Conceptual Engineering Design and Optimization Methodologies using Geometric Programming

by

Berk Öztürk

B.S., Massachusetts Institute of Technology, 2016

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

Master of Science in Aeronautics and Astronautics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2018

© Massachusetts Institute of Technology 2018. All rights reserved.

Author . . . .
Department of Aeronautics and Astronautics
February 1st, 2018

Certified by . . . .
Mark Drela
Professor, Aeronautics and Astronautics
Thesis Supervisor

Accepted by . . . .
Hamsa Balakrishnan
Associate Professor, Aeronautics and Astronautics
Chair, Graduate Program Committee

Conceptual Engineering Design and Optimization Methodologies using Geometric Programming

by

Berk Öztürk

Submitted to the Department of Aeronautics and Astronautics
on February 1st, 2018, in partial fulfillment of the
requirements for the degree of
Master of Science in Aeronautics and Astronautics

Abstract

Geometric programs (GPs) and other forms of convex optimization have recently experienced a resurgence due to the advent of polynomial-time solution algorithms and improvements in computing. Observing the need for fast and stable methods for multidisciplinary design optimization (MDO), previous work has shown that geometric programming can be a powerful framework for MDO by leveraging the mathematical guarantees and speed of convex optimization. However, there are barriers to the implementation of optimization in design. In this work, we formalize how the formulation of non-linear design problems as GPs facilitates design process. Using the principles of pressure and boundedness, we demonstrate the intuitive transformation of physics- and data-based engineering relations into GP-compatible constraints by systematically formulating an aircraft design model. We motivate the difference-of-convex GP extension called signomial programs (SPs) in order to extend the scope and fidelity of the model. We detail the features specific to GPkit, an object-oriented GP formulation framework, which facilitate the modern engineering design process. Using both performance and mission modeling paradigms, we demonstrate the ability to model and design increasingly complex systems in GP, and extract maximal engineering intuition using sensitivities and tradespace exploration methods. Though the methods are applied to an aircraft design problem, they are general to models with continuous, explicit constraints, and lower the barriers to implementing optimization in design.

Thesis Supervisor: Mark Drela
Title: Professor, Aeronautics and Astronautics

Acknowledgments

Firstly, I would like to thank Professor Warren Hoburg for giving me the opportunity to work on research that I believe has great potential. He introduced me to geometric programming and welcomed me to the Hoburg Research Group (now the Convex Engineering Group). Under his mentorship, I learned that the most interesting designs are the ones we don't expect, and that challenging current engineering design methods and norms is the most worthwhile objective of optimization. So thank you for the opportunity to do just that.

I am grateful for the many friends I have in the Aerospace Computational Design Lab (ACDL) who have lightened my work hours with their camaraderie. Of this group, the Convex folks reserve a special place. I am lucky to collaborate with such a talented and dedicated group of researchers, and I hope that we will continue to redefine conceptual engineering design.

This thesis would not have been possible without the amazing work Ned Burnell does as the lead developer of GPkit. He never ceases to amaze me with his creativity and energy. The many debates we have had about design philosophy have inspired this thesis, and I'm glad I've gotten the opportunity to extensively document many of the ideas that were discussed.

I thank my co-advisors Professor Mark Drela and Bob Haimes, who have taken on the burden of advising me in Woody's absence. Thank you for your guidance and wisdom, and pushing me to finish this thesis.

I am indebted to cycling for keeping me fit and sane. Cycling has given me opportunities to escape, even as I miss winter training camp to conclude this thesis. MIT Cyclists have been an immutable source of joy in my life, and are near and dear to my heart. Thanks for enabling me.

I would like to thank my long-time friend Johannes Norheim for being a battle-scarred companion through 5 years of MIT. We have been challenged, and our paths have converged and diverged many times, but our friendship has not wavered. My partner Elise Newman has been a ray of sunshine in every weather, and I am looking forward to our future adventures together.

My parents definitely deserve a mention because I wouldn't exist without them. The longer I live, the more I understand that I am where I am because of the sacrifices they have made. Deniz, you have been my rock. It has been my privilege to be your brother and watch you grow up, and will be my privilege to help you through your struggles and celebrate your future successes. I am confident you will be more than I have been.

Contents

1 Introduction 13

1.1 Distinguishing between design and optimization 14

1.2 Unifying design and optimization with GP 16

2 Engineering inequalities and intuition, from equalities 19

2.1 Making feasibility sets and boundedness explicit 19

2.2 Defining the design problem 22

2.2.1 Objective functions 22

2.2.2 Functional description: constraining the problem 23

2.3 GP modeling from physics 23

2.3.1 Free and fixed variables: weight and lift model 24

2.3.2 Alternate objectives: more performance metrics 26

2.3.3 More physics for boundedness: thrust and drag model 27

2.4 Limits of GP and convexity, and SP modeling 30

2.4.1 Signomial constraints: fuel volume model 30

2.4.2 Arguments for the signomial equality 32

2.4.3 Completing the model: wing structural model 33

2.5 Results of SimPleAC 34

3 Extensibility of GP 37

3.1 Modularization and improved fidelity: engine model 37

3.1.1 Creating an engine submodel 37

3.1.2 Data-based modeling: engine power vs. weight 39

3.1.3 Converting all subsystems into submodels 41
3.2 Mission design and performance modeling form 42
3.2.1 Linking performance models: flight segments 46
3.2.2 Characterizing the environment: atmospheric model 47
3.2.3 Mission objectives 48
3.3 Design exploration through mission design 48
3.4 More modeling improvements before multimission design 51
3.4.1 Environmental effects: engine lapse rate model 52
3.4.2 Making use of sensitivities: engine BSFC model 52
3.5 Multimission design 53
3.5.1 Multimission objective functions 54
3.5.2 Multimission optimization results 55
3.5.3 Potential extensions of multimission design 57

4 Conclusion **59**

A Mathematical Framework **61**
A.1 Geometric Programming 61
A.2 Signomial Programming 62

B Model Resources **65**
B.1 Flight segment model variables 65

List of Figures

1-1 The flow diagrams of two methods of optimization. 17

2-1 The x-y feasibility set of a simple monomial equality. 20
2-2 The x-y feasibility set of lower-bounding monomial, and upper-bounding posynomial. 21
2-3 The x-y feasibility set of upper-bounding monomial, and upper-bounding posynomial. 21

3-1 Engine MSL power versus weight fits for $K = 1, 2$ posynomial terms with underlying data. 40
3-2 Variable and constraint hierarchy of the SimPleAC model for a single flight segment. 41
3-3 Variable and constraint hierarchy of the SimPleAC static+performance model for two flight segments. 42
3-4 Variable hierarchy of thrust constraint 3.4. 46
3-5 The fuel and total weight contours with respect to range and payload. 49
3-6 Fraction of total fuel stored in fuselage with respect to range and payload. 50
3-7 Time cost and time cost index sensitivity contours. 50
3-8 $\frac{\text{BSFC}}{\text{BSFC}_{min}}$ versus $\frac{P_{shaft}}{P_{shaft,alt}}$ data fit. 53
3-9 Multimission uni-directional graph, with $N_{segments} = 2$ and $N_{missions} = 2$. 54

A-1 A signomial inequality constraint and GP approximations about two different points. 64
A-2 The signomial equality constraint $C_D = f(C_L)$ and its approximation. 64

List of Tables

2.1 Variables introduced in the weight and lift model. 25
2.2 Variables introduced to define new performance metrics. 26
2.3 Unbounded variables in the weight and lift model. 27
2.4 Variables introduced in the thrust and drag model. 29
2.5 Unbounded variables in the GP-compatible formulation. 29
2.6 Variables introduced in the fuel model. 32
2.7 Variables introduced in the wing structural model. 34
2.8 Values of free variables in the SimPleAC model. 34
2.10 Sensitivities of parameters in the SimPleAC model. 35

3.1 Variables of SimPleAC in static+performance modeling, detailed in the variable and constraint hierarchy. 43
3.3 Inputs to the design space exploration of the ***Mission*** model. 49
3.4 A selection of sensitivities to design parameters. 51
3.5 Inputs to the M1 and M2 ***Mission*** models. 55
3.6 Results of the single- and multi-mission solutions. 56

B.1 Variables introduced in the flight segment model. 65

Chapter 1

Introduction

Modern engineering design, and particularly aerospace design, has come to rely heavily on optimization. Multidisciplinary Design Optimization (MDO) research has stressed the importance of fast and reliable tools for engineering design [13]. However most MDO tools suffer from poor time performance, due to the multimodal[1] nature of many engineering design problems. Furthermore, these tools act like black boxes since they provide point solutions[2] without additional information about the design problem such as sensitivities, which can provide important engineering insight.

In the Convex Engineering Group (CEG), we have sought to improve the engineering design process by leveraging the mathematical guarantees and speed of convex optimization. Our primary software product is GPkit [3], an open-source, object-oriented software to help build Geometric Program (GP)-compatible models and interface with solvers. The seminal works in the field of convex optimization ([1],[5]) and much of CEG's previous work ([7],[9],[11],[18]) have demonstrated that geometric programming and its non-log-convex extension signomial programming are useful for certain kinds of optimization problems, but have yet to formalize why they facilitate design.

The mathematical restrictions on the form of constraints in the GP formulation remain the biggest barriers in using optimization in engineering design. Chapter 2 will show that the form of the GP actually facilitates the design optimization process and engineering understanding, rather than impeding it, through the disciplined use of inequalities to express constraints. Furthermore, GPkit allows for the monitoring of the boundedness of variables in a model, which facilitates the model building process and allows engineers to have properly conditioned models. Hence this thesis will pass on some of the expertise we have developed in the CEG building GP-compatible models from general non-linear physical models.

---

[1] Multimodal problems have multiple locally optimal solutions.

[2] A point solution is a design that is optimal for a given mission, but does not consider the feasibility of other potentially interesting missions.

Chapter 3 will showcase the extensibility of GP. The formulation of the GP as a 'bag of constraints' instead of a hierarchical set of relations confers advantages when trying to expand the fidelity and scope of models, especially in the conceptual design stage. Furthermore, the solution to the dual of the GP provides optimal sensitivities[3] which allow targeted efforts by engineers to collaboratively improve models.

Chapter 3 will also discuss the features specific to GPkit in facilitating an engineering design process that is streamlined and collaborative, and is compatible with modern engineering design methodologies. The modularity of the models, as well as the ability to create vectorized models, variables and constraints allows for a mission design approach that ensures that requirements both at the sub-system and complete-system levels are satisfied.

The features of GP and convex optimization in general will be discussed in the context of an aircraft MDO problem, but the methods discussed are general to other engineering design problems which have explicit, continuous constraints.

1.1 Distinguishing between design and optimization

It is difficult to find definitions of design and optimization that identify the similarities and differences between the terms. To understand why GPs facilitate design, it's useful to determine what features of optimization create barriers to entry for its use in design.

---

[3] The (optimal) sensitivity is the local expected fractional change in the objective value of the optimal solution per fractional change in a variable or constraint value [1].

In the context of this thesis, I will define design as following: To design is to conceive the form and function of something. In the engineering sense, we can think about the form as the configuration or the parametrization. The form usually defines n, the number of degrees of freedom of the system, which has a direct effect on the size of the feasibility space, as well as the complexity of the problem. On the other hand, the function is the actual purpose of the things being designed. It is oftentimes the aspect of the design that we can quantify (i.e. the performance), and has some physics that can be modeled.

An important aspect of design is that it is a process that explores an n-dimensional feasible space of possible solutions. We can think about the feasible set of a design as all of the designs that satisfy the functional requirements. But without a clear method of comparing the relative performances of designs, the classical definition of design implies a class of feasibility problems satisfying a set of constraints that act on the designer's parametrization of the problem.

In this thesis, optimization will assume the following definition: To optimize is to select an element in a set of feasible solutions with the lowest desired objective value. It is also a process, which is sensitive to the elements contained within the set (related to the form), and the choice of objective function (related to the function). In many ways, optimization is a natural extension of design, because it requires an explicit mathematical representation of the form and function.

By these definitions, both design and optimization explore feasible and infeasible sets, but differ in a fundamental way. Design is based on feasibility, whereas optimization seeks optimality. This observation gives insight as to why there is a barrier to using optimization in design. The distinction allows design to be performed in non-restrictive mathematical forms, since non-linear feasibility problems are much easier to solve than non-linear optimization problems. Optimization is done in specific mathematical forms; since most problems of interest are complex, computational time is a limited resource. These forms can prove an impediment for designers unfamiliar with optimization to use it.

1.2 Unifying design and optimization with GP

Geometric programming has developed "in response to a need to solve problems in the actual world" [5]. GPs and other convex optimization methods have been in development since the 1960's, but have come into the limelight thanks to the development of polynomial-time algorithms for convex programming [14] and improvements in computing. The form of the GP limits its application to certain kinds of design problems, especially since GPs generally require explicit, continuous constraints. But for these problems, GP and convex optimization naturally integrate into the conceptual design process for three primary reasons.

1. *Inequalities help engineering understanding.*

   The mathematical constraints of the GP force designers to have a proper grasp of the fundamental tradeoffs and pressures in a design. Traditionally, physical relations are expressed as equalities. But there is an almost-seamless transition from fundamental physics to GP-compatible inequalities for certain kinds of problems, and the GP-compatible form makes boundedness of variables explicit. This understanding of pressure and boundedness facilitates the conversion of general physical engineering equations into optimization-compatible constraint forms. Certain mathematical restrictions of the GP can be partially overcome through the use of the difference-of-convex (DC) extension of the GP called the Signomial Program (SP). SPs give us the flexibility to model non-log-convex functions, as well as allowing designers to explicitly enforce the tightness of constraints through the use of signomial equalities when the direction of pressure on variables is not clear.

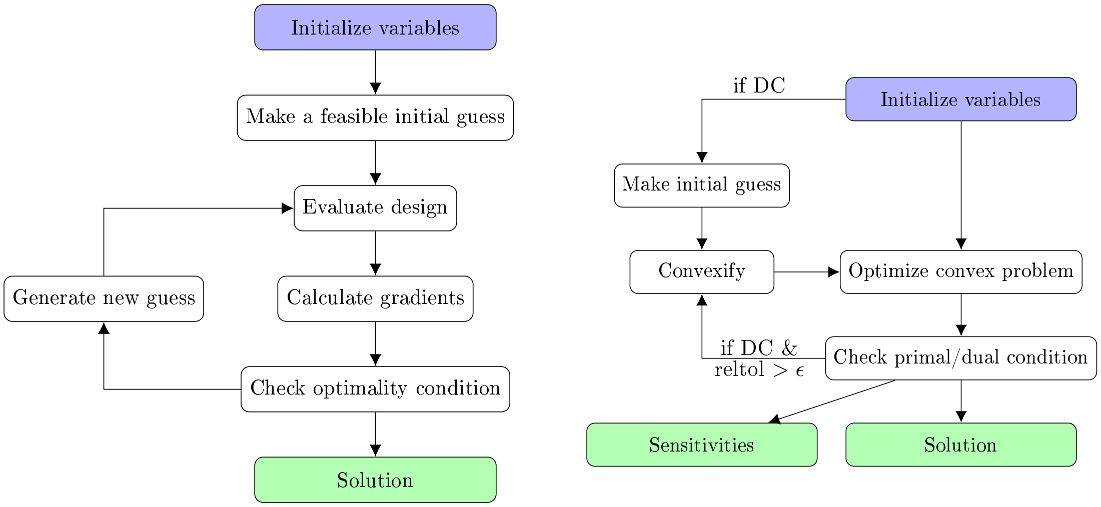
2. *Models are extensible and modular.*

   Models in GP can be made arbitrarily complex. The 'bag of constraints' form of the GP means that there is no need to reformulate the optimization scheme as more constraints are added. This makes incremental modeling improvements straight-forward. The traditional engineering design process is split into conceptual, preliminary and critical design segments. GP modeling facilitates this process by allowing ever-increasing levels of complexity.

Gradient-based optimization methods for multimodal, multicomponent systems often involve convergence loops, as shown in Figure 1-1, which have to be re-engineered when new constraints are introduced. Furthermore, the designer has to tune the module for generating new guesses from gradient information, which is unreliable at best. GPs (and the GP approximations of SPs) are solved all-at-once [13], which means that there are no constraint convergence loops to worry about or parameters to tune. The form of the GP facilitates the addition of variables and constraints while extending model capabilities. Please refer to [13] for a more in-depth analysis of various MDO architectures.

A big advantage of the convexity of the GP is the low-cost computation of constraint and variable sensitivities by leveraging Lagrange duality [6]. This helps determine which parts of the model yield the greatest returns in terms of fidelity to improved modeling, so engineers can target their efforts.

3. *Models are amenable to mission and multi-mission design, and are compatible with modern engineering design methodologies.*


Initialize variables
Make a feasible initial guess
Evaluate design
Generate new guess
Calculate gradients
Check optimality condition
Solution
if DC
Initialize variables
Make initial guess
Convexify
Optimize convex problem
if DC &
reltol > ϵ
Check primal/dual condition
Sensitivities
Solution

(a) Gradient-based optimization

(b) Convex, and difference-of-convex (DC) optimization

Figure 1-1: The flow diagrams of two methods of optimization.

> The design tools available in GPkit make it easy to implement mission design, and build models that are shared between different design problems. Mission design helps engineers gain valuable intuition about the tradeoffs in the performance of a design, and multi-mission design allows designs to be able to satisfy a variety of missions and mission objectives.

This thesis will methodically demonstrate the advantages of GP in modeling and exploring complex engineering trade spaces.

Chapter 2

Engineering inequalities and intuition, from equalities

This section will demonstrate how the form of the GP (monomial equalities and posynomial inequalities) helps engineers develop intuition about the direction each variable is pressured by a given objective function. Please refer to Appendices A.1 and A.2 for detailed descriptions of the forms of GPs and SPs respectively. The intuition gained in turn helps formulate constraints that can properly bound each variable in the optimization model. GPkit, CEG's GP modeling framework, facilitates this process by providing feedback to designers about the boundedness of a model.

A simple aircraft optimization problem named SimPleAC[1] will be derived to demonstrate the intuitive structure of GPs and SPs, and show how we can introduce new constraints to bound the feasibility sets of models.

2.1 Making feasibility sets and boundedness explicit

Consider the simple design problem below, where we define a simple bilinear monomial equality with respect to x and y, and constrain the sum of the variables:

---

[1] 'S' and 'P' have been capitalized to designate that the model will be a 'signomial program', and 'AC' is an acronym for 'aircraft'.

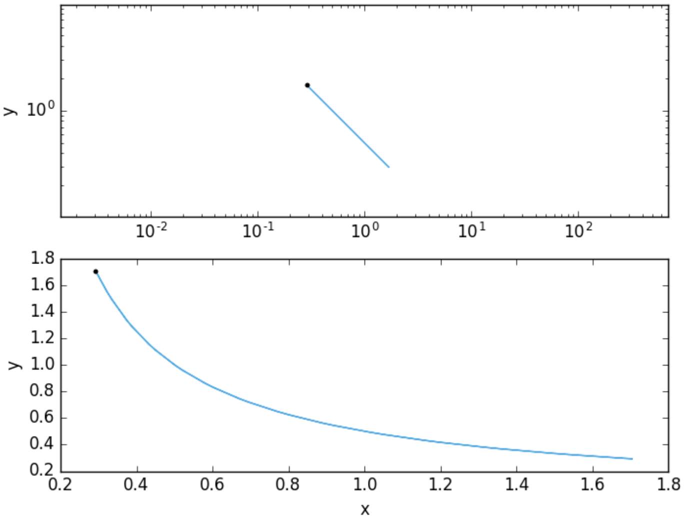


Figure 2-1: The x-y feasibility set of a simple monomial equality.

$$\begin{aligned} &\text{minimize} && x \\ &\text{subject to} && x + y \leq 2, \qquad xy = \frac{1}{2} \end{aligned}$$

We can draw the feasibility set of the above problem in both linear and log-space, as shown in Figure 2-1. One expects that, given only two variables related through an equality and bounded by an inequality, the feasibility space is a finite line segment in log-space, and a finite exponential function in linear space. The black dot in Figure 2-1 is the global optimum of the feasibility set ($x = 1 - \frac{1}{\sqrt{2}}$, $y = 1 + \frac{1}{\sqrt{2}}$).

However, if we had decided to impose $xy \geq \frac{1}{2}$ instead of $xy = \frac{1}{2}$, we would get a new feasibility set as shown in Figure 2-2. Using the GP form, we can always upper-bound posynomials, and lower-bound both posynomials and monomials to get convex feasible sets. Note that the optimal point, which is at $x = 1 - \frac{1}{\sqrt{2}}$, $y = 1 + \frac{1}{\sqrt{2}}$, does not change when the monomial equality is relaxed. This key observation will allow us to turn the equalities in most physical models into GP-compatible posynomial inequalities. The posynomial equality relaxation is explained in greater detail in [6].

However if we convert the monomial equality to $xy \leq \frac{1}{2}$, we will get an unbounded model, whose feasibility set is shown in Figure 2-3. Since minimizing x is our objective, and both variables are only upper-bounded, they both collapse towards numerical pre-

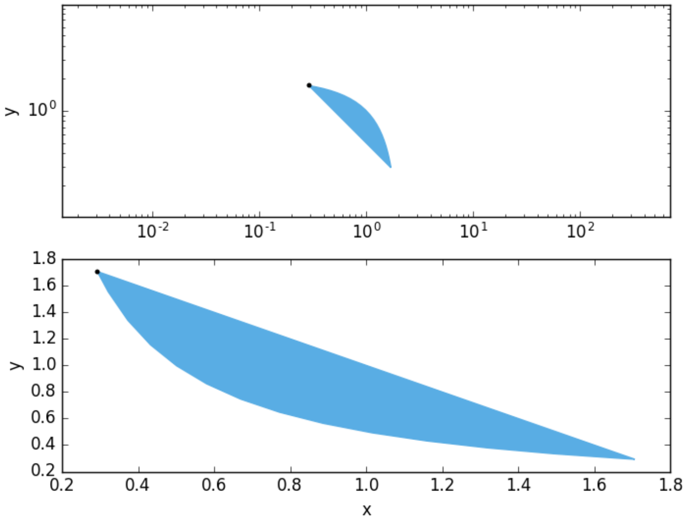


Figure 2-2: The x-y feasibility set of lower-bounding monomial, and upper-bounding posynomial.

cision zero, giving the vanishing feasibility set shown in Figure 2-3. This section will demonstrate methods to create GP-compatible models that are adequately bounded to avoid such singularities towards zero or infinity, which for the most part do not exist in real physical models. The models will be created within GPkit [3].

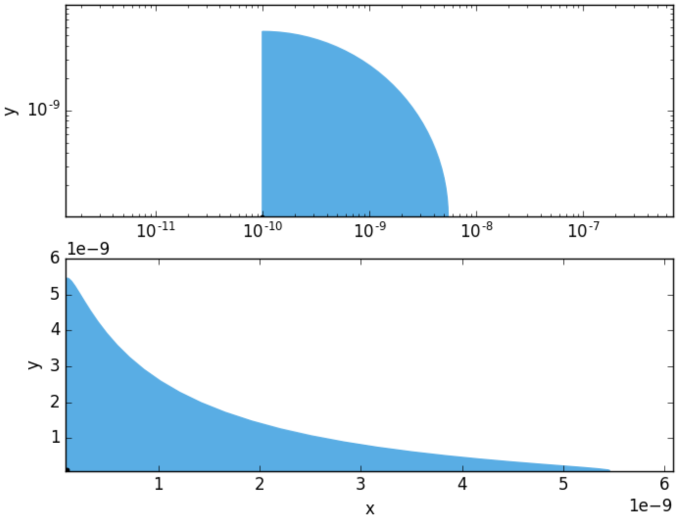


Figure 2-3: The x-y feasibility set of upper-bounding monomial, and upper-bounding posynomial.

2.2 Defining the design problem

GPs are amenable to solving a large variety of design problems (see [1] for an extensive number of examples). This thesis uses aircraft design to demonstrate design methodologies for convex optimization since the author's background is in aerospace engineering. Aircraft epitomize the nature of complex engineering problems. The physical relations describing their motion are nonlinear, and all of their subsystems are coupled through the primary forces in flight (thrust, weight, lift, drag). The goal of the aircraft in question will be to perform a basic 'ferry' mission, that is to carry a given payload over a distance while minimizing an objective function.

2.2.1 Objective functions

Objective functions are the way that a designer puts pressure on the variables in an optimization problem. To begin with, we will consider total fuel weight W_f as our objective, which will put downward pressure on all of the variables that would cause greater fuel burn, namely drag and weight. We must necessarily lower-bound all variables in the objective function that have positive exponents, and upper-bound all variables with negative exponents.

In many design problems formulated as GPs, many different objectives will put pressure on design in the same direction. For example, an aircraft optimized for fuel weight will look different compared to one that has been designed for total weight or payload-fuel consumption[2], but each of these objectives will put a downward pressure on drag and weight. As such, this model will be able to take in a number of objective functions and be properly constrained and bounded. According to Raymer, an important principle of aircraft design is 'that there is no such thing as a free lunch!' [16, pg.26]. An improvement in one objective function will result in reduced performance with respect to others.

---

[2] Payload-fuel consumption is the ratio of fuel weight to payload weight, a useful efficiency parameter.

2.2.2 Functional description: constraining the problem

The typical process for designing anything usually involves doing either a component decomposition or a functional decomposition of the problem. In this case, we will think about the functional decomposition to create a basic list of constraints that our aircraft will need to satisfy to be able to capture the tradeoffs in an aircraft design problem. (In Section 3.1.3, we will examine how the component decomposition can help structure larger problems.)

What does an aircraft need to be able to do to deliver payload over a distance?

- It will need to sustain steady level flight, keeping itself and the payload aloft (Section 2.3.1).
- It will need to overcome drag (Section 2.3.3).
- It will need to contain enough fuel to complete its mission (Section 2.4.1).
- It will need to sustain its structural loads (Section 2.4.3).

Note that these are in no way presented in order of importance, which is reflective of the non-hierarchical nature of GPs. In the basic example, we will choose not to model engines, and leave this as an exercise to complete in Section 3.1 to improve the model.

2.3 GP modeling from physics

Optimization model creation often starts haphazardly, with the designer having a vague idea about the set of physics that govern a problem, and some basic assumptions about the configuration. In this section, we will generate variables and constraints with abandon, and think about how to make sure each variable is adequately bounded later. Each subsection is intended to introduce the reader to an important aspect of GP modeling, accompanied by examples in implementation.

2.3.1 Free and fixed variables: weight and lift model

For this particular design problem, we start by modeling weight and lift, since one of the fundamental functions of the aircraft is to stay aloft. The aircraft has weight, which consists of the payload, wing, and fuel weights.

$$W \geq W_p + W_w + W_f \tag{2.1}$$

Note that we have already had to make determination about the relation between the two sides of the equation. Heavier aircraft burn more fuel, so we are justified to put total weight as greater than the sum of the component weights since W will be pressured downward by the objective. Furthermore, we can always add more weight to the aircraft by adding ballast to it, even if this would likely worsen the objective. So we allow for slackness in this constraint, even if we know intuitively that it will almost always be tight as explained in Section 2.2.1.

The aircraft has to sustain steady level flight, so it needs to generate enough lift. We use the naive lift is greater than weight ($L \geq W$) model below for steady level flight:

$$\frac{1}{2}\rho V^2 S C_L \geq W_p + W_w + 0.5W_f \tag{2.2}$$

where the lift of the aircraft is equal to weight of the aircraft with half-fuel, which is a crude estimate of the average weight of the aircraft throughout the flight. Again, the GP form is seamless here, since lift is related to induced drag, and so it is pressured downward into the posynomial on the right hand side (RHS) of the equation.

We would also like the fully-fueled aircraft to be able to fly at a minimum speed of V_{min} without stalling, so we add the following constraint:

$$W \leq \frac{1}{2}\rho V_{min}^2 S C_{L_{max}} \tag{2.3}$$

Note that, although we could use a monomial equality here, we don't, because this relation does not need to be tight. It is acceptable that the aircraft is able to fly

Variable	**Value**	**Units**	**Description**
ρ	1.23	$\frac{\mathrm{kg}}{\mathrm{m}^3}$	density of air
C_L	...	–	wing lift coefficient
$C_{L,max}$	1.6	–	lift coefficient at stall
S	...	m^2	total wing area
V	...	$\frac{\mathrm{m}}{\mathrm{s}}$	cruising speed
V_{min}	25	$\frac{\mathrm{m}}{\mathrm{s}}$	takeoff speed
W	...	N	total aircraft weight
W_f	...	N	fuel weight
W_w	...	N	wing weight
W_p	6250	N	payload weight

Table 2.1: Variables introduced in the weight and lift model.

at a velocity slower than V_{min} for a given objective function.

At this point, we have introduced a large set of variables, some of which are input parameters, and the others free variables. The decision of whether to keep variables free or fixed can shape the model development process in all forms of optimization. The decision can be influenced by many factors in a GP, with four in particular that stand out in this model. We set the lift coefficient and takeoff speed to be constants since (I) we would need more detailed modeling to determine their values. Payload weight is set because (II) it would always be unbounded towards zero due to the downward pressure from the objective. A designer may also fix variables (III) if he/she knows their values with certainty (e.g. gravitational acceleration g) or (IV) the variable is a normalizing coefficient (which we will see in Section 3.1.2). The variables have been defined in Table 2.1, where the fixed variables have associated values. The remaining variables are free variables, to be optimized once the model is appropriately bounded.

Note that the atmospheric density ρ in Table 2.1, and other atmospheric variables in Section 2 are set to be constant. The behavior of these variables with altitude will be modeled in more detail in Section 3.2 when we introduce an aircraft flight profile.

Variable	**Value**	**Units**	**Description**
C_D	...	–	drag coefficient
C_L	...	–	wing lift coefficient
L/D	...	–	lift-to-drag ratio
$Range$	3000	km	aircraft range
T_{flight}	...	hr	flight time

Table 2.2: Variables introduced to define new performance metrics.

2.3.2 Alternate objectives: more performance metrics

As multi-objective designers, we may also be interested in knowing about additional performance metrics that can serve as part of objective functions. A few that are particularly relevant to aircraft will be introduced here.

The time of flight of the aircraft, which is a useful metric to calculate time cost, is simply the aircraft's range divided by its cruise velocity:

$$T_{flight} \geq \frac{\text{Range}}{V} \tag{2.4}$$

The lift-to-drag ratio is also defined:

$$L/D = \frac{C_L}{C_D} \tag{2.5}$$

The new variables we have introduced are detailed in Table 2.2.

An important note about variables that are potential alternate objectives: these variables must always be lower-bounded (or inverted and upper-bounded) since the general GP is a minimization problem. However, if they are not also upper-bounded, these variables will likely be unbounded for a given model and run off to $+\infty$. This means that performance-quantifying variables will need to be in a monomial form, or must be present in a posynomial objective function as shown in Equation 2.6 for boundedness. We shall see this coming into play in Section 3.2.3.

$$\text{Objective} \geq \sum_{i=1}^{n} c_i \prod_{i=1}^{n} v^{k_{i,j}} \tag{2.6}$$

2.3.3 More physics for boundedness: thrust and drag model

If we attempt to run our current model, we would find that it is unbounded in many variables. The results are shown in Table 2.3.

Unbounded variable	Units	Direction
S	m^2	∞
T_{flight}	hr	∞
V	$\frac{\text{m}}{\text{s}}$	∞
W_f	N	0
W_w	N	0

Table 2.3: Unbounded variables in the weight and lift model.

This is not surprising at all, since none of the defined constraints lower-bound W_f, the objective function. Additionally, without pressure from the objective, any variables that are not both upper- and lower- bounded will tend to blow up. This is indicative usually that more modeling or direct substitutions are required to sufficiently bound the variables.

In this case, we lack a propulsion model which would properly bound fuel weight, velocity, and time of flight. For initial modeling purposes, we assume a naive constant brake specific fuel consumption (BSFC) for the 'engine' of the aircraft, which is assumed to provide as much thrust as needed. Since $T \geq D$:

$$W_f \geq g \times \text{BSFC} \times T_{flight} \times DV \tag{2.7}$$

the fuel weight required is the product of gravitational acceleration, BSFC, time of flight, and the total drag power on the aircraft. The drag is the product of dynamic pressure ($\frac{1}{2}\rho V^2$), planform area S, and the coefficient of drag of the aircraft:

$$D \geq \frac{1}{2}\rho V^2 S C_D \tag{2.8}$$

There are yet more relaxed monomial equalities in Equations 2.7 and 2.8. If the pressure on the left hand side (LHS) or RHS of a monomial equality are clear as in these cases, it is a good practice to relax the equality to leave as many degrees of freedom in the design space as possible. The intuition is that we can almost always spend more fuel or have more drag, but we are confident that the constraints will be tight since our objective will suffer as a consequence.

The drag coefficient of the aircraft is assumed to be the sum of the fuselage drag, the wing profile drag, and the wing induced drag coefficients [7]:

$$C_D \geq C_{D_{fuse}} + C_{D_{wpar}} + C_{D_{ind}} \tag{2.9}$$

The individual components of the drag are represented as monomial equalities, borrowing constraints 2.10 through 2.15 from [7]. The fuselage drag is a function of its drag area CDA_0 and the planform area of the wing:

$$C_{D_{fuse}} = \frac{CDA_0}{S} \tag{2.10}$$

where the CDA_0 is linearly proportional to the volume of fuel in the fuselage:

$$V_{f_{fuse}} = CDA_0 \times 10 \text{ m} \tag{2.11}$$

Note that we correct the dimensionality of the volume here, since GPkit automatically checks units.

The wing profile drag is the product of the form factor, the friction drag coefficient, and the wetted area ratio of the wing [7],

$$C_{D_{wpar}} = kC_f S_{wetratio} \tag{2.12}$$

The Reynolds number of the aircraft wing is approximated

$$Re \leq \frac{\rho}{\mu} V \sqrt{\frac{S}{AR}} \tag{2.13}$$

and used to find the friction drag coefficient of wing. We approximate the C_f by

Variable	**Value**	**Units**	**Description**
$\mathcal{R}$	...	–	aspect ratio
BSFC	400	$\frac{\mathrm{g}}{\mathrm{kW \cdot hr}}$	brake specific fuel consumption
CDA_0	...	m^2	fuselage drag area
C_f	...	–	skin friction coefficient
D	...	N	total drag force
e	0.92	–	Oswald efficiency factor
k	1.17	–	form factor
μ	1.78×10^{-5}	$\frac{\mathrm{kg}}{\mathrm{m \cdot s}}$	viscosity of air
Re	...	–	Reynolds number
$\left(\frac{S}{S_{wet}}\right)$	2.075	–	wetted area ratio
$V_{f_{fuse}}$	...	m^3	fuel volume in the fuselage

Table 2.4: Variables introduced in the thrust and drag model.

Unbounded variable	Units	Direction
$V_{f_{fuse}}$	m^3	0
W_w	N	0

Table 2.5: Unbounded variables in the GP-compatible formulation.

assuming a turbulent flat plate flow:

$$C_f \geq \frac{0.074}{Re^{0.2}} \tag{2.14}$$

The induced drag of the wing is calculated with a span efficiency factor e, and is a function of the C_L and aspect ratio $\mathcal{R}$ of the wing.

$$C_{D_{induced}} = \frac{C_L^2}{\pi \mathcal{R} e} \tag{2.15}$$

The new variables are detailed in Table 2.4.

As shown in Table 2.5, attempting to run the model as is results in both the fuselage fuel volume $V_{f_{fuse}}$ and the wing weight W_w still having no lower bounds. These variables will need to be properly bounded to complete the SimPleAC model.

2.4 Limits of GP and convexity, and SP modeling

Even with the demonstrated strengths of GPs in solving certain classes of problems, it is important to recognize that the mathematical framework has limits. The three distinct types of GP-incompatibility in design problems are detailed by Hoburg [6], which are discreteness, quasi-convexity, and multi-modality. Discreteness in the GP can be approached by coupling discrete programming methods such as branch-and-bound into a sequential GP. This is outside of the scope of this thesis. Quasi-convexity and to a certain extent multimodality can be addressed through a non-log-convex extension of GPs called SPs, where certain constraints are expressed as signomial (or difference-of-posynomial) constraints (described in greater detail in Appendix A.2). Even the addition of a single signomial constraint turns the problem from a GP to a SP, which means that the problems loses convexity and all of the mathematical guarantees associated with it. It takes engineering intuition to recognize where and when improved modeling is worth the loss of the mathematical guarantees. Kirschen [10] describes in greater detail how signomial constraints are often required to capture fundamental design tradeoffs.

2.4.1 Signomial constraints: fuel volume model

In an attempt to put a lower bound $V_{f_{fuse}}$, we will be adding a fuel volume model to SimPleAC, where fuel can be stored in the wing or in the fuselage. The fuel volume will be modeled first instead of the wing weight because the wing weight will be a function of the fuel stored in the fuselage. The reason why this model is GP-incompatible is because of the following constraint which follows logically:

$$V_{f_{avail}} \leq V_{f_{wing}} + V_{f_{fuse}} \tag{2.16}$$

The fuel volume available must be less than the sum of the fuel volume available in the wing and the fuselage. It turns out that volumes that 'contain' free variables can create signomial constraints. (One way around this is potentially creating fuel fraction variables to denote how much fuel is stored in each volume, but other potential parametrizations will not be explored here.)

As such, we can continue to develop the model, since it is important for us to capture the fuel distribution between the wing and the fuselage. Fuel weight is going to influence the lift required of the aircraft, so the weight of the fuel is determined using a density parameter ρ_f.

$$V_f = \frac{W_f}{\rho_f g} \tag{2.17}$$

We need a model of how much fuel volume there is in a wing. Intuitively, we would expect the volume within a wing to be related linearly to its thickness ratio (τ) and span (b), and to the square of its chord (c).

$$V_{f_{wing}} \propto \tau b c^2 \tag{2.18}$$

It is convenient to express relation 2.18 in terms of planform area S, aspect ratio AR and thickness ratio τ only. Using the additional relations $AR = \frac{b^2}{S}$ and $S \propto bc$, we can express $V_{f_{wing}}$.

$$V_{f_{wing}} \propto \tau \left(\frac{AR}{S}\right)^{0.5} \left(\frac{S}{b}\right)^{2} \propto \left(\frac{AR}{S}\right)^{0.5} \frac{S^2}{SAR} \propto \frac{\sqrt{S}\tau}{\sqrt{AR}} \tag{2.19}$$

$$V_{f_{wing}}^2 \leq 9 \times 10^{-4}\ \mathrm{m}^4 \times \frac{S\tau^2}{AR} \tag{2.20}$$

Such variable transformations can be useful to have a minimal parametrization of designs. One can solve the minimal optimization problem, and post-process the solution of the problem to get a complete geometry as necessary. The new variables introduced to bound fuel volume are in Table 2.6. (In Equation 2.20, the constant 9×10^{-4} m^4 was picked as the coefficient in front of the relation by tuning it after the model was developed, but any other coefficient would work.)

Variable	**Value**	**Units**	**Description**
ρ_f	817	$\frac{\mathrm{kg}}{\mathrm{m}^3}$	density of fuel
g	9.81	$\frac{\mathrm{m}}{\mathrm{s}^2}$	gravitational acceleration
τ	0.12	–	airfoil thickness to chord ratio
V_f	...	m^3	fuel volume
$V_{f_{avail}}$	...	m^3	fuel volume available
$V_{f_{wing}}$	...	m^3	fuel volume in the wing

Table 2.6: Variables introduced in the fuel model.

2.4.2 Arguments for the signomial equality

This segue will explain and motivate the use of signomial equalities, as described in [15], in SP modeling. Signomial equalities must be used as a last resort. The signomial equality is the only place where the feasibility set of individual GPs within a SP solve are not guaranteed to be subsets of the feasibility set of the SP. This is because the signomial solution algorithm in GPkit flattens the original signomial equality constraint, a concave curve in log-space in n-dimensions, onto a line in log-space in n-dimensions(Method C in [15]) that intersects the original constraint at the optimal point of the last GP solve. This is undesirable, although the final solution of the SP with equalities is guaranteed to be in the feasibility region of the SP. Furthermore, SPs with signomial equalities have been demonstrated to require more GP solves than SPs without signomial equalities. However, there are a few arguments to be made in defense of signomial equalities.

One good use case of the signomial equality is in constraints in which the direction of pressure on free variables is not clear. This ensures the tightness of constraints that may otherwise have unbounded variables. A good example is in atmospheric models. Although the pressure on air viscosity μ is almost certainly downward since it results in lower drag, the pressure on air density ρ is not clear because of the tradeoffs between aircraft endurance and range.

The second reason is that we are not interested in the 'feasibility set' of the atmosphere, since this has no intuition behind it: at every altitude, the atmospheric quantities can only be be represented by single quantities. Additionally, the computational penalty of implementing signomial equalities in atmospheric models is low, since the monomial approximation to the atmospheric data is not far from the monomial approximations made to the signomial equality. In fact, when we add an atmospheric model to SimPleAC in Section 3.2, we will be implementing signomial equalities to represent both air density ρ and viscosity μ.

2.4.3 Completing the model: wing structural model

Only one unbounded variable remains, which is wing weight W_w. We can think of wing weight as having two components, the skin weight that only grows as a function of wing area and the structural weight which is a function of both the geometry and loading. The surface weight expression is straightforward.

$$W_{w_{surf}} \geq W_{w_{coeff2}} S \tag{2.21}$$

We would like wing structural weight to account for the loading distribution and the geometry of the wing. The wing will have to sustain a maximum bending load (we will neglect shear, since the two are coupled) due to maximum takeoff weight, multiplied by an ultimate structural factor N_{ult} for maneuvering. I have borrowed the wing weight model from [7], and adapted it through a structural weight coefficient $W_{w_{coeff1}}$. Note that this equation captures the major trends in wing structural sizing. We can see this through looking at the partial derivatives of the wing weight with respect to the different free variables. The weight grows with the cube of the span ($AR^{1.5} = \frac{b^3}{S^{1.5}_{constant}}$, derived from the integration of a quadratically increasing bending moment), linearly with the maximum structural factor N_{ult}, and inversely with the surface area[3]. Using similar partial-derivative based analyses, it is often easy to make first-order models for components.

$$W_{w_{strc}} \geq \frac{W_{w_{coeff1}}}{\tau} N_{ult} AR^{1.5} \sqrt{(W_0 + \rho_f g V_{f_{fuse}}) W S} \tag{2.22}$$

---

[3] This can be difficult to see, but since $S \propto b_{constant} c$ and loading is constant, as area grows, the thickness of the wing grows as well at a constant τ. So the weight increases linearly with S, and stiffness increases with the cube of S. Integrated over the whole wing this yields a S^{-1} relation.

Variable	**Value**	**Units**	**Description**
N_{ult}	3.3	–	ultimate load factor
$W_{w_{coeff1}}$	2×10^{-5}	$\frac{1}{\mathrm{m}}$	wing weight coefficient 1
$W_{w_{coeff2}}$	60	Pa	wing weight coefficient 2
$W_{w_{strc}}$	...	N	wing structural weight
$W_{w_{surf}}$	...	N	wing skin weight

Table 2.7: Variables introduced in the wing structural model.

Equation 2.22 takes into account the root bending moment relief due to presence of fuel and weight in the wings by performing a geometric average of the total weight, and the weight excluding wing fuel and wing weight.

The total wing weight is now lower-bounded by its component weights, and we have introduced the final set of variables in Table 2.7.

$$W_w \geq W_{w_{surf}} + W_{w_{strc}} \tag{2.23}$$

2.5 Results of SimPleAC

The benefits of convex optimization and GP in both solution quality (in terms of mathematical guarantees) and the low-cost computation of sensitivities have been detailed in ([7],[11]), so these benefits will not be featured here. However, the values of the free variables, and the sensitivities of the fixed parameters are presented for the reader.

Table 2.8: Values of free variables in the SimPleAC model.

Free Variables	Value	Units
$(CDA0)$	0.004751	m^2
A	23.41	
C_D	0.01928	
C_L	0.7867	
C_f	0.004054	
$C_{D_{fuse}}$	2.902×10^{-4}	

$C_{D_{ind}}$	0.009149	
$C_{D_{wpar}}$	0.009843	
D	237.2	N
L/D	40.8	
Re	2.026×10^6	
S	16.37	m^2
T_{flight}	23.84	hr
V	34.96	$\frac{\mathrm{m}}{\mathrm{s}}$
V_f	0.09678	m^3
$V_{f_{avail}}$	0.09678	m^3
$V_{f_{fuse}}$	0.04751	m^3
$V_{f_{wing}}$	0.04928	m^3
W	1.007×10^4	N
W_f	775.7	N
W_w	3041	N
$W_{w_{strc}}$	2059	N
$W_{w_{surf}}$	982.1	N

Table 2.10: Sensitivities of parameters in the SimPleAC model.

Sensitivities	Value
BSFC	+1.1
$Range$	+1.1
W_p	+1.1
g	+1.1
$\left(\frac{S}{S_{wet}}\right)$	+0.57
k	+0.57
e	-0.53
V_{min}	-0.49
τ	-0.34
N_{ult}	+0.31
$W_{w_{coeff1}}$	+0.31
ρ	-0.3
$C_{L,max}$	-0.24

$W_{w_{coeff2}}$	+0.15
μ	+0.11
ρ_f	-0.044

Chapter 3

Extensibility of GP

Recalling from Figure 1-1, traditional gradient-based design optimization tools implement convergence loops that assume structure within a given design problem. The 'bag of constraints' form of the GP means that constraints can be added to the problem without having to restructure the optimization formulation. This property, coupled with the object-oriented modeling framework of GPkit, allows GP compatible models to be continuously extensible. This section will demonstrate common methods used to extend the capability and improve the fidelity of GP- and SP-compatible models.

3.1 Modularization and improved fidelity: engine model

The aircraft currently has an engine that weighs nothing and magically supplies unlimited power. This is obviously unphysical, and requires refinement.

3.1.1 Creating an engine submodel

Before even thinking about modeling, we would like to leverage the object-oriented GPkit models to put the variables describing the engine into a submodel (currently only consisting of the BSFC variable). In the GPkit software, we do this by creating a new class (an object in the Python language) called **Engine** and creating a *setup* method that returns the constraints within it. The **Model** and **Variable** objects in the sample code are imported from GPkit.

```
class Engine(Model):
    def setup(self):
        # Dimensional constants
        BSFC = Variable("BSFC", 400, "g/(kw*hr)",
                                    "brake specific fuel consumption")
        constraints = []
        return constraints
```

We allow the SimPleAC model to contain the variables and constraints of the engine as follows:

```
class SimPleAC(Model):
    def setup(self):
        self.engine = Engine()
        self.components = [self.engine]
        ...
        return constraints, self.components
```

This restructuring of the model yields the exact same GP formulation as the unstructured problem, but gives us the flexibility to develop submodels collaboratively and in a disciplined manner.

If we think of an engine as an input-output system, we can determine how it would interact with the SimPleAC system, and create appropriately bounded sets of variables. At the most basic level, an engine provides shaft power, consumes fuel, and has weight. The model is missing both the shaft power and weight description of the engine. If we abstract away the propeller (the relation between shaft power and thrust power) through a propeller efficiency, we can perhaps relate maximum power to weight.

3.1.2 Data-based modeling: engine power vs. weight

We can imagine that, for a specific kind of engine, there is a relation between the maximum shaft power available and the mass of the engine, somewhat related to the cube-square law, which describes the relation between the surface area and volume of objects. And let's assume that our knowledge of the internal workings of engines is limited, but we have some knowledge of the technology available in the market and have data to support it. Using GPfit [8], we will try to fit the data to find GP compatible relations between engine weight and maximum power. This section will try to highlight the best practices when making data-based models.

To be able to fit the engine power versus weight data, we take several important steps.

- **Comb the data.** Since we are essentially projecting data with potentially high standard deviation onto a single line, it is important to fit the ranges of data we care about.

- **Normalize the data.** Normalizing the data by some known quantity is preferable, since fits should not be dependent on the units that are used while performing it. This also helps the fit integrate seamlessly into GPkit, since dimensional fits would require units manipulation to avoid errors. The data can be normalized by reference quantities (in this case using the maximum power and weight values from the data set).

- **Choose the type of fit.** In [8], *softmax-affine* (SMA) and *implicit softmax-affine* (ISMA) functions are proposed and implemented as convex approximations to data. Depending on the behavior of the data, one or the other may be appropriate. For engineering relations that are expected to be smooth, SMA functions are often good approximations. However, if kinks are expected in the functions, ISMA functions can locally adjust the softness of the fit to reduce the error of the fit.

- **Choose the number of posynomial terms in the fit.** The number of posynomial terms can be changed to better capture the trends in the data. RMS error can be reduced by including more posynomial terms, but only if the variable of interest has downward pressure on it from the objective function (since it is on the greater side of the inequality). Otherwise fits are limited to monomial equalities in n-dimensions.

After having performed these intermediate steps on the engine data, the relation we obtain for the one-term (monomial) approximation is as follows:

$$\left(\frac{W_{eng}}{W_{eng,max}}\right)^{0.100} = 0.988\left(\frac{P_{shaft}}{P_{shaft,max}}\right)^{0.117} \tag{3.1}$$

This fit, shown by the dashed line in Figure 3-1, has a root mean square error of 0.414, which has to do both with the quality of the fit and the level of variation in the data. Since engine weight will have downward pressure on it from the objective function, we can easily use a two-term posynomial approximation to improve its error.

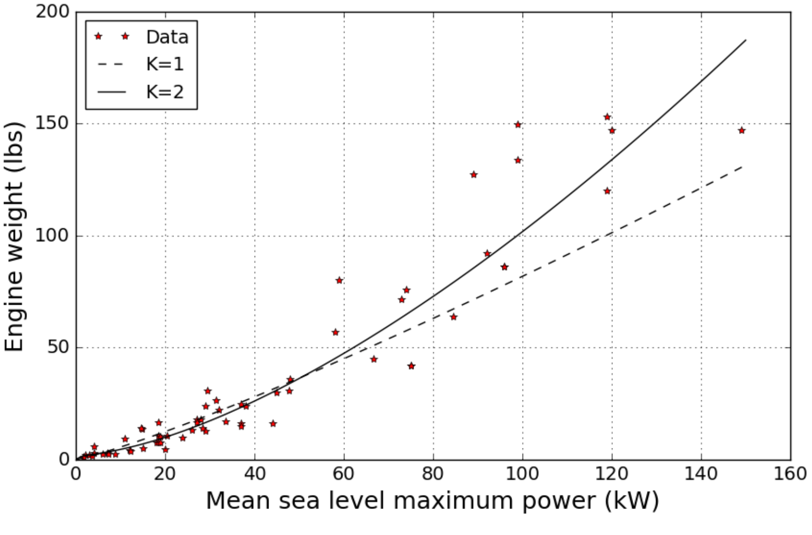


Figure 3-1: Engine MSL power versus weight fits for $K = 1, 2$ posynomial terms with underlying data.

$$\left(\frac{W_{eng}}{W_{eng,max}}\right)^{1.92} \geq 4.41 \times 10^{-3}\left(\frac{P_{shaft}}{P_{shaft,max}}\right)^{0.759} + 1.44\left(\frac{P_{shaft}}{P_{shaft,max}}\right)^{2.90} \tag{3.2}$$

This relation has an RMS error of 0.346, which is a significant improvement. Both fits are shown with the data in Figure 3-1 for comparison.

With a SMA approximation, adding more than two terms to the fit do not improve its RMS error on the given data, due to the large standard deviation of the data used. As such, we will proceed with the 2-term posynomial fit.

Other constraints in engine model

The cruise shaft power is constrained to be 20% of the maximum shaft power of the engine, to account for engine surge power demands and add engineering realism. This rather arbitrary constraint is removed later when the full mission model is integrated.

$$P_{shaft} \leq \frac{1}{5} P_{shaft,max} \tag{3.3}$$

3.1.3 Converting all subsystems into submodels

Within this framework, we can modularize the SimPleAC into wing and fuselage modules as well, with very little additional work. This creates the variable and constraint hierarchy as presented in Figure 3-2, which define all of the constraints required for SimPleAC to fly one flight segment.

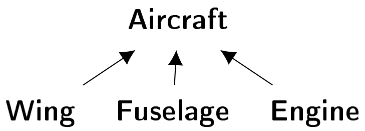


Figure 3-2: Variable and constraint hierarchy of the SimPleAC model for a single flight segment.

Uni-directional graph structures such as in Figure 3-2 are informative, since they provide an intuitive representation of the way constraints and variables are passed between GPkit models. In this basic framework, variables and constraints from one model can only be called by models that are higher and connected in the diagram. This reconciles the fact that object creation in software engineering is serial, whereas the components of the system being optimized are interconnected. The way the SP is solved at the end has no hierarchy, but as we will see in Section 3.2 a hierarchical representation will facilitate the vectorization of constraints required for mission design.

3.2 Mission design and performance modeling form

The SimPleAC defined so far works well to demonstrate the capabilities of SPs in helping explore tradeoffs in engineering design. However, often in the design process, we will want to test the performance of a design in different conditions, and/or during different phases of a mission. This requires the vectorization of constraints that relate to the performance of the design. What we'd like to do is to have a single aircraft optimize both its static sizing variables (having to do with the airframe), and its flight performance simultaneously. This requires a major augmentation of the model graph defined in Figure 3-2, into a uni-directional graph as shown in Figure 3-3.

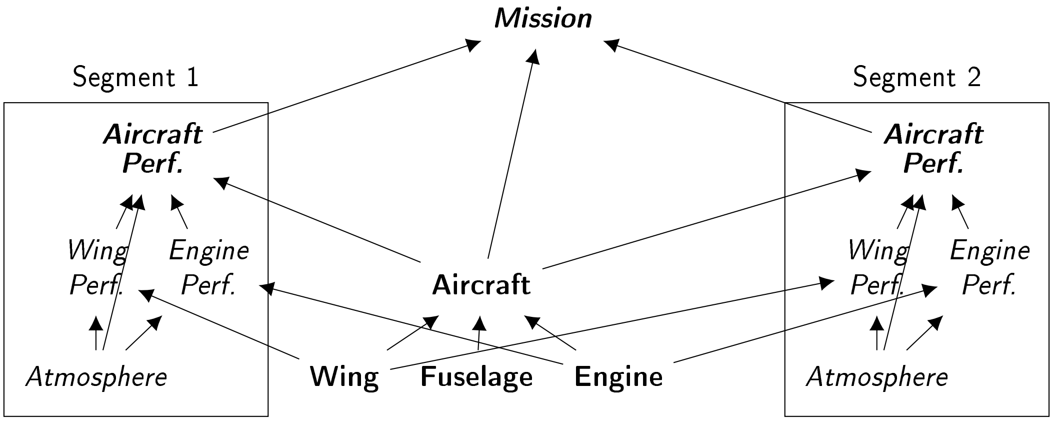


Figure 3-3: Variable and constraint hierarchy of the SimPleAC static+performance model for two flight segments. Models that include sizing variables are bolded while models that include performance variables are italicized. There are models that contain both kinds of variables.

Figure 3-3 represents a model with two flight segments, where the models enclosed in rectangles contain the set of constraints that are vectorized by the number of flight segments, $N_{segments} = 2$. Each one of the performance models contains variables that change between flight segments. Note that the fuselage is the only subcomponent not to have a performance model. This is because the drag coefficient of the fuselage is assumed to be constant between flight segments, making it static. A ***Mission*** model that links flight segments together has been added, as well as an *Atmosphere* model, which describes the conditions in which the aircraft operates.

The static aircraft model and the atmospheric state are passed as arguments to multiple performance models within this framework. To transform our previously static model to the performance-static model hierarchy we have identified, we have to determine which variables belong in which node of the graph. Table 3.1 details the full decomposition of the model into its submodels in the format defined by Figure 3-3. This is as simple as identifying which variables we do not expect to change during flight segments, and which ones we do. Note that some of the variables from the previous sections have been renamed (e.g. $T_{flight} \rightarrow t_m$ and $W_f \rightarrow W_{f_m}$) to clarify their purpose within this framework.

Table 3.1: Variables of SimPleAC in static+performance modeling, detailed in the variable and constraint hierarchy.

Variable	Units	Description
Mission		
C	$\frac{1}{\mathrm{hr}}$	hourly cost index
W_{f_m}	N	total mission fuel
t_m	hr	total mission time
$Range$	km	aircraft range
W_p	N	payload weight
t_s	hr	segment time
R_s	km	segment range
W_{f_s}	N	segment fuel burn
W_{start}	N	segment beginning weight
W_{avg}	N	segment average weight
W_{end}	N	segment end weight
$\frac{dh}{dt}$	$\frac{\mathrm{m}}{\mathrm{hr}}$	climb rate
h	m	flight altitude
V_{min}	$\frac{\mathrm{m}}{\mathrm{s}}$	takeoff speed
Mission/*Atmosphere*		
μ	$\frac{\mathrm{kg}}{(\mathrm{m}\cdot\mathrm{s})}$	dynamic viscosity

μ_{MSL}	$\frac{\mathrm{kg}}{(\mathrm{m \cdot s})}$	dynamic viscosity at MSL
ρ	$\frac{\mathrm{kg}}{\mathrm{m}^3}$	density of air
ρ_{MSL}	$\frac{\mathrm{kg}}{\mathrm{m}^3}$	density of air at MSL
h	m	altitude
h_{top}	m	highest altitude valid
***Mission*/SimPleAC**		
V_f	m^3	maximum fuel volume
$V_{f_{avail}}$	m^3	fuel volume available
W	N	maximum takeoff weight
W_f	N	maximum fuel weight
g	$\frac{\mathrm{m}}{\mathrm{s}^2}$	gravitational acceleration
ρ_f	$\frac{\mathrm{kg}}{\mathrm{m}^3}$	density of fuel
***Mission*/SimPleAC/Engine**		
$P_{shaft,max}$	kW	MSL maximum shaft power
$P_{shaft,ref}$	kW	reference MSL maximum shaft power
W_e	N	engine weight
$W_{e,ref}$	N	reference engine weight
η_{prop}		propeller efficiency
***Mission*/SimPleAC/Fuselage**		
$(CDA0)$	m^2	fuselage drag area
$C_{D_{fuse}}$		fuselage drag coefficient
$V_{f_{fuse}}$	m^3	fuel volume in the fuselage
***Mission*/SimPleAC/Wing**		
A		aspect ratio
S	m^2	total wing area
$\left(\frac{S}{S_{wet}}\right)$		wetted area ratio
$V_{f_{wing}}$	m^3	fuel volume in the wing
W_w	N	wing weight
$W_{w_{strc}}$	N	wing structural weight
$W_{w_{surf}}$	N	wing skin weight
N_{ult}		ultimate load factor
$W_{w_{coeff1}}$	$\frac{1}{\mathrm{m}}$	wing weight coefficient 1
$W_{w_{coeff2}}$	Pa	wing weight coefficient 2
$C_{L,max}$		lift coefficient at stall
k		form factor
e		Oswald efficiency factor

τ		airfoil thickness to chord ratio
Mission/SimPleACP		
C_D		drag coefficient
D	N	total drag force
L/D		lift-to-drag ratio
Re		Reynolds number
V	$\frac{\mathrm{m}}{\mathrm{s}}$	cruising speed
Mission/SimPleACP/*EngineP*		
BSFC	$\frac{\mathrm{g}}{(\mathrm{hr}\cdot\mathrm{kW})}$	brake specific fuel consumption
P_{shaft}	kW	shaft power
T	N	propeller thrust
Mission/SimPleACP/*WingP*		
C_L		wing lift coefficient
C_f		skin friction coefficient
$C_{D_{ind}}$		wing induced drag
$C_{D_{wpar}}$		wing profile drag

Then, using the variable structure in Table 3.1, we can place the constraints in the appropriate locations. Each constraint should be placed in the model that contains the variable in the constraint that is highest in the level of hierarchy. For example, we can consider the constraint for thrust power in Equation 3.4.

$$T \times V \leq \eta_{prop} P_{shaft} \tag{3.4}$$

We expect that thrust (T) and shaft power (P_{shaft}) variables exist in *Engine Performance*. Since our model has no model for propeller efficiency (η_{prop}), we treat it as a static parameter in **Engine**. Velocity (V) is a variable in ***Aircraft Performance***. As a result, the constraint for thrust power would logically reside in the ***Aircraft Performance*** model, the highest level in the hierarchy as shown in Figure 3-4. Since this model is vectorized, the constraint would be vectorized by the number of flight segments we create.

Now we have used a framework to modularize our constraints, which makes it

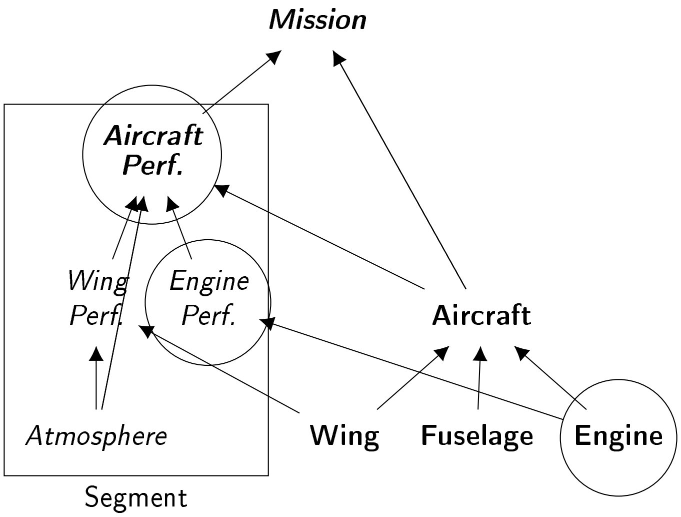


Figure 3-4: Variable hierarchy of thrust constraint 3.4. The models that contain the variables in the constraint are enclosed in circles. Constraint logically resides in ***Aircraft Perf.***. The vectorization of flight segment performance models in the rectangle has been neglected for clarity.

amenable to vectorization and mission design.

3.2.1 Linking performance models: flight segments

Although the variables in the performance models are vectorized, they can be constrained against each other. If each of the ***Aircraft Performance*** models were operating independently of each other simulating different missions, then they would simply be merged in the bag of constraints of ***Mission***. However, we know that the models are related since the aircraft burns fuel throughout the mission, changing its flight characteristics.

The derivation of the SP-compatible flight segment models has been detailed in [11], and used widely within the CEG in aircraft design. It defines segment start, end and average weights, as well as altitude, and all of its relevant constraints are contained in the ***Mission*** model. Please find the full set of variables belonging to the flight segment model in Appendix B.1.

The monomial equality below has been added to the formulation

$$h_{avg_1} = \frac{1}{2}\Delta h_1 \tag{3.5}$$

$$h_{avg_i} = \sqrt{h_i \times h_{i-1}}, \quad i = 2, ..., N_{segments} \tag{3.6}$$

to define an average altitude variable h_{avg} with respect to the segment altitude change variable Δh and segment ending altitude h in Section 3.2.2. This adds conservatism to the density and drag (otherwise, the air density for a flight segment is calculated at the end of the segment, at which the aircraft is at its highest altitude). The cruise altitude (final altitude in every flight segment but the initial segment) has been constrained to be greater than 5000m.

As with most GP approximations, there are limitations to this model. To avoid non-positive altitude change values (Δh), we restrict the aircraft to climb during every segment, and don't model descents. Furthermore, we have binned the flight segments to equal range segments to avoid the potential lower-unboundedness of the lengths of certain segments.

3.2.2 Characterizing the environment: atmospheric model

We have created a mission and flight segment framework without having a model of the environment in which the aircraft operates. So far, we have assumed that the aircraft flies at a constant altitude (sea level) for a single mission segment, and is subject to the same air density and viscosity. An atmospheric model is essential to capturing the tradeoffs between flight altitude, engine performance, and lift and drag characteristics. This simplification is overcome through vectorization.

Tao's atmosphere fits [17] have been borrowed for this purpose. These are 2-term softmax-affine fits of the atmospheric quantities of interest (ρ and μ in this thesis) with respect to altitude. The constraints are guaranteed to be tight through signomial equalities, as explained in Section 2.4.2. The relations are valid between 0-10000m of altitude.

Similar environmental models can be made for other design problems where the environmental variables are inextricably coupled to performance. Another good example of environmental modeling in GP is performed by Burton [4], where wind speeds are integrated into a loitering aircraft optimization problem.

3.2.3 Mission objectives

Recalling from Section 2.3.2, upper-unbounded performance metrics often have to reside in the objective function to be bounded. We combine mission fuel W_{f_m} and mission time t_m into a composite objective function through a cost index C for boundedness,

$$\text{Objective} \geq W_f \frac{1}{\mathrm{N}} + \mathrm{C} \times t_m \tag{3.7}$$

and divide W_{f_m} by newtons to achieve uniform units (non-dimensional). Another method to achieve proper boundedness is to add an arbitrary large upper bound. However this will result in the bounding constraint being tight and giving unphysical results, and so this thesis will implement the objective in Equation 3.7 instead. Cost index C is defined as a separate parameter so that we can observe the sensitivity of the variable post-optimization.

3.3 Design exploration through mission design

There are a few interesting methods that we can use to explore potential designs using GPs. So instead of showing the optimum of the SimPleAC for a single mission, leveraging the speed of convex optimization, we can map out the entire design space with respect to mission parameters. Please refer to [9] for details on the computational advantages of the GP and SP compared to other non-linear optimization methods.

In this case, the SimPleAC has been optimized (for $N_{segments} = 4$) over a range of payload weight (1000-10000 N) and range (1000-5000km), and the mission fuel weight and total weight have been plotted in contour plots in Figure 3-5. Note that every point in the design space represents a fully optimized aircraft. The full list of inputs

Constants	Value	Units	Description
$Range$	[1000-10000]	km	aircraft range
W_p	[1000-5000]	N	payload weight
C	120	$\frac{1}{\text{hr}}$	hourly cost index
$T/Ofactor$	2		takeoff thrust factor
V_{min}	25	$\frac{\text{m}}{\text{s}}$	takeoff speed
h_{cruise}	5000	m	minimum cruise altitude

Table 3.3: Inputs to the design space exploration of the ***Mission*** model.

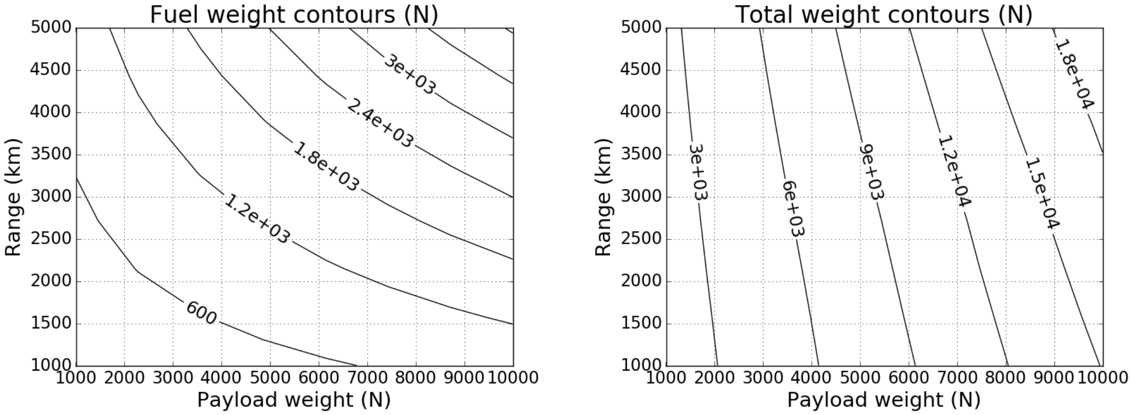


Figure 3-5: The fuel and total weight contours with respect to range and payload.

to the ***Mission*** model are detailed in Table 3.3

In Section 2.4.1 we had to weigh whether or not it was worth losing the mathematical guarantees of convexity to be able to model fuel storage. Now we can use our SP model to understand the tradeoffs in fuel storage, and when it is beneficial to store fuel in the wing versus the fuselage.

Figure 3-6 shows how designs for different range and payload requirements allocate fuel differently within the aircraft. As the mission range increases for a given payload weight (upward movement on the graph), more and more fuel is allocated within the fuselage as a proportion of total fuel. Since fuselage fuel volume is directly related to increased fuselage drag, it is logical that no fuel is put in the fuselage until the fuel volume constraint in the wing becomes tight. And this is the behavior that is observed, since no fuel is allocated in the fuselage towards the lower right of the graph.

For composite objective functions, it can be difficult to have intuitions about how sensitivities to parameters can affect the design, since the parameters act on multiple

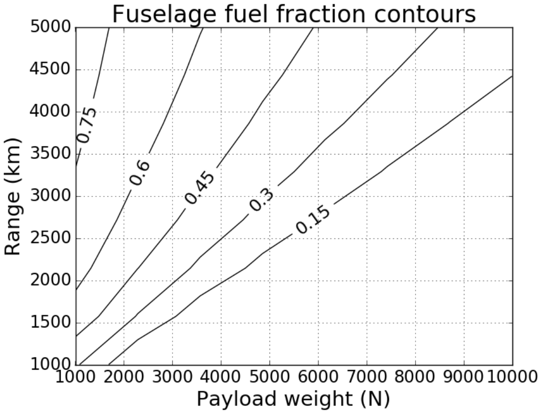


Figure 3-6: Fraction of total fuel stored in fuselage with respect to range and payload.

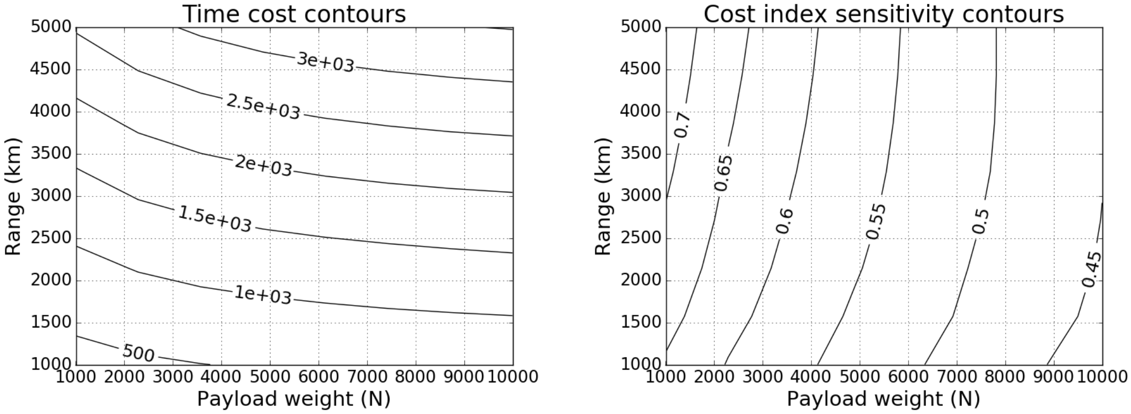


Figure 3-7: Time cost and time cost index sensitivity contours. We can gain intuition about the relative importance of different components of composite objective functions by showing both the costs and their sensitivities together.

variables of interest. A way to attempt to decouple these is to plot both the cost of a variable in the objective and its relevant sensitivity next to each other, as shown in Figure 3-7. Taking a look at point [4000N,3500km], we can follow the time cost (2×10^3) contour to see all of the missions with the same time cost as this mission (same average flight speed), and see how the fraction of time cost versus total cost varies through the sensitivity to the time cost index. This can help a designer gain intuition about the relative importance of different components of cost.

Variable/Model	Sensitivity	Variable description
Mission		
$Range$	+1.1	aircraft range
V_{min}	-0.67	takeoff speed
C	+0.46	hourly cost index
W_p	+0.45	payload weight
h_{cruise}	+0.12	minimum cruise altitude
***Mission*/SimPleAC**		
g	+0.54	gravitational acceleration
ρ_f	-0.042	density of fuel
***Mission*/SimPleAC/Engine**		
η_{prop}	-0.65	propeller efficiency
$P_{shaft,ref}$	-0.067	reference MSL maximum shaft power
$W_{e,ref}$	+0.044	reference engine weight
***Mission*/SimPleAC/Wing**		
$(\frac{S}{S_{wet}})$	+0.45	wetted area ratio
k	+0.45	form factor
$C_{L,max}$	-0.33	lift coefficient at stall
e	-0.17	Oswald efficiency factor
$W_{w_{coeff2}}$	+0.12	wing weight coefficient 2
τ	-0.11	airfoil thickness to chord ratio
N_{ult}	+0.07	ultimate load factor
$W_{w_{coeff1}}$	+0.07	wing weight coefficient 1
Mission*/*SimPleACP*/*EngineP		
$BSFC$	[+0.16 +0.14 +0.14 +0.14]	brake specific fuel consumption

Table 3.4: A selection of sensitivities to design parameters.

3.4 More modeling improvements before multimission design

There are still significant weaknesses in the model relating to the engine of the model that require improvement before we can perform multimission design in Section 3.5. We can see this by observing the sensitivities in Table 3.4.

As we can see, variables internal to the engine model, such as BSFC and η_{prop} have large sensitivities (0.59 [cumulative] and -0.65 respectively). The objective of the model is as sensitive to these variables as mission input variables such as range and payload, so these variables require refinement.

The following sections will take a two-pronged approach to improved engine modeling. The first weakness of the current model is the fact that the engine can supply the same amount of power regardless of altitude. The maximum power of naturally aspirated piston engines drops with altitude; adding a lapse rate will improve how much we trust the engine model. The second weakness is the lack of an engine BSFC model. Empirical data shows that the BSFC of an engine deteriorates at low power outputs. In Sections 3.4.1 and 3.4.2, more data-based modeling will be used to capture BSFC behavior at all throttle ranges.

3.4.1 Environmental effects: engine lapse rate model

Data-based modeling techniques have been detailed in Section 3.1.2. In the same way posynomial fits were created for the relationship between engine maximum power and weight, the lapse rate of the engine was fitted with respect to throttle level. This model requires the insertion of another signomial equality into the model. Consider the posynomial inequality expression below:

$$1 \geq L + \frac{P_{shaft,alt}}{P_{shaft,max}} \tag{3.8}$$

Since the BSFC of a normally aspirated piston engine would be expected to improve as the $P_{shaft} \rightarrow P_{shaft,alt}$, the maximum shaft power at altitude ($P_{shaft,alt}$) has downward pressure on it from the fuel burn objective. This means that the inequality doesn't adequately lower-bound $P_{shaft,alt}$. If we try to flip the inequality in Constraint 3.8, then $P_{shaft,alt}$ is upper-unbounded, so with our current parametrization of the shaft power, we must use a signomial equality.

3.4.2 Making use of sensitivities: engine BSFC model

The BSFC is one of the variables that the model is most sensitive to (total sensitivity over all mission segments of 0.59), and it has yet to be modeled. As stated in Section 3.4.1, the BSFC of a naturally-aspirated piston engine improves as the engine puts out more power relative to its maximum power at a given altitude.

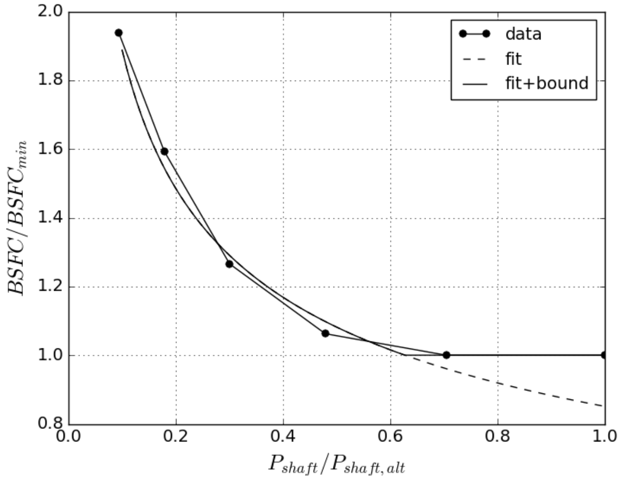


Figure 3-8: $\frac{\text{BSFC}}{\text{BSFC}_{min}}$ versus $\frac{P_{shaft}}{P_{shaft,alt}}$ data fit. The dashed monomial fit is not able to capture the tail ends of the curve, which is resolved by the bounded fit.

BSFC has downward pressure due to the objective function, and therefore must be lower-bounded. This makes it amenable to posynomial fits. A monomial fit of the $\frac{\text{BSFC}}{\text{BSFC}_{min}}$ versus $\frac{P_{shaft}}{P_{shaft,alt}}$ was created. The result is shown by the dashed line in Figure 3-8.

Although the RMS error of the fit is low (0.028) due to the small number of data points, there is a significant deterioration in the quality of the fit as $\frac{P_{shaft}}{P_{shaft,alt}} \to 1$, and increasing the number of posynomial terms in the fit does not alleviate this problem. What we can do is put a lower bound on the BSFC of the engine that depends on the actual lowest BSFC achieved by the engine. The final BSFC relation is shown in Equation 3.9.

$$\left(\frac{\text{BSFC}}{\text{BSFC}_{min}}\right)^{0.1} = .984\left(\frac{P_{shaft}}{P_{shaft,alt}}\right)^{-0.0346}, \quad \left(\frac{\text{BSFC}}{\text{BSFC}_{min}}\right) \geq 1 \tag{3.9}$$

3.5 Multimission design

Having created a mission profile for the SimPleAC, it only takes one extra level of hierarchy to do multimission design. Now we vectorize the missions flown by the same aircraft.

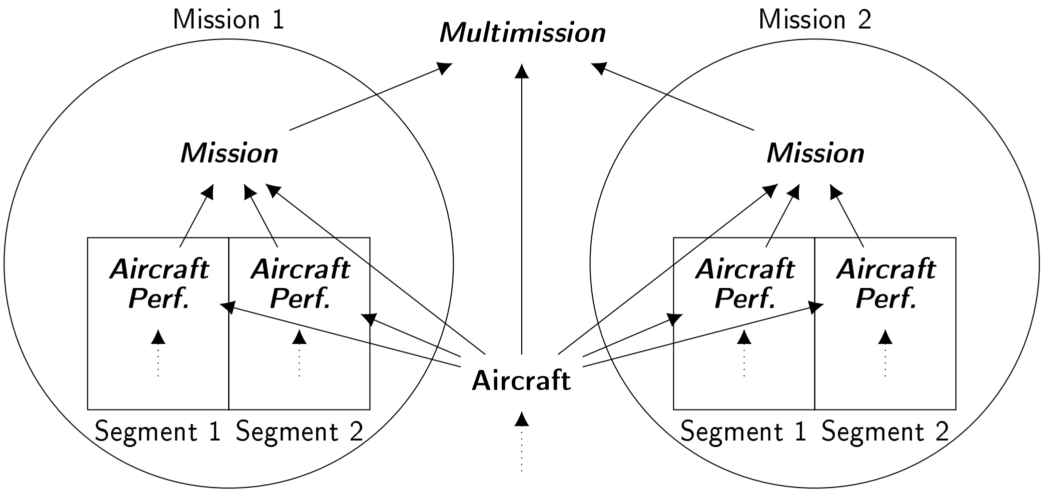

Mission 1
Multimission
Mission 2
Mission
Mission
Aircraft Perf.
Aircraft Perf.
Aircraft Perf.
Aircraft Perf.
Aircraft
Segment 1
Segment 2
Segment 1
Segment 2

Figure 3-9: Multimission uni-directional graph, with $N_{segments} = 2$ and $N_{missions} = 2$.

The vectorization has gone down two levels, where each performance model is vectorized $N_{segments} \times N_{missions}$ times, and each mission model is vectorized $N_{missions}$ times. The single instance of **Aircraft** is passed on as an argument to all of these models, which ensures that both its static and performance constraints are satisfied for each flight segment of every mission.

3.5.1 Multimission objective functions

The objective function of the ***Multimission*** model has to be a posynomial that puts pressure on the variables in every ***Mission*** model. For example, knowing that $W_{f_m}\frac{1}{\mathrm{N}} + C \times t_m$ is a valid objective function to the single mission, the objective function below for $N_{missions}$

$$\text{Objective} \geq \sum_{n=1}^{N_{missions}} W_{f_m}\frac{1}{\mathrm{N}} + \mathrm{C} \times t_m \tag{3.10}$$

will definitely work, since the same aircraft will fulfill two missions that are thus linked. Assuming that $N_{missions} = 2$, note that an objective of the type below will also work

$$\text{Objective} \geq W_{f_{m_1}}\frac{1}{\mathrm{N}} + \mathrm{C}_2 \times t_{m_2} \tag{3.11}$$

but will require some management in variable boundedness since some internal variables, namely $W_{f_{m_2}}$ and t_{m_1}, can and will diverge to numerical infinity since they are not necessarily pressured by the objective function. Simply upper-bounding the variables internally by some large bound will resolve boundedness issues. For this kind of objective, the performance of the aircraft will be a compromise with respect to aircraft optimized purely for mission time for the first mission or fuel burn for the second mission.

3.5.2 Multimission optimization results

The SimPleAC ($N_{segments} = 4$) was optimized for the 3 different mission scenarios:

1. M1: A single mission minimizing fuel burn (W_{f_m}).
2. M2: A single mission minimizing time cost ($\mathrm{C} \times t_m$)
3. MM: A multimission design performing both missions, minimizing fuel burn over M1 ($W_{f_{m_1}}$), and mission time over M2 ($\mathrm{C}_2 \times t_{m_2}$) (objective in Equation 3.11).

The different mission requirements are detailed in Table 3.5. M1 is the same mission as the one specified in Section 3.2, simulating a standard ferry mission where we care about fuel consumption. M2 is a dash mission delivering more payload than M1, where mission time is the objective. Both missions share the same flight altitudes, takeoff thrust factors and minimum takeoff speeds.

Input	M1	M2	Units	Description
C	120	360	$\frac{1}{\mathrm{hr}}$	hourly cost index
$Range$	3000	1000	km	aircraft range
W_p	6250	8000	N	payload weight
h_{cruise}	5000		m	minimum cruise altitude
$T/Ofactor$	2			takeoff thrust factor
V_{min}	25		$\frac{\mathrm{m}}{\mathrm{s}}$	takeoff speed

Table 3.5: Inputs to the M1 and M2 ***Mission*** models.

The results are shown in Table 3.6. The multimission aircraft represents a compromise aircraft that performs both missions with relatively high performance.

Table 3.6: Results of the single- and multi-mission solutions. Italicized values have been post-processed.

Variables	M1	M2	MM1	MM2	Units	Description
W	10453	45199	13843		N	maximum takeoff weight
AR	18.6	1.66	7.53		–	aspect ratio
S	27.8	120.2	36.8		m^2	total wing area
W_e	72	22730	1425		N	engine weight
W_w	3553	7457	2935		N	wing weight
$W_{w_{strc}}$	1885	245	725		N	wing structural weight
$W_{w_{surf}}$	1668	7212	2209		N	wing skin weight
W_{f_m}	577	7012	1252	1484	N	total mission fuel
t_m	*23.3*	4.58	*16.4*	6.30	hr	total mission time
BSFC(avg)	0.320	0.327	0.419	0.326	$\frac{\mathrm{lb}}{\mathrm{hp/hr}}$	average BSFC
V(avg)	35.8	122.3	50.8	88.7	m/s	average flight velocity

Note that for the two aircraft designed for single missions, the other mission is infeasible. For the M1 aircraft, M2 is infeasible due to the wing structural limits. For the M2 aircraft, M1 is infeasible due to inadequate fuel volume. When performing this kind of multimission design, we ensure that the aircraft design is feasible for each mission, meaning it is able to perform all of the missions under the sets of static and performance constraints.

Furthermore, there is evidence to suggest that the solve time of a SP scales sublinearly with the number of variables and constraints. So it might be interesting for a designer to input the entire variety of missions that they expect the design to perform into a single multimission framework to ensure feasibility, especially if the mission parameters vary significantly.

3.5.3 Potential extensions of multimission design

Multimission design and optimization increases the robustness of a design to mission variability. It would be further strengthened by the application of robust optimization to the framework. It can be used in the design of modular systems, and systems with potential present and future extensibility as well. A good example from aircraft design is the design of a family of aircraft, and more specifically 'stretchable' aircraft. In this case, a series of aircraft could be designed that have aerodynamic surfaces and engines in common, but have different length fuselages to accommodate different numbers of passengers over different routes of interest.

Chapter 4

Conclusion

This thesis has demonstrated methods for engineering design and optimization using geometric programming. It is motivated by the need to reduce the activation energy required to use optimization for conceptual design. The GP and SP are the mathematical frameworks that enable the intuitive formulation of general physical relations into optimization-compatible constraints.

In Chapter 2, the ideas of pressure and boundedness were explored in the context of a GP. The basic aircraft design problem was defined, and the importance of objective functions and variable freedom was identified in achieving bounded convex problems. Section 2.4 provided justification for the need to use the non-convex extension of GPs called SPs in design, with some loss of mathematical guarantees and solution speed. Fuel volume models provided a meaningful use case for signomial constraints, and the chapter provided arguments for the use of the signomial equality. The result was a fully bounded SP-compatible aircraft model that captures fundamental tradeoffs in conceptual aircraft design.

Chapter 3 introduced the idea of modularity, and improved the fidelity of the aircraft model by using data-based modeling in Section 3.1. The model was subsequently modularized into component models, and then into performance and static models in order to be able to vectorize certain sets of variables and constraints that are repeated between different mission segments. A logical (but non-unique) hierarchy of models was presented to allow designers to make decisions about where to locate different variables and constraints in the modular mission architecture.

Chapter 3 also presented a case where environmental modeling becomes critical for designs we trust as engineers. While performing tradespace exploration, an intuitive way to analyze the components of a multi-objective GP was presented, in an effort to draw maximal engineering understanding from the tradespaces that were explored.

Before the model was integrated into a multimission framework in Section 3.5, the SimPleAC engine model was improved upon in response to the sensitivity information. Potential new objective functions were identified and implemented in the context of multimission design and optimization. A sample solution of a 2-mission multimission aircraft design was presented alongside two aircraft optimized for each mission alone to further motivate the use of multimission MDO.

This thesis has demonstrated how to model a complex engineering system starting from scratch in GP and SP using component and function based abstractions. Using tools developed in the CEG, it has outlined methods that allow designers to extract engineering understanding of the trade spaces using convex and difference-of-convex methods. Although an aircraft design problem was the focus of this work, these methods are general to engineering design problems with explicit, continuous constraints, and lower the barriers to using optimization in engineering design.

Appendix A

Mathematical Framework

A.1 Geometric Programming[1]

Introduced in 1967 by Duffin et al. [5], a geometric program GP is a type of constrained optimization problem that becomes convex after a logarithmic change of variables. Modern interior point methods allow a typical sparse GP with tens of thousands of decision variables and tens of thousands of constraints to be solved in minutes on a desktop computer [2]. These solvers do not require an initial guess, and guarantee convergence to a *global* optimum, assuming a feasible solution exists. If a feasible solution does not exist, the solver will return a certificate of infeasibility. These impressive properties arise because a GP's objective and constraints consist of only monomial and posynomial functions, which can be transformed into convex functions in log space.

A monomial is a function of the form

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

where $a_j \in \mathbb{R}, c \in \mathbb{R}_{++}$ and $u_j \in \mathbb{R}_{++}$. An example of a monomial is the common expression for lift, $\frac{1}{2}\rho V^2 C_L S$. In this case, $\mathbf{u} = (\rho, V, C_L, S)$, $c = 1/2$, and $a = (1, 2, 1, 1)$.

---

[1] This section has been borrowed from the paper titled *Efficient Aircraft Multidisciplinary Design Optimization and Sensitivity Analysis via Signomial Programming*, by Martin York, Berk Ozturk, Edward Burnell and Warren Hoburg. The paper is undergoing review for publication in the AIAA Journal as of January 24th, 2018.

A posynomial is a function of the form

$$p(\mathbf{u}) = \sum_{k=1}^{K} c_k \prod_{j=1}^{n} u_j^{a_{jk}} \tag{A.2}$$

where $a_{jk} \in \mathbb{R}, c_k \in \mathbb{R}_{++}$ and $u_j \in \mathbb{R}_{++}$. A posynomial is a sum of monomials. Therefore, all monomials are also one-term posynomials.

A GP minimizes a posynomial objective function subject to monomial equality and posynomial inequality constraints. A GP written in standard form is

$$\begin{aligned} &\text{minimize } p_0(\mathbf{u}) \\ &\text{subject to } p_i(\mathbf{u}) \leq 1, i = 1, ..., n_p, \\ &m_i(\mathbf{u}) = 1, i = 1, ..., n_m \end{aligned} \tag{A.3}$$

where p_i are posynomial functions, m_i are monomial functions, and $\mathbf{u} \in \mathbb{R}^n_{++}$ are the decision variables. Once a problem has been formulated in the standard form (Equation A.3), it can be solved efficiently.

A.2 Signomial Programming[2]

It is not always possible to formulate a design problem as a GP. This motivates the introduction of signomials. Signomials have the same form as posynomials

$$s(\mathbf{u}) = \sum_{k=1}^{K} c_k \prod_{j=1}^{n} u_j^{a_{jk}} \tag{A.4}$$

but the coefficients, $c_k \in \mathbb{R}$, can now be any (including non-positive) real numbers.

A signomial program (SP) is a generalization of GP where the inequality constraints can be composed of signomial constraints of the form $s(u) \leq 0$.. The log transform of an SP is not a convex optimization problem, but is a difference of convex optimization problem that can be written in log-space as

---

[2] This section has been borrowed from the paper titled *Efficient Aircraft Multidisciplinary Design Optimization and Sensitivity Analysis via Signomial Programming*, by Martin York, Berk Ozturk, Edward Burnell and Warren Hoburg. The paper is undergoing review for publication in the AIAA Journal as of January 24th, 2018.

$$\begin{aligned} &\text{minimize } f_0(\mathbf{x}) \\ &\text{subject to } f_i(\mathbf{x}) - g_i(\mathbf{x}) \leq 0, i = 1, ..., m \end{aligned} \tag{A.5}$$

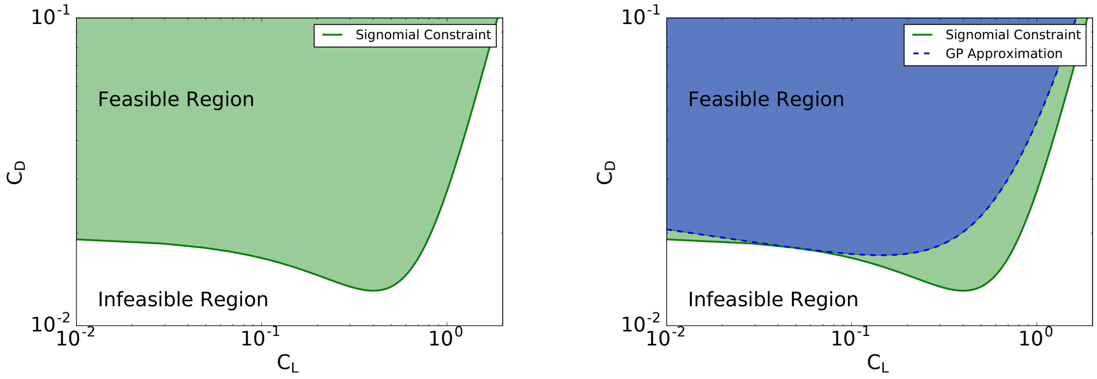
where f_i and g_i are convex.

There are multiple algorithms that reliably solve signomial programs to *local* optima [1, 12]. A common solution heuristic, referred to as difference of convex programming or the convex-concave procedure, involves solving a sequence of GPs, where each GP is a local approximation to the SP, until convergence occurs. It is worth noting that the introduction of even a single signomial constraint to any GP turns the GP into a SP, thus losing the guarantee of solution convergence to a global optimum. Despite the possibility of convergence to a local, not global, optimum, SPs are a powerful tool. The convex approximation, $\hat{f}(x)$, to the non-convex signomial in log-space, $f(x) - g(x)$, is constructed such that it always satisfies

$$\hat{f}(x) \geq f(x) - g(x) \quad \forall \quad x \tag{A.6}$$

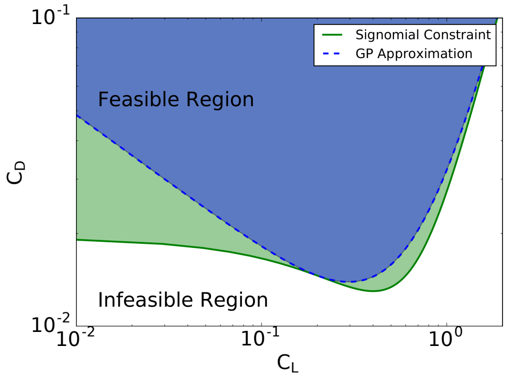
In other words, for each constraint, the feasible set of the convex approximation $\hat{f}(x) \leq 0$ is a subset of the original SP's feasible set, $f(x) - g(x) \leq 0$. This means SP inequalities do not require a trust region, removing the need for trust region parameter tuning and making solving SPs substantially more reliable than solving general nonlinear programs. Figure A-1, where a series of convex (GP compatible) constraints approximates a non-convex parabolic drag polar in log space, illustrates this property.

Signomial equality constraints can be approximated by monomials as shown in Figure A-2 and may require a trust region. Trust regions were not used in the presented model. Signomial equalities are the least desirable type of constraint due the approximations involved. Most constraints in this work were relaxed to inequali-



(a) Non-convex signomial inequality drag constraint

(b) Convex approximation about $C_L = 0.05$.



(c) Convex approximation about $C_L = 0.20$.

Figure A-1: A signomial inequality constraint and GP approximations about two different points.

ties and checked for tightness by GPkit [3]. For additional details on how signomial equalities are approximated, see Opgenoord et al. [15].

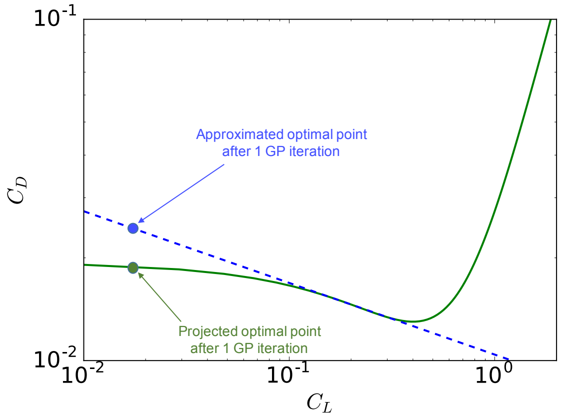


Figure A-2: The signomial equality constraint $C_D = f(C_L)$ and its approximation.

Appendix B

Model Resources

B.1 Flight segment model variables

Variable	Units	Description
$\frac{dh}{dt}$	$\frac{\text{m}}{\text{hr}}$	climb rate
h	m	segment flight altitude
h_{avg}	m	segment average flight altitude
R_s	km	segment range
W_{start}	N	segment beginning weight
W_{end}	N	segment end weight
W_{avg}	N	segment average weight
W_{f_s}	N	segment fuel burn
W_{f_m}	N	total mission fuel
t_s	hr	segment time
t_m	hr	total mission time

Table B.1: Variables introduced in the flight segment model.

Bibliography

[1] Stephen Boyd, Seung-Jean Kim, Lieven Vanderberghe, and Arash Hassibi. A tutorial on geometric programming. *Optim. Eng.*, 2007.

[2] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization.* Cambridge University Press, 7 edition, 2009.

[3] Edward Burnell and Warren Hoburg. Gpkit software for geometric programming. http://github.com/convexengineering/gpkit, 2017. Version 0.6.0.

[4] Michael Burton and Warren Hoburg. Solar-electric and gas powered, long-endurance uav sizing via geometric programming. *AIAA Journal*, 2017.

[5] R.J Duffin, E.L. Peterson, and C. Zener. *Geometric programming: theory and application.* Wiley New York, 1967.

[6] Warren Hoburg. *Aircraft Design Optimization as a Geometric Program.* PhD thesis, University of California, Berkeley, 2013.

[7] Warren Hoburg and Pieter Abbeel. Geometric programming for aircraft design optimization. *AIAA Journal*, 2014.

[8] Warren Hoburg, Philippe Kirschen, and Pieter Abbeel. Data fitting with geometric-programming-compatible softmax functions. *Optim. Eng.*, 2016.

[9] Philippe Kirschen and Warren Hoburg. The power of log transformation: A comparison of geometric and signomial programming with general nonlinear programming techniques for aircraft design optimization. *AIAA SciTech*, 2018.

[10] Philippe G. Kirschen. Signomial Programming for Aircraft Design. Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, 2016.

[11] Philippe G. Kirschen, Edward E. Burnell, and Warren W. Hoburg. Signomial programming models for aircraft design. *AIAA SciTech*, 2016.

[12] Thomas Lipp and Stephen Boyd. Variations and extension of the convex concave procedure. *Optim. Eng.*, 2015.

[13] Joaquim R.R. Martins and Andrew B. Lambe. Multidisciplinary design optimization: A survey of architectures. *AIAA Journal*, 2013.

[14] Yurii Nesterov and Arkadi Nemirovski. *Interior-Point Polynomial Algorithms in Convex Programming.* Society for Industrial and Applied Mathematics, 1994.

[15] Max M.J. Opgenoord, Brian S. Cohen, and Warren W. Hoburg. Equality constraints for signomial programming. *Optim. Eng.*, 2016.

[16] Daniel P. Raymer. *Aircraft Design: A Conceptual Approach.* American Institute of Aeronautics and Astronautics, Reston, Virginia, 4th edition, 2006.

[17] Tony Tao. *Design, Optimization, and Performance of an Adaptable Aircraft Manufacturing Architecture.* PhD thesis, Massachusetts Institute of Technology, 2018, expected.

[18] Martin York, Warren Hoburg, and Mark Drela. Turbofan engine sizing and tradeoff analysis via signomial programming. *AIAA Journal of Aircraft*, 2017.