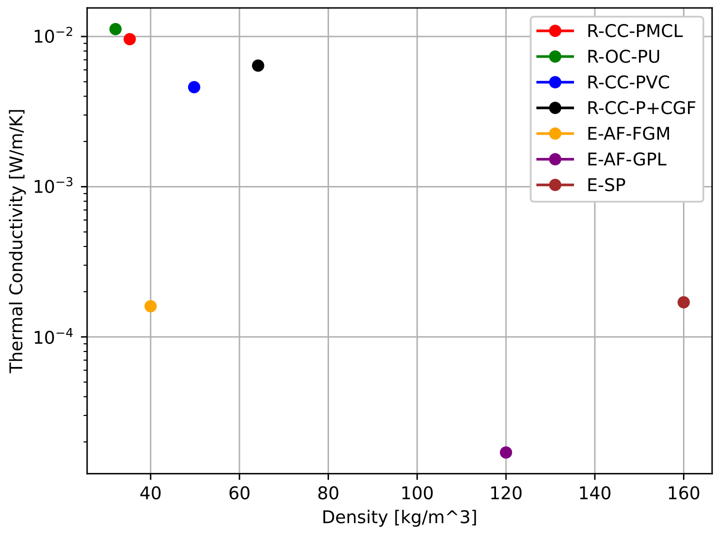


Figure 4-19: Insulation Material Properties

Material	Yield Strength (MPa)	Density (kg/m^3)
Steel	690	7860
Alumnium	410	2800
Titanium	825	4460
Carbon Composite	1900	1530

Table 4.15: Structural Material Material Properties [19]

The final total weight and the total fuel weight for the LR mission is plotted in Figure 4-20. The structural material optimization run results are clustered together as indicated by the different shapes of the markers on the graph. As expected, the carbon composite tanks (shown as circles 'o') offer the lightest mass as well as decreased the total fuel weight. Interestingly, E-AF-GPL was the material that decreased the total

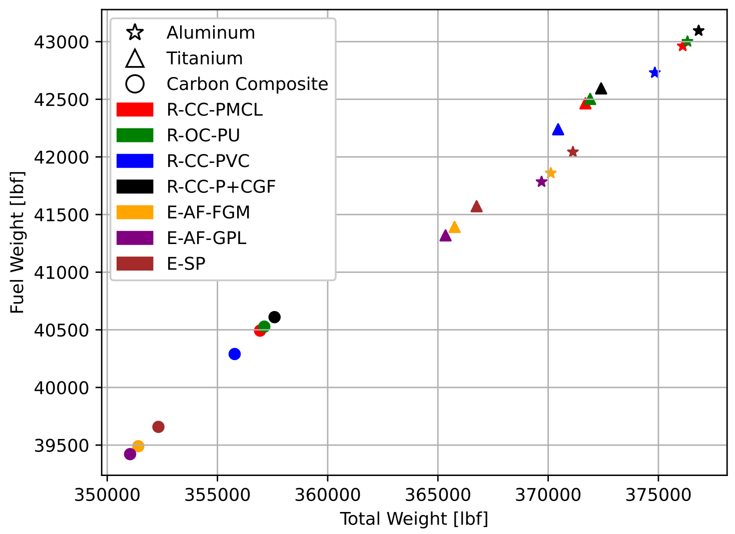


Figure 4-20: Overall Weight Differences

weight and the fuel weight the most. The optimizer prioritized the decrease in thermal conductivity over the increase in density.

Figure 4-21 and 4-22 shows the cryogenic properties. Here we define cryogenic efficiency as the following:

$$\eta_C = \frac{W_{burn}}{W_{cryo}} \tag{4.18}$$

Following the overall weight trends, carbon composite decreases the weight of the tank, and E-AF-GPL showed the lowest boil-off rate. Interestingly, carbon fiber was the most efficient; however, choosing E-AF-GPL also increased the efficiency as well.

The material selection for the cryogenic subsystem did result in expected trends and did not result in any counter-intuitive insights. However, this study did convey the idea that careful material selection for the insulation material and the structural material is important. Future work can also include a cost model in the optimization to consider both fuel burn and manufacturing cost at the same time.

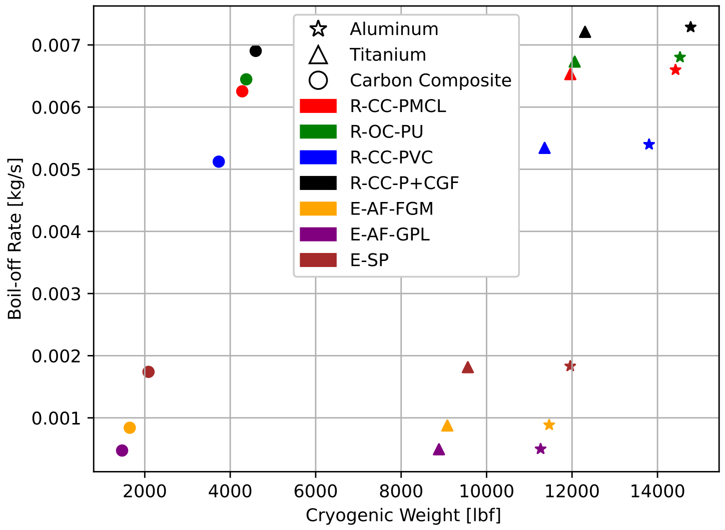


Figure 4-21: Overall Cryogenic Weight Differences

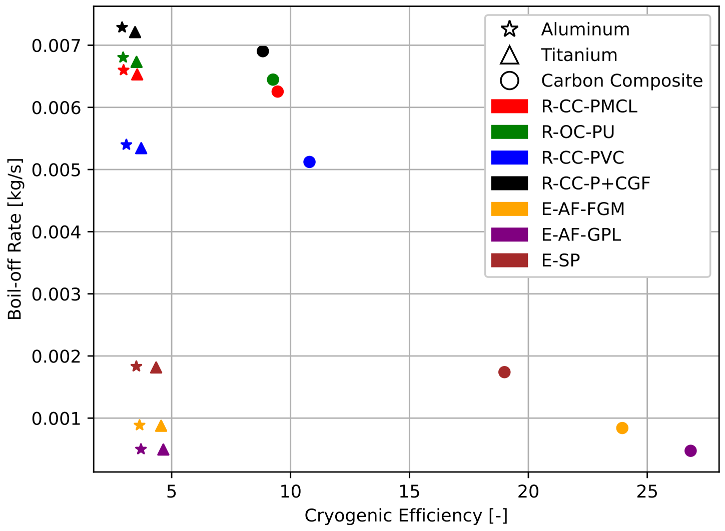
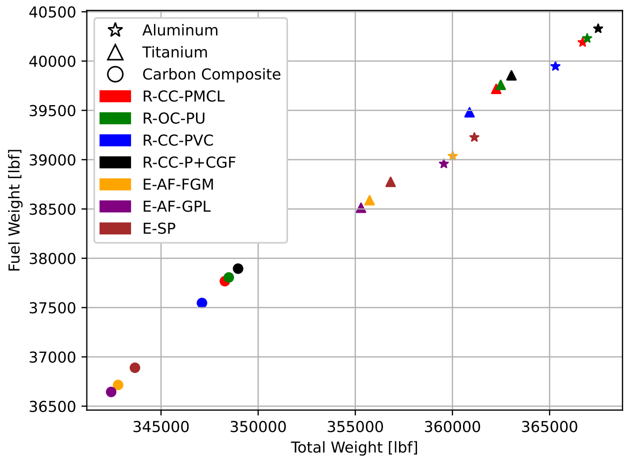


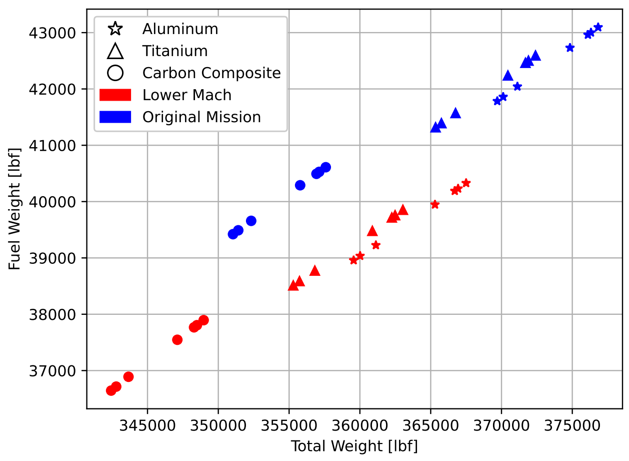
Figure 4-22: Overall Cryogenic Efficiency Differences

4.4.3 Combined Effects

The previous study into cryogenic tank material enforced the original mission limits. As previous sections showed, relaxing the mission constraints also results in design improvements. Therefore, optimization runs where mission parameters were relaxed were produced. However, rather than decreasing the minimum Mach number to 0.6, the minimum cruise Mach number was only decreased to 0.7 due to convergence issues for some of the material combinations.



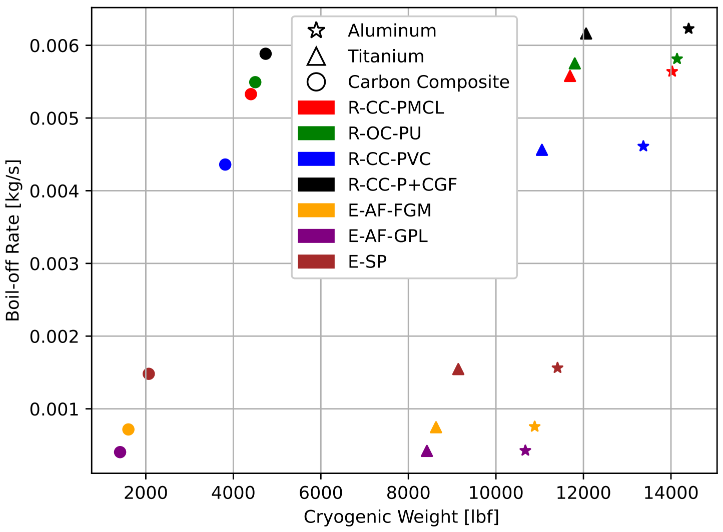
(a) Low Mach Weight Difference



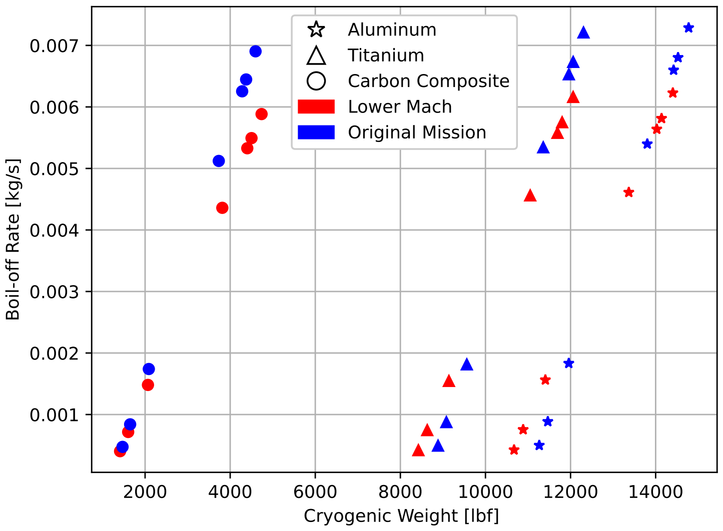
(b) Comparison

Figure 4-23: Weight Difference between Materials and Mission Changes

Figure 4-23 shows the change in total aircraft weight and the fuel weight. Figure 4-23a shows the specific materials selected with the lower Mach cruise number and Figure 4-23b shows the difference between a changed mission limits and the original mission limits. The material trends from the original mission optimization are very similar to the current ones. However, one interesting conclusion can result from Figure 4-23b: decreasing the cruise Mach will decrease the fuel weight a similar amount as if the tank material used is carbon composite. This is an interesting conclusion as carbon composite is a much more expensive technology, and the same effects of the higher technology can be achieved by adjusting the mission profile.

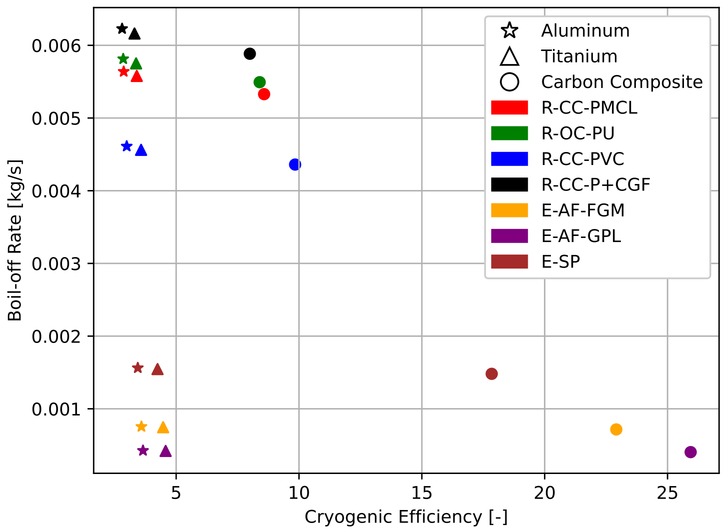


(a) Low Mach Cryo Weight Difference

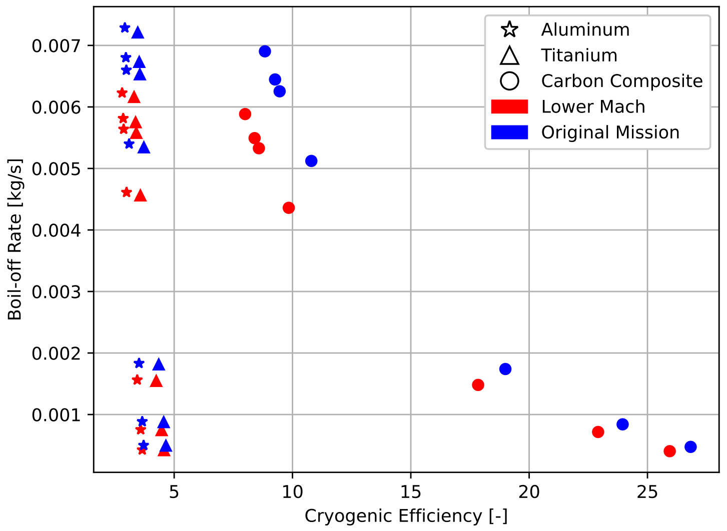


(b) Cryo Weight Comparison

Figure 4-24: Cryo Weight Difference between Materials and Mission Changes



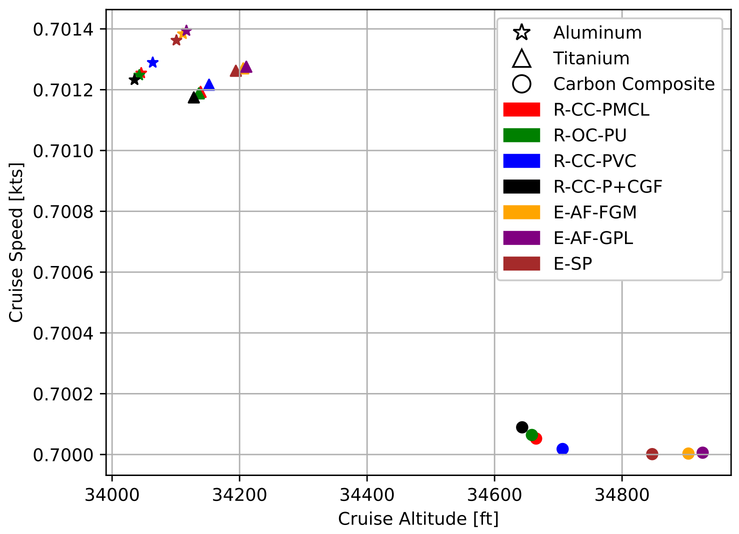
(a) Low Mach Cryo Efficiency Difference



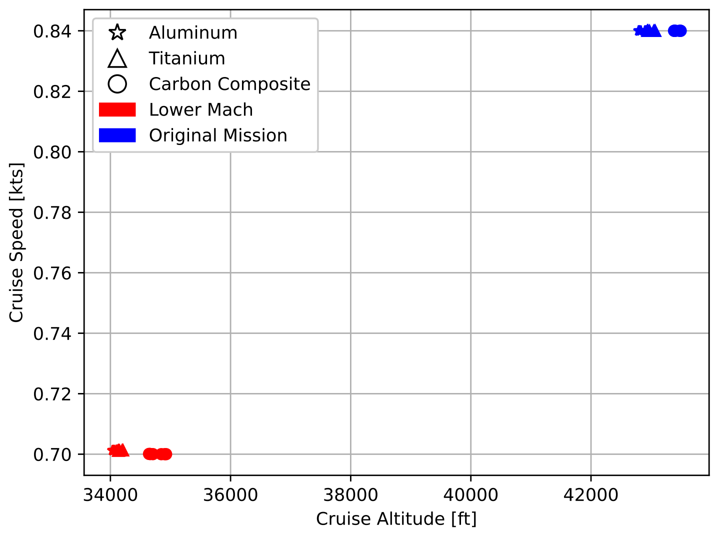
(b) Cryo Efficiency Comparison

Figure 4-25: Cryo Efficiency Difference between Materials and Mission Changes

Figure 4-24 and Figure 4-25 shows the results from the relaxed mission constraints. The relaxed mission constraints does not change the cryogenic tank design and its performance much if at all. This is expected as the decreased fuel required from the relaxed mission constraint negates the need for further improvement in cryogenic tank performance.



(a) Low Mach Mission Difference



(b) Mission Comparison

Figure 4-26: Mission Difference between Materials and Mission Changes

Figure 4-26 shows the final mission results. As expected, the optimizer still takes advantage of the lower Mach number down to Mach 0.7 to decrease the fuel weight regardless of the insulation and structural material.

Overall the model proves useful for answering questions about mission design and cryogenic technology tradeoffs for future hydrogen-powered aircraft. Useful physical insights are gained from the results of the re-optimization of the model.

4.5 Chapter Summary

In this chapter, we use the combined signomial programming (SP) architecture to examine the hydrogen aircraft design problem. We are able to obtain physical insights into the problem when we integrate both the vehicle and mission design, resulting in much of the This model assumed that hydrogen is burned directly in a turbofan engine and did not look at the use of hydrogen fuel cells. However, this could be done as part of future work.

A case study exploring relaxed mission constraints and different cryogenic technology is explored. Physical insights are found through the optimization results. Most importantly, the model shows that modifying the mission can yield the same effect as improving the cryogenics technology. This finding shows that examining the operations is just as important as improving the technology and can sometimes yield the needed results. The best results are obtained by co-optimizing both the vehicle and mission profile at the same time.

Chapter 5

Conclusions and Recommendations

5.1 Thesis Summary

In this thesis, the concept of convex optimization architectures for concurrent design of vehicles and the missions they perform is introduced. The idea of approximating non-convex optimization problems as a convex optimization problem is combined with the idea that holding certain variables constant to create a convex optimization problem. This allows large non-convex optimization problems to potentially be solved using convex optimization.

This concept is demonstrated in two case studies: the design of a sounding rocket and of a future hydrogen aircraft. Each problem is successfully solved, and physical insights are gained from each. Both problems demonstrate the importance of incorporating mission design into the vehicle design process. In the rocket design problem, the trajectory optimization influences the vehicle design significantly through the fuel weight and max thrust requirement. For the hydrogen aircraft optimization problem, the flight profile influenced the cryogenic tank design and the engine performance required. The integrated vehicle and mission design produces better designs (in terms of total mass savings) than separating the two processes.

In summary, this thesis achieved all the objectives listed in the introduction and they are reproduced here:

1. **Convex optimization is a useful tool in solving the integrated vehicle and mission design problem.** The thesis introduces the concept of convex optimization architectures to solve non-convex optimization problems. The thesis demonstrates this concept on two problems and produces meaningful results.

2. **Integrated vehicle and mission design is important in designing vehicles** Both demonstration problems produce results that show that mission design is crucial and can sometime solve the problem at hand with convex optimization architectures. Physical insights such as reduced boil-off as well as MTOW decreases of 25% and above were observed with a integrated vehicle and mission analysis for the hydrogen aircraft case.

5.2 Future Directions

Future work can be categorized into two categories: one pertaining to the convex optimization architectures and another pertaining to the vehicle and mission design problem itself.

5.2.1 Convex Optimization Architectures

There are many tuning parameters that can be used to make these architectures perform better. The number of iterations per side (GP/SCO) could be an interesting knob to turn and experiment with. One of the key features of the integrated architecture is the fact that the one iteration per side allows the optimizer to not spend any extra time on using out-of-date information from the other side. However, with convex optimization, each iteration costs a second or less in some cases. It would be an interesting experiment to see whether increasing the number of iterations per side to say 2, 3 or even 4 would increase the robustness of the solution or make the optimizations smoother. Another interesting idea that could be experimented with would be the initialization. The first couple iterations are usually highly unstable until they settle down. Optimization of the number of first iterations such that a good enough initial guess can be given would be a great way to stabilize the process.

Another avenue of future work is in the exploration of different ways to approximate the design space. In most cases, linearization and convexification was used to approximate the design space; however, other techniques may exist that keep the convexity while producing a better approximation.

Furthermore, on the problem formulation side, these architectures (especially the alternating architecture) are very similar to BCD. There are many flavors of BCD: some BCD variants modify the objective function to ensure both all the coordinate direction optimization stays within the feasible areas of all coordinate descent directions searched. There are also many ways to cycle through the different sub-problems, whether through a cyclic pattern or a set pattern as with BCD. That can be something worth exploring as well. These are not covered in this thesis; however, these show the number of extensions that are possible to this general framework for sequential convex optimization.

Finally, these architectures can be implemented using parallel computing architectures. Combined with the fast solve times of each individual solve times, the computational tractability and solve efficiency can be further improved. The integrated architecture and the alternating architecture are two that can take advantage of the parallel structure of computation to be mapped onto a parallel computation system.

5.2.2 Integrated Vehicle and Mission Design

In general, the robustness of the individual sub-problem convex optimization solves can be improved. During the implementation of the trajectory optimization, the robustness to different vehicle designs was a consistent problem. Furthermore, the trajectory optimization needs a very good starting point, which is not a problem if only the trajectory optimization problem is the only problem being solved; however, with different vehicles being tested each time, this may become a problem.

Another direction is in the capturing of passive effects in the alternating/integrated architecture. Passive effects are effects that indirectly affect the objective function. For example, in the sounding rocket problem, aerodynamics is a passive effect as it does not directly impact the fuel burn itself like the control strategy, but indirectly it does so through the drag adjustment. It is not clear whether those effects are actually being captured in the optimization architectures, so an investigation into that would greatly improve the understanding of not just the problem itself but also the convex optimization architectures.

Finally, a great step forward towards further application of this methodology would be to modify the objective function into something value-based or mission success based. Fuel burn and total system weights are great surrogate variables or representative variables for the actual value that we are interested in (cost, profit, mission success, etc.), and being able to directly represent that in the optimization problem would greatly improve the utility of this formulation.

As aviation contemplates moving from kerosene to hydrogen as a primary fuel source further applications to more advanced vehicle design and mission concepts can be anticipated.

Appendix A

Trajectory Optimization using Convex Optimization: Goddard Problem

As mentioned in the text, the Goddard problem has not been solved in the convex optimization setting. Therefore, the feasibility of solving the problem using sequential convex optimization is demonstrated here. Specifically, a closer look at the problem solve characteristics are examined here.

A.1 Comparison with GPOPS

The implementation, dynamics and details of the problem can be found in Section 3.1.4. The Goddard problem as mentioned is unique in that there is a singular arc section. The solve is compared with GPOPS [60], a MATLAB pseudospectral method. The test problem parameters are shown in Table A.1.

Parameter	Value
m_{wet}	56
m_{dry}	10
h_0	100
v_0	0
h_f	5500
v_f	0
T_{max}	3094
T_{min}	50
C_D	0.15
S_{ref}	.11
I_{sp}	170

Table A.1: Test Problem Parameters

Both GPOPS and SCO results are compared and summarized in the rest of this section. Table A.2 shows the solve times for the both processes. For this specific test case, SCO converges to the same tolerance almost twice as fast despite many more iterations.

Parameter	GPOPS	SCO
Time (s)	80.4	49.5
Tolerance	1e-6	1e-6
Iterations	10	20

Table A.2: Solve Properties

Figure A-1 shows the mass history and Figure A-2 shows the height profile of the vehicle. The mass history shows that both SCO and GPOPS converged to the same mass history. However, both SCO and GPOPS had different height trajectories, with the SCO choosing a slightly steeper/higher altitude trajectory compared to GPOPS.

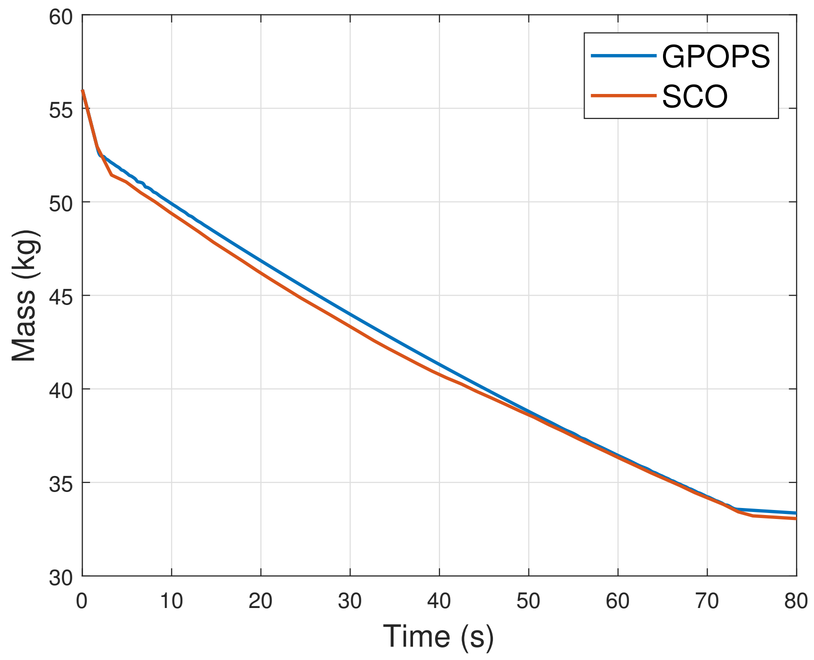


Figure A-1: Mass Profile

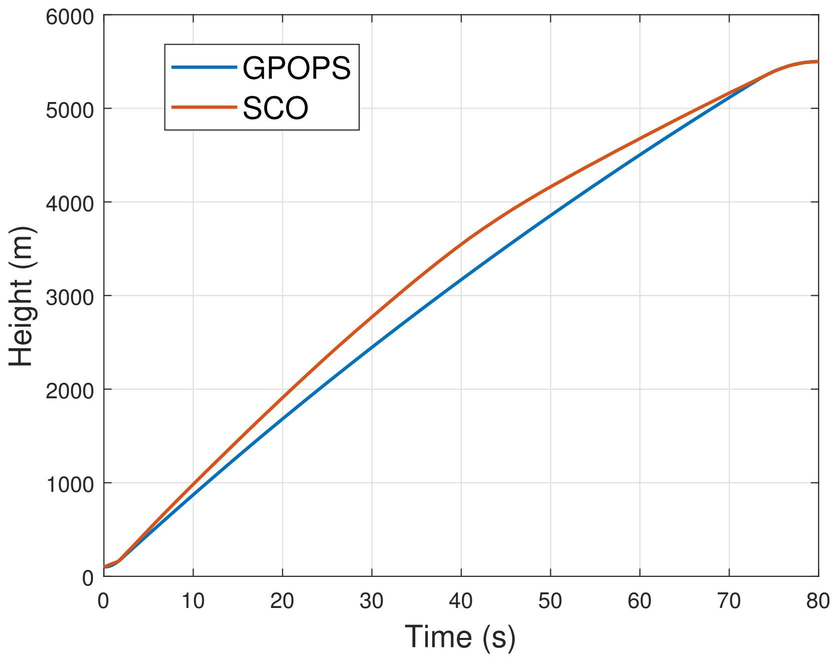


Figure A-2: Height Profile

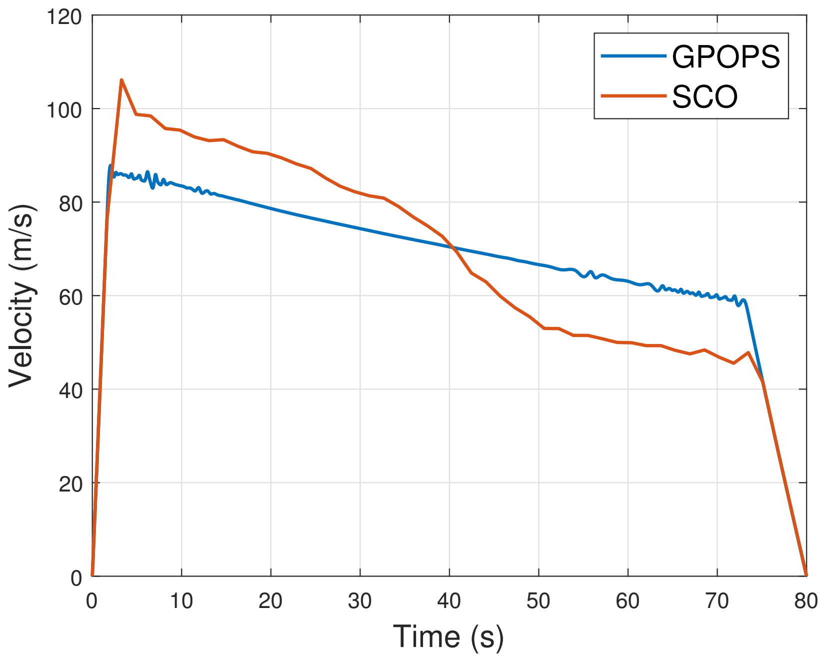


Figure A-3: Velocity Profile

Figure A-3 shows the velocity profile and Figure A-4 shows the thrust profile of the vehicle. The control profile shows almost exactly the same control profile for GPOPS and SCO; however, the velocity profile is very different. SCO goes for higher maximum speed profile that is approximately 20 m/s faster than GPOPS, but shallows down significantly. in the latter portion of the profile. This is particularly interesting as both the control profile and mass histories are the same. However, if we look closely, the SCO thrust proifle holds that high thrust for a couple seconds longer, which most likely resulted in the higher velocity. The hypothesis is that this is due to the collocation method used: this version of SCO used trapezoidal collocation which is a first order collocation method whereas GPOPS uses a pseudospectral collocation which is much more accurate. Also, the SCO is split into 50 points where as GPOPS has thousands of points and mesh refinement for areas that are oscillatory. This is an area for further investigation and future work.

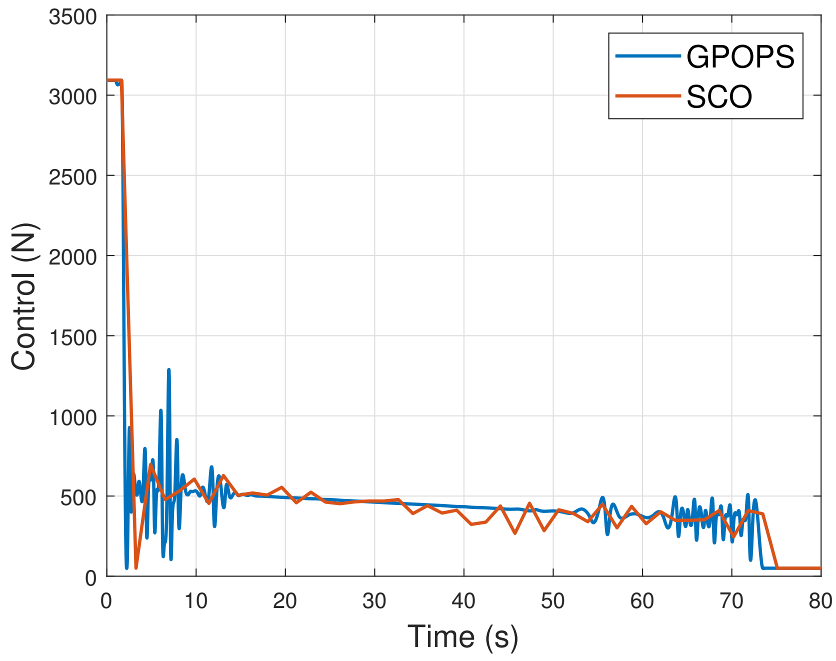


Figure A-4: Thrust Profile

A.2 Singular Arc

The previous results show varying results in the singular arc section. There are many different factors that could affect the behavior of the singular arc. In particular, 4 parameters of interest are investigated: the number of points (K), mass of the vehicle, drag of the vehicle, and the optimization tolerance. Three levels are chosen, and the effects on the control history of the vehicle are examined. To further post process the singular arc behavior, the root-mean-square (RMS) and the standard deviation is calculated for each singular arc. The base configuration is K=50, $S_{ref} = 0.15$, Mass $= 80$, and tolerance $=$ 1e-4.

Number of points K

Figure A-5 shows the optimal control profile of an optimization where the number of points are 50, 100, and 200 points and Table A.3 shows the resulting RMS and

K	RMS	Standard Deviation
50	730	256
100	721	280
200	716	268

Table A.3: Change in Number of Points

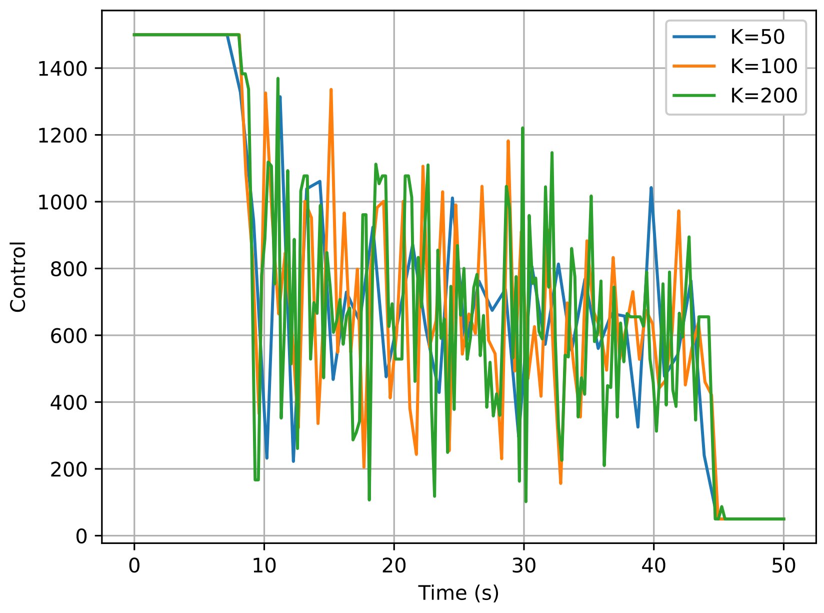


Figure A-5: Changing the number of points K

standard deviation. From both the table and calcuated results, it is clear that the number of points does not change the singular arc behavior.

Vehicle Mass

Figure A-6 shows the optimal control profile of an optimization where the wet mass of the vehicle are 80, 65, and 50 kg and Table A.4 shows the resulting RMS and

Weight	RMS	Standard Deviation
80	730	256
65	585	211
50	449	161

Table A.4: Change in Weight of Vehicle

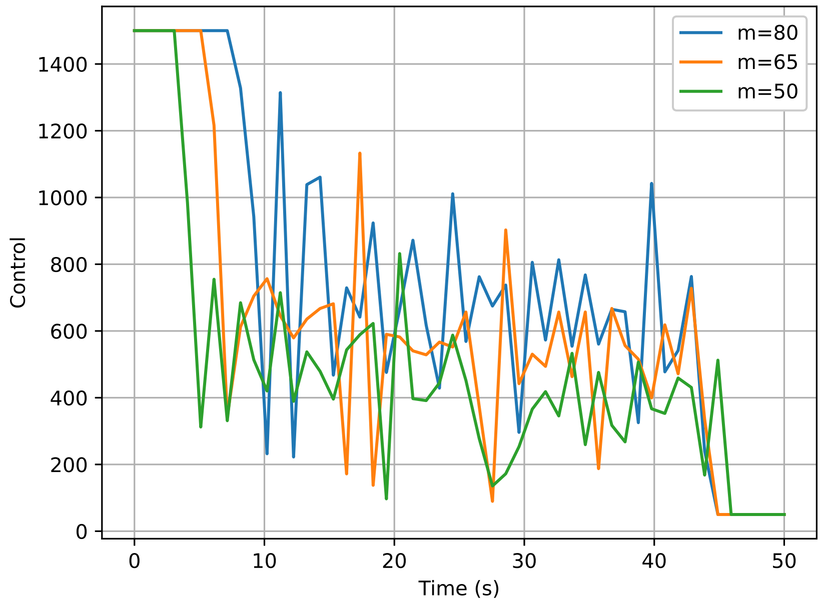


Figure A-6: Changing the mass of the vehicle

Drag	RMS	Standard Deviation
0.15	730	256
0.015	658	163
0.0015	632	116

Table A.5: Change in Drag of Vehicle

standard deviation. As expected, the control profile of the optimal control profile changed to account for the decreased length of the initial boost phase. Furthermore, as expected the RMS also goes down as the optimal control profile changes to account for the lower thrust needed to maintain the vehicle at cruise. However, the vehicle mass has some effect on the standard deviation as it is almost halved as the vehicle weight goes from 80 to 50. This is most likely due to the decrease in thrust magnitude a the singular arc which results in less of a random sampling.

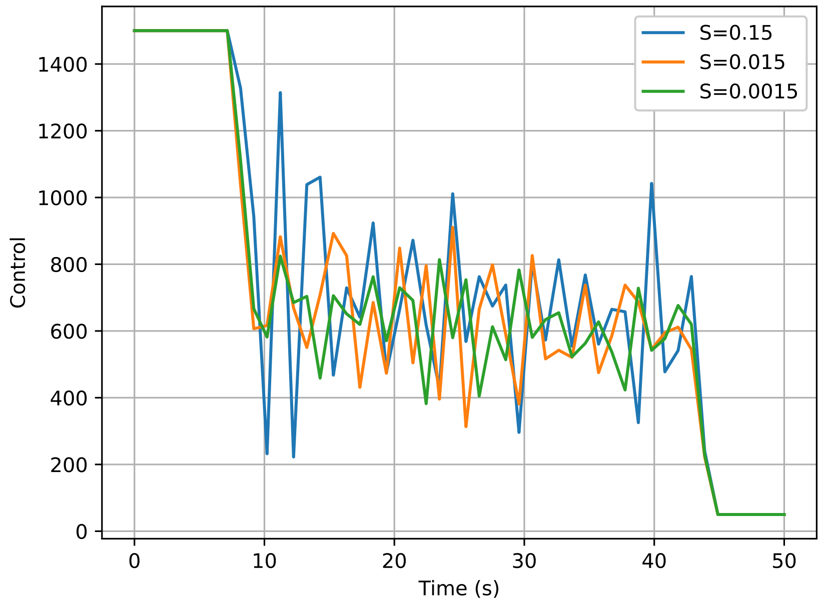


Figure A-7: Changing the drag

Tolerance	RMS	Standard Deviation
1e-4	730	256
1e-5	708	158
1e-6	639	**59**

Table A.6: Change in Convergence Tolerance

Vehicle Aerodynamics

Figure A-7 shows the optimal control profile of an optimization where the S_ref of the vehicle are 0.15, 0.015, 0.0015 m^2 and Table A.5 shows the resulting RMS and standard deviation. Similar to the mass of the vehicle results, RMS decreases to account for the drop in thrust needed and standard deviation also decreases. This is again most likely due to the decrease in thrust magnitude a the singular arc which results in less of a random sampling: the oscillations are most likely proportional to the RMS of the singular arc.

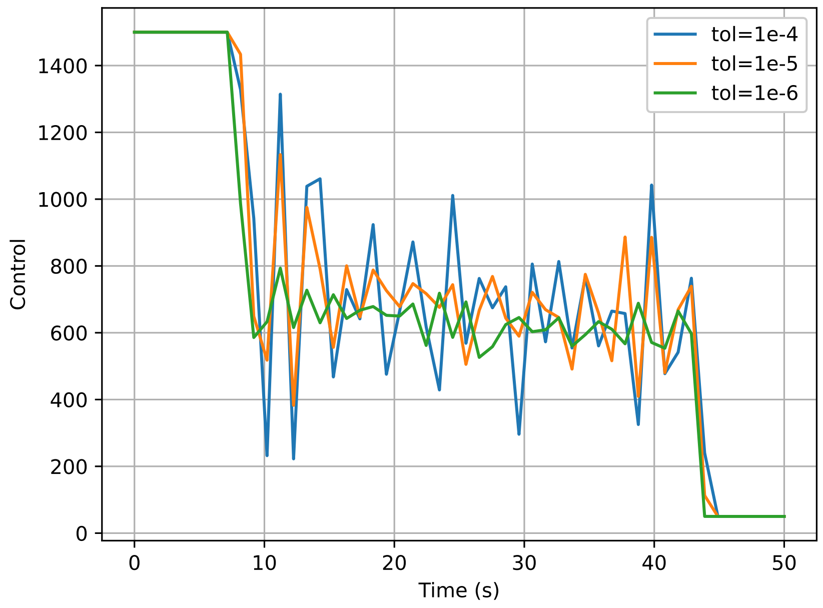


Figure A-8: Changing the convergence tolerance

Optimization Tolerance

Finally, figure A-8 shows the optimal control profile of an optimization where the tolerances of the optimization are 1e-4, 1e-5, and 1e-6 and Table A.6 shows the resulting RMS and standard deviation. This result shows a clear reason for the significant oscillations during the singular arc: decreasing the tolerance by orders of magnitudes decreases the standard deviation of the singular arc profile. This makes sense because by decreasing the tolerance, the optimizer is forced to iterate more such that the trust radius parameter is smaller, which decreases the amount the solution between iterations can vary.

Bibliography

[1] AFRL/RY - Modeling and Simulation.

[2] Liquid hydrogen fuelled aircraft - system analysis (CRYOPLANE) | CRYOPLANE Project | FP5 | CORDIS | European Commission.

[3] NASA Systems Engineering Handbook. page 297.

[4] Program to Optimize Simulated Trajectories II (POST2). Library Catalog: post2.larc.nasa.gov.

[5] QuickShot. Library Catalog: www.spaceworks.aero.

[6] convexengineering/gplibrary, September 2019. original-date: 2015-10-19T19:44:45Z.

[7] Jeremy Agte, Olivier de Weck, Jaroslaw Sobieszczanski-Sobieski, Paul Arendsen, Alan Morris, and Martin Spieck. MDO: assessment and direction for advancement—an opinion of one international group. *Structural and Multidisciplinary Optimization*, 40(1-6):17–33, January 2010.

[8] Darcy L. Allison and Raymond M. Kolonay. Expanded MDO for Effectiveness Based Design Technologies: EXPEDITE Program Introduction. In *2018 Multidisciplinary Analysis and Optimization Conference*, Atlanta, Georgia, June 2018. American Institute of Aeronautics and Astronautics.

[9] James Allison and Daniel R. Herber. Multidisciplinary Design Optimization of Dynamic Engineering Systems. In *54th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Boston, Massachusetts, April 2013. American Institute of Aeronautics and Astronautics.

[10] MOSEK ApS. *MOSEK Optimizer API for Python 9.0.91*, 2017.

[11] Brian Kenichi Bairstow and B S Engineering. Effectiveness of Integration of System-Level Optimization in Concurrent Engineering for Rocket Design. page 111.

[12] Kevin Bowcutt, Geojoe Kuruvila, Thomas Grandine, and Evin Cramer. Advancements in Multidisciplinary Design Optimization Applied to Hypersonic Vehicles to Achieve Performance Closure. In *15th AIAA International Space Planes*

and Hypersonic Systems and Technologies Conference, Dayton, Ohio, April 2008. American Institute of Aeronautics and Astronautics.

[13] Stephen Boyd, Seung-Jean Kim, Lieven Vandenberghe, and Arash Hassibi. A tutorial on geometric programming. *Optimization and Engineering*, 8(1):67–127, May 2007.

[14] Stephen P. Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, Cambridge, UK ; New York, 2004.

[15] R D Braun and I M Kroo. Development and Application of the Collaborative Optimization Architecture in a Multidisciplinary Design Environment. page 19.

[16] G. Daniel Brewer. *Hydrogen Aircraft Technology*. Routledge, 1 edition, November 2017.

[17] Edward Burnell and Warren Hoburg. Gpkit software for geometric programming. https://github.com/convexengineering/gpkit, 2018. Version 0.8.0.

[18] Matthew Clarke, Jordan Smart, Emilio M. Botero, Walter Maier, and Juan J. Alonso. Strategies for Posing a Well-Defined Problem for Urban Air Mobility Vehicles. In *AIAA Scitech 2019 Forum*, San Diego, California, January 2019. American Institute of Aeronautics and Astronautics.

[19] Anthony J Colozza, Analex Corporation, and Brook Park. Hydrogen Storage for Aircraft Applications Overview. page 32, 2002.

[20] Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[21] Quoc Tran Dinh and Moritz Diehl. Local Convergence of Sequential Convex Programming for Nonconvex Optimization. In Moritz Diehl, Francois Glineur, Elias Jarlebring, and Wim Michiels, editors, *Recent Advances in Optimization and its Applications in Engineering*, pages 93–102. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[22] Alexander Domahidi, Eric Chu, and Stephen Boyd. ECOS: An SOCP solver for embedded systems. In *2013 European Control Conference (ECC)*, pages 3071–3076, Zurich, July 2013. IEEE.

[23] Alexander Domahidi, Aldo U. Zgraggen, Melanie N. Zeilinger, Manfred Morari, and Colin N. Jones. Efficient interior point methods for multistage problems arising in receding horizon control. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 668–674, December 2012. ISSN: 0743-1546.

[24] John Duchi. Sequential Convex Programming. page 19.

[25] Robert D. Falck and Justin S. Gray. Optimal Control within the Context of Multidisciplinary Design, Analysis, and Optimization. In *AIAA Scitech 2019 Forum*, San Diego, California, January 2019. American Institute of Aeronautics and Astronautics.

[26] E. Fleeman. *Tactical Missile Design.* Education Series. AIAA, Reston, VA, 2nd edition, 2006.

[27] Nathaniel S. Gates, Kevin R. Moore, Andrew Ning, and John D. Hedengren. Combined Trajectory and Propulsion Optimization for Solar-Regenerative High-Altitude Long Endurance Unmanned Aircraft. In *AIAA Scitech 2019 Forum*, San Diego, California, January 2019. American Institute of Aeronautics and Astronautics.

[28] Michael Grant and Robert Braun. Analytic Hypersonic Aerodynamics for Conceptual Design of Entry Vehicles. In *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, Orlando, Florida, January 2010. American Institute of Aeronautics and Astronautics.

[29] Michael Grant, Ian Clark, and Robert Braun. Rapid Design Space Exploration for Conceptual Design of Hypersonic Missions. In *AIAA Atmospheric Flight Mechanics Conference*, Portland, Oregon, August 2011. American Institute of Aeronautics and Astronautics.

[30] Justin S. Gray, John T. Hwang, Joaquim R. R. A. Martins, Kenneth T. Moore, and Bret A. Naylor. OpenMDAO: an open-source framework for multidisciplinary design, analysis, and optimization. *Structural and Multidisciplinary Optimization*, 59(4):1075–1104, April 2019.

[31] Techbriefs Media Group. Trajectory Optimization: OTIS 4. Library Catalog: www.techbriefs.com.

[32] S Hall. 16.32 lecture: Singular arcs, Spring 2019.

[33] Dustin J. Harper. Operations Analysis Integration for Effectiveness-Based Design in the AFRL EXPEDITE Program. In *AIAA Scitech 2020 Forum*, Orlando, FL, January 2020. American Institute of Aeronautics and Astronautics.

[34] Eric S. Hendricks, Robert D. Falck, and Justin S. Gray. Simultaneous Propulsion System and Trajectory Optimization. In *18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Denver, Colorado, June 2017. American Institute of Aeronautics and Astronautics.

[35] Daniel R. Herber and James T. Allison. Nested and Simultaneous Solution Strategies for General Combined Plant and Control Design Problems. *Journal of Mechanical Design*, 141(1):011402, January 2019.

[36] V. Ho-Huu, S. Hartjes, H. G. Visser, and R. Curran. An optimization framework for route design and allocation of aircraft to multiple departure routes. *arXiv:1908.11086 [cs]*, August 2019. arXiv: 1908.11086.

[37] Warren Hoburg, Philippe Kirschen, and Pieter Abbeel. Fitting geometric programming models to data. page 22.

[38] Warren Hoburg, Philippe Kirschen, and Pieter Abbeel. Data fitting with geometric-programming-compatible softmax functions. *Optimization and Engineering*, 17(4):897–918, December 2016.

[39] Warren Woodrow Hoburg. Aircraft Design Optimization as a Geometric Program. page 113, 2015.

[40] John P. Jasa, Charles A. Mader, and Joaquim Martins. Trajectory Optimization of a Supersonic Aircraft with a Thermal Fuel Management System. In *2018 Multidisciplinary Analysis and Optimization Conference*, Atlanta, Georgia, June 2018. American Institute of Aeronautics and Astronautics.

[41] Michelle Kirby and Dimitri Mavris. A method for technology selection based on benefit, available schedule and budget resources. In *2000 World Aviation Conference*, San Diego,CA,U.S.A., October 2000. American Institute of Aeronautics and Astronautics.

[42] Philippe G Kirschen. Signomial Programming for Aircraft Design. page 127, 2016.

[43] Philippe G. Kirschen and Warren W. Hoburg. The Power of Log Transformation: A Comparison of Geometric and Signomial Programming with General Nonlinear Programming Techniques for Aircraft Design Optimization. In *2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Kissimmee, Florida, January 2018. American Institute of Aeronautics and Astronautics.

[44] Mason Levy and Kevin Choi. Multidisciplinary Design Optimization for Effectiveness-Based Design in the AFRL EXPEDITE Program. In *AIAA Scitech 2020 Forum*, Orlando, FL, January 2020. American Institute of Aeronautics and Astronautics.

[45] Xinfu Liu. Fuel-Optimal Rocket Landing with Aerodynamic Controls. *Journal of Guidance, Control, and Dynamics*, 42(1):65–77, January 2019.

[46] Xinfu Liu and Ping Lu. Solving Nonconvex Optimal Control Problems by Convex Optimization. *Journal of Guidance, Control, and Dynamics*, 37(3):750–765, May 2014.

[47] Xinfu Liu, Ping Lu, and Binfeng Pan. Survey of convex optimization for aerospace applications. *Astrodynamics*, 1(1):23–40, September 2017.

[48] J. Löfberg. Yalmip : A toolbox for modeling and optimization in matlab. In *In Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[49] David Maniaci. Relative Performance of a Liquid Hydrogen-Fueled Commercial Transport. In *46th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, January 2008. American Institute of Aeronautics and Astronautics.

[50] Yuanqi Mao, Michael Szmuk, Xiangru Xu, and Behcet Acikmese. Successive Convexification: A Superlinearly Convergent Algorithm for Non-convex Optimal Control Problems. *arXiv:1804.06539 [math]*, April 2018. arXiv: 1804.06539.

[51] Joaquim Martins and John Hwang. Review and Unification of Methods for Computing Derivatives of Multidisciplinary Systems. In *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference&lt;BR&gt;20th AIAA/ASME/AHS Adaptive Structures Conference&lt;BR&gt;14th AIAA*, Honolulu, Hawaii, April 2012. American Institute of Aeronautics and Astronautics.

[52] Joaquim R. R. A. Martins and Andrew B. Lambe. Multidisciplinary Design Optimization: A Survey of Architectures. *AIAA Journal*, 51(9):2049–2075, September 2013.

[53] Dimitri N. Mavris, Oliver Bandte, and Daniel A. DeLaurentis. Robust Design Simulation: A Probabilistic Approach to Multidisciplinary Design. *Journal of Aircraft*, 36(1):298–307, January 1999.

[54] Kenneth Michael Mull. A Genetic Algorithm Incorporating Design Choice for the Preliminary Design of Unmanned Aerial Vehicles. page 124.

[55] Leland Nicolai and Grant Carichner. *Fundamentals of Aircraft and Airship Design, Volume 1: Aircraft Design*. Number 1st in Education Series. AIAA, Reston, VA, 2010.

[56] Sven Niederberger. EmbersArc/SCvx, November 2019. original-date: 2018-09-07T14:29:27Z.

[57] Jorge Nocedal and Stephen J. Wright. *Numerical optimization*. Springer series in operations research. Springer, New York, 2nd ed edition, 2006. OCLC: ocm68629100.

[58] Michael A Patterson and Anil V Rao. A General-Purpose MATLAB Software for Solving Multiple-Phase Optimal Control Problems Version 2.3. page 71.

[59] Ruben E. Perez, Hugh H. T. Liu, and Kamran Behdinan. *A Multidisciplinary Optimization Framework for Control-Configuration Integration in Aircraft Conceptual Design*.

[60] Anil V. Rao, David A. Benson, Christopher Darby, Michael A. Patterson, Camila Francolin, Ilyssa Sanders, and Geoffrey T. Huntington. *Algorithm 902: GPOPS, A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using the Gauss Pseudospectral Method.* 2010.

[61] Meisam Razaviyayn. Successive Convex Approximation: Analysis and Applications. page 205.

[62] Satadru Roy, William A. Crossley, Kenneth T. Moore, Justin S. Gray, and Joaquim Martins. Next generation aircraft design considering airline operations and economics. In *2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Kissimmee, Florida, January 2018. American Institute of Aeronautics and Astronautics.

[63] Satadru Roy, Kenneth Moore, John T. Hwang, Justin S. Gray, William A. Crossley, and Joaquim Martins. A Mixed Integer Efficient Global Optimization Algorithm for the Simultaneous Aircraft Allocation-Mission-Design Problem. In *58th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Grapevine, Texas, January 2017. American Institute of Aeronautics and Astronautics.

[64] Zhenghui Sha, Kushal A. Moolchandani, Apoorv Maheshwari, Joseph Thekinen, Jitesh Panchal, and Daniel A. DeLaurentis. Modeling Airline Decisions on Route Planning Using Discrete Choice Models. In *15th AIAA Aviation Technology, Integration, and Operations Conference*, Dallas, TX, June 2015. American Institute of Aeronautics and Astronautics.

[65] Hao-Jun Michael Shi, Shenyinying Tu, Yangyang Xu, and Wotao Yin. A Primer on Coordinate Descent Algorithms. *arXiv:1610.00040 [math, stat]*, September 2016. arXiv: 1610.00040.

[66] Jaroslaw Sobieszczanski-Sobieski, Jeremy Agte, and Robert Sandusky, Jr. Bi-level integrated system synthesis (BLISS). In *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, St. Louis,MO,U.S.A., September 1998. American Institute of Aeronautics and Astronautics.

[67] Michael Szmuk, Taylor P. Reynolds, and Behcet Acikmese. Successive Convexification for Real-Time 6-DoF Powered Descent Guidance with State-Triggered Constraints. *arXiv:1811.10803 [math]*, November 2018. arXiv: 1811.10803.

[68] Christine Taylor and Olivier L. de Weck. Coupled Vehicle Design and Network Flow Optimization for Air Transportation Systems. *Journal of Aircraft*, 44(5):1479–1486, September 2007.

[69] L Vandenberghe. The CVXOPT linear and quadratic cone program solvers. page 30.

[70] Dries Verstraete. CRANFIELD UNIVERSITY. page 266.

[71] Christopher Winnefeld, Thomas Kadyk, Boris Bensmann, Ulrike Krewer, and Richard Hanke-Rauschenbach. Modelling and Designing Cryogenic Hydrogen Tanks for Future Aircraft Applications. *Energies*, 11(1):105, January 2018.

[72] Martin A. York, Berk Öztürk, Edward Burnell, and Warren W. Hoburg. Efficient Aircraft Multidisciplinary Design Optimization and Sensitivity Analysis via Signomial Programming. *AIAA Journal*, pages 1–16, September 2018.

[73] J Ziemann, F Shum, and H EBERIUSg. LOW-NO, COMBUSTORS FOR HYDROGEN FUELED AERO ENGINE. page 8.