TECHNIQUE FOR THE OPTIMIZATION OF THE POWERHEAD CONFIGURATION AND PERFORMANCE OF LIQUID ROCKET ENGINES

A Dissertation
Presented to
The Academic Faculty

By

Brad David St. Germain

In Partial Fulfillment
Of the Requirements for the Degree of
Doctor of Philosophy in Aerospace Engineering

Georgia Institute of Technology
July 2003

TECHNIQUE FOR THE OPTIMIZATION OF THE POWERHEAD CONFIGURATION AND PERFORMANCE OF LIQUID ROCKET ENGINES

Approved:

John R. Olds

John R. Olds, Chairman

Timothy C. Lieuwen

Jerry M. Seitzman

Shabbir Ahmed

John A. Blevins

John A. Blevins

Date Approved 7-13-2003

To Lisa

ACKNOWLEDGEMENTS

I would like to thank the many people who have helped me throughout this research. First, I would like to thank my graduate advisor Dr. John Olds. Without your guidance and wisdom this research would have not been possible. I would also like to thank the other members of my committee, Dr. Timothy Lieuwen, Dr. Jerry Seitzman, Dr. Shabbir Ahmed, and Dr. John Blevins. Your advice and insight has proved invaluable.

Thank you also to my friends and coworkers in the Space Systems Design Lab. Many of you have come and gone over the years and each of you has made my time at Tech more enjoyable. Thanks to the old guard for paving the way and to the new guys for keeping the lab fun. A special thanks to Tim for putting up with my endless ramblings about this research over the last few months, can you believe I'm actually done!

I would like to thank my family, Mom, Dad, Nancy and Gabrielle. Thank you for always being there and for supporting me in all my endeavors. Mom, the endless amounts of love and guidance you have given me are immeasurable. Dad, all of your wisdom, advice, and love have helped in ways you can't even imagine.

Finally, I must thank my loving wife Lisa. Thank you for being my biggest cheerleader and best friend. Having you in my life to share my dreams and experiences makes them all the more special.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

C2H2	acetylene
C2H6	ethane
C3H8	propane
CH4	methane
CO	carbon monoxide
CO_2	carbon dioxide
EX	expander cycle
ft^2	square feet
GG	gas generator cycle
H	monatomic hydrogen
H_2	hydrogen
H_2O	water
HP	horsepower
hrs	hours
klbs	thousand pounds
lbm/sec	pounds mass/second
Mlbs	million pounds
O	monatomic oxygen
O_2	oxygen
O/F	oxidizer-to-fuel ratio
OH	hydroxyl radical
psia	pounds per square inch (absolute)
R	Rankine
SC	staged-combustion cycle
sec	second

LIST OF ACRONYMS

B&B	Branch and Bound
CA	Contributing Analysis
CCD	Central Composite Design
CFD	Computational Fluid Dynamics
DOE	Design of Experiments
DSM	Design Structure Matrix
EELV	Evolved Expandable Launch Vehicle
GBD	Generalized Bender’s Decomposition
GOA	Generalized Outer Approximation
HPFP	High-Pressure Fuel Pump
JANAF	Joint Army/Navy/Air Force
JP-4	Jet Propellant 4
LB	Lower Bound
LH2	Liquid Hydrogen
LOX	Liquid Oxygen
LPFP	Low-Pressure Fuel Pump
LPFT	Low-Pressure Fuel Turbine
MINLP	Mixed-Integer Nonlinear Programming
MoFD	Method of Feasible Directions
NAFCOM99	NASA/Air Force Cost Model 99
NASA	National Aeronautics and Space Administration
NASP	National Aerospace Plane
NGLT	Next Generation Launch Technology
NIST	National Institute of Standards and Technology
NLP	Nonlinear Programming Problem

NPSH	Net Positive Suction Head
OA	Outer Approximation
RP-1	Rocket Propellant 1
RSE	Response Surface Equation
SLP	Sequential Linear Programming
SSE	Sum of Squares of the Error
SSM	Sum of Squares of the Mean
SSME	Space Shuttle Main Engine
SSTO	Single-Staged to Orbit
SQP	Sequential Quadratic Programming
TFU	Theoretical First Unit
UB	Upper Bound
US	United States

LIST OF SYMBOLS

a	underestimator coefficient
A_e	exit area
b	RSE coefficient
C_o	isentropic spouting velocity
d	lower bounding shift
Δ	diagonal shift matrix
∇	gradient
e	expansion ratio
$F(\bar{x})$	objective function (only continuous variables)
$f(\bar{x})$	true continuous function
$\tilde{f}(\bar{x})$	response surface equation
$F(\bar{x}, \bar{y})$	objective function (discrete and continuous variables)
g	ratio of specific heats
g_o	gravity
$g(\bar{x})$	inequality constraint (only continuous variables)
$\tilde{g}(\bar{x})$	response surface equation of inequality constraint
$g(\bar{x}, \bar{y})$	inequality constraint (discrete and continuous variables)
h	step size
H	Hessian
h	efficiency
h_{in}	inlet enthalpy
H_p	pump head rise

h_{out_s}	isentropic exit enthalpy
$h(\overline{x})$	equality constraint (only continuous variables)
$h(\overline{x},\overline{y})$	equality constraint (discrete and continuous variables)
I_{sp}	specific impulse
J	Jacobian
l	Lagrange multiplier
$L(\overline{x})$	lower bound of continuous function
$\dot{m}$	mass flow rate
n	number of continuous variables
N_r	pump rotational speed
N_s	stage specific speed
P	pressure
p_{trat}	turbine pressure ratio
r	density
$\dot{Q}$	heat transfer rate
T	temperature
T_{vac}	vacuum thrust
T_{sl}	sea-level thrust
T/W	vacuum thrust-to-weight
U_m	turbine pitchline velocity
x	continuous variable
x^*	optimum
x_l	continuous variable lower bound
x_u	continuous variable upper bound
y	discrete variable

SUMMARY

The development and optimization of liquid rocket engines is an integral part of space vehicle design, since most Earth-to-orbit launch vehicles to date have used liquid rockets as their main propulsion system. Rocket engine design tools range in fidelity from very simple conceptual level tools to full computational fluid dynamics (CFD) simulations. The level of fidelity of interest in this research is a design tool that determines engine thrust and specific impulse as well as models the powerhead of the engine. This is the highest level of fidelity applicable to a conceptual level design environment where faster running analyses are desired.

The optimization of liquid rocket engines using a powerhead analysis tool is a difficult problem, because it involves both continuous and discrete inputs as well as a nonlinear design space. Example continuous inputs are the main combustion chamber pressure, nozzle area ratio, engine mixture ratio, and desired thrust. Example discrete variable inputs are the engine cycle (staged-combustion, gas generator, etc.), fuel/oxidizer combination, and engine material choices.

Nonlinear optimization problems involving both continuous and discrete inputs are referred to as Mixed-Integer Nonlinear Programming (MINLP) problems. Many methods exist in literature for solving MINLP problems; however none are applicable for this research. All of the existing MINLP methods require the relaxation of the discrete variables as part of their analysis procedure. This means that the discrete choices must be evaluated at non-discrete values. This is not possible with an engine powerhead design code. Therefore, a new optimization method was developed that uses modified response surface equations to provide lower bounds of the continuous design space for each unique discrete variable combination. These lower bounds are then used to efficiently solve the optimization problem. The new optimization procedure was used to find optimal rocket engine designs subject to various weight, cost and performance constraints. The results show that the new method efficiently solved the mixed-input optimization problem without requiring discrete variable relaxation.

CHAPTER 1

MOTIVATION

1.1 Historical Background

Liquid rocket engines are a core technology for the exploration and development of space. These engines have played a major role in the United States' (US) space program since its inception. Liquid rocket engines provided the power used in the first sounding rockets, intercontinental ballistic missiles, and the early manned and interplanetary missions [1]. The use of liquid rocket engines is just as prevalent today and they remain necessary for the continuation of assured space access.

Liquid rocket engine design began at the turn of the 20th century with the pioneering works of Dr. Robert Goddard (Figure 1). Goddard, known as the "Father of Modern Rocketry", first began experimenting with liquid propellants for rocket engines in 1909. His first liquid fueled rocket was launched on March 16, 1926. The flight lasted a mere 2.5 seconds but the age of liquid-fueled rockets had begun. Goddard moved to the deserts of New Mexico to continue his work on liquid-fueled rockets. There he pioneered the use of stabilizing fins and gyroscopes to help control his rockets. Most of

Goddard's work went unnoticed in the US. However, others noticed, most notable Wernher von Braun. He took Goddard's plans from various literature sources and used those as a starting point for the V-2 rocket [2,3,4,5].

Figure 1: Dr. Robert Goddard

After WWII, the US rocketry industry began to evolve and grow. With the influx of technology and man-power from Germany, many new rocket vehicles and liquid rocket engines were developed. These new engines were used during different vehicle programs such as the Redstone, Atlas, Thor and later the Apollo program. The current state of the art liquid rocket engine, the Space Shuttle Main Engine (SSME), was developed for the Space Shuttle program. The SSME is currently the only reusable, booster-class, liquid rocket engine. Figure 2 shows three different liquid rocket engines that represent the progression of engine design over the years. The MB-1, developed in the 1950's for the Thor program, was a Liquid Oxygen (LOX)/ Kerosene engine capable of producing ~170 klbs of thrust. The J-2 was used on the second and third stages of the Saturn V moon rocket. This LOX/liquid hydrogen (LH2) engine, developed in the mid 1960’s, was capable of producing ~230 klbs of thrust. The final engine shown is the SSME. This 500 klbs class engine was developed in the late 1970’s and also uses LOX/hydrogen propellants [6].

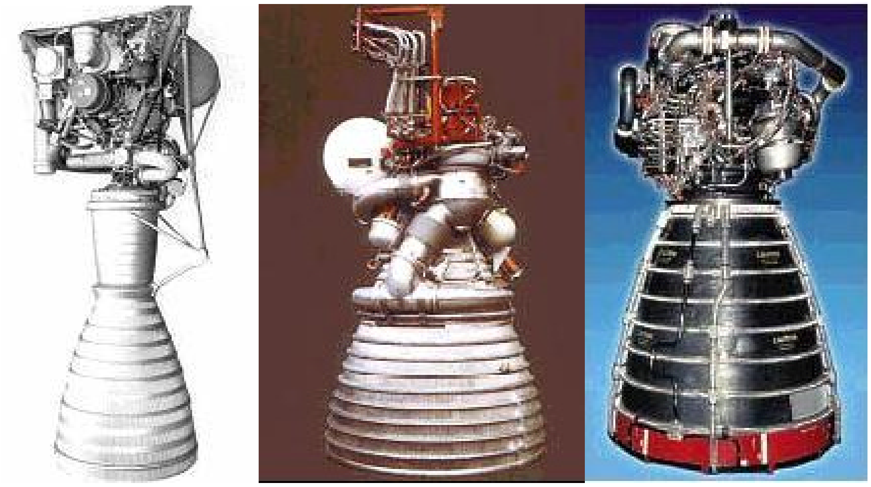


Figure 2: The MB-1, J-2 and SSME

Liquid rocket engines are categorized by their propellant combination and by their engine cycle. The engine cycle refers to the means by which the rocket propellant is moved from the vehicle’s tanks to the main combustion chamber of the engine. The fuel and oxidizer should be at very high pressures (1000 – 3500 psia) when injected into the main combustion chamber. This allows for more engine thrust and better engine performance. Two broad classes of engine cycles exist. The first and simplest is the pressure-fed cycle, where the propellants are stored at very high pressures in their respective tanks and the high pressure drives the propellants to the main combustion chamber. These types of systems tend to be limited to low impulse applications.

The second class of engine cycle is the pump-fed cycle. Here, the propellants are stored at relatively low pressures in their propellant tanks (~ 30 psia). This allows for much lighter tanks and therefore less overall vehicle structural mass. However, the fuel and oxidizer must now be pressurized in order to be injected at high pressures into the combustion chamber. This pressurization is accomplished by using a grouping of turbopumps, mixers, heat exchangers, pre-burners and feed lines called a powerhead. These components work together to pressurize and move the propellants into the combustion chamber. The turbopumps are powered by hot gases obtained by routing propellant through heat exchangers lining the main combustion chamber and nozzle or by using the hot gases resulting from combustion in a preburner. The source of the hot gases for the turbines is the main parameter that defines the different types of pump-fed engine cycles. Since this research deals with larger thrust class engines, only pump-fed liquid rocket engines will be further analyzed.

As space flight approaches 50 years of existence, vehicles are still relying on many of the fundamental technologies developed at the dawn of the space era. Hence, all operational space vehicles today still use rocket power as the primary propulsion means. The vast majority of these vehicles choose liquid propellants over solid propellants because of the associated increase in engine performance. Escaping Earth’s gravity and placing objects in orbit is a difficult challenge. The benefit of using rocket propulsion is that space vehicles can quickly accelerate out of the Earth’s sensible atmosphere and therefore reduce the velocity losses occurred while reaching orbit.

There have been many efforts to develop vehicles for space access that do not use rocket engines as their primary motive power. The most famous, and well funded, of these programs was the National Aerospace Plane (NASP). The NASP was to be powered by scramjets, a form of air-breathing combined cycle hypersonic propulsion. The scramjet engines proposed for NASP were designed to be much more fuel efficient than liquid rocket engines. After spending $1.73B over the life time of the project, the goal of a reusable single-staged to orbit (SSTO) air-breathing vehicle was never met [7]. Since the cancellation of NASP in the early 1990's the development and study of hypersonic space vehicles has been relegated to design studies [8,9,10,11]. Many of these studies show the promise of hypersonic propulsion, but it is highly unlikely that a functioning hypersonic space vehicle will be built within the next 20 years. To reinforce this fact, the next generation launch technology (NGLT) program, which is part of NASA's space launch initiative, has focused it's near term research on technologies for improving space access using liquid rocket engines. The NGLT program has called for the development of the United States' first reusable hydrocarbon fueled liquid rocket engine, the RS-84 (Figure 3) [12]. The RS-84 will represent the first oxidizer-rich staged combustion engine to be built in the US.

The majority of satellites and probes are launched from the Earth using expendable launch vehicles. These rockets, which are the descendants of the first intercontinental ballistic missiles, almost always use liquid rocket engines during their ascent. The desire of the expendable launch vehicle community is to have a fuel efficient, cheap and easy to manufacture engine since it will be discarded after every flight. The current generation of expendable launch vehicles are called the Evolved Expendable Launch Vehicles (EELV). The two EELVs currently being produced are Boeing's Delta IV and Lockheed Martin's Atlas V launch vehicles. While both of these rockets use variations of Pratt & Whitney's LOX/LH2 expander cycle motor, the RL-10, as their upper stage engine, the first stage engines vary greatly. The Atlas V uses a Russian built RD-180 LOX/hydrocarbon staged combustion engine, while the Delta IV uses a LOX/LH2 gas-generator engine, the RS-68.

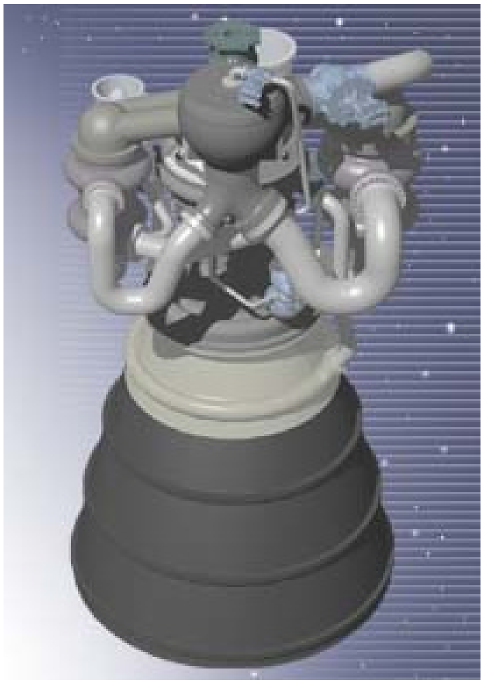


Figure 3: RS-84 Engine

There is a continuing need for the development of new and improved liquid rocket engines. Their use is just as prevalent today as it has ever been. However, different engine applications use different metrics by which to measure "improvement" in an engine design. For an engine being used on a man-rated system, reliability is the dominant factor. For expendable vehicles, cost might be the overwhelming factor. In reality, the design metric chosen will most likely be a combination of performance, weight, economic and safety requirements.

1.2 Design Environment

The design process for most systems, including liquid rocket engines can be divided into three design phases, conceptual, preliminary and detailed design [13]. Conceptual design is the most free form design phase where the design is initially being formed. Addressed during this time are the design options, requirements, and other top-level design choices. If the conceptual design phase produces a feasible product the preliminary design phase begins. Here, major design changes are frozen and more detailed analysis and work is done to limit the uncertainty of the design. After the preliminary design phase is complete the detailed design phase begins. Now all the individual components must be developed along with all the requirements for the tooling and fabrication process. The performance metrics are also finalized in the detailed design phase [13].

The application of this work will be in the conceptual design phase of rocket powered space vehicles. To design these vehicles, various individual disciplines are combined in a collaborative, multidisciplinary design process. These disciplines include aerodynamics, flight mechanics, structural and weight analysis, propulsion, cost, operations and reliability. This combination forms an integrated design team where each team member is responsible for a specific discipline. Team members execute an individual disciplinary analysis tool, and these disciplines are coupled in an iterative conceptual design process in which information about each candidate design is exchanged between the disciplines until the vehicle's design converges. The design process is most conveniently represented by the design structure matrix (DSM) shown in Figure 4 [14]. Design structure matrices are useful because they show the coupling between the various disciplines used in the design process. Each box in the DSM represents a specific discipline and is called a contributing analysis (CA). The feed forward links on the top of the CAs show where information must be fed downstream in the design process. The feed back links underneath the CAs show information that must be relayed back upstream in the design process. These feed back loops cause the design process to be iterative. Some CAs are more strongly coupled than others, with the strongest coupling occurring between the propulsion, trajectory, and weights and sizing disciplines. Typically 6-8 system level iterations are required to get a converged vehicle design.

Since the propulsion discipline is in the main iteration loop, it must be executed many times to get the final vehicle solution. This makes it desirable to not have to spend a lot of time executing the propulsion discipline so vehicle designs can be completed quickly. If one wishes to optimize the propulsion design inside the iteration loop, a problems exists. Typical propulsion design tools have both discrete and continuous inputs. Most deterministic optimization routines can only handle continuous variables. Therefore, this optimization routine would have to be run for each discrete input variable combination in order to determine the optimal engine design. The result is that the engine design code must be run many times to complete one analysis of the propulsion discipline. If forced to optimize the engine design in this way, the design time constraint will limit the tool choices for the propulsion discipline to very fast running propulsion codes. Typically, low fidelity level design tools or meta-models of higher fidelity tools would have to be used. Both of these choices increase the uncertainty of the propulsion results. The lower fidelity tools make simplifying assumptions in an effort to decrease computational time. Meta-models of higher fidelity tools have inherent inaccuracies because of the error associated with fitting the model to the given data.

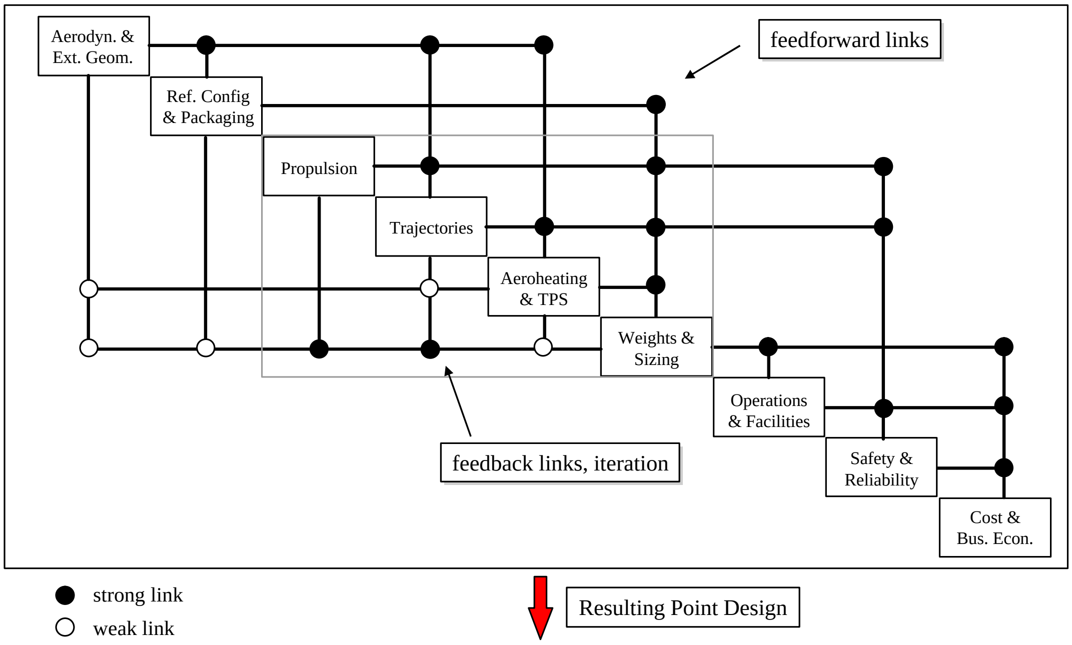


Figure 4: Design Structure Matrix (DSM)

If a higher fidelity code is used, there is typically little, if any optimization done on the engine design. The design would either be fixed early in the iteration process by a knowledgeable propulsion engineer, or only a limited search of the design space would occur. The latter would be the case if the discrete variables, such as engine cycle and the fuel/oxidizer combination are fixed, and then an optimization process is applied to the continuous design space.

The ideal case would to be able to use a higher fidelity code for the propulsion analysis and still search the entire design space (including the discrete space) without greatly increasing the design time. In order to do this, a new, more efficient optimization approach is needed. This approach needs to be able to handle both discrete and continuous variables and be able to find the optimal solution in an appropriate amount of time.

1.3 Optimization Problem

Many different types of rocket engine design codes currently exist [15 - 20]. These range in fidelity and scope from simple, system level tools to high fidelity CFD analyses. This research is focused on applying optimization techniques to a liquid rocket engine design tool that models the powerhead of the engine as well as determines the engine's performance. Given the current level of available computing power, this type of engine analysis tool probably represents the highest level of fidelity suitable for a conceptual level design environment. The optimization of liquid rocket engine designs using a powerhead analysis tool poses an interesting problem, because codes of this type have both discrete and continuous design inputs. Traditionally, optimization schemes use gradient information to guide the search of the design space [21]. This gradient information is not available for discrete variables, so a non-traditional optimization approach must be used.

The discrete inputs into the engine code include different material choices, the fuel/oxidizer combination, the basic engine cycle, and variations of these cycles (i.e. whether to use boost pumps, the turbine configuration…). The continuous inputs common for every cycle include desired chamber pressure, nozzle expansion ratio, propellant mixture ratio (oxidizer to fuel ratio by mass), and desired vacuum thrust.

Nonlinear optimization problems involving both discrete and continuous inputs are called Mixed-Integer Nonlinear Programming (MINLP) problems. The optimization of a rocket engine design using a powerhead analysis tool is an MINLP problem. Many methods exist for solving MINLP problems. These include Generalized Bender's Decomposition (GBD), Outer Approximation (OA) and its variants, and methods based on a Branch and Bound (B&B) framework [22]. Unfortunately, none of the existing methods will work for the problem presented here. The reason for this is the inability to easily relax the discrete variables (evaluate at non-discrete settings).

A new optimization scheme was developed to solve the problem of interest. The new method finds the lower bound of the continuous design space for each discrete variable combination. The solution of the lower bounding space will provide a solution that is equal to or better than the solution of the true space. The discrete variable combination with the most promising lower bound is solved for its optimum in the continuous space using a traditional local nonlinear optimization method. This solution of the true space is then used to determine what, if any, discrete variable combination needs to be solved next in order to find the optimal solution. The lower bounds of the continuous space for each discrete variable combination are found by fitting response surface equations (RSE) to the gathered data and then modifying these equations using underestimating techniques pioneered in global optimization. The underestimating techniques turn the RSE into a lower bound.

This new method has been applied to two different engine design codes. The first proof-of-concept application was to a quick running rocket engine performance code on a test problem involving two discrete variables (13 discrete variable combinations) and three continuous variables. This proof-of-concept showed that the proposed method solves the rocket optimization problem without requiring the relaxation of the discrete variables. The final application was to a more detailed and hence slower running rocket engine powerhead design code. This optimization problem involved three discrete variables (110 discrete variable combinations) and four continuous variables. Again, the method was able to efficiently find the optimal solution and did so within a time frame appropriate for a conceptual design environment.

1.4 Goals and Objectives

The objective of this research is to develop an efficient optimization method that is able to optimize designs that have both discrete and continuous inputs, without requiring the relaxation of the discrete space. This optimization method will be applied to a liquid rocket engine design code that not only determines the performance characteristics of the engine, but also analyzes the engine’s powerhead. A code of this fidelity will allow a more complete analysis of the rocket engine because details about specific engine components will be available. An existing liquid rocket engine powerhead analysis code, PHATCAT, is available to the author [16]. For this research, some current inadequacies in PHATCAT will be addressed. Also, inaccuracies in the code will be mediated and the code’s capability will be expanded to increase the number of different engine cycles it is able to analyze. Listed below are several specific goals of this research.

- Goal 1: Determine the applicability of existing optimization techniques.

A great deal of research has been conducted in the area of numerical optimization. A portion of this research is dedicated to optimization methods designed specifically for problems with mixed-inputs (both continuous and discrete). The applicability of these schemes to this research needs to be determined. Also, there are a plethora of nonlinear optimization schemes that are designed for continuous variable inputs. Their usefulness for this research also needs to be decided. Finally, the performance of a baseline optimization method needs to be established to give a benchmark for future improvements. The baseline method will use a nonlinear optimization routine to find the optimum of the continuous variable space for every discrete variable combination.

- Goal 2: Develop a new optimization method that is able to solve mixed-input problems where the discrete variables can not be relaxed.

The rocket engine powerhead design code that will be used for this research has both continuous and discrete inputs. This code is unable to run when the discrete input variable settings are not at one of the appropriate discrete values. Therefore, any new optimization method developed for this research must not require that these discrete inputs be set to a non-discrete setting.

- Goal 3: Use the new optimization procedure to obtain the same solution found by the baseline method in a time appropriate for a conceptual design environment.

The new optimization method is intended to allow the use of the higher fidelity powerhead analysis code in a conceptual design environment. To accomplish this, the new method must be able to determine the optimal engine design, using a standard desktop computer, in less than 2 hours of computational time during the design process.

CHAPTER 2

BACKGROUND

2.1 Optimization Background

2.1.1 Local Optimization Methods

The area of numerical optimization deals with the problem of finding the minimum or maximum of a particular objective function subject to various design constraints. Many different optimization techniques have been developed and they are applicable to various problem types. There are methods designed to handle functions with multiple extrema and those that only guarantee finding a local optimum. Some methods are able to handle design constraints directly, while others must be modified to solve constrained problems. Optimization problems are typically presented in the standard form outlined in Equation 1.

$$\min_{\bar{x}} F(\bar{x}) \tag{1a}$$

$$s.t.\ h(\bar{x}) = 0 \tag{1b}$$

$$g(\bar{x}) \leq 0 \tag{1c}$$

$$\bar{x}_l \leq \bar{x} \leq \bar{x}_u \tag{1d}$$

Standard form for an Optimization Problem

The function that is being minimized is called the objective function. There are three different ways to represent constraints on the objective using standard form. Constraints can either be represented as equality [$h(\bar{x})$], inequality constraints [$g(\bar{x})$], or side constraints [21].

The type of classical numerical optimization problem that is of interest in this research is the constrained optimization of a continuous function. This is one of the most common optimization problems and there are many methods that specialize in its solution. In order for these methods to work properly several conditions are imposed on the design space. First, as mentioned earlier, the design space must be continuous. This is because these numerical optimizers must be able to obtain information about the objective function and constraints at any place within the domain of the solution. Also, most of the common constrained optimization methods require that the design space be unimodal. If the design space is multi-modal, these algorithms will tend to terminate at different extrema depending on the location of the initial starting point. Even with these restrictions, these local optimization methods are very powerful and they will be used in this research to find the optimum of the design space represented by the continuous input variables.

Determining whether the current point being evaluated is an extrema is a critical concept in optimization. For an unconstrained function this is a relatively simple task. At an extrema, the gradient of the function with respect to the design variables will vanish (Equation 2). However, this condition does not guarantee that the point is a minimum [21]. The signs of the eigenvalues of the Hessian matrix must also be checked (Equation 3). At the minimum of the objective function the values of the eigenvalues of the Hessian formed by the second derivatives of this function will be positive (i.e. the Hessian will be positive definite [23]).

$$\nabla F(\bar{x}) = \begin{Bmatrix} \dfrac{\partial}{\partial x_1} F(\bar{x}) \\ \dfrac{\partial}{\partial x_2} F(\bar{x}) \\ \vdots \\ \dfrac{\partial}{\partial x_n} F(\bar{x}) \end{Bmatrix} \tag{2}$$

$$H = \begin{bmatrix} \dfrac{\partial^2 F(\bar{x})}{\partial x_1^2} & \dfrac{\partial^2 F(\bar{x})}{\partial x_1 \partial x_2} & \cdots & \dfrac{\partial^2 F(\bar{x})}{\partial x_1 \partial x_n} \\ \dfrac{\partial^2 F(\bar{x})}{\partial x_2 \partial x_1} & \dfrac{\partial^2 F(\bar{x})}{\partial x_2^2} & \cdots & \dfrac{\partial^2 F(\bar{x})}{\partial x_2 \partial x_n} \\ \cdots & \cdots & \ddots & \cdots \\ \dfrac{\partial^2 F(\bar{x})}{\partial x_n \partial x_1} & \dfrac{\partial^2 F(\bar{x})}{\partial x_n \partial x_2} & \cdots & \dfrac{\partial^2 F(\bar{x})}{\partial x_n^2} \end{bmatrix} \tag{3}$$

While the above conditions define the minimum of an unconstrained design, they are insufficient to define the minimum of a constrained design space. The location of the minimum of the objective function, as defined by the gradient and Hessian calculations might not be in the feasible region of the design space (Figure 5) [21].

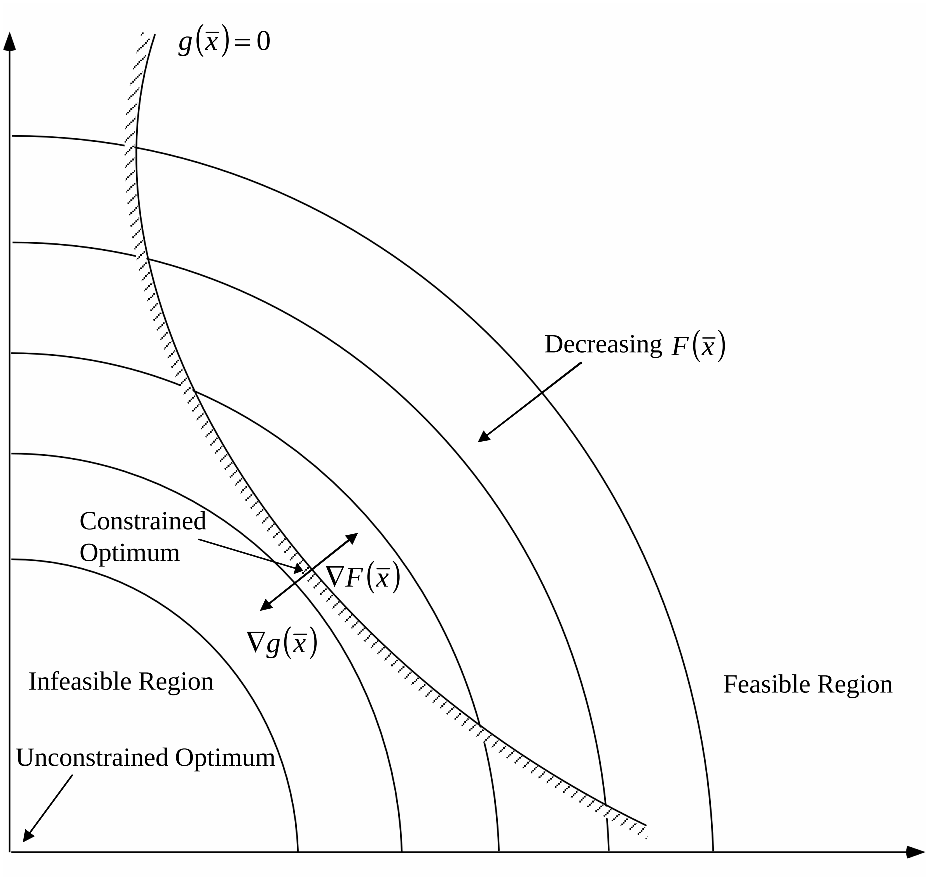


Figure 5: Constrained Optimum

In order to determine whether the current point is a constrained optimum three conditions must be met [21]. These conditions, known as the Kuhn-Tucker conditions, are outlined below. The geometric interpretation of the Kuhn-Tucker conditions is shown in Figure 6 [21]. The first condition states that the current point must be feasible. The second condition states that if an inequality constraint is not active, then its Lagrange multiplier must be zero. The final Kuhn-Tucker condition implies that at the constrained optimum there is no longer a feasible and usable direction in which to improve the design. For an unconstrained problem, this final condition simplifies to the gradient of the objective is zero at the optimum.

1. $\bar{x}^*$ is feasible

2. $l_j g_j\left(\bar{x}^*\right) = 0 \qquad j = 1, m \qquad l_j \geq 0$

3. $\nabla F\left(\bar{x}^*\right) + \sum_{j=1}^{m} l_j \nabla g_j\left(\bar{x}^*\right) + \sum_{k=1}^{l} l_{m+k} \nabla h_k\left(\bar{x}^*\right) = 0$

$l_j \geq 0 \qquad l_{m+k}$ unrestricted in sign

Kuhn-Tucker Conditions

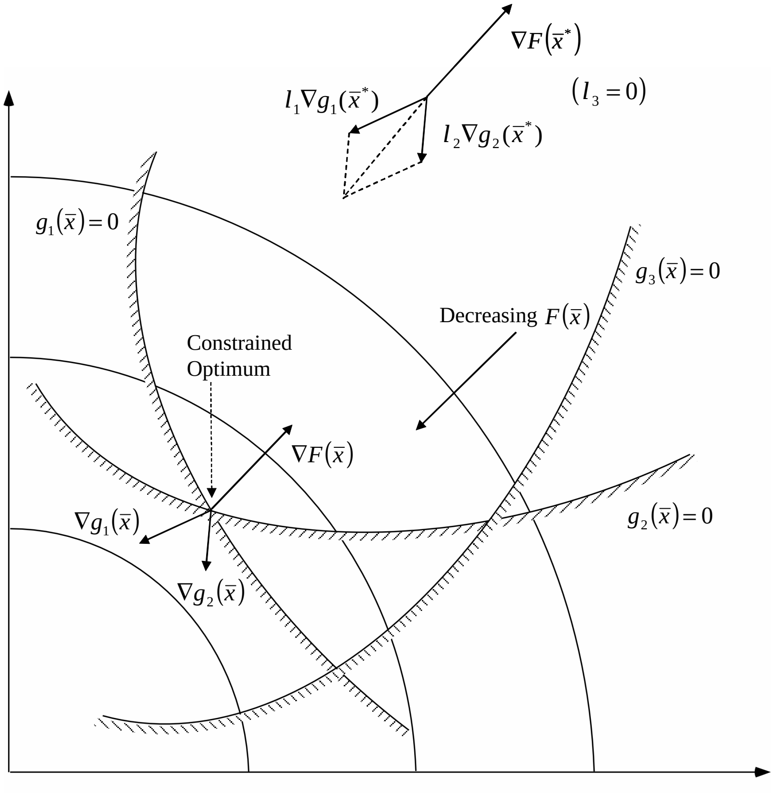


Figure 6: Graphical Representation of the Kuhn-Tucker Conditions

Now that the basic properties of numerical optimization have been established, the features of three popular constrained optimization methods will be discussed. These three methods are the Method of Feasible Directions (MoFD), Sequential Linear Programming (SLP), and Sequential Quadratic Programming (SQP).

Method of Feasible Directions (MoFD):

The method of feasible directions is a popular numerical optimization scheme [21,24]. Many different optimization codes include MoFD as a solution choice [25,26,27]. The method of feasible directions attempts to directly deal with the design constraints by finding search directions that improve the solution without violating the constraints. This method attempts to search in a usable (decreases objective function value) and feasible (no violated constraints) direction. The method essentially tries to follow the constraint boundaries until it reaches a point in the design space where there is no longer a usable and feasible direction in which to proceed. This point should be the constrained optimum.

Sequential Linear Programming (SLP):

Constrained linear programming problems are relatively simple and easy to solve and many solution methods exist [28,29,30]. Sequential linear programming attempts to take advantage of this by solving the nonlinear optimization problem as a series of linear programming problems [21,31]. The design space is linearized about an initial point and this linear programming problem is solved for the approximate solution of the design space. Then a new linear programming problem is formed around this approximate solution and the process continues until the true solution is reached. In practice, move limits must be placed on the design variables during the linear programming problem to eliminate the potential that the problem is unbounded [21].

Sequential Quadratic Programming (SQP):

Sequential quadratic programming differs from the previous methods in the way in which the search direction of the design space is formed. In SQP the search direction is formed by using a quadratic approximation of the objective function and linear approximations of the constraints. Many optimization techniques are available to solve this quadratic direction finding problem [32,33]. After the search direction is determined, the minimum of an augmented objective function is found along this direction. The objective function is augmented by adding penalty terms to account for the design constraints; therefore this search is not directly constrained. The process of finding a search direction and then minimizing the augmented objective function along that direction is continued until a converged solution is reached.

Numerical Gradients:

The methods outlined above use gradients of the objective and constraint functions to help solve the optimization problem. Analytical gradients of these functions can be supplied if they are known. Alternatively, numerical gradients can be calculated if analytical gradients are unavailable. This is the case for the current research and two types of numerical gradients will be investigated. Numerical gradients are calculated by taking the difference of function values obtained at points in the design space that are slightly different from one another. For example, Equation 4 shows the formula for a forward difference gradient. $F(x)$ is the value of the function at the current design point and $F(x+h)$ is the function value at the point x + h, where h is the step size. Step sizes are usually on the order of ~10^{-6}. As the step size is made smaller the approximation error of the gradient decreases. However, if step sizes are made too small, computer round-off error can become significant and cause an increase in the gradient calculation error. Values on the order of 10^{-6} represent a compromise between these two competing factors. Equation 4 is derived by solving the Taylor series approximation of $F(x+h)$ (Equation 5) to get $F^{'}(x)$ and then dropping the higher order terms.

$$F'(x) = \frac{F(x+h) - F(x)}{h} \quad \text{with error of Order (h)} \qquad (4)$$

$$F(x+h) = F(x) + hF'(x) + \frac{h^2}{2!}F''(x) + \frac{h^3}{3!}F'''(x) + \ldots \qquad (5)$$

The second numerical gradient option presented is a central difference gradient (Equation 6). This gradient is formed by subtracting the Taylor series approximation of $F(x+h)$ from the approximation of $F(x-h)$ (Equations 7 and 8). The central difference gradient has one-tenth the error of the finite difference gradient for the same step size, but it requires twice the number of function calls.

$$F'(x) = \frac{F(x+h) - F(x-h)}{2h} \quad \text{with error of Order (h}^2\text{)} \tag{6}$$

$$F(x+h) = F(x) + hF'(x) + \frac{h^2}{2!}F''(x) + \frac{h^3}{3!}F'''(x) + \ldots \tag{7}$$

$$F(x-h) = F(x) - hF'(x) + \frac{h^2}{2!}F''(x) - \frac{h^3}{3!}F'''(x) + \ldots \tag{8}$$

2.1.2 MINLP Methods

The optimization problem posed in this research falls under the broad class of Mixed Integer Nonlinear Programming (MINLP). Problems of this nature are found in all classes of engineering and science [34,35]. The main thrust of work done in this area, however, has been concentrated in the field of chemical engineering, specifically in chemical process/plant design [36-51], The coupling of the discrete and continuous variables, with their inherent nonlinearities, make MINLP problems hard to solve [22]. The standard form for MINLP problems is shown in Equation 9.

$$\min_{\bar{x},\bar{y}} F(\bar{x},\bar{y}) \tag{9a}$$

$$s.t.\ h(\bar{x},\bar{y}) = 0 \tag{9b}$$

$$g(\bar{x},\bar{y}) \leq 0 \tag{9c}$$

$$\bar{x}_l \leq \bar{x} \leq \bar{x}_u \tag{9d}$$

$$\bar{y} \in Y \text{ integer} \tag{9e}$$

Standard form of MINLP problems

In the above formulation, $F(\bar{x},\bar{y})$ represents the objective function, $h(\bar{x},\bar{y})$ represents equality constraints, $g(\bar{x},\bar{y})$ represents inequality constraints, the $\bar{x}'s$ are the continuous variables, and the $\bar{y}'s$ are the discrete variables. Typically the discrete variables are reformulated and represented as binary variables as outlined in reference 22.

The main reason that MINLP problems are difficult to solve is that as the number of discrete variables increase, any potential solution algorithm is faced with having to solve a large combinatorial problem in the y-space while at the same time coping with the nonlinearities of the x & y-spaces. Several solution procedures for basic MINLP problems will be discussed. These procedures all make the assumptions that the discrete variables can be relaxed (i.e. evaluated at non-integer values) and that the objective and constraint functions are convex. The three methods that will be mentioned are Branch and Bound (B&B), Generalized Bender's Decomposition (GBD), and Generalized Outer Approximation (GOA).

Branch and Bound:

Branch and bound algorithms were developed to solve mixed integer linear programming problems but have successfully been employed for nonlinear problems. This algorithm was one of the first developed to optimize problems involving discrete inputs [52,53]. B&B algorithms are formulated as tree searches where the feasible region is partitioned into subdomains and valid upper and lower bounds are generated throughout the tree search. For MINLP problems the B&B algorithm starts by solving a continuous relaxation of the problem at the root node. Information from this solution is used to choose the discrete variables to branch on, and hence divide the computational domain. These subdomains are explored and if the lower bound of a resulting subdomain is not greater than the currently available upper bound, the remaining subdomain is not explored because it will not lead to a better solution. In this way only a portion of the design space needs to be investigated. The following example is given to further clarify the Branch and Bound algorithm. It is taken from reference 22.

$$\min_{\bar{x},\bar{y}} 2x_1 - 3y_1 - 2y_2 - 3y_3 \tag{10a}$$

$$\text{s.t.} \quad 2 - x_1 - y_1 - y_2 - y_3 \leq 0 \tag{10b}$$

$$10x_1 + 5y_1 + 3y_2 + 4y_3 - 10 \leq 0 \tag{10c}$$

$$x_1 \geq 0 \tag{10d}$$

$$y_1, y_2, y_3 = 0{,}1 \tag{10e}$$

This problem has three binary variables, y_1, y_2, and y_3 and one continuous variable, x_1. The first step in the algorithm is to find the solution at the root node of the Branch and Bound tree. This value is found by relaxing the binary constraints on the y-variables and in this case solving the resulting linear programming problem. The value of the objective at the root node is -6.8. This becomes the lower bound of the problem. Next variable y_1 is branched on. Node one is then searched next providing a new lower bound and branching variable. This process continues until node 3 is reached. Node 3 provides an upper bound on the solution because all of the binary variables have binary values. The solution is now bounded between (-5, -6.667). Node 3 isn't explored further since the binary variables have values of either 0 or 1. Node 4 is infeasible so the rest of tree following that branch is ignored. Node 5 provides a new upper bound for the solution which in now bounded by the interval (-6, -6.667). The only remaining node that needs to be investigated is node 6. The solution of this node is greater than the current upper bound, so the branch and bound algorithm is complete and the final solution is $F(\bar{x}, \bar{y})$ = -6.0, at $y_1 = 1$, $y_2 = 0$, $y_3 = 1$, and $x_1 = 0$. This example used a depth first tree search. There are many variations on the basic Branch and Bound algorithm and details can be found in several other sources [54 - 61].

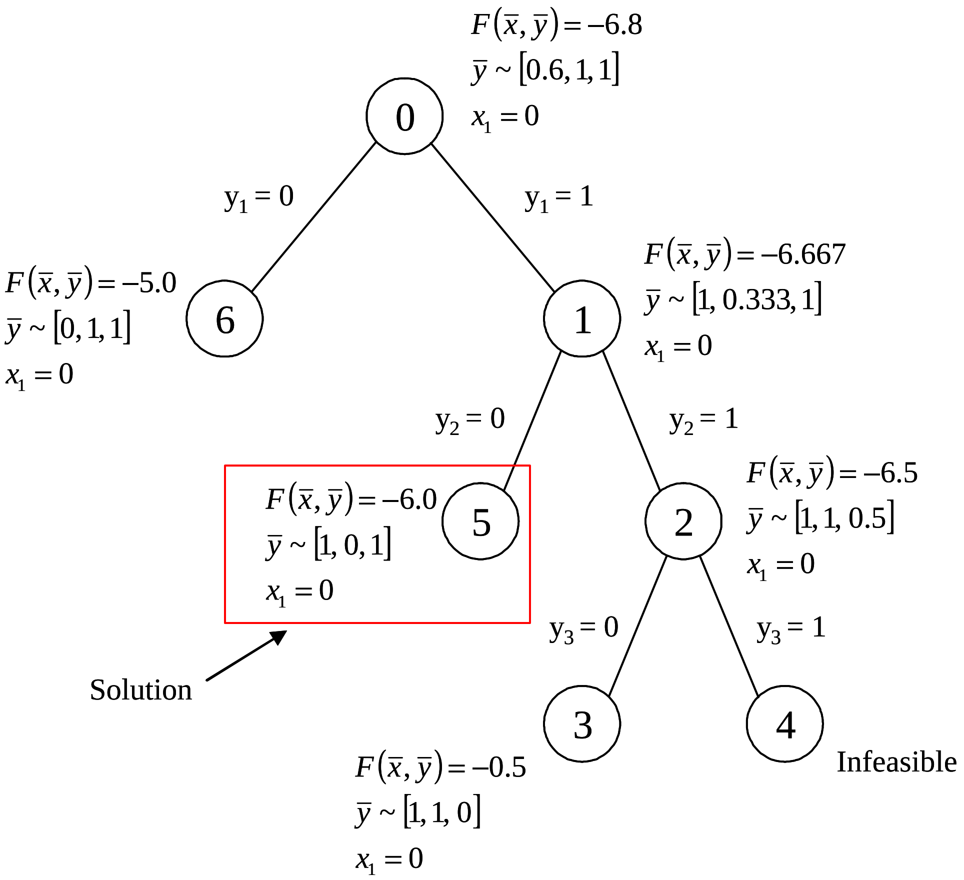

$F(\bar{x},\bar{y}) = -6.8$
$\bar{y} \sim [0.6, 1, 1]$
$x_1 = 0$
0
$y_1 = 0$
$y_1 = 1$
$F(\bar{x},\bar{y}) = -5.0$
$\bar{y} \sim [0, 1, 1]$
$x_1 = 0$
6
1
$F(\bar{x},\bar{y}) = -6.667$
$\bar{y} \sim [1, 0.333, 1]$
$x_1 = 0$
$y_2 = 0$
$y_2 = 1$
$F(\bar{x},\bar{y}) = -6.0$
$\bar{y} \sim [1, 0, 1]$
$x_1 = 0$
5
2
$F(\bar{x},\bar{y}) = -6.5$
$\bar{y} \sim [1, 1, 0.5]$
$x_1 = 0$
$y_3 = 0$
$y_3 = 1$
Solution
3
4
Infeasible
$F(\bar{x},\bar{y}) = -0.5$
$\bar{y} \sim [1, 1, 0]$
$x_1 = 0$

Figure 7: Branch and Bound Example

Generalized Bender's Decomposition:

Generalized Bender's Decomposition was first introduced by Geoffrion [62]. He generalized an approach first described by Bender in 1962 [63]. This method creates a non-increasing sequence of upper bounds and non-decreasing sequence of lower bounds for the solution of the MINLP. The upper bounds are easily found by fixing the discrete variables to a chosen value and solving the resulting nonlinear programming problem (NLP) in the continuous space only. This will either provide the optimal solution (if the discrete variables are at their optimal values) or an upper bound on the optimal solution. The lower bounds are based on duality theory. This solution method is an iterative process, where first the discrete variables are set to an initial value and an upper bound is obtained. Then the lower bounding procedure calculates a lower bound and also provides the next set of discrete variables for the next upper bound. This sequence continues until the upper and lower bounds converge to a desired tolerance. Algorithms of this type are called decomposition algorithms. Additional information on GBD can be found in references 64, 65, and 66.

Generalized Outer Approximation:

Generalized outer approximation is another decomposition algorithm used to solve convex MINLP problems. GOA is an extension of the original outer approximation method proposed by Duran and Grossman which makes it applicable to MINLP problems with a nonlinear objective function and nonlinear equality/inequality constraints [67]. GOA is similar to the GBD algorithm described earlier. The difference between the two is in the forming of the lower bounds. In GOA the lower bounds are formed by linearizing the objective and constraint functions at the solution of the current upper bounding NLP. The lower bounds are based on the accumulation of the linearized objective and constraint functions at the various solutions of the upper bounding NLP. These linearizations will provide a lower bound of the solution if both the objective and constraint functions are convex. As more linearizations are added, the algorithm will calculate a non-decreasing set of lower bounds of the objective function. Reference 68 provides the algorithm outlined below.

```
REPEAT
    1.  Solve the subproblem formed by fixing the discrete variables. (Use nonlinear
        optimization scheme to find optimum of continuous space)
    2.  Linearize the objective and constraint functions about the solution obtained in
        Step 1.
    3.  IF (Solution from Step 1 is Feasible & < Current Upper Bound) THEN
        Mark the current solution as the best point and set the Upper Bound equal to
        that value.
    4.  Solve the current relaxation $M^i$ of the lower bounding problem, giving new
        discrete variable assignments $y^{i+1}$ to be tested in the algorithm. Set $i = i + 1$.
UNTIL (Upper Bound = Lower Bound)
```

The lower bounding problem M^i is formed by the accumulation of the linearization of the objective and constraint values. If an infeasible upper bounding problem is encountered, a feasibility problem is solved instead. The solution of the feasibility problem is the values of the continuous variables that minimize the violation in the constraints. A graphical depiction, of how adding more linearizations forms a sequence of non-decreasing lower bounds, is shown in Figure 8. More details about GOA are in references 69, 70, and 71.

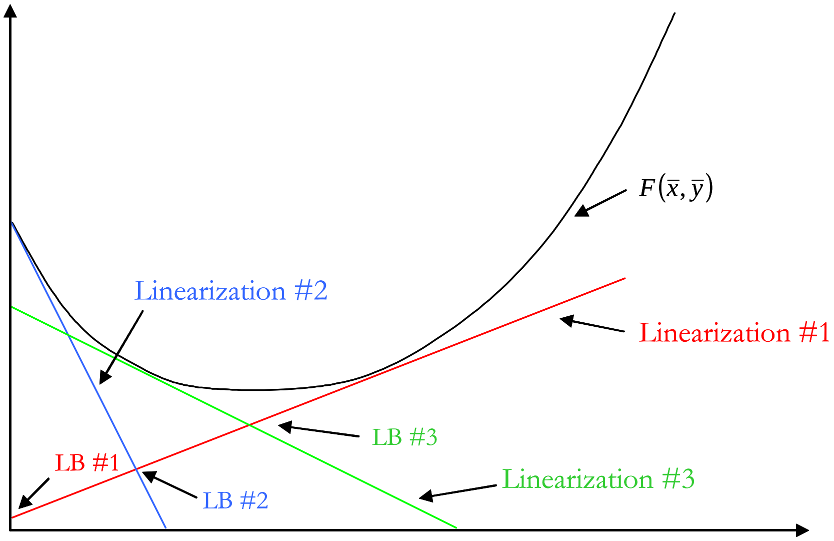


Figure 8: GOA Lower Bound Formation

2.1.3 Lower Bounding Techniques

Nonconvex MINLP Methods:

Unfortunately, all of the MINLP methods described above will only work on problems that are convex in nature. However, most engineering optimization problems tend to behave in a manner that isn't purely convex. These non-convexities can cause all of the aforementioned algorithms to terminate at non-optimal solutions. Several nonconvex MINLP schemes have been developed. These include branch and reduce [72], nonconvex outer approximation [73], SMIN-αBB, and GMIN-αBB [74,75,76,77]. All of these methods employ convex underestimators to solve nonconvex MINLP problems. Convex underestimators are convex functions that provide lower bounds of the nonconvex function they are underestimating. An example convex underestimator for a generic nonconvex function is shown in Figure 9.

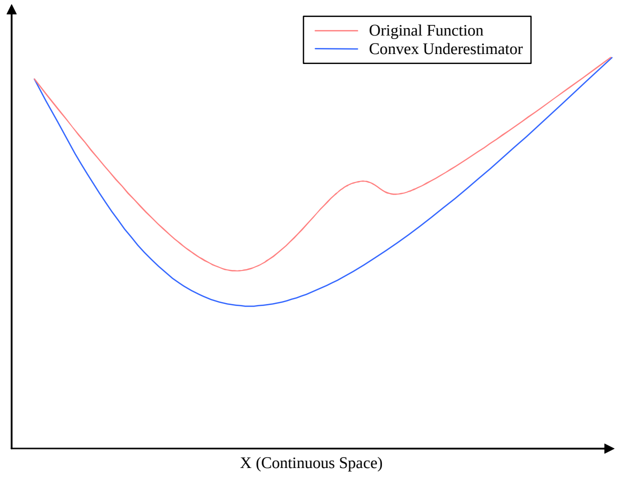


Figure 9: Example Convex Underestimator

A nonconvex function does not have a unique convex underestimator. In fact, any valid convex underestimator will work for the solution procedures mentioned above. However, these solution procedures will be more efficient the closer the convex underestimator is to the convex envelope of the original nonconvex function.

Convex Underestimators:

Convex underestimators will be discussed further because of their use in the final solution approach. These underestimators will be used to determine the lower bounding equations of the continuous design space for each discrete variable combination. The lowering bounding equations will be formed by combining a meta-model of the continuous space with an underestimator term. The meta-model used for this research is a quadratic response surface equation which is already convex. Therefore, the lower bounding properties of the convex underestimators are more important for this application than their ability to convexify the design space. Reference 74 outlines several different convex underestimating strategies. Most of these are for underestimating specific nonconvex terms found in the equation to be underestimated (bilinear, trilinear, fractional, and concave terms). However, there is also a general convex underestimator that will be used for this research. Equation 11 shows the equation for this general convex underestimator.

$$L(\bar{x}) = f(\bar{x}) + \sum_{i=1}^{n} \alpha_i \left(x_i^L - x_i\right)\left(x_i^U - x_i\right) \qquad (11)$$

Here the function $f(\bar{x})$ is underestimated over the entire domain [x^L, x^U] by the function $L(\bar{x})$. The α_i's are positive scalars and therefore the summation term in Equation 11 is always negative. This guarantees that $L(\bar{x})$ is an underestimator of $f(\bar{x})$. If the original $f(\bar{x})$ is nonconvex, the values of the α_i's can be set so that $L(\bar{x})$ becomes convex. This is because the summation term in Equation 11 is convex and therefore all the non-convexities in the original function can be "overpowered" with large enough α_i's. A function is convex only if its Hessian matrix is positive semi-definite throughout the design space, and the Hessian of $L(\bar{x})$ can be related to the Hessian of $f(\bar{x})$ using

Equation 12, where Δ is a diagonal shift matrix whose diagonal elements are the α_i's. Equation 12 leads directly to Theorem 1 also from reference 74.

$$H_L(\bar{x}) = H_f(\bar{x}) + 2\Delta \tag{12}$$

> **Theorem 1:** $L(\bar{x})$*, as defined in Equation 11, is convex if and only if* $H_f(\bar{x})$ + 2Δ = $H_f(\bar{x})$ + 2 diag(α_i) *is positive semi-definite for all* $\bar{x} \in \left[\bar{x}^L, \bar{x}^U\right]$.

There are many methods available to determine the proper diagonal shift matrix needed to form an $L(\bar{x})$ which is a convex underestimator of a nonconvex $f(\bar{x})$ [78]. However, in this research the α_i's need to be set to values which turn the convex meta-model of the continuous design space for each discrete variable combination into a lower bounding model of each discrete variable combination's true continuous design space. Therefore, a new approach was developed to find the required α_i's. This approach will be outlined in Chapter 5.

2.2 Rocket Engine Modeling

2.2.1 SCORES Background

The SpaceCraft Object-oriented Rocket Engine Simulation (SCORES) is a tool that was originally developed at The Georgia Institute of Technology to perform conceptual level “quick-look” trade studies of liquid rocket engine performance characteristics [15]. The code has since been updated by John Bradford of SpaceWorks Engineering and this updated version was used for the proof-of-concept application for the new optimization method [20]. SCORES has the ability to simulate rocket engine combustion using various propellant choices. Available fuels include liquid hydrogen, various hydrocarbons, as well as hydrazine. Oxidizers that can be used are liquid oxygen, hydrogen peroxide, and nitrogen tetroxide. Additional inputs to SCORES include the desired engine chamber pressure, the oxidizer to fuel ratio of the propellants, the nozzle expansion ratio, and the desired vacuum thrust. However, SCORES is not a powerhead analysis code. It uses efficiencies on the thrust and I_{sp} calculations to account for the different engine thermodynamic cycles. SCORES simulates the combustion process in the main combustion chamber as an adiabatic, constant pressure equilibrium calculation with the assumption that all of the species involved behave as perfect gases [15]. To calculate the equilibrium composition exiting the combustor, SCORES minimizes the Gibbs free energy of the mixture subject to mass conservation constraints [79,80,81]. The method used to minimize Gibbs free energy is based on the method described in reference 79.

Two different analysis methods are commonly used to calculate the equilibrium composition at a specified thermodynamic state. These two methods are the equilibrium constant method and the minimization of energy method. For the equilibrium constant method, a specific set of reactions must be specified in order to determine the equilibrium composition. The minimization of free energy method does not require this reaction set and therefore this method was used in SCORES. Equilibrium can be defined using three different thermodynamic properties. It is determined by maximizing entropy, or minimizing either Gibbs energy or Helmholtz energy. For the current analysis, pressure and temperature are used to determine the thermodynamic state of the composition inside the combustion chamber and therefore it is appropriate to use the minimization of Gibbs energy approach.

The inputs into the equilibrium analysis are the combustion temperature and pressure and the number of atoms of each element present. For a specified list of products, the equilibrium composition is one in which Gibbs energy is minimized at the given temperature and pressure subject to a conservation of atoms constraint. For the analysis in SCORES it is necessary to determine the adiabatic flame temperature of the combustor at a given chamber pressure. This is done by using a bisection algorithm which iterates on combustion temperature until the enthalpy of the equilibrium composition equals the initial enthalpy of the reactants entering the combustion chamber.

After the equilibrium composition and combustion temperature are found, the nozzle is analyzed. First, the nozzle exit Mach number is determined. Several assumptions are made for this calculation. The expansion process is modeled as a steady, inviscid, quasi-1D, isentropic process. Also, the flow is assumed to be either chemically frozen or in equilibrium in the nozzle. In chemically frozen flow, the reaction rates of the species in the nozzle are assumed to be zero. This implies that the composition of the flow is “frozen” at the throat conditions throughout the nozzle. Equilibrium flow makes the opposite assumption that the composition in the nozzle has infinitely fast reaction rates. Therefore, the flow composition is assumed to be at the equilibrium composition associated with the temperature and pressure at the nozzle exit. The equilibrium assumption allows for the occurrence of recombination reactions inside the nozzle and an increased nozzle exit temperature. The exit temperature increases because recombination reactions are exothermic and release energy back into the flow. Figure 10 shows a plot of exit temperature versus expansion ratio for a 500 klbs, LOX/LH2 engine with an oxidizer-to-fuel ratio (O/F) of 6.0 and a chamber pressure of 3000 psia.

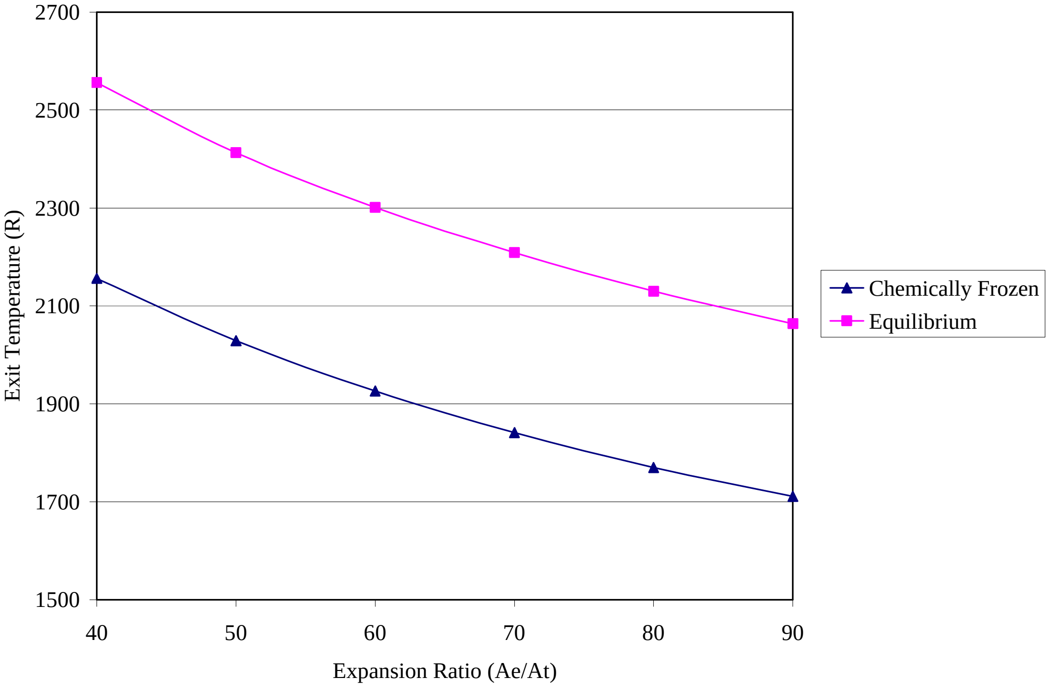
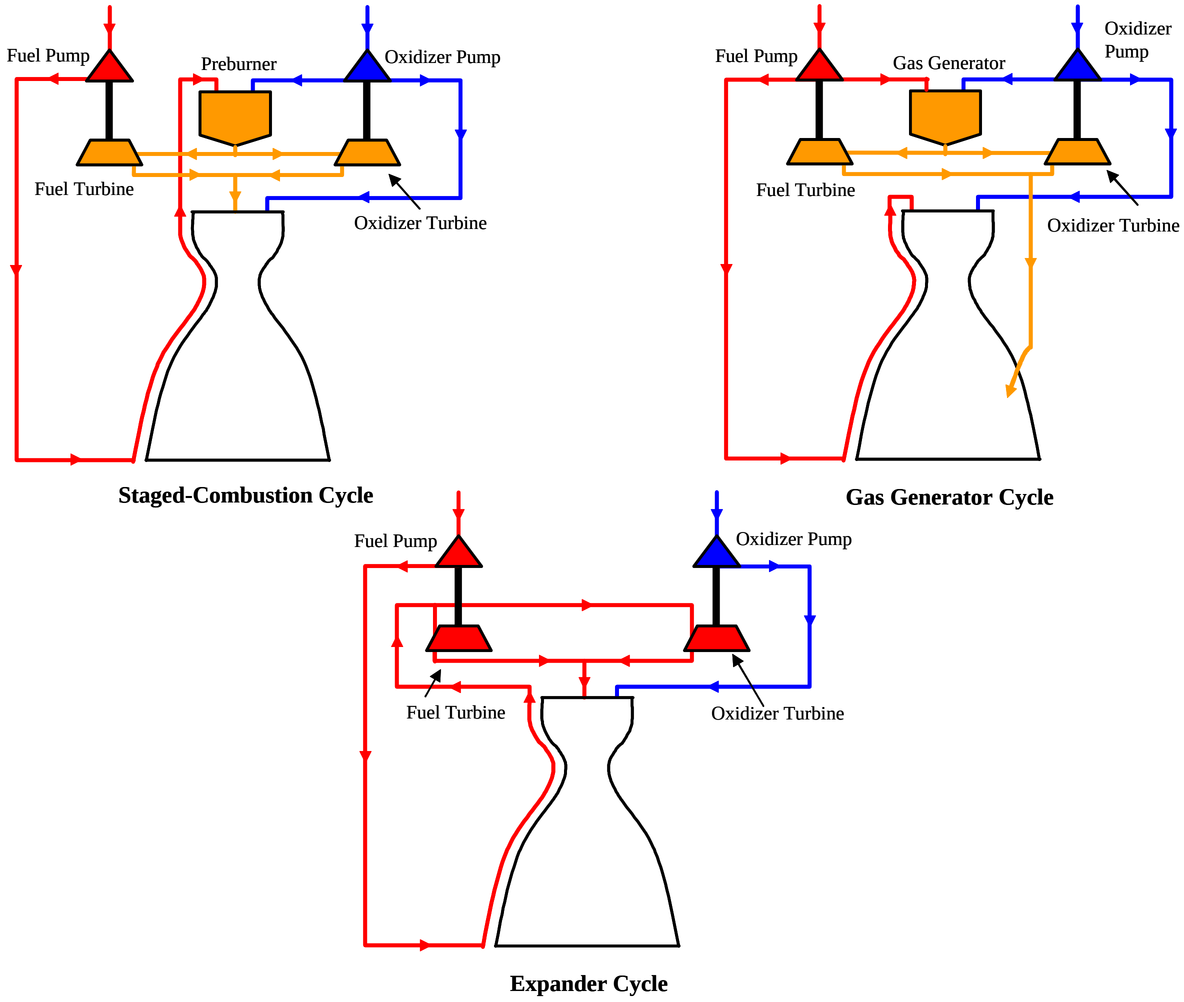


Figure 10: Exit Temperature (Frozen vs. Equilibrium Flow)

Since the recombination reactions release more heat to the flow and allow for greater flow acceleration in the nozzle, the equilibrium assumption provides an upper bound on engine performance. The chemically frozen flow assumption gives a conservative estimate of performance. True nozzle performance is somewhere between that predicted by these two assumptions. The nozzle exit conditions obtained by assuming either chemically frozen or equilibrium flow are used along with the chamber conditions to determine engine thrust and specific impulse.

2.2.2 Engine Cycle Background

Two broad categories of pump-fed powerhead cycles are used to move the propellants from their tanks to the high pressure combustion chamber. These are open and closed cycles. In closed cycles, all of the propellant leaving the vehicle's tanks passes through the throat of the engine. For open cycles, some of the propellant does not pass through the engine's throat. Open cycles tend to be less complicated because fewer powerhead components are in series with each other, but they also tend to have a lower specific impulse when compared to an equivalent closed cycle. The open cycle that will be included in this research is the gas generator cycle. The closed cycles included are the staged-combustion and the expander cycle. Representative flow paths for these cycles are shown in Figure 11 [82]. Detailed discussion of rocket engine cycles can be found in references 82, 83, and 84.



Staged-Combustion Cycle

Gas Generator Cycle

Expander Cycle

Figure 11: Engine Powerhead Cycle Diagrams

Staged-Combustion Cycle:

The staged-combustion cycle has the highest potential performance of the three cycles discussed above. It is able to provide high thrust levels while maintaining a high specific impulse. In staged-combustion engines, a preburner is used to combust propellants to produce the working fluid needed to power the turbines. This preburner is typically a small combustion chamber that is operated either very fuel rich or very fuel lean depending on the fuel/oxidizer combination and engine design heritage. The most well known staged-combustion engine is probably the Space Shuttle Main Engine (SSME). The SSME uses a fuel-rich dual preburner staged-combustion cycle. In this cycle, the hydrogen fuel is used to regeneratively cool the engine nozzle and combustion chamber. After the regenerative cooling, the majority of the fuel is sent to the preburners, while a small fraction is used to power the fuel boost pump turbine. The fuel in the preburners is mixed with enough oxidizer to produce the energy required to power the turbines that drive the main fuel and oxidizer pumps. For the SSME the fuel preburner is operated at a mixture ratio (oxidizer to fuel ratio by mass) of ~ 0.98 while the oxidizer preburner operates at a mixture ratio of ~ 0.68 [83]. The main combustion chamber O/F is near 6.0. After the preburner exhaust gases exit the turbines they are injected into the main combustion chamber with the remaining oxidizer flow. Since the turbines in a staged-combustion cycle are in series with the main combustion chamber, the turbine pressure ratios should be limited to 1.5 – 2.0 [85]. The turbine pressure ratio is the ratio of the turbine inlet pressure to the turbine exit pressure (Equation 13). Since the turbine pressure ratio is limited, the turbine inlet temperature must be relatively high to produce enough turbine power to achieve high combustion pressures.

$$P_{trat} = \frac{P_{inlet}}{P_{exit}} \tag{13}$$

Expander Cycle:

The other closed cycle to be modeled is the expander cycle. Examples of expander cycle engines are the RL-10 family of upper stage engines made by Pratt & Whitney. In an expander cycle, the hot gas used to drive the turbines is produced using the heat transferred to the fuel while it is regeneratively cooling the nozzle. Expander cycles tend to be power limited. This is because the turbine drive gases receive limited energy from the heat transfer process and therefore lower chamber pressures are found in expander engines. With a typical expander design, the available chamber pressure decreases as thrust increases. This is because increased thrust brings an associated increase in mass flow rate. This increased mass flow rate requires more power to pressurize the flow to the required chamber pressure and decreases the turbine inlet temperature if the heat transfer from the combustor remains constant. There are ways to increase the heat transfer to the turbine drive gases as thrust and hence, fuel mass flow rate increases. These include using different combustion chamber materials and increasing the chamber dimensions, but there are practical limitations on these increases. Making the chamber longer will increase the heat transfer, but the pressure drop through the regenerative cooling passages will also increase. The highest thrust true expander engine currently being proposed is the RL-60. This 50 klbs – 65 klbs vacuum thrust class engine is under development at Pratt & Whitney [86,87,88]. Even with the limited thrust constraint, true expander cycles have many advantages. They offer good I_{sp}, are simpler than staged-combustion cycles, have an inherently smooth start-up sequence, provide a benign turbine environment, and have lighter turbopump weights when compared to staged-combustion or gas generator cycles.

An alternative to a true expander cycle is the split expander. Split expander cycles offer the promise of a higher thrust engine with the same beneficial characteristics of a true expander. A split expander engine attempts to reduce the required fuel pump power without reducing the available turbine power. The fuel flow is split after the first stage of the main fuel pump. Part of the fuel is diverted to the main combustion chamber, bypassing the regenerative cooling passages, while the rest of the fuel is pressurized by the second stage of the fuel pump. This reduces the overall power requirements of the fuel pump, since less fuel encounters the pressure drop through the regenerative cooling passage, but also reduces the fuel flow through these cooling passages. The turbines now have less working fluid, but since less fuel is used to cool the chamber and nozzle, the turbine inlet temperatures increase. This increase in temperature offsets the decrease in mass flow and therefore there is no theoretical decrease in turbine work potential. Since the pump power has decreased without an associated decrease in turbine power, the potential exists for higher chamber pressures at higher thrust levels. The amount of flow that can be diverted and sent directly to the main combustion chamber without being used to cool the engine is limited. There must be enough regenerative flow to assure that the engine can still be adequately cooled without the turbine inlet temperatures increasing dramatically. Pratt & Whitney has proposed the RLX split-expander engine as a propulsion option for the 2nd generation launch vehicle program. The proposed engine design had a vacuum thrust level of ~ 350 klbs [89,90].

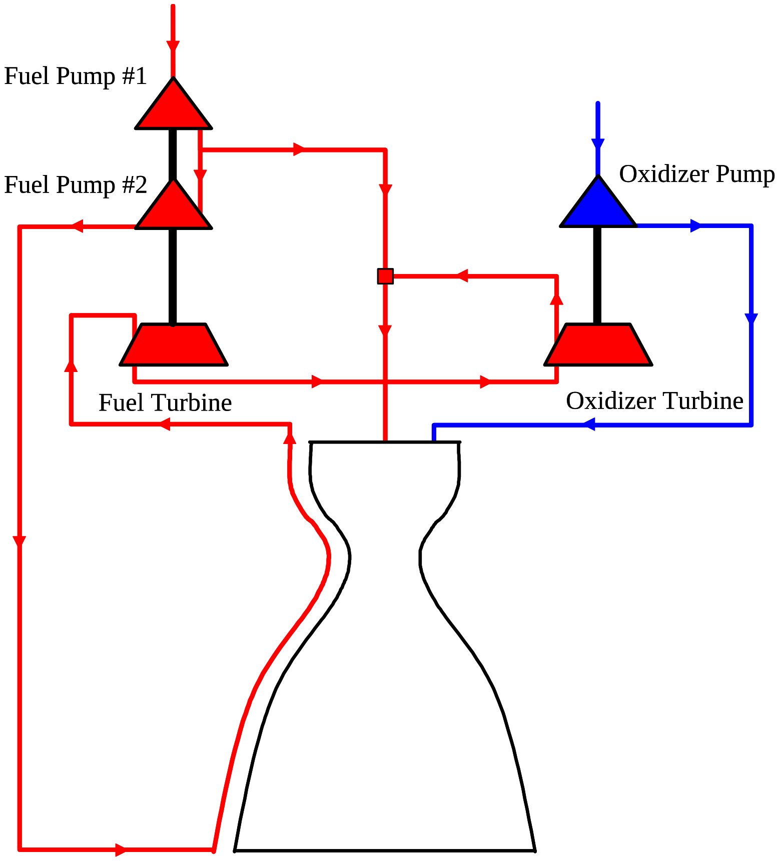


Figure 12: Split-Expander Cycle

Gas Generator Cycle:

The gas generator cycle is similar to the staged-combustion cycle in that it uses a preburner to supply the turbine working fluid. However, a gas generator cycle is an open cycle and therefore the turbine drive gases do not pass through the throat of the engine. Many engines employ the gas generator cycle including the Vulcain 2 used on the Ariane V and the F-1 engine used to power the first stage of the Saturn V. The preburner in a gas generator cycle is usually fuel rich to provide a non-oxidative environment for the turbine blades. The mixture ratio of the preburner in a gas generator, as well as the staged-combustion cycle, is controlled in order to provide turbine drive gases with a temperature that does not exceed the allowable turbine inlet temperature. Since the gas generator is an open cycle, the turbine drive gases are not introduced into the engine's main combustion chamber. They are usually expanded through a smaller separate nozzle or introduced into the main nozzle to provide film cooling and allow for further expansion. This causes a decrease in engine performance since the propellants used to power the preburner are not fully expanded from the high chamber pressure in the engine's nozzle. Since the preburner exhaust flow is not introduced into the main combustion chamber, the turbine and chamber flows are not in series. This allows a much greater pressure drop across the turbine so less mass flow is needed to produce a given power level. Typical pressure ratios seen across gas generator turbines are on the order of 4.0 – 8.0. If turbines are placed in series, the total pressure ratio across all the turbines can be as high as 20 [85]. The performance decrease associated with gas generator cycles is offset by their simpler design. This simplicity tends to yield engines that are more reliable and cheaper to build and develop.

Variations exist for all the cycles discussed above. These include the use of boost pumps, number of preburners, number of turbines, and whether the pumps and/or turbines are geared. These variations lead to a large number of possible engine cycles, with each one of these cycles representing a discrete variable choice in the optimization procedure.

2.2.3 Powerhead Code Background

There are many existing powerhead analysis tools. They typically use a modular analysis approach, where individual engine components are modeled separately using thermodynamic and other appropriate relationships and these separate modules are then linked to form the desired engine cycle to be analyzed [16,18,91]. The main components that must be modeled are the combustors (main and preburners), pumps, turbines, heat exchangers, and the nozzle. PHATCAT is the existing in-house Georgia Tech powerhead analysis tool. Following is an overview of the analysis techniques used in the original version of PHATCAT. In Chapter 3, specific modifications and additions made to PHATCAT for the current research will be addressed.

Pumps:

Pumps are used to pressurize the fuel and oxidizer to pressures that are high enough to achieve the desired chamber pressure. Every cycle modeled for this work contains at least one fuel and oxidizer pump. Many of the cycles contain more than this and most include boost pumps. The boost pumps are used for the initial pressurization from the low tank pressures. These pumps produce a relatively small pressure increase and their main purpose is to help prevent cavitation in the larger pumps. The inputs into the pump analysis used in PHATCAT are the fluid pressure and temperature at the pump inlet, the pump discharge pressure, and the propellant mass flow rate. The pump exit conditions and power requirements are determined by comparing the operation of the real pump being modeled to an isentropic pump. This is accomplished by applying an efficiency (Equation 14) to the calculated isentropic pump work [92]. This efficiency is supplied by the user and typical values are around 65% for boost pumps, 70% - 75% for the main LH2 pumps, and between 75% - 80% for LOX and hydrocarbon main pumps [83].

$$h_{pump} = \frac{work_{isentropic}}{work_{actual}} \tag{14}$$

The first step in calculating the power required by the pump is to calculate the isentropic work. The isentropic work is calculated by first setting the ideal exit entropy equal to the inlet entropy. Then, this ideal exit entropy and assumed pump exit pressure are used to lookup the ideal exit temperature in a propellant property database. The ideal exit temperature and exit pressure are then used to determine the ideal exit enthalpy, again by using a property database. The isentropic work of the pump is then calculated by subtracting the ideal exit enthalpy from the inlet enthalpy, which is found using the inlet temperature and pressure. The real pump work is calculated using the pump isentropic efficiency and isentropic work, as shown in Equation 15. Once the real pump work is known, the required pump power is simply calculated by multiplying the pump work by the propellant mass flow rate (Equation 16).

$$work_{actual} = \frac{\left(h_{out_s} - h_{in}\right)}{h_{pump}} \tag{15}$$

$$Power_{required} = \dot{m}\frac{\left(h_{out_s} - h_{in}\right)}{h_{pump}} \tag{16}$$

The efficiency of the pump is a user input, but empirical relations exist that can be used to estimate pump efficiency if it is unknown. Reference 85 outlines a procedure that uses stage-specific speed to determine pump efficiency. Stage-specific speed is a similarity parameter used to characterize pumps and is determined using Equation 17.

$$N_s = \frac{N_r\sqrt{Q}}{\left(\frac{H_p}{n}\right)^{0.75}} \qquad \left(\frac{{}^{rad}\!/\!{}_{s}\sqrt{{}^{m^3}\!/\!{}_{s}}}{m^{.075}}\right) \tag{17}$$

where: N_r – pump rotational speed (rad/s)

Q – propellant volumetric flow rate (m^3/s)

n – number of pump stages

H_p – pump’s head rise (m)

The head rise is the height that a column of propellant would have to be in order to have a pressure at the bottom of the column equal to the pump discharge pressure (Equation 18). Reference 85 gives estimates for the total allowable pressure rise over a single pump stage for various propellants. These values can be used, along with a total pump pressure rise to determine the required number of pump stages.

$$H_p = \frac{\left(P_{exit} - P_{inlet}\right)}{g_o r} \quad \text{(m)} \tag{18}$$

For a given pump rotational speed, pump design, fluid flow rate and required head, the pump efficiency can be determine using Figure 13 [85].

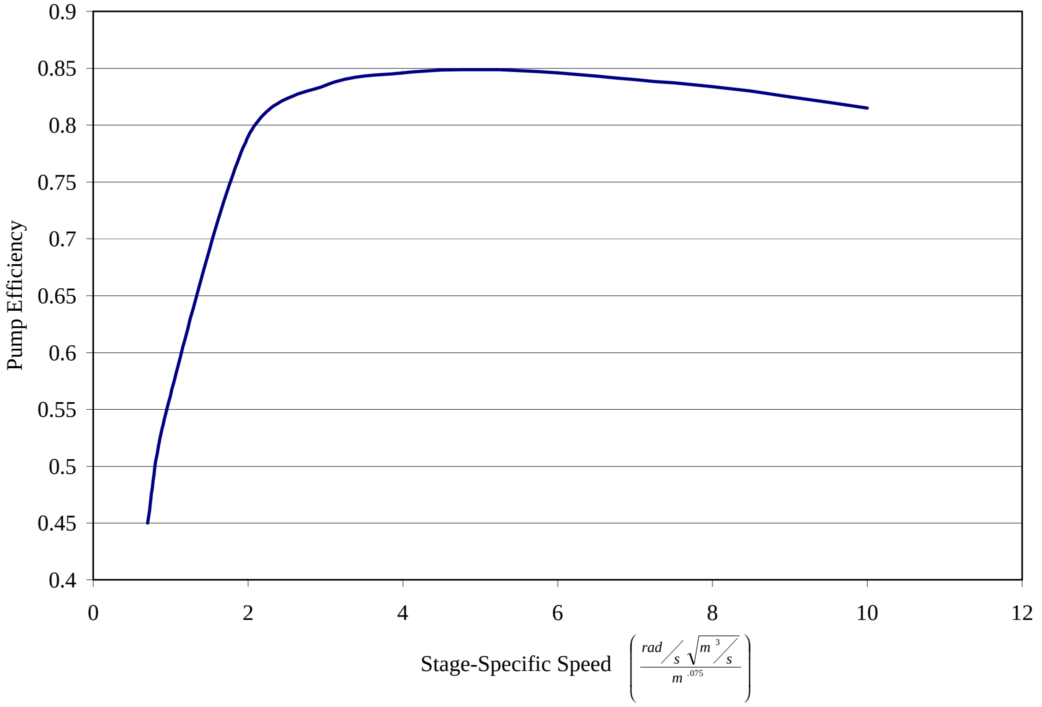


Figure 13: Pump Efficiency vs. Stage-Specific Speed

If the pump rotational speed is not known, a stage-specific speed can be assumed and Equation 17 can be used to solve for the pump rotational speed. Suggested values for stage-specific speeds are 2.0 for liquid hydrogen and 3.0 for other propellants [85]. Equation 17 can only be used to calculate the pump's rotational speed if a boost pump is present. If no boost pump is present, the lesser of the values obtained from Equations 17 and 19 should be used in order to prevent cavitation in the main pump.

$$N_r = \frac{u_{ss} NPSH^{0.75}}{\sqrt{Q}} \quad \text{(rad/s)} \qquad (19)$$

where: u_{ss} – suction specific speed

u_{ss} = 130 for LH2

u_{ss} = 90 for other cryogens

u_{ss} = 70 for other propellants

$$NPSH = \frac{P_{inlet} - P_{vapor}}{g_o r} \quad \text{(m)} \qquad (20)$$

Equation 19 calculates the rotational speed of the pump based on cavitation effects. The net positive suction head (NPSH) is a measure of the margin available in the pump to prevent cavitation. More details about pump analysis and design can be found in references 85, 93, 94.

Turbines:

Turbines are used to provide the power needed by the pumps. The turbines are driven by hot gases obtained by combusting some fuel and oxidizer in a preburner/gas generator or from heat transferred to the fuel during the regenerative cooling process. The analysis of the turbine in PHATCAT is similar to the pump calculations. The required inputs into the turbine model are the working fluid inlet pressure and temperature, the working fluid's mass flow rate, its composition, the turbine pressure ratio, and the turbine efficiency. The turbine pressure ratio was defined in Equation 13 which is repeated here for convenience.

$$P_{trat} = \frac{P_{inlet}}{P_{exit}} \tag{13}$$

As with the pump analysis, the isentropic work provided by the turbine is first calculated and then this work is modified by the turbine efficiency to get the actual work. The inlet pressure and temperature are used to find the inlet enthalpy and entropy. The ideal exit entropy is then set equal to the inlet entropy and this ideal entropy along with the defined exit pressure are used to find the ideal exit temperature. The ideal exit enthalpy can then be found using the exit pressure and ideal exit temperature. The ideal exit enthalpy and the inlet enthalpy are then used to calculate the ideal turbine work. This ideal work is adjusted by the turbine efficiency in order to obtain the actual turbine work (Equations 21 & 22). Now that the actual turbine work is known, the power produced by the turbine can be found by multiplying this work by the working fluid mass flow rate (Equation 23).

$$h_{turbine} = \frac{work_{actual}}{work_{isentropic}} \tag{21}$$

$$work_{actual} = h_{turbine}\left(h_{in} - h_{out_s}\right) \tag{22}$$

$$Power_{produced} = \dot{m}h_{turbine}\left(h_{in} - h_{out_s}\right) \tag{23}$$

As with the pump efficiency, if the turbine efficiency is unknown, it can be estimated using empirical relations. The procedure to calculate turbine efficiency again follows the analysis outlined in Reference 85. The first parameter that must be calculated in order to determine the turbine efficiency is the turbine isentropic spouting velocity (Equation 24). This is the velocity the turbine working fluid would have if it was expanded isentropically from the turbine inlet conditions to the turbine exit pressure.

$$C_o = \sqrt{2c_pT_i\left[1-\left\{\frac{1}{P_{trat}}\right\}^{\frac{g-1}{g}}\right]} \quad \text{(m/s)} \tag{24}$$

where: T$_i$ – turbine inlet temperature (K)

P$_{trat}$ – turbine pressure ratio (Equation 13)

The next parameter that is needed is the maximum allowable turbine pitchline velocity. This velocity is a function of the turbine inlet temperature and the turbine blade material. Figure 14 shows data for allowable pitchline velocities versus inlet temperature for various turbine blade materials [95].

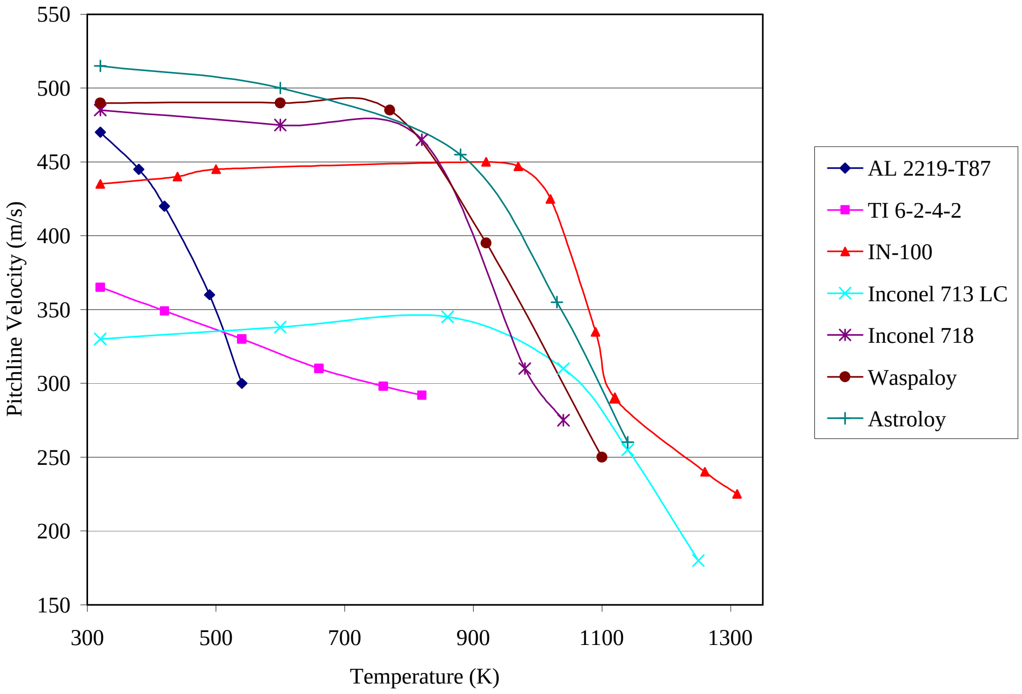


Figure 14: Pitchline Velocity versus Turbine Inlet Temperature

Once turbine pitchline velocity and isentropic spouting velocity are found, the only remaining design decision is to determine what type of turbine to use. Most turbine designs are limited to 2-rotors or less to reduce weight and complexity. For gas-generator cycles, where large turbine pressure ratios are desired, a pressure compounded turbine is usually used. However, if the inlet blade heights of the pressure compound turbine are too small a designer may be forced to switch to a velocity compound turbine because they usually have taller inlet blades [85]. The reaction turbine design is typically used in expander and staged-combustion engines. These turbines are designed for higher pitchline to spouting velocity ratios, which are common in closed cycle engines because of the desire to limit the turbine pressure ratio. The following figures, from reference 96, allow the determination of turbine efficiency using the pitchline velocity, isentropic spouting velocity, and turbine type. These efficiencies do not take into account losses from viscous effects, friction, or leakage. Therefore, for large turbines, the efficiency should be reduced by 5% and for small turbines it should be decreased by 25% [85]. Additional information about liquid rocket engine turbines can be found in reference 97.

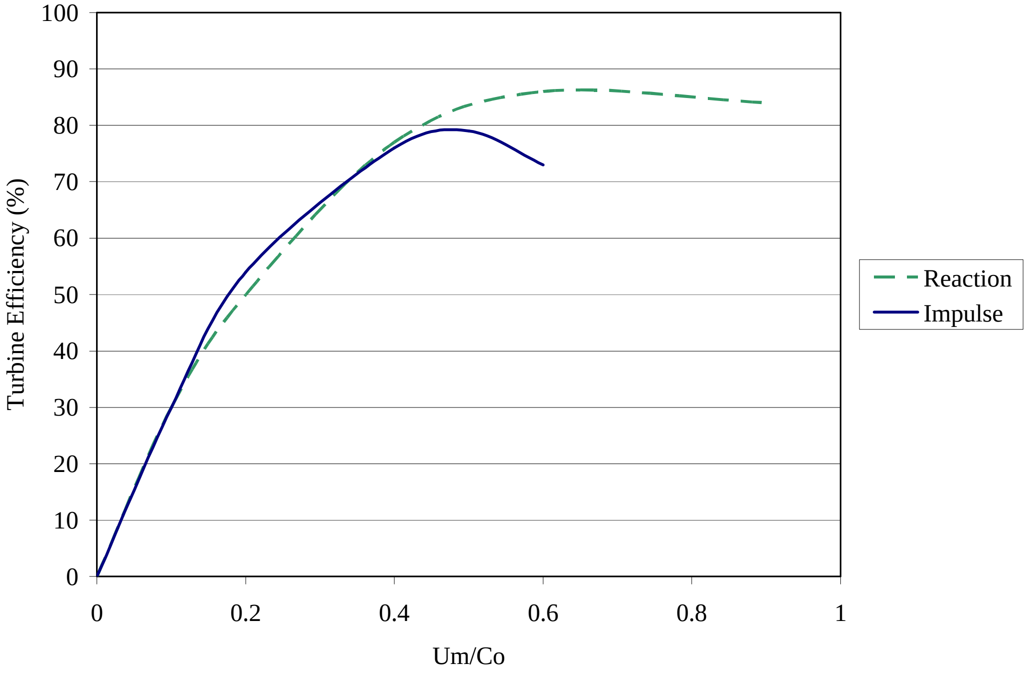


Figure 15: Turbine Efficiency versus Velocity Ratio (One Rotor Turbines)

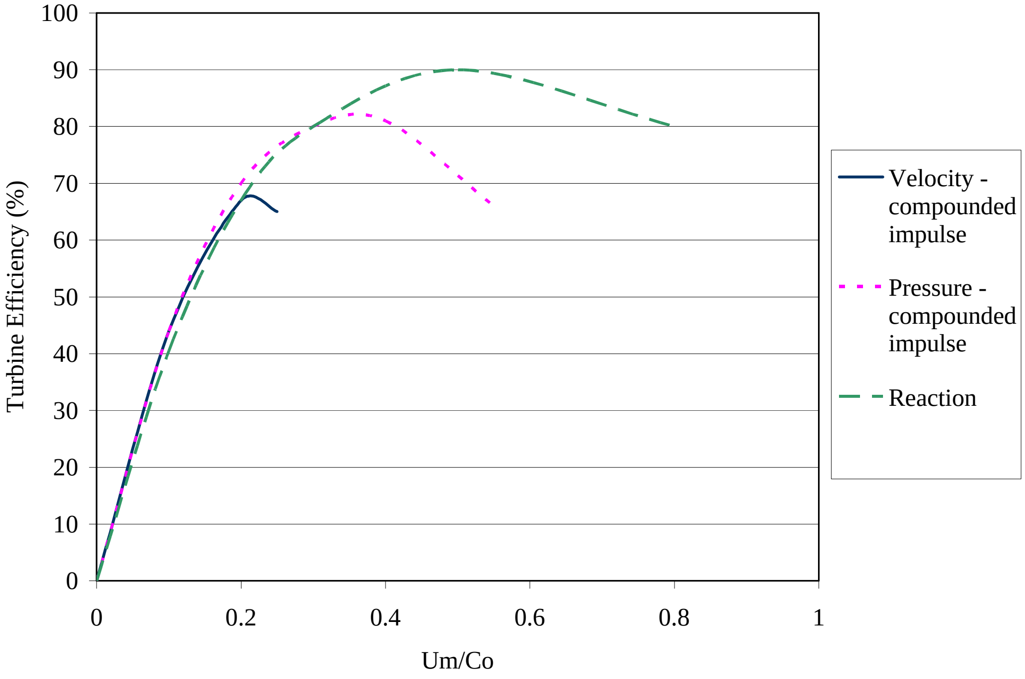


Figure 16: Turbine Efficiency versus Velocity Ratio (Two Rotor Turbines)

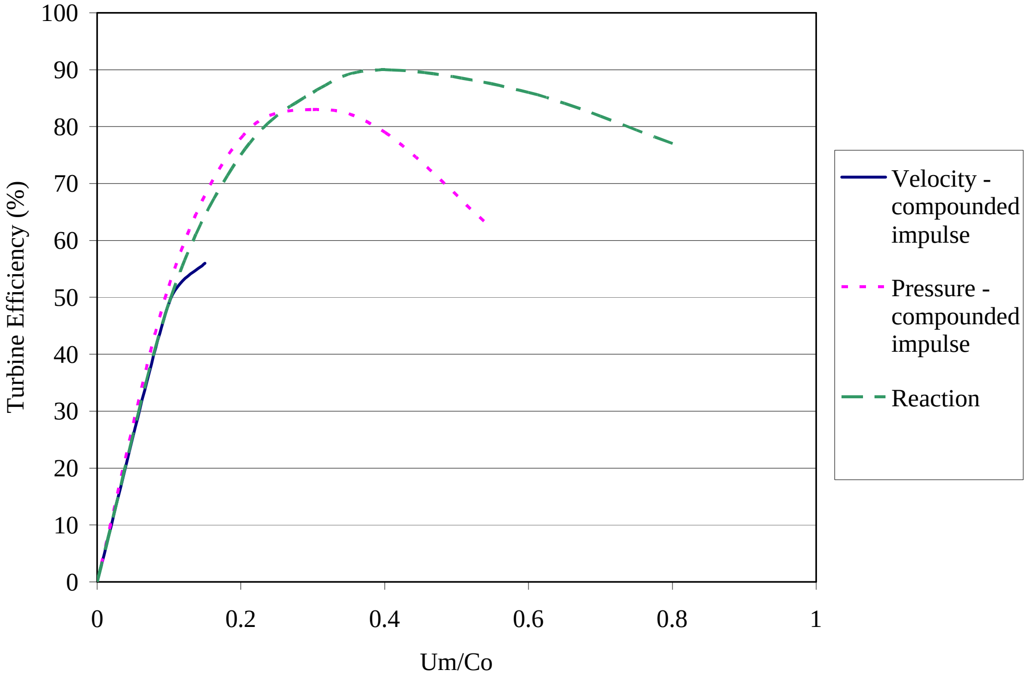


Figure 17: Turbine Efficiency versus Velocity Ratio (Three Rotor Turbines)

Heat Exchangers:

Heat exchangers are used in several places in an engine powerhead cycle. The most obvious place is the use of a heat exchanger to regeneratively cool the engine nozzle and main combustion chamber. For expander cycles, this heat transfer is what produces the high temperature gas used to power the turbines. Also, the regenerative cooling passages must transfer enough heat to the propellant to keep the nozzle and chamber below their structural temperature limits. Another use for heat exchangers in an engine cycle is to add energy to the small amount of propellant that is commonly used to maintain the pressure in the main propellant tanks.

A heat exchanger is modeled in PHATCAT using the first law of thermodynamics. The inputs into the heat exchanger analysis are the inlet pressure, temperature and mass flow rate, along with the fluid pressure drop across the heat exchanger, the efficiency of the exchanger, and the baseline heat transfer rate. The heat exchanger efficiency is a measure of how much of the energy leaving the chamber and nozzle flow is captured by the fluid in the exchanger. The energy capture is quantified by the increase in fluid enthalpy after it passes through the heat exchanger. Using the known fluid inlet conditions, the heat exchanger exit enthalpy can be calculated using Equation 25. Then this exit enthalpy is used, along with the exit pressure, to find the fluid exit temperature.

$$h_{out} = h\left(\frac{\dot{Q}}{\dot{m}}\right) + h_{in} \tag{25}$$

The heat transfer rate used in Equation 25 is not necessarily equal to the baseline heat transfer rate input by the user. The baseline heat transfer rate is scaled depending on the current main combustion chamber target. This scaling is done using an empirical relationship outlined in reference 84 (Equation 26).

$$\dot{Q} = \dot{Q}_{base}\left(\frac{P}{P_{base}}\right)^{0.8} \tag{26}$$

Combustors:

Two different combustion devices are use in engine cycles. The first is the main combustion chamber which is usually operated at conventional mixture ratios. The other combustion device is the preburners found in staged-combustion and gas generator cycles. These preburners run either very fuel rich or fuel lean depending on the engine cycle and propellant combination. Both of these combustion devices are modeled by the same minimization of Gibb's energy equilibrium analysis used in SCORES. Several modifications to the standard equilibrium analysis had to be made for it to be able to model both fuel rich and fuel lean combustion. The convergence of the equilibrium product species concentrations is dependent on the initial guesses of those concentrations. Therefore, for both the fuel rich and fuel lean combustion devices the product species initial guesses were changed to better reflect the final solution. For fuel rich combustion, the initial guesses for the concentrations of the more fuel oriented products were increased. Similarly, for lean combustion, the concentrations of the more oxidizer oriented products were increased. More specific information about both main combustion chambers and preburners is available in references 98 and 99.

Flow Mixes and Splits:

Not all of the propellant leaving the vehicle’s tank follows the same path to the main combustion chamber. There are many instances where part of the main propellant flow needs to be diverted or mixed together. For example, in the split-expander cycle (Figure 12) the fuel flow is split after the first pump and the mixed back together after the split flow has passed through the turbines. The flow splits are modeled in PHATCAT as lossless devices. The fluid properties leaving the split are the same as those entering. The amount of flow exiting each outlet of the split is determined by a split percentage that is input by the user. Mixers are also modeled as lossless devices and the mixer exit enthalpy is found by taking the mass average of the input enthalpies (Equation 27). The exit enthalpy and pressure are then used to determine the remaining exit conditions.

$$h_{out} = \frac{\dot{m}_{in_1} h_{in_1} + \dot{m}_{in_2} h_{in_2}}{\dot{m}_{in_1} + \dot{m}_{in_2}} \qquad (27)$$

Valves and Injectors:

Valves and injectors are both modeled as isenthalpic devices across which there is a large pressure decrease. The user specifies the pressure drop across the injectors and valves, and the exit temperature is then found by using the calculated exit pressure and the inlet enthalpy. The pressure drop across an injector should be about 20% for unthrottled engines and as high as 30% for throttled engines. This pressure drop helps isolate the combustion chamber from the rest of the powerhead so combustion pressure oscillations do not propagate back through propellant feed system [85].

Nozzle:

The final component to be addressed is the nozzle. Nozzle calculations in a powerhead code must take into account heat transfer to the propellant during the regenerative cooling process. If this heat transfer from the nozzle is not accounted for, an overly optimistic assessment of engine performance will result. The nozzle is divided at the throat and two different analysis methods are used for each half. For the convergent section, isentropic relations are assumed to hold true, and they are used to find the throat conditions assuming choked flow at the throat.

The divergent section of the nozzle is where the heat transfer to the cooling flow is taken into account. Therefore, the divergent section is not adiabatic, and isentropic relations can not be used. The method of potentials outlined in reference 100 was used to determine nozzle performance. This method allows for the modeling of flow with multiple driving potentials. For this nozzle analysis, two driving potentials exist, the nozzle area change and the regenerative heat transfer.

The variable that is solved for using the method of potentials is the nozzle exit Mach number. The Mach number is found by using a Runge-Kutta numerical integration routine that solves for the flow Mach number as a function of axial distance from the throat. The contours of the nozzle must be known in order to determine the exit Mach number. Both conical and bell nozzles are available in PHATCAT. For more details on the nozzle analysis see references 16 and 100.

Fluid Properties:

All of the modular analysis routines described above require the availability of fluid properties at various temperatures and pressures in order to complete their calculations. Two different property databases were included with the original version of PHATCAT. The first is tabular data for hydrogen and oxygen that was used to determine the properties needed for any analysis routine encountered before a combustion device. The source for the original cold flow data used in PHATCAT is the NIST properties code SUPERTRAPP [101]. After the fluid passes through a combustion device, the second fluid properties database is used. This database consists of curve fits of various species included in the JANAF tables [102]. The species that were included are H2, O2 and the possible products of their equilibrium reaction. For both the JANAF curve fits and the tabular NIST data, the fluid properties included are enthalpy, entropy, and specific heat as functions of temperature and pressure. The JANAF curve fits also supply Gibb's energy which is needed in the equilibrium routine.

Two different ways are needed by the modular analyses to determine fluid properties. The first way is by supplying a temperature and pressure and receiving an enthalpy, entropy, specific heat, or Gibb's value. This is very straightforward because temperature and pressure are the natural inputs for both the curve fits and the 2-D tabular interpolation routine. The second way in which properties need to be determined is by supplying a pressure and either an enthalpy or entropy and receive temperature as the output. This requires a root finding method to be employed. The original version of PHATCAT uses a Newton-Raphson root finding method to determine the correct temperature. Details of this method are provided in the next section.

Power Balancing:

In order to complete the analysis of an engine cycle, it must be "balanced." A balanced engine cycle is one in which the power required by the pumps is produced by the turbines. Also, a balanced cycle will have the desired chamber pressure. Typically, the pump and turbine powers are balanced by changing the pressure ratio across the turbines until the turbine power matches the pump power. Simultaneously the pump exit pressures will be varied until the desired chamber pressure is achieved. The pump exit pressures must be high enough to make up for the pressure drop that occurs as the propellant is traveling from the pump to the chamber. The pressure drop comes from injector pressure losses, regenerative cooling pressure losses, and depending on the cycle, the turbine pressure drop. Other balances exist and many are cycle dependent. Total propellant mass flow rate can be varied to achieve a desired thrust level. Also, gas generator mixture ratio or regenerative flow split percentages can be varied to set a specific turbine inlet temperature.

A multi-dimensional Newton-Raphson root finding scheme is used in PHATCAT to balance the engine. The method requires unimodal, continuous and differentiable functions. Since analytical derivatives of the engine analysis are not available, numerical derivatives are used. To begin the Newton-Raphson iteration process, an initial guess is made for the independent variable. Then the analysis function is evaluated at that point and the difference between the target for the dependent variable and its actual value is calculated. Figure 18 shows a graphical representation of one iteration of a 1-D Newton-Raphson problem [16]. After the function is evaluated at the initial guess, the x-intercept of the line tangent to the function at the initial point is found. This x-intercept provides the updated solution (Equation 28).

$$x_{n+1} = x_n - \frac{F(x_n)}{F'(x_n)} \tag{28}$$

This process continues until the root of the function is found. To extend the Newton-Raphson method to multiple dimensions, the points x_{n+1}, x_n and the function value $F(x_n)$ become vectors and the function's derivative, $F'(x_n)$ becomes the Jacobian matrix. The multi-dimensional form is given in Equation 29.

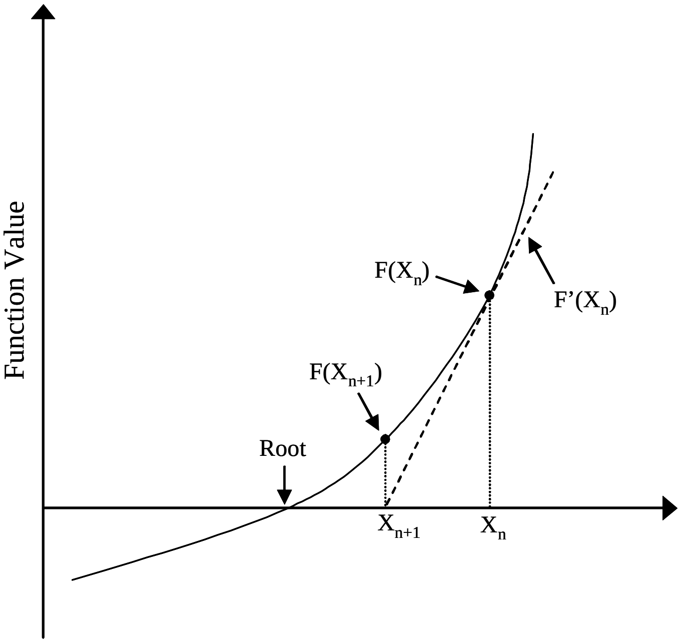


Figure 18: Newton-Raphson Graphical Example

$$\bar{x}_{n+1} = \bar{x}_n - \frac{\bar{F}}{J} \tag{29}$$

where:

$$\bar{x}_n = \begin{bmatrix} x_{n,1} \\ \vdots \\ x_{n,m} \end{bmatrix}$$

$$\bar{F} = \begin{bmatrix} F_1 \\ \vdots \\ F_m \end{bmatrix}$$

$$J = \begin{bmatrix} \frac{\partial F_1}{\partial x_{n,1}} & \cdots & \frac{\partial F_1}{\partial x_{n,m}} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_m}{\partial x_{n,1}} & \cdots & \frac{\partial F_m}{\partial x_{n,m}} \end{bmatrix}$$

CHAPTER 3

ENGINE ANALYSIS DETAILS

3.1 Additions to PHATCAT

PHATCAT was used to simulate the powerhead and determine the performance characteristics of the various engine cycles studied for this research. The original version of PHATCAT was written in July, 2001, by Tim Cormier at The Georgia Institute of Technology. In its original state, PHATCAT does not provide the desired accuracy or capability required for this research. Therefore, several changes/additions were made to the program to tailor it for this use. A list of these modifications follows, and more detailed information about certain changes will be provided.

- The Newton-Raphson root finding method used to balance the powerhead was made more robust, allowing the code to be used more effectively in an optimization scheme.
- The hydrogen property database was changed, which fixed errors in the power and exit temperature results supplied by the pump and turbine routines.
- The JANAF curve fits for the specific heat of H_2, H and OH were fixed.

- RP-1, methane (CH4), and propane (C3H8) were added as possible fuel choices. Original version only had the choice of hydrogen fuel.
- The equilibrium combustion model was adjusted to accurately model RP-1, CH4 and C3H8 combustion. This included stoichiometric, fuel-rich and fuel-lean combustion.
- An option to scale the regenerative heat transfer rate and boost pump exit pressures with chamber pressure was added. This allowed for balanced engine designs, for specific engine cycles, to be achieved over a wider range of input variables.
- Twenty-one new engine cycles were added as input options to PHATCAT. The original version included three.

Two separate fluid property databases are used in PHATCAT. The first is the cold flow database, which is used to get properties of fluids that haven't gone through a combustion device. This database originally consisted of tables of property data, for hydrogen and oxygen, obtained from the NIST properties code SUPERTRAPP [101]. The second property database is the hot flow database, which is used to get properties of fluids and mixtures either during or after combustion. This database consists of curve fits of data from the JANAF tables [102].

As will be shown in the next section, the original version of PHATCAT gave significant errors when comparing hydrogen turbopump powers and exit temperatures to data obtained from literature. These discrepancies were found to be caused by errors in the cold flow properties database. After researching the problem, it was discovered that the NIST SUPERTRAPP code was not the best source for properties of hydrogen as a function of temperature and pressure. The code is designed mainly to provide properties of hydrocarbon mixtures. Therefore, a new source of hydrogen data was found. The new source was also a code from NIST, called REFPROPS [103]. This code uses the equation of state by Youngblood to determine hydrogen fluid properties [104]. The replacement of the old hydrogen properties fixed the errors associated with the pump and turbine analyses.

One of the discrete variable choices, available in the final application of the optimization method, is the engine fuel choice. In the original version of PHATCAT only one fuel choice is available. Therefore, the number of fuels was expanded to include RP-1, methane, propane, and the original hydrogen fuel. Both property databases had to be expanded to include these new fuels. Also, missing properties for some of the possible combustion products of the new fuels were added to the hot flow database. Cold flow data for methane, propane and RP-1 was obtained from SUPERTRAPP. Methane and propane are included as default inputs to SUPERTRAPP, but RP-1 is not. Several methods are available for simulating a mixture of hydrocarbons such as RP-1 in SUPERTRAPP. These include inputting RP-1 as a single pseudo-fluid, a mixture of defined hydrocarbons, or as a petroleum fraction. All three methods were investigated and it was found that the petroleum fraction method gave results that more closely agreed with that published in literature. Since various properties were missing from the hot flow database for some of the hydrocarbon combustion products, additional curve fits were added. One curve fit that needed to be included was the enthalpy of RP-1 as a function of temperature. The enthalpy of RP-1 is not available from the JANAF tables, so other sources for this data were found. The enthalpy of formation for RP-1 was taken from reference 79. Sensible enthalpy was determined using specific heat data found in reference 105 for JP-4 (Jet Propellant 4), which is very similar to RP-1. This enthalpy data was used for the hot flow database, instead of the values obtained from SUPERTRAPP, because the equilibrium results for RP-1 combustion will be compared to those obtained from reference 79 and this comparison is more applicable if the initial enthalpy of RP-1 is the same in both codes.

In order to determine the equilibrium composition of the combustion products of RP-1, propane or methane with LOX, the equilibrium analysis routine in PHATCAT had to be adjusted. The original equilibrium analysis model included H_2, O_2, H_2O, OH, H, O, CO, and CO_2 as possible combustion products. These are all the products that are needed for hydrogen/oxygen combustion at various mixture ratios, and for hydrocarbon/oxygen combustion at either fuel-lean or near stoichiometric mixture ratios. For fuel-rich hydrocarbon combustion more product species are needed. Therefore, for propane combustion, C3H8 was added as a possible product, and for methane combustion, CH4 was added as a product choice. For fuel-rich RP-1 combustion, CH4, C3H8, C2H2, and C2H6 were included as additional products. Also, the initial guesses for the product concentrations were changed depending on the mixture ratio. This helped eliminate convergence problems at very fuel-rich or fuel-lean conditions.

Originally, PHATCAT only contained three specific engine cycle choices. These were the RL-10 expander cycle, the SSME dual preburner fuel-rich staged-combustion cycle, and the Vulcain gas generator cycle. Several new engine cycles were added in order to give more discrete variable choices for the final application of the optimization method. The SSME cycle was replaced with two faster running LH2 staged-combustion cycles. The first has a powerhead design very similar to the SSME, while the second only has a single preburner and one main turbine. Variations of the Vulcain cycle were also added to PHATCAT. These variations included the use of different numbers of turbines and boost pumps, and the use of hydrocarbon fuels. The RL-10 expander cycle was replaced with a higher thrust class split expander cycle similar to the RLX. Finally, an oxidizer-rich single preburner staged-combustion cycle was included, which used RP-1, CH4 and C3H8 as fuels. This cycle is similar to the RD-180, except it only has one main combustion chamber. These additions increased the number of engine cycle choices in PHATCAT from three to twenty-two.

3.2 Verification of Additions

3.2.1 Hydrogen Component Verification

The performance of the various modular components used to form the engine cycles in the original version of PHATCAT, were compared to powerhead data for the SSME taken from reference 83. These comparisons, which can be found in the original PHATCAT documentation (reference 16), show that the original version of the code was able to match the performance data for the LOX side of the SSME cycle, but was unable to correctly predict the LH2 component properties. For brevity, an interested reader is referred to reference 16 for the LOX component verification. The LH2 component verifications will be repeated here along with the results from the updated version of PHATCAT. The tables below show the comparisons of the results predicted by the original version of PHATCAT for the SSME low-pressure fuel pump (LPFP) and high-pressure fuel pump (HPFP). There is noticeable error in the exit temperature calculations for both pumps, and the power calculations for the low-pressure pump.

Table 1: SSME LH2 Pump Exit Temperatures (Original Results)

	P_{in} (psia)	*T_{in} (R)*	*P_{out} (psia)*	*m_{dot} (lbm/s)*	h	*Original PHATCAT T_{out} (R)*	*Ref. 83 T_{out} (R)*	*% error*
LPFP	30.02	37.08	282.82	161.40	.65	39.6	40.9	3.18%
HPFP	282.82	40.86	7055.95	161.40	.73	91.5	103.1	11.25%

Table 2: SSME LH2 Pump Powers (Original Results)

	P_{in} (psia)	*T_{in} (R)*	*P_{out} (psia)*	*m_{dot} (lbm/s)*	h	*Original PHATCAT Power (HP)*	*Ref. 83 Power (HP)*	*% error*
LPFP	30.02	37.08	282.82	161.40	.65	3412	3694	7.63%
HPFP	282.82	40.86	7055.95	161.40	.73	77235	77497	0.34%

The new version of PHATCAT includes updated hydrogen properties. These new hydrogen properties, corrected the errors found for both fuel pumps. The update results are shown in Table 3 and Table 4. The error has been reduced to less than 0.5% for both the exit temperature and power calculations.

Table 3: SSME LH2 Pump Exit Temperatures (New Results)

	P_{in} (psia)	*T_{in} (R)*	*P_{out} (psia)*	*m_{dot} (lbm/s)*	h	*New PHATCAT T_{out} (R)*	*Ref. 83 T_{out} (R)*	*% error*
LPFP	30.02	37.08	282.82	161.40	.65	41.0	40.9	0.24%
HPFP	282.82	40.86	7055.95	161.40	.73	103.4	103.1	0.29%

Table 4: SSME LH2 Pump Powers (New Results)

	P_{in} (psia)	*T_{in} (R)*	*P_{out} (psia)*	*m_{dot} (lbm/s)*	h	*New PHATCAT Power (HP)*	*Ref. 83 Power (HP)*	*% error*
LPFP	30.02	37.08	282.82	161.40	.65	3682	3694	0.32%
HPFP	282.82	40.86	7055.95	161.40	.73	77516	77497	0.02%

Similar errors are noticed in the hydrogen turbine results. The tables below show the comparison of the original and new results, for output temperature and power of the low-pressure fuel turbine (LPFT), to those published in reference 83. The original PHATCAT analysis accurately predicted the turbine exit temperature, but over-predicted the turbine power output. The new hydrogen properties added to the cold flow database fixed the turbine output power results, while maintaining an accurate prediction of turbine exit temperature.

Table 5: SSME LH2 Turbine Exit Temperature (Original Results)

	P_{in} (psia)	*T_{in} (R)*	*DP/P*	*m_{dot} (lbm/s)*	h	*Original PHATCAT T_{out} (R)*	*Ref. 83 T_{out} (R)*	*% error*
LPFT	5011.06	455.04	1.3219	34.08	.52	441.7	440.6	0.25%

Table 6: SSME LH2 Turbine Power (Original Results)

	P_{in} (psia)	*T_{in} (R)*	*DP/P*	*m_{dot} (lbm/s)*	h	*Original PHATCAT Power (HP)*	*Ref. 83 Power (HP)*	*% error*
LPFT	5011.06	455.04	1.3219	34.08	.52	3955	3694	7.07%

Table 7: SSME LH2 Turbine Exit Temperature (New Results)

	P_{in} (psia)	*T_{in} (R)*	*DP/P*	*m_{dot} (lbm/s)*	h	*New PHATCAT T_{out} (R)*	*Ref. 83 T_{out} (R)*	*% error*
LPFT	5011.06	455.04	1.3219	34.08	.52	439.5	440.6	0.25%

Table 8: SSME LH2 Turbine Power (New Results)

	P_{in} (psia)	*T_{in} (R)*	*DP/P*	*m_{dot} (lbm/s)*	h	*New PHATCAT Power (HP)*	*Ref. 83 Power (HP)*	*% error*
LPFT	5011.06	455.04	1.3219	34.08	.52	3685	3694	0.24%

The original PHATCAT analysis of the regenerative heat exchangers used for the cooling of the nozzle and chamber of the SSME, over-predicted the heat exchanger exit temperatures. The following tables show that correcting the hydrogen properties again fixed these results.

Table 9: SSME LH2 Heat Exchanger Exit Temperatures (Original Results)

	P_{in} (psia)	*T_{in} (R)*	*DP/P*	*m_{dot} (lbm/s)*	*Q_{dot} (BTU/sec)*	*Original PHATCAT T_{out} (R)*	*Ref. 83 T_{out} (R)*	*% error*
Nozzle Cooling	6719.9	106.38	1.074	63.69	88349	500.0	464.0	7.76%
Chamber Cooling	6719.9	106.38	1.341	34.08	45308	494.1	455.0	8.59%

Table 10: SSME LH2 Heat Exchanger Exit Temperatures (New Results)

	P_{in} (psia)	*T_{in} (R)*	*DP/P*	*m_{dot} (lbm/s)*	*Q_{dot} (BTU/sec)*	*New PHATCAT T_{out} (R)*	*Ref. 83 T_{out} (R)*	*% error*
Nozzle Cooling	6719.9	106.38	1.074	63.69	88349	462.4	464.0	0.34%
Chamber Cooling	6719.9	106.38	1.341	34.08	45308	455.4	455.0	0.09%

The calculations of the exit temperatures for the valves and mixers found in the hydrogen fuel lines also showed noticeable error when compared to the results from reference 83. The following tables show that these errors were again eliminated by changing the hydrogen properties in the cold flow fluid properties database.

Table 11: SSME LH2 Valve Exit Temperatures (Original Results)

	P_{in} (psia)	*T_{in} (R)*	*DP/P*	*m_{dot} (lbm/s)*	*Original PHATCAT T_{out} (R)*	*Ref. 83 T_{out} (R)*	*% error*
Main Fuel Valve	7055.95	103.14	1.050	161.42	105.3	106.4	1.03%
Nozzle Cooling Bypass	6719.90	106.38	1.074	63.69	109.3	110.5	1.09%
LH2 Tank Repressurization	3790.71	440.64	74.46	0.77	470.0	453.8	3.57%

Table 12: SSME LH2 Valve Exit Temperatures (New Results)

	P_{in} *(psia)*	T_{in} *(R)*	*DP/P*	m_{dot} *(lbm/s)*	*New PHATCAT* T_{out} *(R)*	*Ref. 83* T_{out} *(R)*	*% error*
Main Fuel Valve	7055.95	103.14	1.050	161.42	106.38	106.4	0.019%
Nozzle Cooling Bypass	6719.90	106.38	1.074	63.69	110.7	110.5	0.18%
LH2 Tank Repressurization	3790.71	440.64	74.46	0.77	454.9	453.8	0.24%

Table 13: SSME LH2 Mixer Exit Temperature (Original Results)

	$P_{in,1}$ *(psia)*	$T_{in,1}$ *(R)*	$m_{dot,1}$ *(lbm/s)*	$P_{in,2}$ *(psia)*	$T_{in,2}$ *(R)*	$m_{dot,2}$ *(lbm/s)*	*Original PHATCAT* T_{out} *(R)*	*Ref. 83* T_{out} *(R)*	*% error*
Nozzle Regenerator Bypass Mixer	6256.9	110.52	63.69	6256.9	464.1	63.69	276.2	293.9	6.0%

Table 14: SSME LH2 Mixer Exit Temperature (New Results)

	$P_{in,1}$ *(psia)*	$T_{in,1}$ *(R)*	$m_{dot,1}$ *(lbm/s)*	$P_{in,2}$ *(psia)*	$T_{in,2}$ *(R)*	$m_{dot,2}$ *(lbm/s)*	*New PHATCAT* T_{out} *(R)*	*Ref. 83* T_{out} *(R)*	*% error*
Nozzle Regenerator Bypass Mixer	6256.9	110.52	63.69	6256.9	464.1	63.69	296.1	293.9	0.8%

The final major powerhead component calculation that needs to be verified is equilibrium combustion analysis. For hydrogen fueled engines, the combustion calculations need to be checked at mixture ratios near 6.0 and at very fuel-rich mixture ratios. Table 15 gives a comparison between the results obtained from PHATCAT and the industry standard equilibrium analysis code, CEA (Ref. 79). The adiabatic combustion temperature and mole fractions were determined for a mixture ratio of 6.0, a chamber pressure of 3000 psia, and fuel and oxidizer inlet temperatures of 536.14 R. These conditions approximate typical main combustion chamber conditions found in the SSME. The results in Table 15 show good agreement between PHATCAT and CEA.

Table 15: Equilibrium Calculation Comparisons (O/F = 6.0, P_c = 3000 psia)

	New PHATCAT	*CEA*
H_2	0.2497	0.2497
O_2	0.0037	0.0036
H_2O	0.6623	0.6633
OH	0.0467	0.0462
H	0.0341	0.0336
O	0.0036	0.0036
T_{ad} (R)	6726.9	6716.6
MW_{mix}	13.438	13.447

The combustion equilibrium calculations were also checked at conditions typical for a fuel-rich preburner, found in a staged-combustion or gas-generator hydrogen fueled engine. Table 16 shows the results for a mixture ratio of 0.8, chamber pressure of 5500 psia, and fuel and oxidizer inlet temperatures of 536.14 R. Again, good agreement is seen between PHATCAT and CEA.

Table 16: Equilibrium Calculation Comparisons (O/F = 0.8, P_c = 5500 psia)

	New PHATCAT	*CEA*
H_2	0.8992	0.8992
O_2	0.0000	0.0000
H_2O	0.1008	0.1008
OH	0.0000	0.0000
H	0.0000	0.0000
O	0.0000	0.0000
T_{ad} (R)	1990.17	1980.05
MW_{mix}	3.629	3.629

Finally, Table 17 shows the results for the performance analysis of the SSME engine. The balancing routine in PHATCAT was used to vary engine mass flow rate until the desired vacuum thrust was achieved. Also, the pump output pressures were varied until the desired pressure in the main combustion chamber was achieved. The performance results produced by the updated version of PHATCAT are very similar to those found in reference 83.

Table 17: SSME Performance Comparison

	New PHATCAT Results	*Ref. 83 Results*	*% error*
Vacuum Thrust (lbf)	511350	511350	0.0%
Mixture Ratio	6.0	6.0	0.0%
Chamber Pressure (psia)	3264.8	3264.8	0.0%
Mass Flow Rate (lbm/s)	1129.2	1127.5	0.2%
Vacuum Isp	452.8	453.5	0.2%
Throat Area (in^2)	81.75	81.38	0.5%
Exit Area (ft^2)	43.88	44.02	0.3%

3.2.1 Hydrocarbon Component Verification

Three new hydrocarbon fuels were added to PHATCAT for this research. However, detailed engine cycle data is only available from literature for RP-1 fueled engines. Therefore, only results for modules using RP-1 will be presented in this section. Reference 83 provides cycle data for the RD-170 engine. This data will be used to determine the performance of the pump, turbine, heat exchanger, mixer and valve analyses. CEA will be used along with reference 84 to provide data for the equilibrium combustion analysis. Overall engine performance values predicted by PHATCAT for the RD-180 are compared to results published in reference 106.

The exit temperatures and powers for the LPFP, HPFP, and LPFT of the RD-170 were calculated using the modular analyses in PHATCAT. The tables below show the comparison of these results to those published in literature. Excellent agreement is seen for all three components.

Table 18: RD-170 RP-1 Pump Exit Temperatures

	P_{in} (psia)	*T_{in} (R)*	*P_{out} (psia)*	*m_{dot} (lbm/s)*	h	*New PHATCAT T_{out} (R)*	*Ref. 83 T_{out} (R)*	*% error*
LPFP	43.51	540.0	435.11	1466.69	.74	541.47	541.08	0.07%
HPFP	435.11	547.38	6392.68	1703.55	.74	569.09	564.48	0.82%

Table 19: RD-170 RP-1 Pump Powers

	P_{in} (psia)	*T_{in} (R)*	*P_{out} (psia)*	*m_{dot} (lbm/s)*	h	*New PHATCAT Power (HP)*	*Ref. 83 Power (HP)*	*% error*
LPFP	43.51	540.0	435.11	1466.69	.74	4056	4034	0.55%
HPFP	435.11	547.38	6392.68	1703.55	.74	70777	71275	0.70%

Table 20: RD-170 RP-1 Turbine Exit Temperature

	P_{in} (psia)	*T_{in} (R)*	*DP/P*	*m_{dot} (lbm/s)*	h	*New PHATCAT T_{out} (R)*	*Ref. 83 T_{out} (R)*	*% error*
LPFT	6392.68	564.48	14.692	236.86	.55	577.04	586.44	1.60%

Table 21: RD-170 RP-1 Turbine Power

	P_{in} (psia)	*T_{in} (R)*	*DP/P*	*m_{dot} (lbm/s)*	h	*New PHATCAT Power (HP)*	*Ref. 83 Power (HP)*	*% error*
LPFT	6392.68	564.48	14.692	236.86	.55	4023	4034	0.27%

The regenerative cooling of the nozzle and chamber of the RD-170 was modeled using the heat exchanger module in PHATCAT. The results for this analysis are shown in Table 22. There is a noticeable, yet tolerable, difference in the heat exchanger exit temperatures. This difference is not entirely unexpected, given the many different ways RP-1 can be represented when trying to determine its thermodynamic properties.

Table 22: RD-170 RP-1 Heat Exchanger Exit Temperatures

	P_{in} (psia)	*T_{in} (R)*	*DP/P*	*m_{dot} (lbm/s)*	*Q_{dot} (BTU/sec)*	*New PHATCAT T_{out} (R)*	*Ref. 83 T_{out} (R)*	*% error*
Nozzle Cooling	5811.6	569.16	1.2	348.44	23716	703.08	728.28	3.46%
Chamber Cooling	4842.9	728.28	1.2	348.44	23716	845.31	885.96	4.59%

The exit temperatures for the valves and mixers encountered on the fuel side of the RD-170's powerhead were modeled in PHATCAT. These results are found in Table 23 and Table 24. Good agreement is seen for all the components tested.

Table 23: RD-170 RP-1 Valve Exit Temperatures

	P_{in} (psia)	*T_{in} (R)*	*DP/P*	*m_{dot} (lbm/s)*	*New PHATCAT T_{out} (R)*	*Ref. 83 T_{out} (R)*	*% error*
Main Fuel Valve	6392.68	564.48	1.100	348.44	568.15	569.16	0.18%
Preburner Fuel Valve	8992.48	572.04	1.100	72.91	577.21	578.70	0.26%

Table 24: RD-170 RP-1 Mixer Exit Temperature

	$P_{in,1}$ (psia)	*$T_{in,1}$ (R)*	*$m_{dot,1}$ (lbm/s)*	*$P_{in,2}$ (psia)*	*$T_{in,2}$ (R)*	*$m_{dot,2}$ (lbm/s)*	*New PHATCAT T_{out} (R)*	*Ref. 83 T_{out} (R)*	*% error*
Fuel Recirculation Flow Mixer	435.1	541.08	1466.7	435.1	586.4	236.86	547.50	547.4	0.02%

In order to analyze all the engine cycles needed for this research, the equilibrium combustion model for RP-1 must be able to handle fuel-rich, fuel-lean and stoichiometric mixtures. Table 25 provides the results for RP-1/LOX combustion at an O/F of 2.72 and a P_c of 3700 psia. These conditions are very similar to those found in the main combustion chamber of the RD-180. The results of this equilibrium combustion analysis show good agreement with those obtained from CEA. The fuel-lean preburner results given in Table 26 are for the LOX-rich preburner found on the RD-170. Table 27 shows results for a fuel-rich gas generator found in reference 84. The results for both these cases are also good.

Table 25: Equilibrium Calculation Comparisons (O/F = 2.72, P_c = 3700 psia)

	New PHATCAT	*CEA*
H_2	0.0681	0.0673
O_2	0.0255	0.0255
H_2O	0.3365	0.3380
OH	0.0682	0.0679
H	0.0242	0.0237
O	0.0135	0.0134
CO_2	0.1689	0.1682
CO	0.2951	0.2961
T_{ad} (R)	7116.58	7107.25
MW_{mix}	24.117	24.131

Table 26: Equilibrium Calculation Comparisons (Fuel-Lean Preburner)

	P_c (psia)	*T_{fuel} (R)*	*T_{oxid} (R)*	*O/F*	*New PHATCAT T_{ad} (R)*	*CEA T_{ad} (R)*	*% error*
Fuel-Lean Preburner	7400	536.4	220.0	52.31	1687.96	1682.95	0.30%

Table 27: Equilibrium Calculation Comparisons (Fuel-Rich GG)

	P_c (psia)	*T_{fuel} (R)*	*T_{oxid} (R)*	*O/F*	*New PHATCAT T_{ad} (R)*	*Ref. 84 T_{ad} (R)*	*% error*
Fuel-Rich Gas Generator	612.1	536.4	220.0	0.342	1663.6	1659.7	0.23%

As a last test, the actual performance of the RD-180 engine was compared to the performance calculation obtained from PHATCAT (Table 28). In order to get the required results, the mass flow rate of the engine was adjusted until the desired vacuum thrust was achieved, and the pump exit pressures were set at values that gave the desired chamber pressure. The values obtained by PHATCAT are similar to the published results for the RD-180. There is a slight over-prediction of specific impulse, but this can be attributed to not having detailed information about the engine cycle of the RD-180. Without this detailed information, several engine parameters had to be determined from other sources. These included pump and turbine efficiencies, regenerative heat flow rates, flow split percentages, and other cycle parameters. The differences between the values used and the actual values for many of these parameters could easily account for the slight performance difference.

Table 28: RD-180 Performance Comparison

	New PHATCAT Results	*Ref. 106 Results*	*% Error*
Vacuum Thrust (lbf)	933400	933400	0.0%
Mixture Ratio	2.72	2.72	0.0%
Chamber Pressure (psia)	3734.0	3734.0	0.0%
Mass Flow Rate (lbm/s)	2719.4	2763.2	1.6%
Vacuum Isp	343.2	337.8	1.6%
Throat Area (in^2)	133.2	135.1	1.4%
Exit Area (ft^2)	34.1	34.6	1.4%

3.3 Weight and Cost Models

PHATCAT does not supply engine weight or cost as one of its outputs. However, these are important metrics that need to be considered when trying to optimize an engine design for a particular application. Therefore, a way to determine engine cost and weight had to be found for this research. Two existing engine cost models for liquid rocket engines are included in the NASA/Air Force Cost Model 99 (NAFCOM99) [107]. The top-level, summary cost model was used for this research. This model employs historically based equations to predict engine cost. The NAFCOM99 model gives both the development and production costs of the engine. The production costs are output as an average production cost per engine and as a theoretical first unit (TFU) cost. The TFU is the cost required to produce the first engine off the assembly line. The average production cost is less than the TFU because it accounts for cost reductions from the learning curve effects associated with producing many engines. The inputs into the cost model include the engine's thermodynamic cycle, vacuum thrust, chamber pressure, design and manufacturing inputs, and engine production rate. The engine cycle is input as a complexity factor that influences the engine's costs. Development cost complexity factors for several engine cycles are shown in Table 29.

Table 29: Development Cost Complexity Factors

Engine Cycle	*Development Cost Complexity Factor*
Staged-Combustion	1.2
Gas Generator	0.8
Expander	0.7
Combustion Tap-off	0.6

No existing engine weight model was available, so a new engine weight model was developed using historical engine data. Since PHATCAT provides information about the individual components of the engine's powerhead, an attempt was made at forming a component based weight model. However, there is not enough data on individual engine component weights in open literature. Most data of this nature is proprietary. Therefore, a top-level engine weight model was made. The top-level weight model was formed using thrust-to-weight engine data for 35 different liquid rocket engines. These engines were categorized by their basic engine cycle and fuel type. At least six data points were available for each engine grouping. The engine groups that were included are LH2 and RP-1 staged-combustion engines, LH2 and RP-1 gas generator engines, and LH2 expander cycle engines. A linear fit of engine weight versus total propellant mass flow rate was made for each of the five engine classes. The linear fits for each engine class, which are shown in the figures below, form the basis of the weight model.

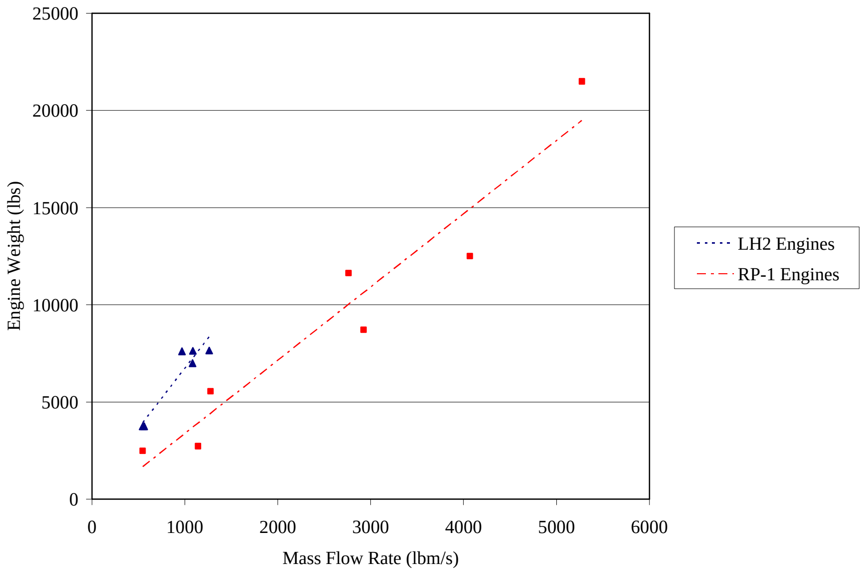


Figure 19: Engine Weight vs. M_{dot} (Staged-Combustion Cycle)

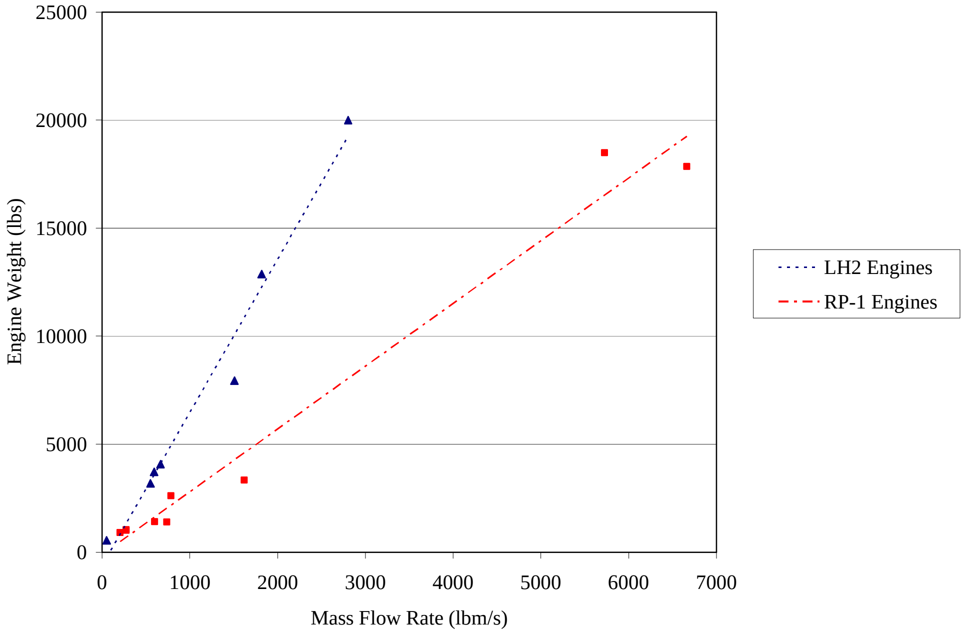


Figure 20: Engine Weight vs. M_{dot} (Gas Generator Cycle)

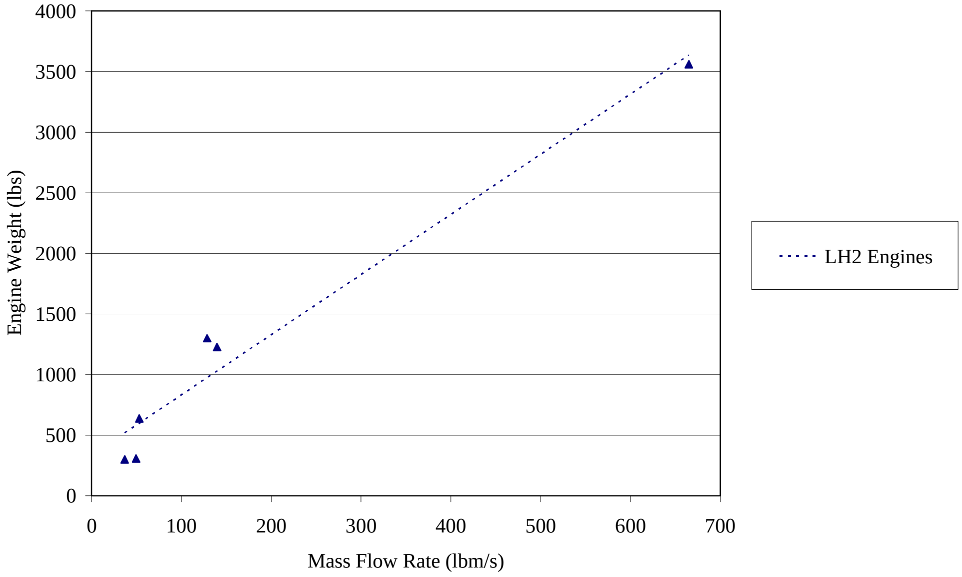


Figure 21: Engine Weight vs. M_{dot} (Expander Cycle)

The chamber pressure of the engine also has a significant effect on engine weight. To account for this, a chamber pressure adjustment is applied to the linear fit. This chamber pressure trend was taken from reference 84 and is shown in Figure 22. For a fixed engine thrust, the combustion chamber weight decreases with chamber pressure, because a smaller chamber size is required. However, turbopump weight increases with chamber pressure, because more power is required in order to pump the same mass flow up to the higher pressure. These components show the most weight sensitivity with chamber pressure, and their influences combine to give the total engine weight versus chamber pressure trend. For typical pump-fed engines, engine weight reaches a minimum for chamber pressures between 2000 and 3000 psia [84].

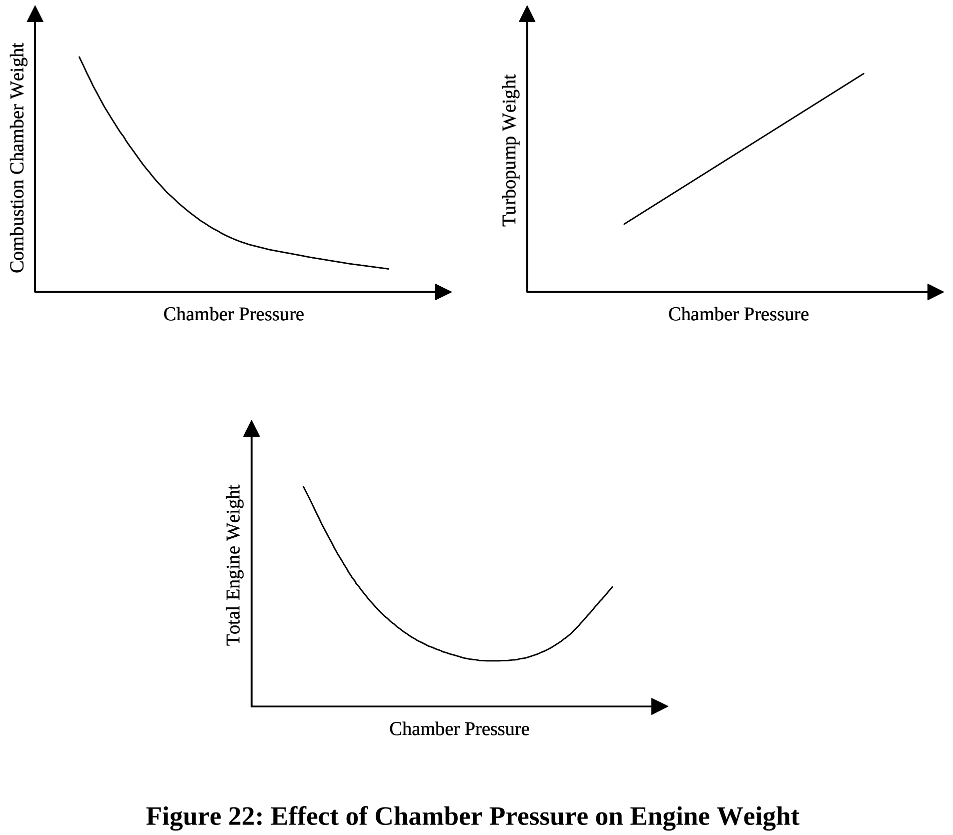


Figure 22: Effect of Chamber Pressure on Engine Weight

For each fuel type and engine cycle, the chamber pressure value giving the minimum engine weight was set, so that the error for the weight predictions of the engines used to fit the model was minimized. This new weight model receives as inputs the engine cycle, fuel choice, total propellant mass flow rate (lbm/s), chamber pressure (psia), and a weight multiplication factor. The weight multiplication factor is used to get weight for propane and methane fueled engines and for cycles that differ from the baseline cycles used to form the model. For instance, a dual-preburner dual turbine staged-combustion engine cycle is the baseline cycle used to form the staged combustion weight model. Therefore, a single-preburner, single turbine staged-combustion engine might have a multiplication factor of 90%, to represent that fact that this engine would be lighter than a similar dual-preburner dual turbine design.

CHAPTER 4

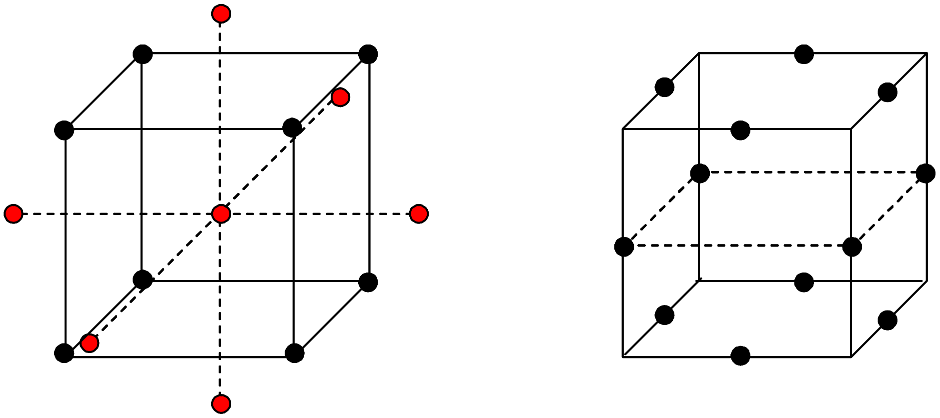
RESPONSE SURFACE METHODOLOGY

4.1 Response Surface Equations

A key component in the new optimization procedure developed for this research is the use of response surface equations (RSE's) to help form the lower bounds of the continuous design space for each discrete variable setting [108,109]. Response surface equations are formed by performing a least squares fit on data that represents the design space of interest. For this research, 2nd order RSE's are used and the number of terms in the equation depends on the number of design variables under consideration. Equation 30 shows the general form of the RSE. The x-variables represent the design variables and the β terms are coefficients determined by the least squares fit.

$$RSE = b_o + \sum_{i=1}^{n} b_i x_i + \sum_{i=1}^{n} b_i x_i^2 + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} b_{ij} x_i x_j \tag{30}$$

A separate RSE must be formed for each possible output of the engine code that might be used as an objective function or constraint in the optimization process. In order to form the RSE, data must be collected from the design space. This is done by running the desired engine design code at specific input variable settings. These settings are determined by the bounds on the input variables and the type of response surface design that is chosen. In order to fit a quadratic model of the design space, data must be available at three distinct settings of each input variable. There are several common response surface designs that are typically used to fit a quadratic RSE. These are a three-level full or fractional factorial, a central composite, and a Box-Behnken design [110]. Each of these designs has features that make them useful in specific cases. The full factorial design requires the most data points but also allows for the most accurate approximation. The central composite design (CCD) combines a two-level fractional (or full) factorial with center and star points to provide curvature to the space, while the Box-Behnken design provides data without requiring that the input values be evaluated at the endpoints of their defined range (Figure 23). A summary of the number of design points needed for each of these designs is provided in Table 30.



Central Composite Design

Box-Behnken Design

Figure 23: Response Surface Designs

Table 30: Design of Experiments

Response Surface Design	*Number of Experiments* *(n = number input variables)*
Central Composite Design (2-Level Full Factorial)	$2^n + 2n + 1$
Box-Behnken Design	½ (n + 1)(n + 2)
3-Level Full Factorial	3^n

For this application a full factorial central composite design was chosen. This represents a good compromise between a 3-level full factorial and a Box-Behnken design. The star points are set on the face of the CCD cube because for each specific engine cycle, bounds on the input variables were found for which the engine power balance would converge. Outside of these bounds, convergence is not guaranteed and therefore the on-face design was chosen so the input variables are never set to values outside their applicable range.

4.2 RSE Example

One particular engine that was modeled for this research was the Vulcain LH2/LOX (liquid hydrogen/liquid oxygen) gas generator engine. Table 31 shows the bounds used for the input variables for this engine. An RSE was generated for each output variable that was used as either the objective or as a constraint in the optimization process. The RSE coefficients generated for the Vulcain vacuum I_{sp} are given in Table 32.

Table 31: Vulcain Input Variable Bounds

Input Variable	*Lower Bound*	*Upper Bound*
T_{vac} (Vacuum Thrust)	80000 lbs	750000 lbs
ε (Area Ratio)	15	125
O/F (Mixture Ratio)	5.0	7.0
P_c (Chamber Pressure)	500 psia	3500 psia

The accuracy of the values predicted by the response surface, compared to the data used to fit the equation, can be quantified by the RSquared value (R^2). The value is computed using the calculated sum of squares of the error and the total sum of squares of the mean (Equation 31) [111]. The value of R^2 is between 0 – 1, with 1 representing a perfect fit of the sampled points. The R^2 value for the Vulcain's vacuum I_{sp} RSE is .997.

Table 32: Vulcain Vacuum I_{sp} RSE Coefficients

Response Surface Term	*Coefficients*
Intercept	427.610051
T_{vac}	6.209670
ε	18.782296
O/F	-14.623416
P_c	1.757800
T_{vac} * ε	0.640364
T_{vac} * O/F	0.247773
T_{vac} * P_c	3.708853
ε * O/F	-0.439573
ε * P_c	0.034244
O/F * P_c	0.450687
T_{vac} * T_{vac}	-5.288926
ε * ε	-10.462974
O/F * O/F	-0.658320
P_c * P_c	-2.350497

$$R^2 = 1 - \frac{SSE}{SSM} \tag{31}$$

where: $SSE = \sum_{i=1}^{n} \left(predicted\, value_i - actual\, value_i \right)^2$

$$SSM = \sum_{i=1}^{n} \left(predicted\, value_i - mean \right)^2$$

The R^2 value is calculated by comparing the fit of the RSE to the data points used for the fit. A good test of a response surface is to see how it performs away from these specific points. To demonstrate this for the Vulcain example, a sweep of each input variable was performed and the vacuum I_{sp} value predicted by the RSE was compared to the actual vacuum I_{sp} obtained from the rocket engine code. These one dimensional sweeps were used since it isn't possible to show a graph of the 4-D response surface. The settings for the input variables not currently being swept are set to match the values of the actual Vulcain engine. The default input values used were a vacuum thrust of 256 klbs, an expansion ratio of 45, a mixture ratio of 5.3, and a chamber pressure of 1595 psia. These settings do not correspond to any of the data points used to fit the RSE with the central composite design. Graphical comparisons of the predicted versus actual vacuum I_{sp} for a swept of each of the input variables are shown in Figures 24 - 27.

While these graphs show that the response surface equations capture the general trend in vacuum I_{sp}, they do not predict the actual value with great accuracy. This is not what one might expect with such a high R^2 value, but these graphs are produced at a point in the design space not near those points used to fit the surface. This illustrates the problem with using RSE's directly to predict the performance of each engine being

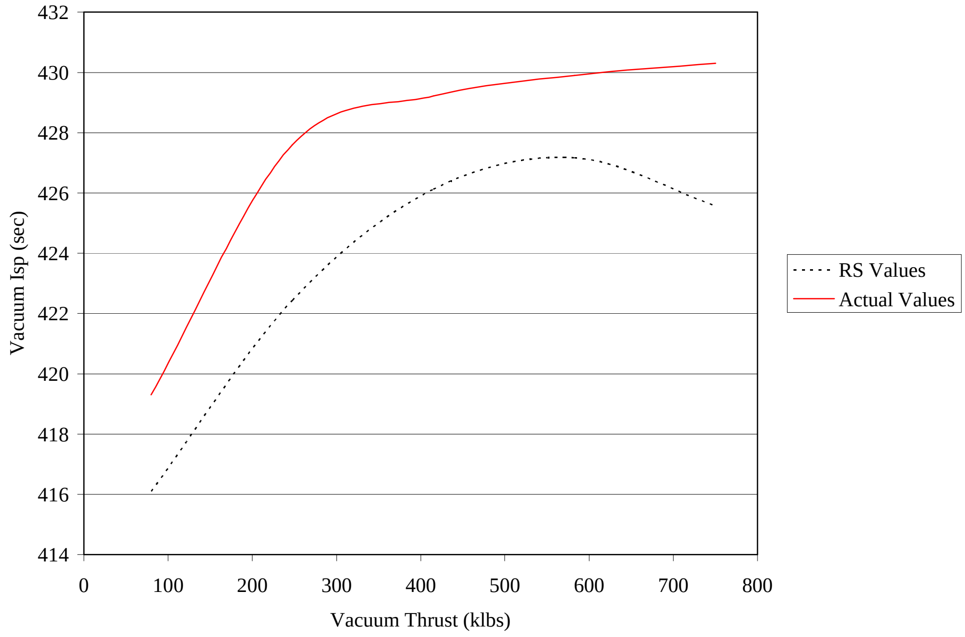


Figure 24: Vacuum I_{sp} vs. T_{vac}

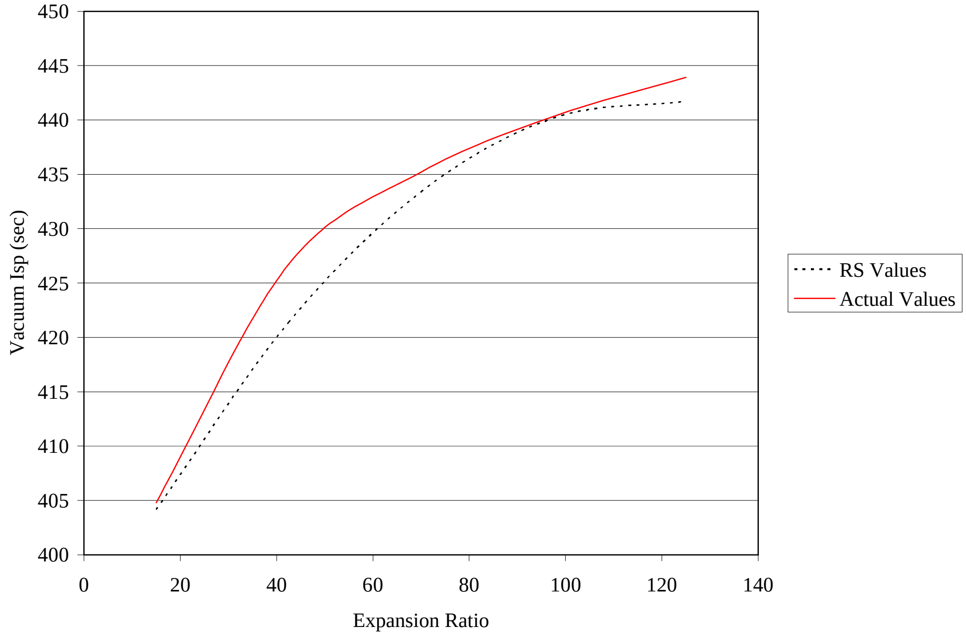


Figure 25: Vacuum I_{sp} vs. e

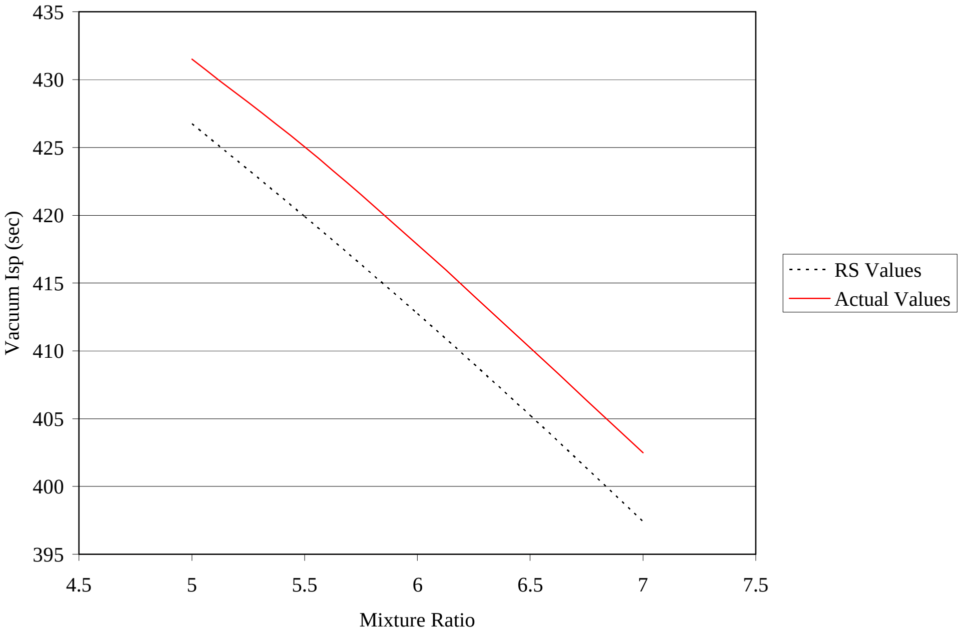


Figure 26: Vacuum I_{sp} vs. O/F

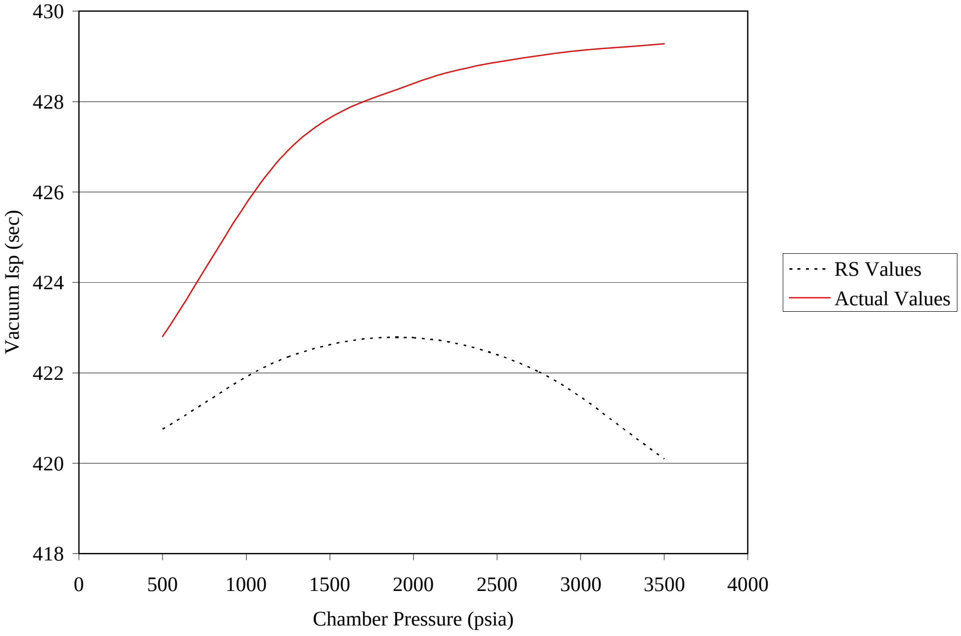


Figure 27: Vacuum I_{sp} vs. P_c

modeled. Since the RSE’s do not necessarily provide accurate results, error in the optimization procedure could occur. These errors could be quite serious, possibly causing the optimal engine design found using the RSE’s to actually be infeasible in the true design space. Even with their associated inaccuracies, these response surfaces will be useful in this research. They provide a good starting point for the lower bounds of the continuous space that will be used in the new optimization procedure to be outlined next.

CHAPTER 5

OPTIMIZATION METHOD

5.1 Applicability of Current MINLP Methods

Several MINLP solution procedures have been outlined in Chapter 2. However, none of these methods meet the requirements for this project. All of these standard solution methods require that the discrete variables be relaxed (take on non-discrete values) at some point during the solution process. The branch and bound algorithm requires relaxation to solve the nonlinear optimization problem at each node in the tree, while the GBD and GOA algorithms require the relaxation of the discrete variables in order to solve their respective lower bounding problems.

These algorithms have been developed to solve problems where the objective and constraints functions are easily represented as analytical equations. Therefore, it is of no concern to require that the discrete variables be relaxed. However, when the values of the objective and constraints are being supplied by an engineering code, the proposition of relaxing the discrete variables becomes more daunting. For example, if y = 1 represents the choice of a staged-combustion cycle and y = 2 represents the choice of an expander cycle, it is not obvious how y = 1.3 would be evaluated. An attempt was made at relaxing the discrete variables by going deeper into the code and finding the continuous variables that the discrete variables set and then using these continuous variables to evaluate the relaxed discrete choices, but this leads to poorly behaved nonlinear programming (NLP) problems. Table 33 shows the features of the existing MINLP solution methods discussed in Chapter 2 and illustrates why none are appropriate for this research.

Table 33: Summary of MINLP Solution Methods

Method	*Requires Relaxing Discrete Variables*	*Only Solves Strictly Convex Problems*
Branch & Bound	Yes	Yes
Generalized Bender's Decomposition	Yes	Yes
Generalized Outer Approximation	Yes	Yes
Branch & Reduce	Yes	No
Nonconvex Outer Approximation	Yes	No
SMIN-αBB	Yes	No
GMIN-αBB	Yes	No
Desired Optimization Scheme	**No**	**No**

5.2 Details of New Optimization Method

All of the major MINLP optimization approaches are ill-suited for this research. Therefore, a new method had to be conceived. Since the discrete input variables can not be relaxed, the first obvious optimization scheme is that for every discrete variable combination, solve the resulting continuous NLP, and take the minimum value of all the NLPs as the optimal solution. This solution procedure requires an extremely large number of function calls to find the optimum. Therefore, it is not an acceptable method. However, this optimization procedure will be used to determine the “correct” optimal solution and the baseline optimization time for each engine design problem investigated. These baseline solutions will be used to determine the success of the newly developed optimization approach.

Some of the NLPs associated with certain discrete variable combinations will have to be solved. This is the only way to get the optimal solution. The goal then is to minimize the number of discrete variable combinations that must to be investigated. This is accomplished by first solving an approximation of the continuous space for each discrete variable combination. This approximation is fast running and is designed to provide a solution for each discrete variable setting that is actually a lower bound of the true solution (a better solution). In order to ensure that the solution of the approximation is a lower bound, the model of the continuous space must have a feasible region that is no smaller than the feasible region of the true continuous space. After the lower bounds of all the discrete variable combinations are found, the combination having the lowest feasible lower bound is determined. Then, the solution of the true continuous space of this discrete variable choice is found using a standard nonlinear optimization technique and the actual engine code. The solution of this NLP becomes the current upper bound of the final solution. Any remaining discrete variable combinations with feasible lower bounds greater than the current upper bound can be eliminated from further consideration. This is because the true solution of the continuous space for each discrete variable combination must be greater than or equal to the lower bound provided by the approximate solution. Now, the discrete variable combination providing the next smallest lower bound is investigated. If its true solution provides a better optimum than the current upper bound, the upper bound is replaced. The process continues until none of the remaining discrete variable combinations have feasible lower bounds less than the current best upper bound. This final upper bound is the optimal solution. Figure 28 shows a simple flowchart outlining the procedure.

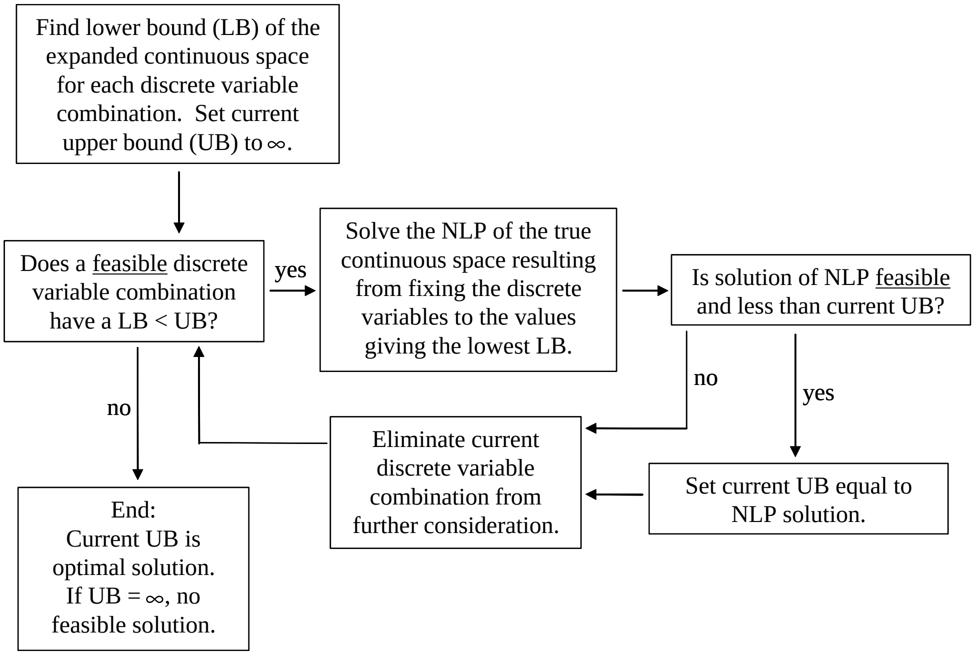

Find lower bound (LB) of the expanded continuous space for each discrete variable combination. Set current upper bound (UB) to ∞.
Does a feasible discrete variable combination have a LB < UB?
yes
Solve the NLP of the true continuous space resulting from fixing the discrete variables to the values giving the lowest LB.
Is solution of NLP feasible and less than current UB?
no
yes
no
Eliminate current discrete variable combination from further consideration.
Set current UB equal to NLP solution.
End:
Current UB is optimal solution.
If UB = ∞, no feasible solution.

Figure 28: Flowchart for New Optimization Procedure

Now the issue of how to form the lower bounding continuous space must be addressed. This space is formed by making use of convex underestimation techniques developed for global optimization problems. If the underestimator of the objective function is used along with underestimated constraints values, the optimal solution of this space will be a lower bound of the true objective function's solution. The underestimators of the constraints have the effect of expanding the feasible space. Therefore, if a discrete variable combination doesn't yield a feasible solution when the continuous space is expanded, it will not have a feasible solution in the more restrictive original space and therefore, that combination can be eliminated from consideration.

The new method that has been developed allows for the calculation of the desired lower bounds without any calls to the rocket engine code during the optimization process. All the needed function calls can be done beforehand and stored in an appropriate format. This is accomplished by making response surfaces of the continuous variable design space for each discrete variable setting. These response surfaces are then transformed into lower bounds of the continuous space using underestimation techniques. Specifically for this application, response surfaces are formed for every discrete variable combination that is included in the optimization problem. The engine code input variables used to form these response surfaces are vacuum thrust, nozzle expansion ratio, engine mixture ratio, and the engine chamber pressure. Response surfaces are found for all the relevant engine outputs. These include vacuum and sea-level I_{sp}, sea-level thrust, engine vacuum thrust-to-weight (T/W), the maximum turbine inlet temperature, and the engine's development costs.

The response surfaces are formed by using a central composite design of experiments (DOE). The CCD for four input variables requires 25 function evaluations (Table 34). The bounds of each input variable are predetermined for every discrete variable combination. The values for the input variables in the DOE are -1, 0, or 1. These represent the respective variable's lower bound, the mid-point between its bounds and the upper bound.

Table 34: Central Composite Design of Experiments

Vacuum Thrust	*Expansion Ratio*	*Mixture Ratio*	*Chamber Pressure*
-1	-1	-1	-1
-1	-1	-1	1
-1	-1	1	-1
-1	-1	1	1
-1	0	0	0
-1	1	-1	-1
-1	1	-1	1
-1	1	1	-1
-1	1	1	1
0	-1	0	0
0	0	-1	0
0	0	0	-1
0	0	0	0
0	0	0	1
0	0	1	0
0	1	0	0
1	-1	-1	-1
1	-1	-1	1
1	-1	1	-1
1	-1	1	1
1	0	0	0
1	1	-1	-1
1	1	-1	1
1	1	1	-1
1	1	1	1

As part of the preprocessing, the engine design code is run at the inputs specified in the DOE for each discrete variable combination to be included in the analysis. Results of these runs are stored in a database file that allows for their retrieval during the optimization process. When response surfaces of a specific engine are needed, this database file is queried and the outputs for the cases in the DOE are used to perform the least squares fit which determines the RSE for each output variable.

Now that the RSE equations for each variable of interest are known, they must be turned into lower bounds of the outputs they are trying to predict. As mentioned earlier, this will be accomplished using convex underestimation techniques. The convex underestimator for a general function presented in Reference 74 will be used to form the lower bounds (Equation 32).

$$L(\bar{x}) = f(\bar{x}) + \sum_{i=1}^{n} a_i \left(x_i^L - x_i\right)\left(x_i^U - x_i\right) \tag{32}$$

The $f(\bar{x})$ term in Equation 32 is replaced with the response surface equation generated for the current output variable of interest and $L(\bar{x})$ is the calculated lower bound. For the current application, n is equal to four in the summation term, since four input variables are used to fit the RSE. Also, the lower and upper bounds for the input variables in Equation 32 can be replaced with -1 and 1 respectively. Equation 33 reflects these adjustments.

$$L(\bar{x}) = b_o + \sum_{i=1}^{4} b_i x_i + \sum_{i=1}^{4} b_i x_i^2 + \sum_{i=1}^{3} \sum_{j=i+1}^{4} b_{ij} x_i x_j + \sum_{i=1}^{4} a_i (-1 - x_i)(1 - x_i) \qquad (33)$$

Another change must be made to the lower bounding equation for it to be correct for this research. The convex underestimating terms (the α terms) in Equation 33 have no effect on the lower bounding value at points in the design space where all the input variables are at either their lower or upper bounds, because one of the terms inside the parentheses will be zero. This is not a problem, if $f(\bar{x})$ was used in Equation 33, because $f(\bar{x})$ provides the true function value for every $\bar{x}$. However, for this research, $f(\bar{x})$ is now replaced with an approximation (the RSE, given the symbol $\tilde{f}(\bar{x})$). Therefore, the lower bound equation will have to account for any discrepancies between $\tilde{f}(\bar{x})$ and the true function values at design points where all the input variables are at their bounds. To do this, an additional delta (δ) term is subtracted from the lower bounding equation (Equation 34).

$$L(\bar{x}) = b_o + \sum_{i=1}^{4} b_i x_i + \sum_{i=1}^{4} b_i x_i^2 + \sum_{i=1}^{3} \sum_{j=i+1}^{4} b_{ij} x_i x_j + \sum_{i=1}^{4} a_i (-1 - x_i)(1 - x_i) - d \qquad (34)$$

This δ term is the maximum positive difference found by subtracting the true function value from the response surface value for all the points in the CCD where every input value is at a bound. This term shifts $\tilde{f}(\bar{x})$ down, so that at the endpoints, the value predicted by $\tilde{f}(\bar{x}) - d$ is less than or equal to the true value. A graphical representation of this shift for a one-dimensional response surface is shown in Figure 29.

In this example the original response surface over predicts the true objective function value for some points in the design space, including one of the endpoints. The δ term is used to shift the response surface so that at the endpoints of the design space, $\tilde{f}(x)-d$ predicts values less than or equal to the true function value.

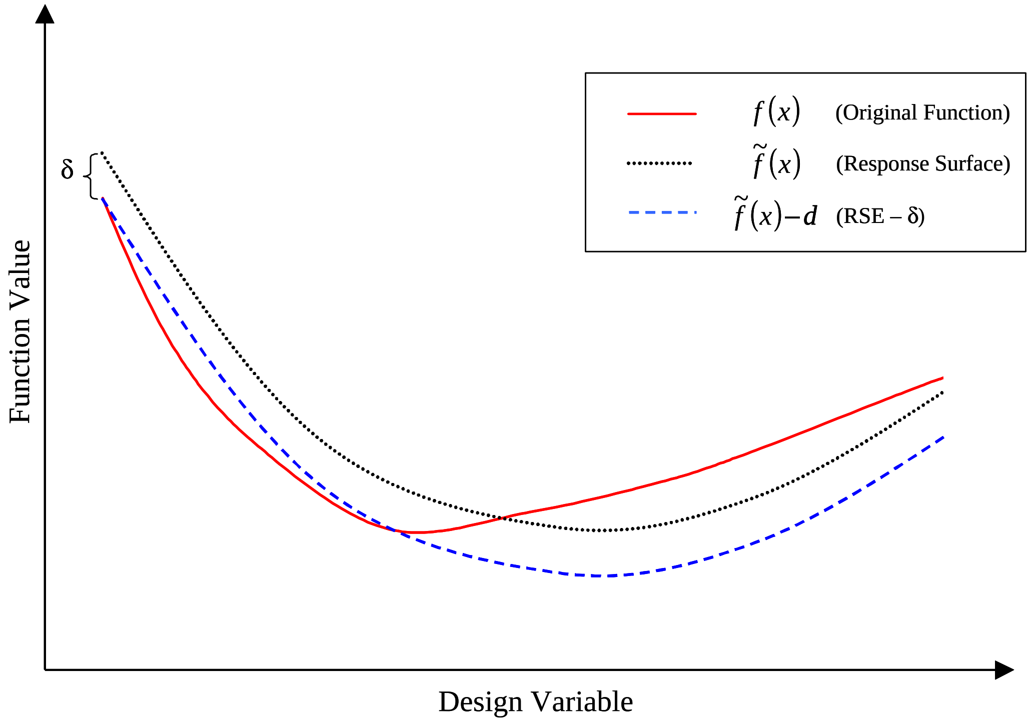


Figure 29: Shifting RSE by d

Even after $\tilde{f}(\bar{x})$ is shifted by the δ term, it is still possible that at some points in the design space it will over-predict the true function value. This over prediction is corrected by the convex underestimating terms. The values of the α_i's in the underestimation terms need to be set appropriately in order to make $L(\bar{x})$ a lower bound of the entire design space. Figure 30 shows the effect the addition of the convex underestimating terms have on $\tilde{f}(x)-d$. At the endpoints, the lower bound equals the value of $\tilde{f}(x)-d$; this is again because the convex underestimating terms do not have an effect at points in the design space where all the input values are at an endpoint. Away from the endpoints, Figure 30 shows the effect of the underestimators. When the appropriate α_i's are found, $L(\bar{x})$ provides values that are equal to or below the values of the original function throughout the entire design space.

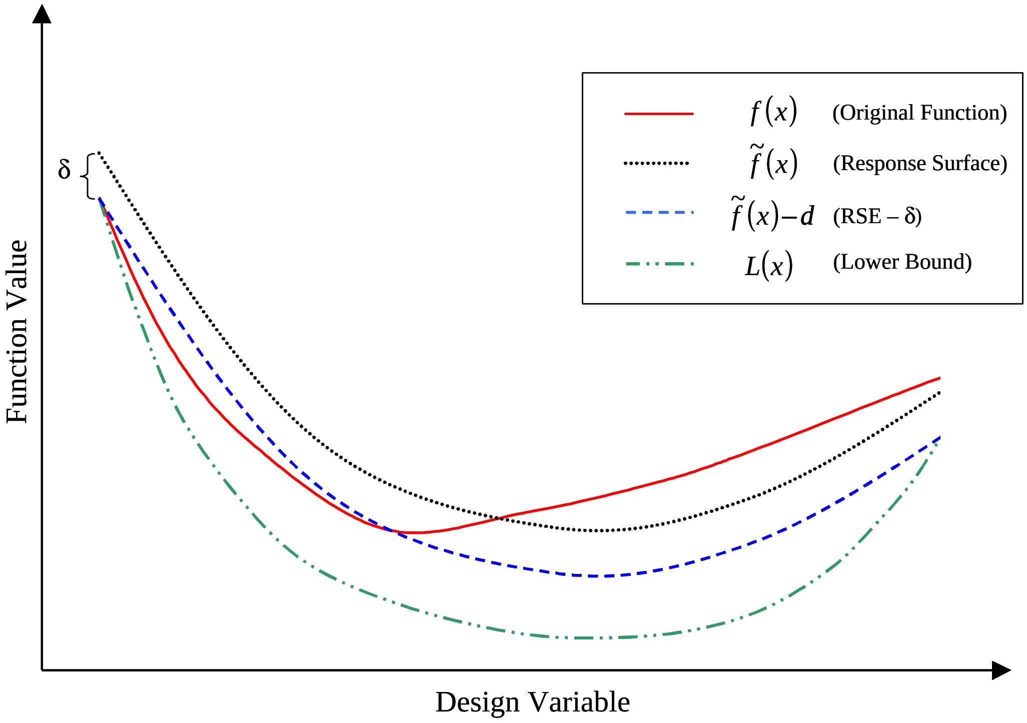


Figure 30: Formation of Lower Bound

A method for the determination of the appropriate values for the α_i's in the lower bounding equation had to be developed for this research. This proves to be an interesting problem because information about the design space is needed in order to determine if the values provided by the lower bounding equations are in fact lower bounds. The convex underestimating equation used for this research was developed for applications where a closed form analytical equation is used to determine the function value ($f(\bar{x})$). This has many advantages, including being able to easily determine the original value of the function anywhere in the design space. Now that these underestimating techniques are being applied to an engineering code, the ability to quantify the entire design space before the optimization problem begins is not possible. Therefore, a process had to be developed that allows for the forming of the lower bounds with only limited knowledge of the design space. In order to form the response surface equations for a specific engine choice, the design code was run at the 25 unique input settings specified in the central composite design of experiments (Table 34). This provides some knowledge of the true design space and the lower bound can be compared with these points. However, these points only provide information about the center and edges of the design space. Additional information is needed about the interior to be able to determine the proper underestimation terms. The desire for information about the entire design space has to be balanced by the time considerations involved with evaluating the engine design code at many distinct input settings. This balance led to the decision to also run the engine code at an additional 2^n points, where n is the number of input variables. These points were placed in the design space in the areas covered the least by the original CCD. The input values for these points were the full-factorial combination of the input variables being set at either a normalized value of -0.5 or 0.5. Again, bounds are established for each input variable and $\tilde{f}(\bar{x})$ is fit using normalized versions of these bounds where -1 represents a lower bound and 1 represents an upper bound. Therefore, -0.5 represents a point one-quarter of the way between the lower and upper bound, and 0.5 represents a point that is three-quarters of the way between the two bounds. Figure 31 shows the points that would be used to form $\tilde{f}(\bar{x})$ (the RSE) and check the lower bounding equation for a 2-D design.

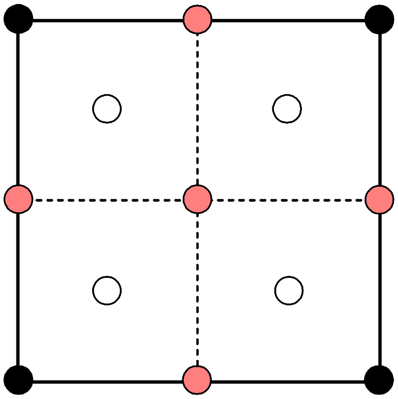


Figure 31: Points Used to Form Lower Bound (2-D)

Table 35: Additional Points Used to Establish Lower Bounds

Vacuum Thrust	*Expansion Ratio*	*Mixture Ratio*	*Chamber Pressure*
-0.5	-0.5	-0.5	-0.5
-0.5	-0.5	-0.5	0.5
-0.5	-0.5	0.5	-0.5
-0.5	-0.5	0.5	0.5
-0.5	0.5	-0.5	-0.5
-0.5	0.5	-0.5	0.5
-0.5	0.5	0.5	-0.5
-0.5	0.5	0.5	0.5
0.5	-0.5	-0.5	-0.5
0.5	-0.5	-0.5	0.5
0.5	-0.5	0.5	-0.5
0.5	-0.5	0.5	0.5
0.5	0.5	-0.5	-0.5
0.5	0.5	-0.5	0.5
0.5	0.5	0.5	-0.5
0.5	0.5	0.5	0.5

The corner points are the full factorial portion of the central composite design, the points on the dotted lines are the center and on-face star points, and the open circles represent the ½ points (+/- 0.5). For the current work, which includes four input variables, Table 35 lists the additional points run for each engine choice in order to establish the lower bounding equations. For *n* continuous variables, the number of total design points that must be run to determine the lower bounding equations is $2^n + 2^n + 2n + 1$ (includes CCD and ½ points). The output values obtained from the rocket engine code for the points in Table 35 are stored in the same results database as the outputs from the points used to fit the RSE. Therefore, these ½ points can also be run before the engine optimization process.

When a particular engine cycle is being optimized, the first step in forming the lower bounds of its continuous space will be to access the results database and fit RSE's ($\tilde{f}(\bar{x})$) to the output variables of interest. Then, the δ term for each $\tilde{f}(\bar{x})$ is determined by finding the maximum positive difference between the value predicted by $\tilde{f}(\bar{x})$ and the value found in the results database for the points at which all the input variables are at either their upper or lower bound. Now the values for the α coefficients in the lower bounding equation can be found. These are set by comparing the lower bounding value and true function value of each point in Table 34 and Table 35, except those points where all the input variables are at an endpoint. Each lower bounding equation will have an α value for each continuous input variable. For the final application of the new optimization method, four continuous input variables are used, so each lower bounding equation will have four α terms. The α values are increased until the lower bound predictions for all non-bound points are less than the true function values stored in the response database. This guarantees that the lower bound equation provides a lower bound of the design space only at the specific points that were analyzed. Therefore, a margin is then chosen where the difference between the true function value and the values obtain by the lower bound equation must be greater than this margin for all the points used to find the α_i's. The larger this margin is made, the more likely the lower bound equation will be valid throughout the entire design space. However, the optimization method will also be less efficient because the lower bound equation will most likely be greatly under-predicting the real function values. This will cause more infeasible designs in the true design space to be feasible in the lower bounding space and also cause less designs to be eliminated by the upper bounds found during the optimization routine. However, if the margin is set to zero, there is the chance that the lower bound equations will not be valid through the design space. This is especially true if the design space is not well behaved, such as in multi-modal spaces or spaces in which the value of the global minimum is significantly less than values obtained in the majority of the design space. The value chosen for the margin is therefore problem dependent.

For both the proof-of-concept and the final application, the design space for both rocket engine codes proved to be relatively well behaved. Therefore, a smaller value for the margin could be used without sacrificing the lower bounding properties of the approximation. For each application, the final margin value that was decided on was 1% of the average value for a particular output over all the cases in the results database. So if the average vacuum I_{sp} for a certain engine, over all the points in the CCD and the ½ points, was 300 seconds, the lower bounding equation for vacuum I_{sp} would have to give values that were at a minimum of 3 seconds lower than the true values.

For the optimization problems solved for this research, the proper values for each α term were determined using a nonlinear optimization technique, specifically method of feasible directions. The objective function of the NLP was to minimize the sum of all the α terms subject to the constraint that at the ½ points and the non-bound points in the CCD the lower bounding equations under predict the true values by at least the margin percentage. The procedure is designed to find the lower bounding equations that best approximate the true design space.

Recall the graphs comparing the true value of the Vulcain's vacuum I_{sp} to that predicted by the response surface equation (pages 88 and 89). These graphs are repeated below except now they show the negative of vacuum I_{sp} and the lower bounds on negative I_{sp} predicted using the method outlined above. The negative of I_{sp} is used because specific impulse is commonly an objective function that is maximized. In the optimization problem, negative I_{sp} is minimized, instead of positive I_{sp} being maximized, because the standard form of optimization problems require that the objective function be minimized. As these graphs show, the response surface equation has now been transformed into a valid lower bound of negative vacuum I_{sp} over the entire design space. This lower bound is used together with the lower bounds of the other output variables to form the lower bounding space used in the optimization process.

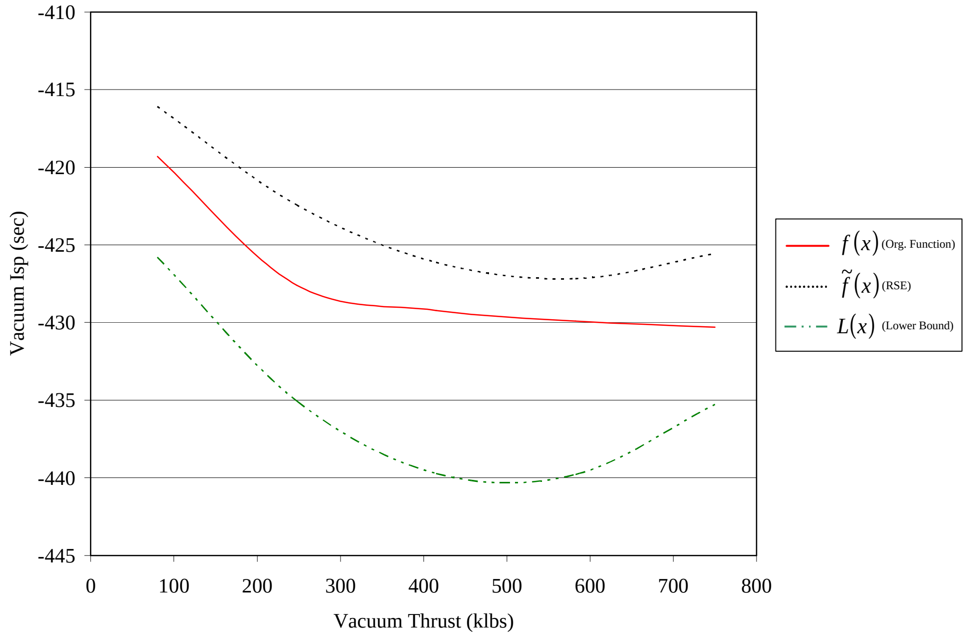


Figure 32: Vacuum I_{sp} vs. Tvac (Lower Bound)

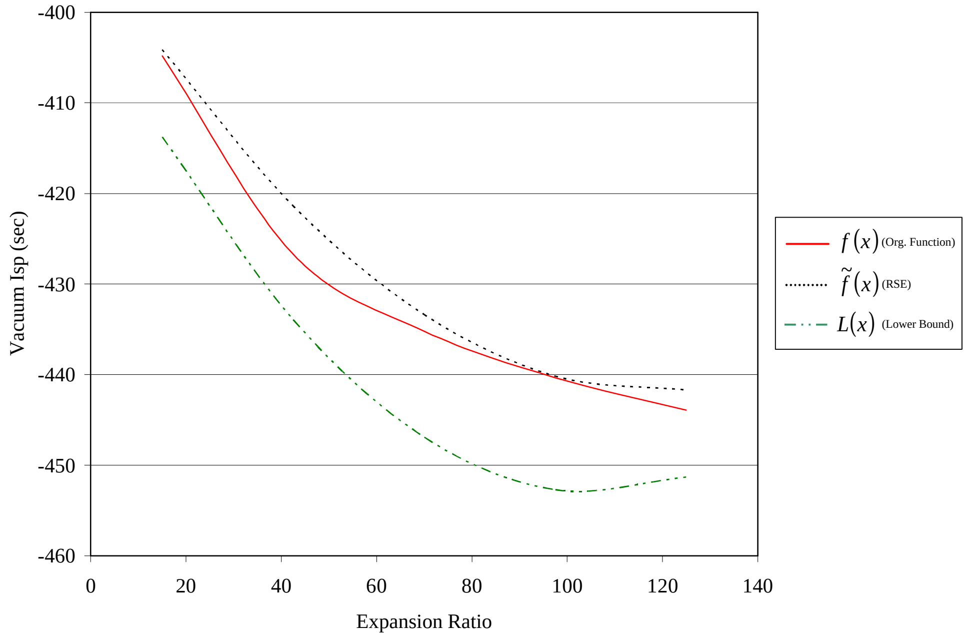


Figure 33: Vacuum I_{sp} vs. e **(Lower Bound)**

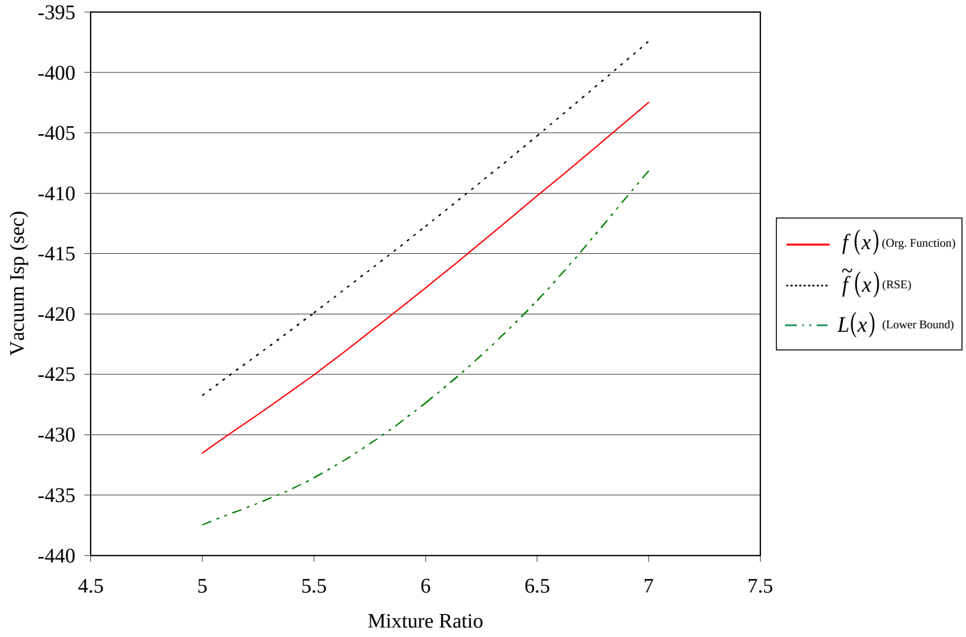


Figure 34: Vacuum I_{sp} vs. O/F (Lower Bound)

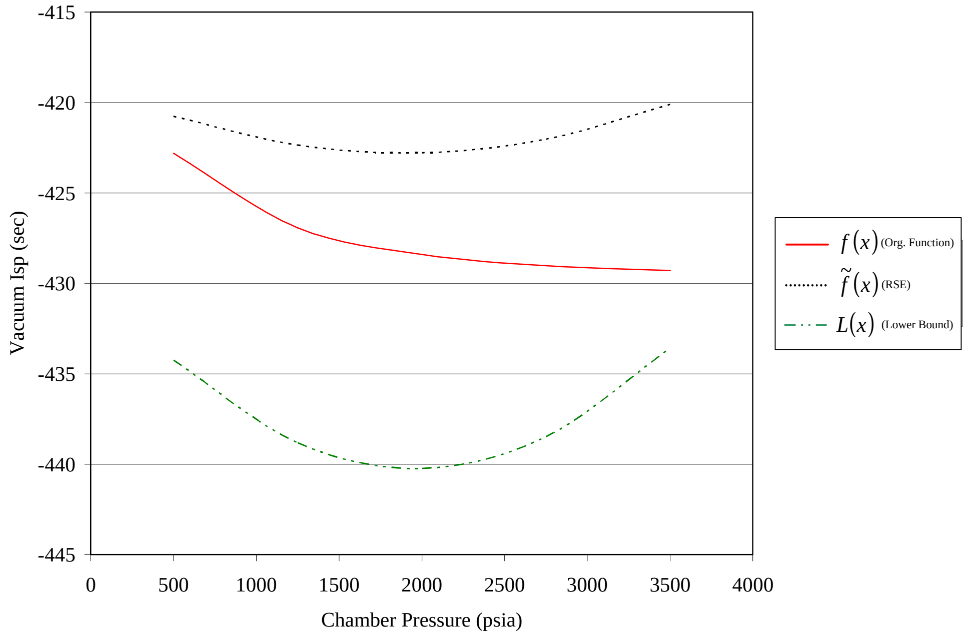


Figure 35: Vacuum I_{sp} vs. Pc (Lower Bound)

It has been mentioned previously that the lower bounding space expands the feasible region of the true design space. Figure 36 illustrates this fact by showing a plot of sea-level thrust versus expansion ratio for the Vulcain engine. Also included in this graph, are the upper and lower bounds of T_{sl} used in the lower bound optimization process. The lower bound values of T_{sl} are found by performing the lower bounding procedure outlined above on the positive values of T_{sl}. This yields an equation that under predicts sea-level thrust. The upper bound on sea-level thrust is found by performing the above procedure on the negative values of T_{sl}. This gives an equation that under predicts the negative of the values of sea-level thrust (i.e. gives more negative values). The results are then multiplied by -1 to get the positive upper bounds.

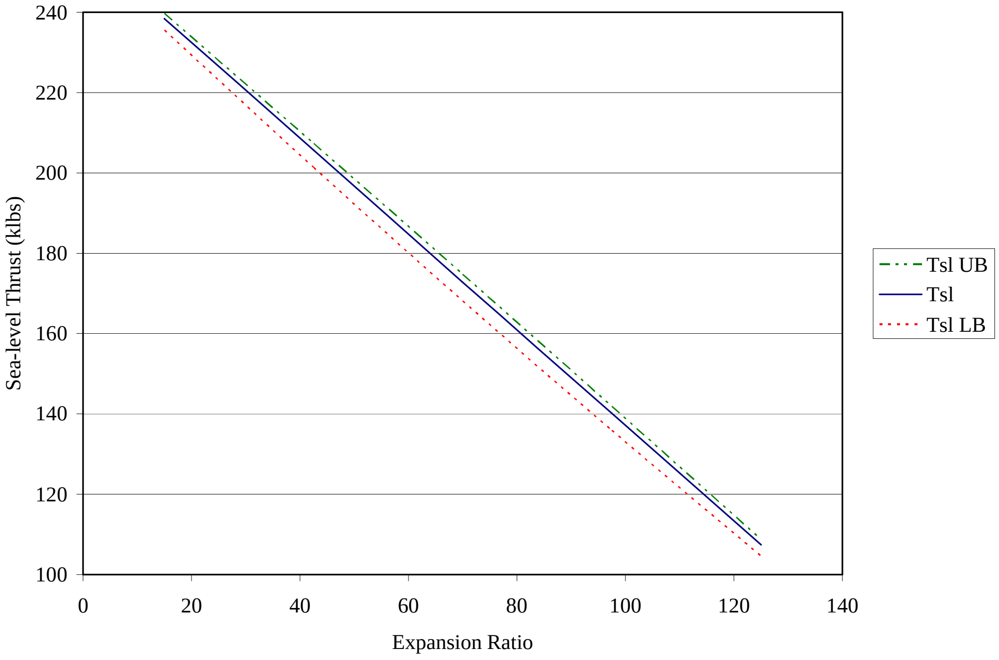


Figure 36: Upper and Lower Bounds on T_{sl}

For each output of the engine code, upper and lower bounding equations are formed. In the lower bounding optimization procedure, the lower and upper bounds developed for all the output values are used to either determine the objective function or constraint values. If a constraint on the optimization problem requires that a particular output be greater than or equal to a certain value, the upper bound on that output is used to evaluate the constraint. Likewise, if a constraint requires that a particular output be less than or equal to a certain value, the lower bound on that output is used. In this way, the feasible space is expanded which aids in finding optimal values that are lower bounds of the true design space.

There are alternate methods by which the response surface equations of the continuous design spaces could be used to find the optimal mixed-input solution. The simplest approach would be to use unadjusted response surfaces to predict the solutions of the continuous spaces. However, this could easily lead to the optimal solution being missed, since there will most likely be error associated with the predictions of the RSE's. Constraints could be relaxed to provide a larger feasible space when using the unaltered RSE's, but this will still not guarantee that the optimal solution will be found. Turning the response surface equations into lower bounding equations provides many advantages over using unaltered RSE's with relaxed constraints. The ability to determine bounds on the optimal solution provides an exact way to establish which true continuous design spaces must be investigated. A simpler method for forming the lower bounding equations by simply shifting the RSE's by a larger δ term and eliminating the underestimator terms (α terms) was considered. However, this has the potential to provide less accurate lower bounds when compared to using both the δ and α terms to form the lower bounding equations. This is because using both the δ and α terms allows the lower bounds to be tailored in order to more specifically match each portion of the design space.

5.3 Example Problem

An example problem will now be solved to further illustrate the new optimization scheme. This example, which is a modified version of one listed in reference 22, was chosen to provide a simple problem on which to demonstrate the features of the new optimization method. This problem can be solved by existing MINLP methods because the discrete variables can be relaxed. Therefore, when compared to existing MINLP algorithms, the new method will most likely be less efficient at solving this example. However, the solution process will still provide insight into the new optimization scheme.

Equation 35 shows the problem which will be solved. The objective function that will be minimized is nonlinear with respect to both the discrete and continuous variables. The first constraint is nonconvex with respect to x_1 and linear with respect to x_2. The second constraint is nonlinear in the continuous space and the final constraint is linear in both the continuous and discrete space. The optimal solution for this problem is an objective function value of 11.272 at $x_1 = 0.2$, $x_2 = -1$, and $y = 1$.

$$\min_{x,y}\ 0.8+5\left(x_2-\sqrt{x_1}\right)^2-0.7\ln(y) \tag{35a}$$

$$\text{s.t.}\quad g_1=-e^{(x_1-0.2)}-x_2\le 0 \tag{35b}$$

$$g_2=x_1^{1.2}+x_2+1.1y-1.3\le 0 \tag{35c}$$

$$g_3=x_1-1.2y-0.2\le 0 \tag{35c}$$

$$0.2\le x_1\le 1 \tag{35d}$$

$$-2.22554\le x_2\le -1 \tag{35d}$$

$$y=1,2,3 \tag{35e}$$

The first step in solving the example is to determine the lower bounding space. To do this, the objective function and constraints must be evaluated at the points needed to form the central composite design of experiments and at the ½ points used to determine the values of the α terms. Thirty-nine function calls are required to find these points for the example problem. After these points are determined, response surfaces are made for the objective function and constraints using the CCD points calculated for y = 1, 2, and 3. The values obtained by the response surfaces are then compared to the true values from the CCD and ½ points to determine the appropriate α and δ settings. Tables 36 - 41 show these comparisons. The RSE's of the objective function ($\tilde{f}(\bar{x})$), for all the discrete variable settings, provide a good model of the true space. However, there are points where the $\tilde{f}(\bar{x})$'s over-predict the true function value. The RSE's for the first and second constraints ($\tilde{g}(\bar{x})$) also effectively model the true space, but their predictions are not exact. For all the discrete variable settings, the $\tilde{g}(\bar{x})$'s for constraints 1 and 2 match the data used to fit them exactly, but some error is seen at the ½ points. For all the discrete variable settings, the response surface equations for constraint 3 provide an exact fit compared to all the data points sampled. This is because the constraint is linear and it is therefore easy for $\tilde{g}(\bar{x})$ to approximate this constraint.

Table 36: Real Values and RSE Values for Y=1

X_1	X_2	$f(\bar{x})$	$g_1(\bar{x})$	$g_2(\bar{x})$	$g_3(\bar{x})$	$\tilde{f}(\bar{x})$	$\tilde{g}_1(\bar{x})$	$\tilde{g}_2(\bar{x})$	$\tilde{g}_3(\bar{x})$
-1	-1	36.52	1.226	-2.281	-1.20	36.62	1.226	-2.281	-1.20
-1	0	22.02	0.613	-1.668	-1.20	22.02	0.613	-1.668	-1.20
-1	1	11.27	0.000	-1.055	-1.20	11.17	0.000	-1.055	-1.20
0	-1	45.80	0.734	-1.884	-0.80	45.60	0.734	-1.884	-0.80
0	0	29.30	0.121	-1.271	-0.80	29.30	0.121	-1.271	-0.80
0	1	16.55	-0.492	-0.658	-0.80	16.75	-0.492	-0.658	-0.80
1	-1	52.82	0.000	-1.426	-0.40	52.92	0.000	-1.426	-0.40
1	0	34.93	-0.613	-0.813	-0.40	34.93	-0.613	-0.813	-0.40
1	1	20.80	-1.226	-0.200	-0.40	20.70	-1.226	-0.200	-0.40
-0.5	0.5	19.60	0.085	-1.173	-1.00	19.55	0.091	-1.171	-1.00
-0.5	-0.5	33.35	0.698	-1.786	-1.00	33.12	0.703	-1.784	-1.00
0.5	0.5	25.02	-0.516	-0.741	-0.60	25.16	-0.522	-0.743	-0.60
0.5	-0.5	40.38	0.097	-1.354	-0.60	40.42	0.091	-1.356	-0.60

Table 37: Difference Between True Values and RSE for Y=1

X_1	X_2	*Obj D*	g_1 *D*	g_2 *D*	g_3 *D*
-1	-1	0.3%	0.0%	0.0%	0.0%
-1	0	0.0%	0.0%	0.0%	0.0%
-1	1	-0.9%	0.0%	0.0%	0.0%
0	-1	-0.4%	0.0%	0.0%	0.0%
0	0	0.0%	0.0%	0.0%	0.0%
0	1	1.2%	0.0%	0.0%	0.0%
1	-1	0.2%	0.0%	0.0%	0.0%
1	0	0.0%	0.0%	0.0%	0.0%
1	1	-0.5%	0.0%	0.0%	0.0%
-0.5	0.5	-0.2%	7.1%	0.2%	0.0%
-0.5	-0.5	-0.7%	0.7%	0.1%	0.0%
0.5	0.5	0.6%	-1.2%	-0.3%	0.0%
0.5	-0.5	0.1%	-6.2%	-0.2%	0.0%

Table 38: Real Values and RSE Values for Y=2

X_1	X_2	$f(\bar{x})$	$g_1(\bar{x})$	$g_2(\bar{x})$	$g_3(\bar{x})$	$\tilde{f}(\bar{x})$	$\tilde{g}_1(\bar{x})$	$\tilde{g}_2(\bar{x})$	$\tilde{g}_3(\bar{x})$
-1	-1	36.03	1.226	-1.181	-2.40	36.14	1.226	-1.181	-2.40
-1	0	21.53	0.613	-0.568	-2.40	21.53	0.613	-0.568	-2.40
-1	1	10.79	0.000	0.045	-2.40	10.68	0.000	0.045	-2.40
0	-1	45.32	0.734	-0.784	-2.00	45.11	0.734	-0.784	-2.00
0	0	28.81	0.121	-0.171	-2.00	28.81	0.121	-0.171	-2.00
0	1	16.06	-0.492	0.442	-2.00	16.27	-0.492	0.442	-2.00
1	-1	52.34	0.000	-0.326	-1.60	52.44	0.000	-0.326	-1.60
1	0	34.45	-0.613	0.287	-1.60	34.45	-0.613	0.287	-1.60
1	1	20.31	-1.226	0.900	-1.60	20.21	-1.226	0.900	-1.60
-0.5	0.5	19.11	0.085	-0.073	-2.20	19.06	0.091	-0.071	-2.20
-0.5	-0.5	32.87	0.698	-0.686	-2.20	32.63	0.703	-0.684	-2.20
0.5	0.5	24.53	-0.516	0.359	-1.80	24.67	-0.522	0.357	-1.80
0.5	-0.5	39.90	0.097	-0.254	-1.80	39.94	0.091	-0.256	-1.80

Table 39: Difference Between True Values and RSE for Y=2

X_1	X_2	*Obj D*	g_1 *D*	g_2 *D*	g_3 *D*
-1	-1	0.3%	0.0%	0.0%	0.0%
-1	0	0.0%	0.0%	0.0%	0.0%
-1	1	-1.0%	0.0%	0.0%	0.0%
0	-1	-0.5%	0.0%	0.0%	0.0%
0	0	0.0%	0.0%	0.0%	0.0%
0	1	1.3%	0.0%	0.0%	0.0%
1	-1	0.2%	0.0%	0.0%	0.0%
1	0	0.0%	0.0%	0.0%	0.0%
1	1	-0.5%	0.0%	0.0%	0.0%
-0.5	0.5	-0.3%	7.1%	2.7%	0.0%
-0.5	-0.5	-0.7%	0.7%	0.3%	0.0%
0.5	0.5	0.6%	-1.2%	-0.6%	0.0%
0.5	-0.5	0.1%	-6.2%	-0.8%	0.0%

Table 40: Real Values and RSE Values for Y=3

X_1	X_2	$f(\bar{x})$	$g_1(\bar{x})$	$g_2(\bar{x})$	$g_3(\bar{x})$	$\tilde{f}(\bar{x})$	$\tilde{g}_1(\bar{x})$	$\tilde{g}_2(\bar{x})$	$\tilde{g}_3(\bar{x})$
-1	-1	35.75	1.226	-0.081	-3.60	35.85	1.226	-0.081	-3.60
-1	0	21.25	0.613	0.532	-3.60	21.25	0.613	0.532	-3.60
-1	1	10.50	0.000	1.145	-3.60	10.40	0.000	1.145	-3.60
0	-1	45.04	0.734	0.316	-3.20	44.83	0.734	0.316	-3.20
0	0	28.53	0.121	0.929	-3.20	28.53	0.121	0.929	-3.20
0	1	15.78	-0.492	1.542	-3.20	15.99	-0.492	1.542	-3.20
1	-1	52.05	0.000	0.774	-2.80	52.16	0.000	0.774	-2.80
1	0	34.16	-0.613	1.387	-2.80	34.16	-0.613	1.387	-2.80
1	1	20.03	-1.226	2.000	-2.80	19.93	-1.226	2.000	-2.80
-0.5	0.5	18.83	0.085	1.027	-3.40	18.78	0.091	1.029	-3.40
-0.5	-0.5	32.58	0.698	0.414	-3.40	32.35	0.703	0.416	-3.40
0.5	0.5	24.25	-0.516	1.459	-3.00	24.39	-0.522	1.457	-3.00
0.5	-0.5	39.61	0.097	0.846	-3.00	39.65	0.091	0.844	-3.00

Table 41: Difference Between True Values and RSE for Y=3

X_1	X_2	*Obj D*	g_1 *D*	g_2 *D*	g_3 *D*
-1	-1	0.3%	0.0%	0.0%	0.0%
-1	0	0.0%	0.0%	0.0%	0.0%
-1	1	-1.0%	0.0%	0.0%	0.0%
0	-1	-0.5%	0.0%	0.0%	0.0%
0	0	0.0%	0.0%	0.0%	0.0%
0	1	1.3%	0.0%	0.0%	0.0%
1	-1	0.2%	0.0%	0.0%	0.0%
1	0	0.0%	0.0%	0.0%	0.0%
1	1	-0.5%	0.0%	0.0%	0.0%
-0.5	0.5	-0.3%	7.1%	0.2%	0.0%
-0.5	-0.5	-0.7%	0.7%	0.5%	0.0%
0.5	0.5	0.6%	-1.2%	-0.1%	0.0%
0.5	-0.5	0.1%	-6.2%	-0.3%	0.0%

The delta percentages shown in the tables above are the calculated using Equation 36. Therefore, a positive delta indicates that the RSE is over-predicting a value. This occurs for both the objective function and constraints 1 and 2. The over prediction of the constraint values causes the feasible region represented by the RSE to be smaller than the true feasible region in some portions of the design space.

$$\Delta = \frac{\left(\tilde{f}(\bar{x}) - f(\bar{x})\right)}{abs\left(f(\bar{x})\right)} \text{ or } \Delta = \frac{\left(\tilde{g}(\bar{x}) - g(\bar{x})\right)}{abs\left(g(\bar{x})\right)} \qquad (36)$$

At all the endpoints of the design space the constraint values predicted by the $\tilde{g}(\bar{x})$'s match the true values. Therefore, the δ terms in all the constraint lower bounding equations are set to zero. However, for each discrete variable setting, there are endpoints for which the $\tilde{f}(\bar{x})$ of the objective function over-predicts the true function value, so their will be nonzero δ terms in the objective function lower bounding equations. For the objective function and constraints 1 and 2, a margin value of 0.5% was chosen. The RSE's for these values show some error, when compared to the true values, but the overall good agreement allows this margin value to be set relatively low. The $\tilde{g}(\bar{x})$'s for constraint 3 predict the values so well that a very small margin can be used to determine the α values of the lower bounding equations. A margin of 0.1% was chosen for this example. In practice no margin is needed for this constraint term and the α's could be set to zero since $\tilde{g}_3(\bar{x})$ exactly matches the true space. It should also be pointed out that in this particular example the first constraint is not a function of the discrete variable and that is why its value is the same for all the discrete variable settings. Now that all the lower bounding information is available, the lower bounds of the design space can be formed using the RSE’s as a starting point. These lower bounds can then be used in the optimization procedure. Tables 42 - 47 show the lower bound results.

Table 42: Real Values and Lower Bound Values for Y=1

X_1	X_2	$f(\bar{x})$	$g_1(\bar{x})$	$g_2(\bar{x})$	$g_3(\bar{x})$	$L(\bar{x})$	$L(\bar{x})$	$L(\bar{x})$	$L(\bar{x})$
						Obj	g_1	g_2	g_3
-1	-1	36.52	1.226	-2.281	-1.20	36.52	1.226	-2.281	-1.200
-1	0	22.02	0.613	-1.668	-1.20	21.84	0.610	-1.676	-1.201
-1	1	11.27	0.000	-1.055	-1.20	11.06	0.000	-1.055	-1.200
0	-1	45.80	0.734	-1.884	-0.80	45.30	0.725	-1.893	-0.801
0	0	29.30	0.121	-1.271	-0.80	28.94	0.109	-1.289	-0.802
0	1	16.55	-0.492	-0.658	-0.80	16.46	-0.501	-0.668	-0.801
1	-1	52.82	0.000	-1.426	-0.40	52.82	0.000	-1.426	-0.400
1	0	34.93	-0.613	-0.813	-0.40	34.76	-0.616	-0.821	-0.401
1	1	20.80	-1.226	-0.200	-0.40	20.59	-1.226	-0.200	-0.400
-0.5	0.5	19.60	0.085	-1.173	-1.00	19.25	0.081	-1.184	-1.002
-0.5	-0.5	33.35	0.698	-1.786	-1.00	32.82	0.694	-1.797	-1.002
0.5	0.5	25.02	-0.516	-0.741	-0.60	24.86	-0.531	-0.757	-0.602
0.5	-0.5	40.38	0.097	-1.354	-0.60	40.13	0.081	-1.369	-0.602

Table 43: Difference Between True Values and Lower Bound Values for Y=1

X_1	X_2	*Obj D*	g_1 *D*	g_2 *D*	g_3 *D*
-1	-1	0.0%	0.0%	0.0%	0.0%
-1	0	-0.8%	-0.5%	-0.5%	-0.1%
-1	1	-1.9%	0.0%	0.0%	0.0%
0	-1	-1.1%	-1.2%	-0.5%	-0.1%
0	0	-1.2%	-9.9%	-1.4%	-0.3%
0	1	-0.5%	-1.8%	-1.5%	-0.1%
1	-1	0.0%	0.0%	0.0%	0.0%
1	0	-0.5%	-0.5%	-1.0%	-0.3%
1	1	-1.0%	0.0%	0.0%	0.0%
-0.5	0.5	-1.8%	-4.7%	-0.9%	-0.2%
-0.5	-0.5	-1.6%	-0.6%	-0.6%	-0.2%
0.5	0.5	-0.6%	-2.9%	-2.2%	-0.3%
0.5	-0.5	-0.6%	-16.5%	-1.1%	-0.3%

Table 44: Real Values and Lower Bound Values for Y=2

X_1	X_2	$f(\bar{x})$	$g_1(\bar{x})$	$g_2(\bar{x})$	$g_3(\bar{x})$	$L(\bar{x})$	$L(\bar{x})$	$L(\bar{x})$	$L(\bar{x})$
						Obj	g_1	g_2	g_3
-1	-1	36.03	1.226	-1.181	-2.40	36.03	1.226	-1.181	-2.400
-1	0	21.53	0.613	-0.568	-2.40	21.36	0.607	-0.572	-2.403
-1	1	10.79	0.000	0.045	-2.40	10.58	0.000	0.045	-2.400
0	-1	45.32	0.734	-0.784	-2.00	44.82	0.728	-0.788	-2.002
0	0	28.81	0.121	-0.171	-2.00	28.46	0.109	-0.179	-2.005
0	1	16.06	-0.492	0.442	-2.00	15.98	-0.498	0.438	-2.002
1	-1	52.34	0.000	-0.326	-1.60	52.34	0.000	-0.326	-1.600
1	0	34.45	-0.613	0.287	-1.60	34.28	-0.619	0.283	-1.603
1	1	20.31	-1.226	0.900	-1.60	20.11	-1.226	0.900	-1.600
-0.5	0.5	19.11	0.085	-0.073	-2.20	18.77	0.081	-0.077	-2.204
-0.5	-0.5	32.87	0.6977	-0.686	-2.20	32.34	0.694	-0.690	-2.204
0.5	0.5	24.53	-0.516	0.359	-1.80	24.38	-0.531	0.351	-1.804
0.5	-0.5	39.90	0.097	-0.254	-1.80	39.65	0.081	-0.262	-1.804

Table 45: Difference Between True Values and Lower Bound Values for Y=2

X_1	X_2	*Obj D*	g_1 *D*	g_2 *D*	g_3 *D*
-1	-1	0.0%	0.0%	0.0%	0.0%
-1	0	-0.8%	-1.0%	-0.7%	-0.1%
-1	1	-1.9%	0.0%	0.0%	0.0%
0	-1	-1.1%	-0.8%	-0.5%	-0.1%
0	0	-1.2%	-9.9%	-4.7%	-0.2%
0	1	-0.5%	-1.2%	-0.9%	-0.1%
1	-1	0.0%	0.0%	0.0%	0.0%
1	0	-0.5%	-1.0%	-1.4%	-0.2%
1	1	-1.0%	0.0%	0.0%	0.0%
-0.5	0.5	-1.8%	-4.7%	-5.5%	-0.2%
-0.5	-0.5	-1.6%	-0.5%	-0.6%	-0.2%
0.5	0.5	-0.6%	-2.9%	-2.2%	-0.2%
0.5	-0.5	-0.6%	-16.5%	-3.1%	-0.2%

Table 46: Real Values and Lower Bound Values for Y=3

X_1	X_2	$f(\bar{x})$	$g_1(\bar{x})$	$g_2(\bar{x})$	$g_3(\bar{x})$	$L(\bar{x})$	$L(\bar{x})$	$L(\bar{x})$	$L(\bar{x})$
						Obj	g_1	g_2	g_3
-1	-1	35.75	1.226	-0.081	-3.60	35.75	1.226	-0.081	-3.600
-1	0	21.25	0.613	0.532	-3.60	21.08	0.607	0.525	-3.604
-1	1	10.50	0.000	1.145	-3.60	10.29	0.000	1.145	-3.600
0	-1	45.04	0.734	0.316	-3.20	44.54	0.727	0.308	-3.203
0	0	28.53	0.121	0.929	-3.20	28.17	0.109	0.913	-3.207
0	1	15.78	-0.492	1.542	-3.20	15.70	-0.498	1.533	-3.203
1	-1	52.05	0.000	0.774	-2.80	52.05	0.000	0.774	-2.800
1	0	34.16	-0.613	1.387	-2.80	33.99	-0.619	1.380	-2.804
1	1	20.03	-1.226	2.000	-2.80	19.82	-1.226	2.000	-2.800
-0.5	0.5	18.83	0.085	1.027	-3.40	18.49	0.081	1.018	-3.405
-0.5	-0.5	32.58	0.6977	0.414	-3.40	32.06	0.694	0.405	-3.405
0.5	0.5	24.25	-0.516	1.459	-3.00	24.10	-0.531	1.445	-3.005
0.5	-0.5	39.61	0.097	0.846	-3.00	39.36	0.081	0.832	-3.005

Table 47: Difference Between True Values and Lower Bound Values for Y=3

X_1	X_2	*Obj D*	g_1 *D*	g_2 *D*	g_3 *D*
-1	-1	0.0%	0.0%	0.0%	0.0%
-1	0	-0.8%	-1.0%	-1.3%	-0.1%
-1	1	-2.0%	0.0%	0.0%	0.0%
0	-1	-1.1%	-1.0%	-2.5%	-0.1%
0	0	-1.3%	-9.9%	-1.7%	-0.2%
0	1	-0.5%	-1.2%	-0.6%	-0.1%
1	-1	0.0%	0.0%	0.0%	0.0%
1	0	-0.5%	-1.0%	-0.5%	-0.1%
1	1	-1.0%	0.0%	0.0%	0.0%
-0.5	0.5	-1.8%	-4.7%	-0.9%	-0.1%
-0.5	-0.5	-1.6%	-0.5%	-2.2%	-0.1%
0.5	0.5	-0.6%	-2.9%	-1.0%	-0.2%
0.5	-0.5	-0.6%	-16.5%	-1.7%	-0.2%

For the constraint functions, the Δ's found by comparing the lower bounds to the true values are negative, except at the endpoints of the design space. Those Δ's are zero because the δ term was set to zero in the lowering bounding equations, since the $\tilde{g}(\bar{x})$'s exactly matched the true space at all the endpoints. However, the lower bounding values for the objective functions at some of the endpoints are negative. This is because the δ term in the lower bounding equations used to model the objective function is nonzero, for all the discrete variable settings. For the objective function and constraints 1 and 2, the maximum Δ between the lower bounds and the true space, not including the endpoints, is -0.5%. This is the value that the margin was set to for these lower bounds for all the discrete variable choices. Likewise, the maximum Δ for constraint 3, for every discrete variable choice, is -0.1% (also excluding the endpoints). This value is equal to the margin set for constraint 3.

The following tables show the values used to get the lower bounding equation for the objective function when y = 1. These tables are shown to further illustrate the solution method. Table 48 gives the β terms used to make the response surface equation ($\tilde{f}(\bar{x})$). The α and δ terms used to transform $\tilde{f}(\bar{x})$ into a lower bound ($L(\bar{x})$) are shown in Table 49. As a reminder, the inputs into $\tilde{f}(\bar{x})$ and $L(\bar{x})$ are the values of X_1 and X_2 that have been normalized to be between -1 and 1.

Table 48: $\tilde{f}(\bar{x})$ Coefficients for Y = 1

Response Surface Term	*Coefficients*
Intercept	29.29760
X_1	6.45759
X_2	-14.42077
X_1* X_1	-0.82235
X_1* X_2	-1.69365
X_2* X_2	1.87744

Table 49: $L(\bar{x})$ Constants for Y = 1

Response Surface Term	*Coefficients*
δ	0.1041
a_{X_1}	0.1869
a_{X_2}	0.0705

Now that all the lower bounding equations are known, they can be used to get the lower bound of the optimal solution for the discrete variable choices. The lower bounding equations for y = 3 do not provide a feasible solution. This means that this discrete variable choice is also infeasible in the true design space. Therefore, this discrete variable choice can be eliminated as a possible optimal solution. The lower bounding results for y = 1 and y = 2 are shown in Table 50 and Table 51. Since the lowest objective function value is provided by the optimum of the lower bounding space of y = 1, this discrete variable setting is used to find the first upper bound, which is found by solving the true space with y = 1. After investigating the continuous space, the upper bound on the solution is 11.272 and the lower bound is 11.064. The current upper bound is less than the solution of the lower bound of the y = 2 space, which is 18.040. Therefore, the true space obtained when y = 2 does not have to be investigated since it will not provide a solution that improves the upper bound. Since y = 2 and y = 3 have been eliminated as solution choices, the optimal solution of the optimization problem is the upper bound, 11.272, found when y = 1, x_1 = 0.2 and x_2 = -1.0.

Table 50: Solutions for Lower Bound and True Space for Y = 1

	X_1	X_2	*Obj*	g_1	g_2	g_3
Lower Bound Solution	0.200	-1.000	11.064	0.000	-1.055	-1.200
True Solution	0.200	-1.000	11.272	0.000	-1.055	-1.200

Table 51: Solutions for Lower Bound and True Space for Y = 2

	X_1	X_2	*Obj*	g_1	g_2	g_3
Lower Bound Solution	0.421	-1.251	18.040	0.000	0.000	-2.183
True Solution	0.448	-1.282	19.354	0.000	0.000	-2.152

5.4 Applicability of New Optimization Method

The new optimization method developed for this research has been tailored to work well with the rocket engine design codes to which it was applied. However, this method will work well on any other optimization problems with similar features. These include problems with the following characteristics.

- Mixed-Integer inputs where the discrete variables can not be relaxed
- A continuous variable design space that is smooth and free of discontinuities
- Objective and constraint values supplied by an analysis that is expensive (time consuming) to run

While the method is designed for problems with the above features, it will work for other applications, but perhaps less efficiently. For problems where the discrete inputs can be relaxed and the analysis isn't time consuming, like the example problem outlined above, other optimization methods will be able to find the solution in less computational time. However, this method is still able to solve such problems. Also, the method can be adapted to handle problems where the continuous input variable design space is not well-behaved. This could include problems with sharp, deep extrema or possibly even multi-modal spaces. For these types of continuous spaces, the margin used to find the α terms would have to be dramatically increased to ensure that the lower bounding procedure gives a lower bound. Also, the NLP method used to find the optimal solutions of the true continuous spaces would most likely have to be changed to an optimization method capable of handling multi-modal spaces. Some possible methods include genetic algorithms, simulated annealing, multilevel coordinate search, or Branch and Reduce [72,112,113].

Even if the domain of applicability of this method is limited to problems with the original features mentioned above, problems with these characteristics still represent difficult applications for which very few, if any, optimization techniques can solve. It should also be noted that no restrictions are placed on the discrete variables. This allows for the solutions of problems with highly nonlinear, discrete variable spaces without any additional difficulty.

CHAPTER 6

PROOF – OF – CONCEPT

6.1 Problem Description

The optimization technique outlined in Chapter 5 needs to be tested on a relevant problem to validate the new method. In order to be able to quickly test the optimization method on many different problems, a fast running design code was needed. The use of a fast running code, allows the optimal solution to be found relatively quickly, by solving all the nonlinear optimization problems associated with every discrete variable combination. The test code should also have features similar to PHATCAT, which will be used in the final application. These include the use of mixed-inputs, where the discrete variables can not be relaxed, and a continuous variable design space that is well-behaved. The code that was chosen for the proof-of-concept demonstration is SCORES. Recall that SCORES is a relatively simple engine design tool that determines rocket performance characteristics very quickly. It uses efficiencies on the ideal performance outputs to model different rocket engine powerhead cycles. SCORES has both continuous and discrete inputs and the discrete input variables are not able to be relaxed. This, combined with its fast execution time, makes SCORES a good choice for a code on which to test the new optimization scheme.

The SCORES discrete input variables that were used for this analysis were the fuel and oxidizer choices. The continuous variables used were the desired chamber pressure, oxidizer to fuel mixture ratio and the nozzle expansion ratio. As mentioned above, SCORES uses efficiencies to scale its performance outputs depending on the engine cycle being analyzed. To avoid choosing a particular cycle, the ideal performance values were used for the optimization metrics. The objective function for all the optimization test problems was the ideal specific impulse. However, I_{sp} should be maximized. To accomplish this, the negative of the ideal I_{sp} was minimized. Variables used as constraints on the optimization problem were sea-level thrust (T_{sl}), nozzle exit area (A_e) and engine vacuum thrust to weight (T/W). The SCORES inputs and outputs that were used for the proof-of-concept problems are shown below in Table 52.

Table 52: Input and Output Descriptions for the Proof-of-Concept Problem

Input	*Discrete/Continuous*	*Output*	*Objective/Constraint*
Fuel	Discrete	Ideal Isp	Objective
Oxidizer	Discrete	Sea-level Thrust	Constraint
Chamber Pressure	Continuous	Nozzle Exit Area	Constraint
Mixture Ratio	Continuous	Engine T/W	Constraint
Expansion Ratio	Continuous		

The version of SCORES used for this research does not supply engine T/W as one of its outputs. However, the engine’s T/W is typically a very important metric for rocket powered space vehicles. In order to include engine weight in this analysis, a simple calculation for T/W was created. This calculation was based on empirical equations found in reference 85. However, the original equations given in the reference only provide engine weight as a function of vacuum thrust, which was fixed at a value of 1 Mlbs for the proof-of-concept problems. Therefore, in order to make T/W a function of the input variables used for these example problems, the model was adjusted to make the calculated engine weight a function of engine mixture ratio, chamber pressure and fuel choice.

The discrete input variables used for the proof-of-concept cases are the fuel and oxidizer combinations. Thirteen different fuel and oxidizer combinations were selected for this test problem, with each one representing a discrete variable choice in the new optimization scheme.

Table 53: Fuel and Oxidizer Combinations

Combination	*Fuel*	*Oxidizer*
1	Hydrogen	Oxygen
2	Methane	Oxygen
3	Propane	Oxygen
4	RP-1	Oxygen
5	$C_{7.76}H_{13.1}$	Oxygen
6	$C_{8.26}H_{15.5}$	Oxygen
7	$C_{10.8}H_{18.7}$	Oxygen
8	Methane	H_2O_2
9	Propane	H_2O_2
10	RP-1	H_2O_2
11	$C_{7.76}H_{13.1}$	H_2O_2
12	$C_{8.26}H_{15.5}$	H_2O_2
13	$C_{10.8}H_{18.7}$	H_2O_2

SCORES has several built-in fuel and oxidizer choices and also has the option to add new fuels if desired. The fuels first listed for combinations 5, 6, and 7 in Table 53 are new fuels that were added to SCORES. New fuels are added by specifying the number of carbon and hydrogen atoms in one molecule of the fuel, along with the fuels' initial enthalpy. The fuels in combinations 5 and 6 are types of gasoline and the fuel in combination 7 is diesel. The values used to model these fuels in SCORES were taken from reference 80. Though the new fuel choices are non-traditional rocket fuels, which would most likely not be considered for an actual engine design, they were added to the analysis in order to give more discrete variable choices and therefore provide a better test problem.

6.2 Proof-of-Concept Test Cases

Ten different test cases were used to study the performance of the new optimization method. These test cases ranged from unconstrained optimization to cases involving constraints on all of the output variables. Equality constraints were also included. These cases were chosen to try to illustrate the different types of problems that can be solved with the new optimization procedure. The cases used are listed in Table 54.

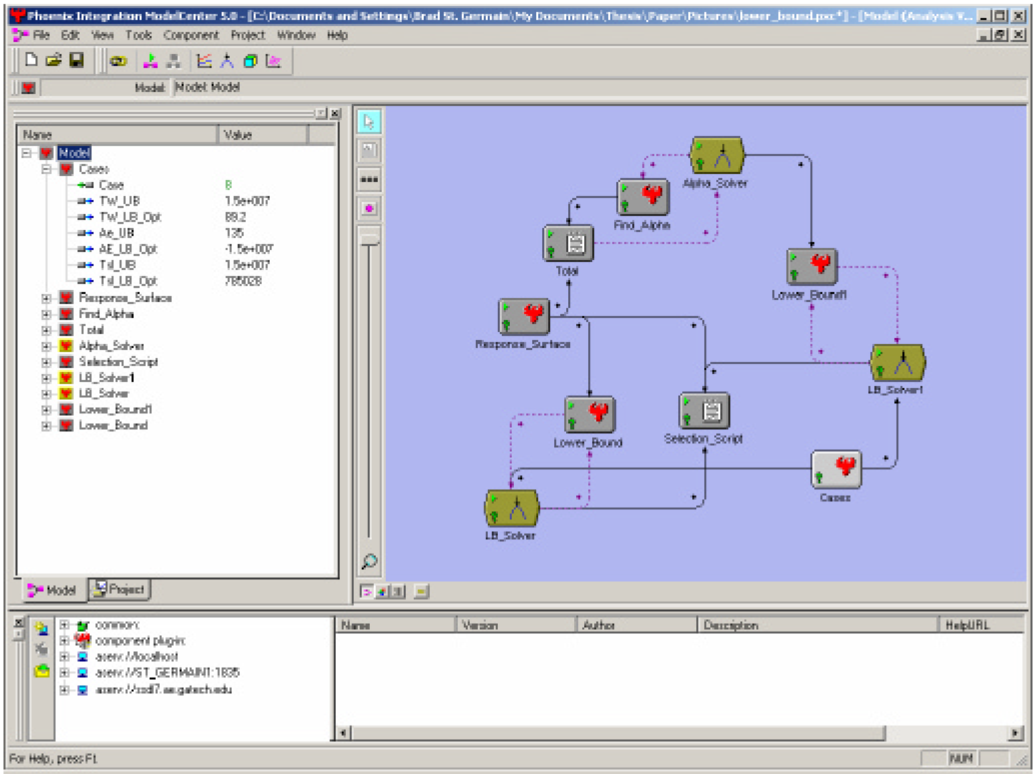
To solve these cases using the proposed optimization method, a C-code was written that ran and stored the results for the engine analyses needed to populate the design points used to determine the lower bounding equations. Then, another C-code took the generated data and formed the response surface equations that represent the SCORES output variables of interest. These response surface equations were then turned into lower bounding equations by finding the correct δ and α terms using the methods described in Chapter 5.

Table 54: Constraints used for Proof-of-Concept Problems

Case	*Constraint 1*	*Constraint 2*	*Constraint 3*
1	N/A	N/A	N/A
2	$T/W \geq 83.5$	N/A	N/A
3	$T/W \geq 95$	N/A	N/A
4	$T/W \geq 90$	$Tsl \geq 800000 lbs$	N/A
5	$Tsl \geq 895123 lbs$	N/A	N/A
6	$Ae \leq 50\,ft^2$	N/A	N/A
7	$Tsl \leq 902568 lbs$	$Tsl \geq 863789 lbs$	N/A
8	$T/W \geq 89.2$	$Ae \leq 135\,ft^2$	$Tsl \geq 785028 lbs$
9	$T/W \geq 87.3$	$Tsl = 698000 lbs$	N/A
10	$Tsl \geq 900000 lbs$	N/A	N/A

To obtain the optimized solutions of the true continuous design space, the lower bounding C-code and SCORES was used in conjunction with a program called ModelCenter [114]. ModelCenter is a frameworks code that allows the easy integration of various codes regardless of the platform on which they are running. Also, ModelCenter has a built-in optimization routine, which includes three constrained nonlinear optimization methods. Sequential linear programming was used to find the optimal values of the true continuous design space and the method of feasible directions was used find the appropriate α values and the optimum of the lower bounding space. Figure 37 shows a screenshot of the ModelCenter application used to solve the lower bounding problem.

To determine the abilities of the new optimization procedure, the solution of each case shown in Table 54 was found using both the new optimization method, and by finding the optimal solution of the continuous design space for every discrete variable choice. The success of the new optimization method was determined by seeing if it could get the same optimal solution obtained by solving every discrete variable combination, but with significantly fewer function calls to SCORES.



Name	Value
Model	
Cases	
Case	8
TW_UB	1.5e+007
TW_LB_Opt	89.2
Ae_UB	135
AE_LB_Opt	-1.5e+007
Tsl_UB	1.5e+007
Tsl_LB_Opt	785028
Response_Surface	
Find_Alpha	
Total	
Alpha_Solver	
Selection_Script	
LB_Solver1	
LB_Solver	
Lower_Bound1	
Lower_Bound	

commons
component plugins
aserv://localhost
aserv://ST_GERMAIN1:1835
aserv://codi7.ae.gatech.edu

Name	Version	Author	Description	HelpURL

Figure 37: ModelCenter Interface

To find the lower bounding equations of the continuous design space for a single discrete variable combination, 15 function calls were needed to fill in the CCD and an additional 8 were needed to find the output values at the ½ points. Therefore, in order to determine the lower bounding equations for all the discrete variable combinations, SCORES had to be run 299 times. Remember, however, that information used to get the lower bounding equations was saved in a database file and therefore the information only had to be generated for the first optimization case solved.

It took an additional 860 SCORES function calls to evaluate the continuous variable nonlinear optimization problems (NLPs) needed to solve all ten cases using the new optimization method. As a comparison, to solve all ten cases by finding the solution to the continuous variable nonlinear optimization problems for all the fuel and oxidizer combinations took 5687 SCORES function calls. Including the function calls needed to find the lower bounds, the new method only takes 20% of the function calls required by the enumeration method. Also, the enumeration method and the new optimization method being proposed for this research got the same answer for each case. Table 55 shows the results for the number of function calls needed for each of the ten cases. As can be seen by these results, for every case, the proposed optimization method is much more efficient than enumerating all the discrete choices.

Table 55: Proof-of-Concept Results

Case	*Function Calls (Solving all NLPs)*	*Function Calls w/ new method*	*Number of NLPs evaluated w/ new method*	*% of Baseline Function Calls*
1	295	30	1	10.2%
2	394	53	2	13.5%
3	398	62	3	15.6%
4	661	188	3	28.4%
5	583	53	1	9.09%
6	645	60	1	9.30%
7	685	46	1	6.72%
8	699	177	3	25.3%
9	788	174	2	22.1%
10	539	17	1	3.15%

More detailed results will now be presented for Case 8 and Case 10. These results are intended to lend further insight into the optimization procedure. Table 56 shows the values of ideal I_{sp} obtained by the lower bounding procedure ($L(\bar{x})$) and by optimizing the true continuous design space ($f(\bar{x})$) for all 13 discrete variable combinations. The true continuous space was optimized for all the discrete variable combinations to help further illustrate the new optimization method. The new method only requires that the true continuous space of three discrete variable combinations be investigated. As expected, the values of ideal I_{sp} (actually the negative of ideal I_{sp}, because I_{sp} is being maximized) obtained by the lower bounding procedure are less than the values obtained by solving the NLP of the true space. Also, the same five cases are infeasible for each solution method.

Table 56: Results for Proof-of-Concept - Case 8

Combination	$L(\bar{x})$ *for Ideal I_{sp} (sec)*	$f(\bar{x})$ *for Ideal I_{sp} (sec)*
1	infeasible	infeasible
2	infeasible	infeasible
3	-326.9773	-325.2923
4	infeasible	infeasible
5	-342.1558	-340.6288
6	-347.8846	-345.1155
7	-346.5635	-344.1238
8	infeasible	infeasible
9	-362.5682	-359.8209
10	infeasible	infeasible
11	-372.4407	-366.8554
12	-372.0730	-367.8708
13	-370.7460	-367.7762

According to Table 55, three NLP solutions were required to solve Case 8 using the new optimization method. These three NLP solutions were for combinations 11, 12, and 13. Combination 11 provides the lowest lower bound, so its NLP was solved first, yielding an upper bound of -366.8554 seconds. Combinations 12 and 13 are the only combinations to have lower bounds less than this upper bound. Therefore, the rest of the combinations can be eliminated from consideration. Next, the NLP for combination 12 was solved, and its solution provides a better upper bound than the one obtained from combination 11. Therefore, the new upper bound value is -367.8708. Finally, the NLP for combination 13 was solved and its solution does not provide a better upper bound. Therefore, the solution of the NLP for combination 12 is the optimum (-367.87 sec). This solution occurred at the following continuous variable settings: O/F = 6.0, expansion ratio = 103.197, and chamber pressure = 3600.523 psia. Before the optimization procedure, upper and lower bounds were established for each of the continuous input variables. These bounds are listed in Table 57. At the final solution, the O/F is set to its upper bound but the other input variables are not at either of their bounds. The constraint values at the optimal solution are shown in Table 58.

Table 57: Continuous Input Variable Bounds

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Mixture Ratio (O/F)	4.5	6
Expansion Ratio (ε)	50	150
Chamber Pressure (P_c)	2500 psia	4000 psia

Table 58: Constraint Values at Optimum (Case 8)

Constraint	*Value*
Sea-Level Thrust (T_{sl})	785156 lbs
Exit Area (A_e)	101.5 ft^2
Thrust-to-Weight (T/W)	89.2

As a note, the reason some of the more conventional fuels performed poorly in this case was because of the T/W constraint. As mentioned earlier, an equation for T/W was created that was a function of fuel type, O/F and chamber pressure. The dependence on fuel type favored fuels with molecules that contained more C and H atoms. This was done under the notion that fuels with larger molecules would be denser and therefore easier to pump, leading to lighter turbo machinery. Therefore, the gasoline and diesel fuels have better T/W's and the optimizer can choose high chamber pressures, which has a negative effect on T/W, to increase the performance without violating the T/W constraint. The more traditional fuels must have lower chamber pressures to avoid violating the T/W constraint and therefore their performance suffers. The notional T/W equation was only created to show interesting trends for the example problem.

The solution to Case 8 demonstrates why every discrete variable setting with a lower bound better than the current upper bound must be investigated. For this case, if the discrete variable choice yielding the lowest lower bound (combination 11) was chosen as the optimal combination, without investigating the remaining valid discrete variable choices, the true optimal solution would be missed. The next case that will be discussed is Case 10. Two sets of results for Case 10 are presented below. Table 59 shows the results obtained using the new optimization procedure with the standard lower bounding equations ($L(\bar{x})$), while Table 60 shows the results found using the new optimization procedure with only the original response surface equations ($\tilde{f}(\bar{x})$).

Table 59: Results for Proof-of-Concept - Case 10

Combination	$L(\bar{x})$ *for Ideal* I_{sp} *(sec)*	$f(\bar{x})$ *for Ideal* I_{sp} *(sec)*
1	-464.3736	-459.9229
2	-363.1333	-361.6224
3	-347.7168	-346.3031
4	-340.2295	-338.7311
5	-337.5242	-336.1536
6	-340.3359	-338.9547
7	-336.5650	-335.3158
8	-356.4762	-346.1678
9	-356.7792	-354.9096
10	-360.2676	-356.9867
11	-363.8689	-357.8790
12	-362.0663	-358.0523
13	-360.9506	-356.5423

Table 60: Results for Proof-of-Concept - Case 10 (Response Surface Only)

Combination	$\tilde{f}(\bar{x})$ *for Ideal* I_{sp} *(sec)*	$f(\bar{x})$ *for Ideal* I_{sp} *(sec)*
1	infeasible	-459.9229
2	-361.2703	-361.6224
3	-345.9736	-346.3031
4	-338.4380	-338.7311
5	-335.8734	-336.1536
6	-338.6713	-338.9547
7	-335.0275	-335.3158
8	infeasible	-346.1678
9	-352.1760	-354.9096
10	-356.2002	-356.9867
11	-357.0584	-357.8790
12	-357.0861	-358.0523
13	-355.4782	-356.5423

The results obtained using the lower bounding equations confirm the data presented in Table 55. Combination 1 provides the best lower bound (-464.3736) and the upper bound found by solving the true continuous space for that combination (-459.9229) is lower than any remaining lower bounds. Therefore, only one NLP had to be solved to find the optimal. However, if only the response surfaces were used, instead of the lower bounding equations, the true optimal is not found. Instead of expanding the feasible space and providing a lower bound, the response surface equations for combination 1 actually decrease the feasible space. The solution of the response surface is infeasible, which means combination 1 would never be investigated and the optimal solution would therefore be missed. This illustrates why the using lower bounding equations to represent the continuous design space, instead of just traditional response surfaces, provides a more robust optimization method. Detailed results for all the optimization cases used for this proof-of-concept analysis can be found in Appendix A.

CHAPTER 7

FINAL APPLICATION

7.1 Problem Description

The proof-of-concept problem in the previous chapter shows the promise of the lower bounding optimization method developed for this research. However, since SCORES is a fast running code, an engine designed using SCORES could easily be optimized without the new method by finding the optimal solution of the continuous design space for all the discrete variable combinations. The main objective of this research is to apply the method to a higher fidelity, and therefore slower running code. PHATCAT is the code that is used for this final application. Optimizing engine designs using the new version of PHATCAT will show the full potential of the lower bounding method. The longer the code supplying the engine performance results takes to run, the more time will be saved by switching to the new optimization scheme. Also, as the number of discrete variable choices increases, the more computation time will be saved by using the new method versus solving the NLP for each discrete variable setting.

The modifications to PHATCAT detailed in Chapter 3 allow its use in this final test of the new optimization method. The new version of PHATCAT is robust enough that a nonlinear optimizer can be used to solve for the optimum of its continuous space. Also, twenty-two different engine cycles are now available as input choices. These help provide enough discrete variable choices to adequately test the new method. These new engine cycles are formed from four basic cycle designs. The four main cycles are a fuel-rich dual-preburner staged-combustion cycle based off the SSME, a fuel-rich gas generator cycle based off the Vulcain, an oxidizer-rich single-preburner staged-combustion cycle based off the RD-180, and a split-expander cycle similar to the RLX. Variations of these basic cycles are also included as inputs and these combine to form the twenty-two cycles available in PHATCAT. The variations made to the cycles include changing the fuel type and changing the number of fuel boost pumps, turbines, and pre-burners/gas generators. These changes alter the engine performance by possibly changing the engines I_{sp}, thrust level, available chamber pressure, weight, cost, or more likely, a combination of all these effects.

The twenty-two engine cycles now available in PHATCAT are listed in Table 61. Each of these cycles represents a discrete variable choice for the optimization method. However, to further demonstrate the abilities of the new lower bounding method, more discrete variable choices are desired. Therefore, the choice of turbine blade material was included as another discrete variable. The turbine blade material affects the performance of the engine by limiting the allowable turbine inlet temperatures. At higher turbine inlet temperatures, less turbine working fluid mass flow is needed to produce the same amount of turbine work. Also, the turbine blade material choice was used to adjust the development and production costs of the engines. Engines using the higher temperature turbine blade materials incurred a cost penalty, while engines using lower temperature materials received a slight cost credit.

Table 61: Engine Cycles Included in PHATCAT

Cycle Name	*Baseline Cycle*	*Fuel Type*	*Uses Fuel Boost Pump*	*# of Main Turbines*	*# of Preburners/ GGs*
SSME	SC	LH2	Yes	2	2
SSME_1PB_1T	SC	LH2	Yes	1	1
Vulcain_LH2	GG	LH2	Yes	2	1
Vulcain_RP1	GG	RP-1	Yes	2	1
Vulcain_CH4	GG	CH4	Yes	2	1
Vulcain_C3H8	GG	C3H8	Yes	2	1
Vulcain_LH2_1T	GG	LH2	Yes	1	1
Vulcain_RP1_1T	GG	RP-1	Yes	1	1
Vulcain_CH4_1T	GG	CH4	Yes	1	1
Vulcain_C3H8_1T	GG	C3H8	Yes	1	1
Vulcain_LH2_NBP	GG	LH2	No	2	1
Vulcain_RP1_NBP	GG	RP-1	No	2	1
Vulcain_CH4_NBP	GG	CH4	No	2	1
Vulcain_C3H8_NBP	GG	C3H8	No	2	1
Vulcain_LH2_1T_NBP	GG	LH2	No	1	1
Vulcain_RP1_1T_NBP	GG	RP-1	No	1	1
Vulcain_CH4_1T_NBP	GG	CH4	No	1	1
Vulcain_C3H8_1T_NBP	GG	C3H8	No	1	1
RD180_RP1	SC	RP-1	Yes	1	1
RD180_CH4	SC	CH4	Yes	1	1
RD180_C3H8	SC	C3H8	Yes	1	1
RLX	EX	LH2	Yes	2	0

Reference 85 gives allowable turbine inlet temperatures as a function of the turbine blade mean pitchline velocity for various turbine blade materials. Since the current version of PHATCAT does not calculate turbine rotational speed, a single value for the maximum allowable turbine inlet temperature was chosen for each of the five turbine blade materials included as discrete variable choices. The five materials used and their associated maximum turbine inlet temperatures are shown in Table 62.

Table 62: Turbine Blade Materials

Turbine Blade Material	*Maximum Allowable Turbine Inlet Temperature (R)*
Inconel 718	1870
Waspaloy	1976
Astroloy	2068
Inconel 713LC	2250
IN-100	2358

These five turbine blade material choices, along with the twenty-two different engine cycles, provide 110 discrete variable combinations for the final application. These are combined with four continuous input variables to form the mixed-integer input set. The four continuous variables used are vacuum thrust, oxidizer to fuel mixture ratio, nozzle expansion ratio and chamber pressure. Bounds are placed on each of these input variables to stop the optimization routine from evaluating an engine cycle at input variable settings outside of the range which guarantees the powerhead will balance. The continuous input variable bounds for each engine cycle are listed in Appendix B.

Lower bounding equations must be formed for the continuous design space of each of the 110 discrete variable inputs. The number of PHATCAT function calls required to form each lower bound space is 41. These calls include the points used to populate the central composite design and those used to get the output values at the ½ points. The total number of function calls required to get all the lower bounding equations is 4510. It takes just over 12 hrs to run all of these design points using a Dell Dimension 8200 PC with a 2.25 GHz, Pentium 4 processor and 512 MB of RAM. While this represents a significant amount of processing time, all of these runs are done before the engine design process and stored in a database file. As long as the database file remains intact, these cases do not have to be re-run and each time an engine design is optimized, the appropriate lower bounding data can be quickly accessed.

Like the proof-of-concept problems, this final application was solved using ModelCenter. The same lower bounding C-code used in the proof-of-concept was altered to allow its use with PHATCAT instead of SCORES. The nonlinear optimizers in ModelCenter were again used to solve for the α terms, the optimum of the lower bounding space, and the optimum of the true continuous variable space. The method of feasible directions was used to find the α terms and optimize the lower bounding equations, while sequential linear programming was used to optimize the true continuous design space. In order to optimize the true space, PHATCAT has to be evaluated many times. Depending on the engine cycle being analyzed, each run can take up to one minute to finish. Therefore, it is desirable to use the nonlinear optimization method that requires the least number of PHATCAT evaluations to find the optimum of the true continuous design space. Typically, sequential quadratic programming is the optimization method that performs best when the optimization function (in this case PHATCAT) doesn’t run very quickly. However, the SQP implementation in ModelCenter does not behave as expected. Many times it was found that SQP would not be able to find a feasible solution, when MoFD and SLP could. SLP was usually able to find the optimal solution with fewer PHATCAT function calls, compared to MoFD, and that is why it was the optimization method used to find the optimal continuous variable settings for the true design space.

7.2 Final Test Cases

Six different rocket engine optimization problems were investigated using both the new lower bounding method and the baseline method, which solves the continuous NLP optimization problem for all 110 discrete variable combinations. For five of the six cases, the objective of the optimization routine was to maximize the vacuum specific impulse of the engine (minimize the negative is I_{sp}). The objective of the final case was to minimize the engine's design, development, testing and evaluation cost (DDTE). The number of constraints on the designs range from zero to four and include equality constraints. Table 63 shows the details of the six cases solved.

Table 63: Six Optimization Problems Solved

Case	*Objective*	*Constraint 1*	*Constraint 2*	*Constraint 3*	*Constraint 4*
1	Max Isp_{vac}	N/A	N/A	N/A	N/A
2	Max Isp_{vac}	$Tsl \geq 395000 lbs$	$Tsl \leq 4500000 lbs$	N/A	N/A
3	Max Isp_{vac}	$Tsl \geq 395000 lbs$	$Tsl \leq 4500000 lbs$	$T_{vac} / W \geq 75$	N/A
4	Max Isp_{vac}	$Tsl \geq 395000 lbs$	$Tsl \leq 4500000 lbs$	$T_{vac} / W \geq 75$	$DDTE \leq \$695M$
5	Max Isp_{vac}	$Tsl \geq 395000 lbs$	$Tsl \leq 4500000 lbs$	$T_{vac} / W \geq 75$	$DDTE \leq \$665M$
6	Min DDTE	$Tsl = 400000 lbs$	$Isp_{sl} \geq 350 \sec$	$T_{vac} / W \geq 60$	N/A

For each discrete variable combination, there is an additional constraint on the maximum allowable turbine inlet temperature, which depends on the current selection for the turbine blade material. The maximum turbine inlet temperature is one of the outputs of PHATCAT for which a response surface and lower bounding equation is formed. This allows the lower bounding problem to determine if any continuous variable settings will provide an engine design with turbine inlet temperatures that do not exceed the allowable limit. It is possible that some engine cycles will be infeasible, even for Case 1, because low enough turbine temperatures could not be achieved. This is particularly true for the low-temperature turbine blade materials. However, it is not known beforehand which cycle/turbine blade material combinations will be infeasible. Therefore, all 110 combinations need to be investigated.

The constraint settings in Cases 1 – 5 were chosen to show the progression in the optimal engine design choice as the optimization problem becomes more constrained. This is meant to mimic the addition of more requirements on the engine design as the overall vehicle solution becomes more defined. The optimization problem, presented in Case 6, is meant to simulate the situation where a replacement engine is being sought for an existing application. Here, many of the design requirements are fixed and the propulsion engineer is charged with finding the cheapest solution to the problem.

The goal of the new optimization method is to be able to find the same optimal solution, obtained by optimizing the continuous space of all the discrete variable combinations, but in a much more time efficient manner. To test this goal, a baseline optimization time must be established by exhaustively optimizing all 110 true continuous subspaces for each case listed in Table 63. Then, the time required to find the optimum using the new lower bounding procedure is found. This time includes that needed to solve the lower bounding equations for every discrete variable combination, and the time required to find the optimum of the true continuous space for those discrete variable combinations providing the best lower bounds. The total time for the new method does not include the 12 hrs needed for preprocessing, since these calculations were completed before the optimization process began.

The first test of the new optimization method will be to see if it can get the same solution found by optimizing the continuous design space for each discrete variable combination. Table 64 shows the comparison of the optimal discrete variable combination found using both methods. This combination consists of the cycle choice and the turbine blade material. For all six cases, both methods chose the same optimal discrete variable combination.

Table 64: Optimal Discrete Variable Settings

Case	*Optimal Discrete Variable Setting (Baseline Method)*	*Optimal Discrete Variable Setting (New Method)*
1	RLX / Inconel 718	RLX / Inconel 718
2	SSME / Inconel 713LC	SSME / Inconel 713LC
3	SSME_1PB_1T / Inconel 713LC	SSME_1PB_1T / Inconel 713LC
4	RD180_CH4 / Inconel 718	RD180_CH4 / Inconel 718
5	RD180_RP1 / Inconel 718	RD180_RP1 / Inconel 718
6	Vulcain_LH2_1T_NBP / Waspaloy	Vulcain_LH2_1T_NBP / Waspaloy

For the same discrete variable combination, both methods will find the same optimal continuous variable settings. This is because, once a discrete variable setting is chosen for further investigation using the new method, the true continuous design space is solved in order to find the continuous variable settings. This is the same true continuous design space that is solved using the baseline method. Since the same nonlinear optimization method, (SLP) is used to solve for the optimum of the true design space in both methods, the same continuous variable settings are found. The continuous input variable settings of the optimal solution found for each of the six optimization cases are listed in Table 65. Also shown in the table are the values of the output variables that represent constraints for some of the optimization cases.

Table 65: Objective, Constraint, and Continuous Input Vales at Optimum

Case	*Obj.*	*(Cont. Input Variable Settings)*				*(Constraint Values at Optimum)*			
	Func. Value	*T_{vac} (klbs)*	*O/F*	*e*	*P_c (psia)*	*T_{sl} (klbs)*	*T/W*	*DDTE ($M)*	*I_{sp} (sec)*
1	472.9 sec	425.0	4.75	125	1750	276.2	92.9	577.2	307.3
2	464.0 sec	598.5	5.75	125	3500	449.6	70.2	930.8	348.6
3	461.7 sec	598.8	5.75	125	2850	449.8	76.6	891.1	346.8
4	371.6 sec	579.9	3.54	125	4500	431.0	75.7	695.0	276.2
5	355.2 sec	546.6	3.09	125	4500	397.8	83.2	665.0	258.5
6	$729.5M	417.9	5.10	15	1559	400.1	67.8	729.5	381.5

The first optimization problem is an unconstrained problem. The optimal engine cycle chosen by both methods is the RLX split-expander engine. The chamber pressure, expansion ratio, and vacuum thrust are set to their upper bound values, while the mixture ratio is set to its lower bound at the optimum. This split-expander cycle has a low turbine inlet temperature, so all five turbine blade material choices provide the same maximum

I_{sp} value. The discrete variable combination using the lowest temperature turbine blade material is listed as the solution because it has the least development costs. However, for this problem, all five blade materials provide an equally optimal solution.

The second case adds an upper and lower bound constraint on sea-level thrust. For this case, the RLX engine cycle provides an infeasible design, because it is not able to meet the lower bound constraint. The optimal discrete variable combination found is the SSME with Inconel 713LC. The SSME engine and two different turbine blade material combinations provide equally good solutions for this case and the lowest temperature material is listed as the solution for the same reasons described above. The chamber pressure and expansion ratio are set to their upper bounds, while the mixture ratio is again set to its lower bound. The vacuum thrust level was adjusted in order to meet the sea-level thrust upper bound constraint.

The addition of the vacuum T/W lower bound constraint in Case 3 eliminates the SSME cycle as an option. Therefore, the optimal discrete variable combination is the modified SSME cycle. This cycle only has one preburner and one main turbine. Therefore, this engine is given a weight reduction factor and it is consequently able to provide a feasible solution for Case 3. The expansion ratio and chamber pressure are again set to their upper bounds and the mixture ratio is at its lower bound. The vacuum thrust is once again set to meet the upper bound sea-level thrust constraint. Also, the two highest temperature turbine blade materials provide the same optimal solution and the lower temperature material, Inconel 713LC, is listed as the best blade material choice.

Case 4 adds an upper bound constraint on the allowable engine development cost, while maintaining the previous sea-level thrust and T/W constraints. The optimal discrete variable combination for this case is the CH4 version of the RD-180 using turbine blades made of Inconel 718. A development cost reduction is applied to engine designs using the lower temperature turbine blade materials, but a development cost increase is applied to engines using CH4 and C3H8 as their fuel choice. The increase in development cost, associated with fuel type, is included to reflect the limited experience in using these fuels in operational engines. The CH4 version of the RD-180 provides higher I_{sp} values when compared to the RP-1 version and the cost decrease associated with the low temperature turbine blade material off sets the increase associated with using CH4 and allows the engine to meet the new development cost constraints.

The fifth case is similar to Case 4, except that the allowable development cost has been lowered by $30M. The optimal discrete variable combination for this case is the baseline RD-180 cycle and Inconel 718. The cost constraint is too low for the CH4 version of the RD-180 to provide a feasible design. The baseline version of the RD-180 does not incur the development cost increase, since RP-1 is the fuel of this cycle. The lowest temperature turbine blade material must be used to meet the cost requirement, even though there is a performance decrease associated with the lower turbine inlet temperatures.

Cases 1 – 5 show the progression of the optimal value of the objective function as more constraints are added to the optimization problem. The optimal value of I_{sp} that is found by the optimization methods decreases as the problems become more constrained. This is because the optimization method is forced to choose lower performing engines to meet the increasingly stringent design constraints.

The final case has a different objective function than Cases 1 – 5. The objective of Case 6 is to minimize the development costs of an engine design subject to sea-level thrust, sea-level I_{sp} and vacuum thrust to weight constraints. The sea-level thrust constraint is actually an equality constraint. This was modeled in the optimization methods by including two separate constraints, one requiring T_{sl} to be greater than or equal to 400 klbs and the other requiring it to be less than or equal to 400 klbs. The optimal discrete variable combination for this case is an LH2 version of the Vulcain with no fuel boost pump and only one main turbine. This variation of the Vulcain was given a development cost decrease when compared to the dual turbine, boost pump version. However, there is also an associated decrease in the engine's performance. Even with the performance decrease, this engine is able to meet the constraints on performance imposed in the optimization problem.

Included in Appendix C, for each optimization case, are the values of the objective function obtained by optimizing the lower bounding space and true continuous space for each discrete variable combination. The discrete variable combinations whose true space had to be optimized to solve the problem using the new optimization method are highlighted in blue and bordered by dotted lines. The optimal discrete combination for each problem is noted by a solid border.

The final test of the new optimization method is to compare the solution times required for the new and baseline method. This comparison is given in Table 66. The times listed in the table are those required to solve the problem using a Dell Dimension 8200 PC with a 2.25 GHz, Pentium 4 processor and 512 MB of RAM. The time given for the baseline method is the time it takes to solve the true continuous design space for all 110 discrete variable combinations. The time required for the new solution method is sum of two separate values. The first is the time required to solve for the optimum of the lower bounding space for every discrete variable combination. The second value is the time it takes to solve the true continuous space of the discrete variable combinations identified by the new method as possible candidates for the optimal discrete setting. Table 66 also shows the percentage of the original baseline time required by the new method to solve the optimization problem.

Table 66: Comparison of Solution Times

Case	*# of NLPs Solved for Baseline Method*	*# of NLPs Solved for New Method*	*% of Baseline NLPs*	*Solution Time (Baseline Method)*	*Solution Time (New Method)*	*% of Baseline Time*
1	110	5	4.55%	20.56 hrs	1.06 hrs	5.15%
2	110	6	5.45%	34.32 hrs	1.49 hrs	4.33%
3	110	3	2.73%	50.60 hrs	1.13 hrs	2.23%
4	110	1	0.91%	88.14 hrs	1.16 hrs	1.32%
5	110	1	0.91%	98.66 hrs	1.04 hrs	1.05%
6	110	1	0.91%	31.36 hrs	1.29 hrs	4.11%

The new method is able to solve each of the optimization problems in significantly less time than that required by the baseline solution procedure. Optimization times are reduced from one to four days to less than two hours. This reduction in time does not sacrifice accuracy, since both methods find the same optimal solution. The times for the new optimization method shown in Table 66 do not include the 12 hrs of preprocessing needed to generate the data used to form the lower bounding procedure. This time was not included because the preprocessing is only done once and it is completed before the optimization process begins. However, if the 12 hrs are included in the time required to solve all six cases using the new method, a time savings of over 94% is still realized.

CHAPTER 8

CONCLUSIONS AND RECOMMENDATIONS

8.1 Conclusions

Three specific goals for this research were identified in Chapter 1. The first goal was to establish a baseline solution method and determine the applicability of existing optimization techniques. The second goal was to develop a new optimization method that could solve the mixed-input rocket engine optimization problem without requiring the relaxation of the discrete variables. The third goal required that this new method be able to optimize the problem of interest in a time frame appropriate for a conceptual design environment. The following paragraphs will illustrate how each of these goals was met.

The original contribution of this research was the development of a new optimization method that can solve mixed-input nonlinear optimization problems. This new optimization method allows the determination of the optimal liquid rocket engine design using a powerhead analysis code. This method is able to optimize problems having mixed-inputs but does not require the relaxation of the discrete input variables. The steps involved in the optimization method are outlined below.

1. Evaluate the engine design code at all points necessary to form the central composite design of experiments and at those points needed to determine the lower bounding equations of the continuous space. This requires that the engine design code be run $\left(2^n + \left(2^n + 2n + 1\right)\right) * y$ times, where n is the number of continuous input variables and y is the number of discrete variable combinations.
2. Form response surface equations of the continuous space ($\tilde{f}(\bar{x})$), which predict all the relevant outputs of the engine design code using the data from the central composite design of experiments.
3. Use the RSE's to make lower bounding equations ($L(\bar{x})$) that expand the feasible continuous design space. These equations are formed by adding two terms to the original response surface equations. The first shifts $\tilde{f}(\bar{x})$ down, so that at all the corner points, the value predicted by $\tilde{f}(\bar{x}) - d$ is less than or equal to the true value ($f(\bar{x})$). The second term transforms this shifted RSE into a lower bound by applying a convex underestimation term. The lower bounding equation is repeated here for convenience (Equation 37).

$$L(\bar{x}) = \boldsymbol{b}_o + \sum_{i=1}^{n} \boldsymbol{b}_i x_i + \sum_{i=1}^{n} \boldsymbol{b}_i x_i^2 + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \boldsymbol{b}_{ij} x_i x_j + \sum_{i=1}^{n} \boldsymbol{a}_i (-1 - x_i)(1 - x_i) - \boldsymbol{d} \quad (37)$$

4. For every discrete variable combination, solve the lower bounding equations to determine a lower bound of the optimal solution for that combination's continuous space.
5. Optimize the true continuous space of the discrete variable combination providing the best lower bound. This value provides an upper bound on the solution.
6. Evaluate the next best lower bound that is less than the current upper bound. If there are no lower bounds less than the current upper bound, the algorithm terminates and the current upper bound is the optimum. Steps five and six are repeated until the solution is found. Typically a maximum of 3 – 4 iterations are required before the optimal solution is found.

This new optimization method does not require that the discrete variables be relaxed, since a lower bounding space is formed for every discrete variable combination. This allows it to be used on engineering codes which can not be executed at non-discrete settings for some of their input values. The flow chart, first shown in Chapter 5, which outlines the solution process developed for this research, is presented again in Figure 38.

The new optimization procedure successfully integrates techniques from many different optimization fields. The basic strategy of using a set of non-increasing upper bounds and non-decreasing lower bounds to solve a mixed-input optimization problem is common to mixed-integer nonlinear programming methods. Local numerical optimization methods are used to find the optimum of the true continuous space and the lower bounding space for each discrete variable combination. Using design of experiments to form response surface equations is an approximation technique typically employed when trying to optimize time consuming analyses. Finally, the convex underestimator used to form the lower bounds was originally developed for use in the field of global optimization.

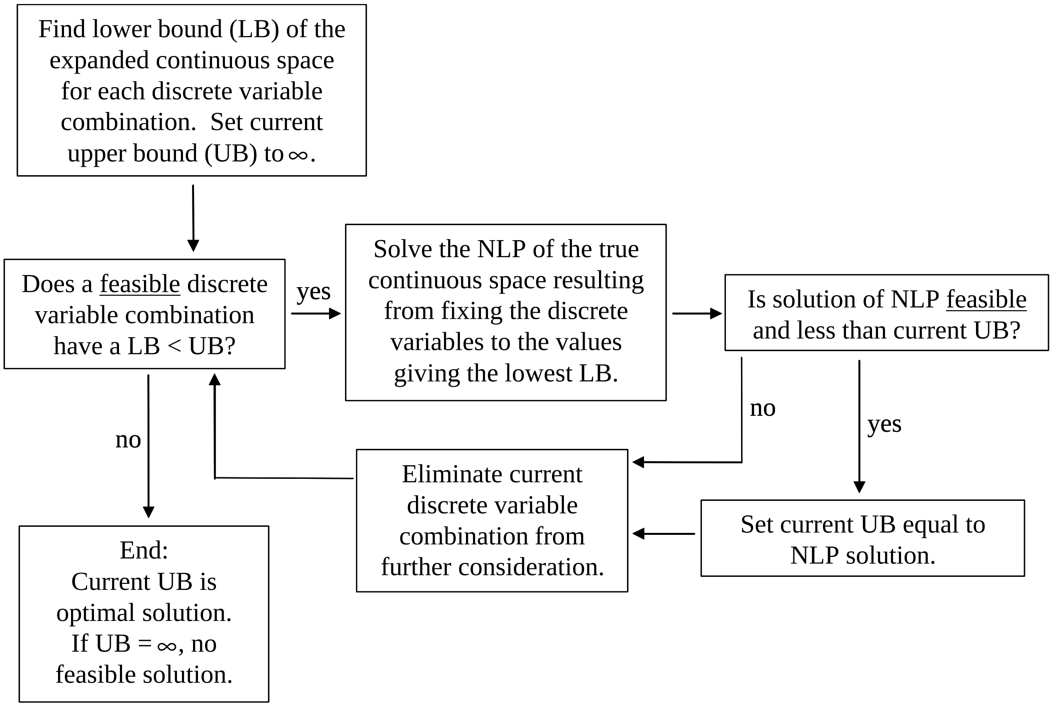

Find lower bound (LB) of the expanded continuous space for each discrete variable combination. Set current upper bound (UB) to ∞.
Does a feasible discrete variable combination have a LB < UB?
yes
Solve the NLP of the true continuous space resulting from fixing the discrete variables to the values giving the lowest LB.
Is solution of NLP feasible and less than current UB?
no
yes
no
Eliminate current discrete variable combination from further consideration.
Set current UB equal to NLP solution.
End: Current UB is optimal solution. If UB = ∞, no feasible solution.

Figure 38: Flowchart for New Optimization Procedure (Repeated)

The accuracy of the new optimization procedure was tested by comparing it to a baseline method also implemented for this research. The baseline method's solution strategy was to simply solve for the optimum of the true continuous design space for all the discrete variable combinations and then chose the best optimum as the final solution. This method was easily implemented, since the optimum of the continuous design space can be found using one of the many available constrained nonlinear optimization techniques. Six different optimization problems were solved by both the baseline method and the new method. The objective of five of the six problems was to maximize the specific impulse of the engine, while the objective for the sixth problem was to minimize the engine's development cost. The number of constraints for each of these problems varied from zero to four and included equality constraints. For all six cases, the optimal solution found by the new method agreed with the solution found using the baseline method.

The solution time required to solve the engine optimization problems using the baseline optimization method varied between one and four days for each of the six cases tested. This makes this method impractical for use in a conceptual design environment. The new optimization method, which was developed for use in such an environment, only takes between 1 to 1 ½ hours to optimize each of the six cases. Compared to the baseline optimization method, this represents an average time savings of 97% for each design case.

8.2 Comments and Recommendations

This new optimization method is flexible enough that it can be applied to other optimization problems not involving liquid rocket engine design. The method will work best, compared to other methods, on problems where the optimization values are supplied by an analysis with the following features.

- Mixed-Integer inputs where the discrete variables can not be relaxed
- A continuous variable design space that is smooth and free of discontinuities
- Objective and constraint values supplied by an analysis that is expensive (time consuming) to run

The new method makes no assumptions on the behavior of the discrete design space. Therefore, discrete design spaces that are highly nonlinear or even multi-modal pose no problem. Also, the new method is able to optimize problems where the continuous variables can be relaxed, however other methods that can take advantage of the relaxation might prove more efficient. The method is also able to optimize problems using fast running analyses, although the true potential of the new method is seen when used in conjunction with a more time consuming code. For this research it was assumed that the optimum of the true continuous design space could be found using sequential linear programming. However, if the continuous design space was not well-behaved, the local optimization technique (SLP) used to find the optimal solutions of the true continuous design spaces may fail. The new method could be adjusted to handle poorly behaved continuous spaces by using global optimization methods instead of local optimization methods and by using a different meta-model type as the basis for the lower bounding equations.

The response surface equations of the continuous spaces will most likely not provide a completely accurate representation of the true continuous spaces being modeled. Therefore, using just the original response surface equations to determine which discrete variable combinations to investigate further, could lead to the algorithm terminating at a sub-optimal solution. There are several potential ways in which the RSE’s could be adjusted in order to make it more likely that the true optimal solution will be found. For this research the RSE’s were turned into lower bounding equations using underestimation techniques. The ability to determine both upper and lower bounds of the optimum for each discrete variable combination is valuable because there is a body of knowledge regarding decomposition methods which make use of these bounds. Other possible ways to adjust the RSE’s do not fit into this body of knowledge and are therefore not as valuable.

The test problems solved in the final application included 110 distinct discrete variable combinations. However, this does not represent the upper limit that can be handled by the new procedure. This number could easily be expanded without having an adverse effect on the algorithm’s performance. In fact, as the number of discrete variable combinations is increased the new method should further outperform the baseline method because a larger percentage of the discrete variable combinations can be eliminated from further consideration as a result of their lowering bounding values. As the number of discrete variable combinations increase the amount of preprocessing time also increases. However, this increase is linear and therefore problems involving a large number of discrete variable combinations (possibly several hundred) may still be efficiently solved by the new method.

There were four continuous input variables for the final application of the new optimization method. This number could be expanded as needed to meet the input requirements of the analysis being optimized. However, the growth in the number of continuous variables has the potential to significantly increase the amount of preprocessing time required to form the lower bounding equations. This upfront investment of computational time must be manageable and this will most likely limit this new method to applications with a maximum of about ten continuous input variables.

This method could also have an application in the preliminary phase of design, where higher fidelity codes are typically used. The longer it takes the analysis code to execute, the fewer number of times it can be called during the optimization procedure, if a solution is desired in a timely fashion. If one is forced to use the baseline method to optimize the mixed-inputs of a high fidelity code, the optimization process might have to be avoided. However, the new optimization scheme only requires the solution of the true design space for a limited number of discrete variable combinations and this has the potential to greatly reduce the optimization time for a high fidelity code. The preprocessing time required to form the lower bounding equations will increase as the analysis code execution time increases, but this preprocessing only needs to be done once and can be completed off-line before the optimization process begins.

The new optimization method developed for this research was used to find the optimal liquid rocket engine design subject to various constraints. This provides an optimal solution but no measure of the sensitivity or robustness of the final answer.

Knowledge of these characteristics is desirable because many uncertainties exist in the early phases of design. One way to determine a more robust engine design is to use a Monte Carlo analysis, which evaluates the engine code several thousand times at different input settings that are determined by distributions on the input variables. This allows for confidence levels to be established for the final solution. Since the Monte Carlo analysis requires that the engine design code be run many times, this type of robust design is most likely limited to a faster running engine code such as SCORES. The new optimization method could be used in conjunction with a Monte Carlo analysis to find an optimal engine design, using SCORES, that meets a desired confidence level. If a more detailed engine design code, such as PHATCAT is used, a Monte Carlo analysis will not be practical. However, even with these slower running codes, sensitivities of the final solution with respect to the various input variables could be determined. These sensitivities can then be used to decide which aspects of the rocket engine warrant further attention as the design matures.

There exists the possibility of improving the efficiency of the new optimization method by changing the meta-modeling technique that is used as the basis for the lower bounding equations. The current response surface equations could be replaced by a different model, which might allow the lower bounds to more accurately represent the true design space. One alternate meta-modeling technique that might be explored is the use of neural networks [115,116]. Neural networks could offer an advantage since the mathematical form of the meta-model (i.e. choosing a 2nd order RSE) does not have to be specified beforehand. Possible drawbacks of using a neural network include the amount of information about the design space that is needed to sufficiently train the network.

The potential also exists to improve the efficiency and robustness of the new optimization method by monitoring the values produced while optimizing the true continuous design spaces and then comparing these values to the lower bound values at those points. This has two benefits. First, if a point in the design space is found where the lower bounding equations do not produce a true lower bound, the equations can be adjusted to provide the proper bounds at this design point. This would help make the method more robust. Conversely, if it is found that the lower bounding equations are severely underestimating the design space, they can be adjusted to more closely represent the true space. This adjustment will help increase the efficiency of the method. Over time, for a specific analysis, these two techniques could be used to refine the lower bounding equations and increase the method's overall performance.

APPENDIX A

PROOF – OF – CONCEPT CASE RESULTS

Table 67: Results for Proof-of-Concept - Case 1

Combination	$L(\overline{x})$ *for Ideal* I_{sp} *(sec)*	$f(\overline{x})$ *for Ideal* I_{sp} *(sec)*
1	479.0166	476.3058
2	380.0762	379.2494
3	363.7709	363.0051
4	355.9897	355.1814
5	353.3577	352.5439
6	356.3197	355.5087
7	352.3348	351.5297
8	369.5238	368.0115
9	368.8630	367.1405
10	372.5755	371.2316
11	373.9608	372.6618
12	373.8997	372.6352
13	372.3244	371.0496

Table 68: Results for Proof-of-Concept - Case 2

Combination	$L(\overline{x})$ *for Ideal* I_{sp} *(sec)*	$f(\overline{x})$ *for Ideal* I_{sp} *(sec)*
1	infeasible	infeasible
2	353.5836	351.8992
3	355.8054	354.0851
4	327.0032	325.0494
5	353.2061	352.1000
6	356.3152	355.1791
7	352.3348	351.4132
8	368.0744	368.0260
9	368.8487	367.1549
10	370.7559	370.5969
11	373.9608	372.6496
12	373.8997	372.6352
13	372.3244	371.0496

Table 69: Results for Proof-of-Concept - Case 3

Combination	$L(\overline{x})$ *for Ideal* I_{sp} *(sec)*	$f(\overline{x})$ *for Ideal* I_{sp} *(sec)*
1	infeasible	infeasible
2	infeasible	infeasible
3	infeasible	infeasible
4	infeasible	infeasible
5	337.0844	335.0312
6	347.5704	345.7721
7	351.5737	350.8995
8	infeasible	infeasible
9	infeasible	infeasible
10	infeasible	infeasible
11	373.5289	372.0338
12	373.8353	371.9839
13	372.3223	370.3816

Table 70: Results for Proof-of-Concept - Case 4

Combination	$L(\overline{x})$ *for Ideal* I_{sp} *(sec)*	$f(\overline{x})$ *for Ideal* I_{sp} *(sec)*
1	infeasible	infeasible
2	infeasible	infeasible
3	323.5633	322.0904
4	infeasible	infeasible
5	339.1115	337.4974
6	345.8222	343.3574
7	345.4972	342.5479
8	infeasible	infeasible
9	360.5188	358.3632
10	infeasible	infeasible
11	371.3709	365.5147
12	371.1657	366.6566
13	370.1574	366.7363

Table 71: Results for Proof-of-Concept - Case 5

Combination	$L(\bar{x})$ *for Ideal* I_{sp} *(sec)*	$f(\bar{x})$ *for Ideal* I_{sp} *(sec)*
1	464.1924	462.4594
2	363.6720	362.6143
3	348.2165	347.2133
4	340.6895	339.6517
5	337.2956	337.0605
6	340.1170	339.8688
7	336.3533	336.2290
8	356.3242	350.0854
9	357.9620	355.6865
10	360.7904	357.7760
11	365.2410	358.6827
12	363.3465	358.8387
13	362.3694	357.3051

Table 72: Results for Proof-of-Concept - Case 6

Combination	$L(\bar{x})$ *for Ideal* I_{sp} *(sec)*	$f(\bar{x})$ *for Ideal* I_{sp} *(sec)*
1	464.4992	462.6082
2	363.3671	362.7511
3	347.9542	347.3428
4	340.3828	339.7855
5	337.8098	337.1908
6	340.6285	340.0003
7	336.9488	336.3676
8	356.6278	350.4507
9	358.3828	355.7970
10	360.8482	357.8891
11	365.0047	358.7840
12	363.1426	358.9493
13	362.1946	357.4098

Table 73: Results for Proof-of-Concept - Case 7

Combination	$L(\bar{x})$ *for Ideal* I_{sp} *(sec)*	$f(\bar{x})$ *for Ideal* I_{sp} *(sec)*
1	470.1176	466.6425
2	369.3971	367.6127
3	353.6824	351.9483
4	346.0350	344.2982
5	343.2854	341.6823
6	346.1744	344.5318
7	342.3242	340.8128
8	360.9450	358.4752
9	362.8528	359.5017
10	365.1701	361.8100
11	368.4710	362.8184
12	367.0542	362.9297
13	365.9269	361.4098

Table 74: Results for Proof-of-Concept - Case 8

Combination	$L(\bar{x})$ *for Ideal* I_{sp} *(sec)*	$f(\bar{x})$ *for Ideal* I_{sp} *(sec)*
1	infeasible	infeasible
2	infeasible	infeasible
3	326.9773	325.2923
4	infeasible	infeasible
5	342.1558	340.6288
6	347.8846	345.1155
7	346.5635	344.1238
8	infeasible	infeasible
9	362.5682	359.8209
10	infeasible	infeasible
11	372.4407	366.8554
12	372.0730	367.8708
13	370.7460	367.7762

Table 75: Results for Proof-of-Concept - Case 9

Combination	$L(\bar{x})$ *for Ideal* I_{sp} *(sec)*	$f(\bar{x})$ *for Ideal* I_{sp} *(sec)*
1	infeasible	infeasible
2	infeasible	infeasible
3	340.4908	335.6902
4	infeasible	infeasible
5	349.7881	347.4321
6	354.2168	351.7300
7	352.1923	350.2133
8	infeasible	infeasible
9	367.6816	363.2751
10	infeasible	infeasible
11	373.9597	372.0077
12	373.8753	372.5209
13	372.3201	370.9242

Table 76: Results for Proof-of-Concept - Case 10

Combination	$L(\bar{x})$ *for Ideal* I_{sp} *(sec)*	$f(\bar{x})$ *for Ideal* I_{sp} *(sec)*
1	464.3736	459.9229
2	363.1333	361.6224
3	347.7168	346.3031
4	340.2295	338.7311
5	337.5242	336.1536
6	340.3359	338.9547
7	336.5650	335.3158
8	356.4762	346.1678
9	356.7792	354.9096
10	360.2676	356.9867
11	363.8689	357.8790
12	362.0663	358.0523
13	360.9506	356.5423

APPENDIX B

FINAL APPLICATION VARIABLE BOUNDS

Table 77: Continuous Input Variable Bounds (SSME)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	200000	750000
Expansion Ratio	15	125
Mixture Ratio	5.75	6.8
Chamber Pressure (psia)	1500	3500

Table 78: Continuous Input Variable Bounds (SSME_1PB_1T)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	275000	750000
Expansion Ratio	15	125
Mixture Ratio	5.75	6.8
Chamber Pressure (psia)	1500	2850

Table 79: Continuous Input Variable Bounds (Vulcain_LH2)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	80000	750000
Expansion Ratio	15	125
Mixture Ratio	5	7
Chamber Pressure (psia)	500	3500

Table 80: Continuous Input Variable Bounds (Vulcain_RP1)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	80000	750000
Expansion Ratio	15	125
Mixture Ratio	3	5
Chamber Pressure (psia)	500	2000

Table 81: Continuous Input Variable Bounds (Vulcain_CH4)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	80000	750000
Expansion Ratio	15	125
Mixture Ratio	3	5
Chamber Pressure (psia)	500	2000

Table 82: Continuous Input Variable Bounds (Vulcain_C3H8)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	80000	750000
Expansion Ratio	15	125
Mixture Ratio	3	5
Chamber Pressure (psia)	500	2000

Table 83: Continuous Input Variable Bounds (Vulcain_LH2_1T)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	80000	750000
Expansion Ratio	15	125
Mixture Ratio	5	7
Chamber Pressure (psia)	500	3500

Table 84: Continuous Input Variable Bounds (Vulcain_RP1_1T)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	80000	750000
Expansion Ratio	15	125
Mixture Ratio	3	5
Chamber Pressure (psia)	500	2000

Table 85: Continuous Input Variable Bounds (Vulcain_CH4_1T)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	80000	750000
Expansion Ratio	15	125
Mixture Ratio	3	5
Chamber Pressure (psia)	500	2000

Table 86: Continuous Input Variable Bounds (Vulcain_C3H8_1T)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	80000	750000
Expansion Ratio	15	125
Mixture Ratio	3	5
Chamber Pressure (psia)	500	2000

Table 87: Continuous Input Variable Bounds (Vulcain_LH2_NBP)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	80000	750000
Expansion Ratio	15	125
Mixture Ratio	5	7
Chamber Pressure (psia)	500	2500

Table 88: Continuous Input Variable Bounds (Vulcain_RP1_NBP)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	80000	750000
Expansion Ratio	15	125
Mixture Ratio	3	5
Chamber Pressure (psia)	500	2000

Table 89: Continuous Input Variable Bounds (Vulcain_CH4_NBP)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	80000	750000
Expansion Ratio	15	125
Mixture Ratio	3	5
Chamber Pressure (psia)	500	2000

Table 90: Continuous Input Variable Bounds (Vulcain_C3H8_NBP)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	80000	750000
Expansion Ratio	15	125
Mixture Ratio	3	5
Chamber Pressure (psia)	500	2000

Table 91: Continuous Input Variable Bounds (Vulcain_LH2_1T_NBP)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	80000	750000
Expansion Ratio	15	125
Mixture Ratio	5	7
Chamber Pressure (psia)	500	2500

Table 92: Continuous Input Variable Bounds (Vulcain_RP1_1T_NBP)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	80000	750000
Expansion Ratio	15	125
Mixture Ratio	3	5
Chamber Pressure (psia)	500	2000

Table 93: Continuous Input Variable Bounds (Vulcain_CH4_1T_NBP)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	80000	750000
Expansion Ratio	15	125
Mixture Ratio	3	5
Chamber Pressure (psia)	500	2000

Table 94: Continuous Input Variable Bounds (Vulcain_C3H8_1T_NBP)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	80000	750000
Expansion Ratio	15	125
Mixture Ratio	3	5
Chamber Pressure (psia)	500	2000

Table 95: Continuous Input Variable Bounds (RD180_RP1)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	300000	1250000
Expansion Ratio	15	125
Mixture Ratio	2	4
Chamber Pressure (psia)	2500	4500

Table 96: Continuous Input Variable Bounds (RD180_CH4)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	300000	1250000
Expansion Ratio	15	125
Mixture Ratio	3	5
Chamber Pressure (psia)	2500	4500

Table 97: Continuous Input Variable Bounds (RD180_C3H8)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	300000	1250000
Expansion Ratio	15	125
Mixture Ratio	3	5
Chamber Pressure (psia)	2500	4500

Table 98: Continuous Input Variable Bounds (RLX)

Continuous Input Variable	*Lower Bound*	*Upper Bound*
Vacuum Thrust (lbs)	220000	425000
Expansion Ratio	35	125
Mixture Ratio	4.75	6.5
Chamber Pressure (psia)	500	1750

APPENDIX C

FINAL APPLICATION CASE RESULTS

Table 99: Results for Final Application - Case 1

Discrete Variable Combination	*Engine Cycle*	*Maximum Allowable Turbine Inlet Temperature (R)*	*Solution of Lower Bounding Space*	*Solution of True Continuous Space*
1	Vulcain_LH2	1485	infeasible	infeasible
2	Vulcain_LH2	1870	infeasible	infeasible
3	Vulcain_LH2	1976	infeasible	infeasible
4	Vulcain_LH2	2068	462.06	453.09
5	Vulcain_LH2	2250	462.06	453.85
6	Vulcain_RP1	1485	infeasible	infeasible
7	Vulcain_RP1	1870	331.67	329.92
8	Vulcain_RP1	1976	331.67	329.92
9	Vulcain_RP1	2068	331.67	329.92
10	Vulcain_RP1	2250	331.67	329.92
11	Vulcain_CH4	1485	infeasible	infeasible
12	Vulcain_CH4	1870	infeasible	infeasible
13	Vulcain_CH4	1976	infeasible	infeasible
14	Vulcain_CH4	2068	360.49	354.91
15	Vulcain_CH4	2250	364.54	361.22
16	Vulcain_C3H8	1485	infeasible	infeasible
17	Vulcain_C3H8	1870	infeasible	infeasible
18	Vulcain_C3H8	1976	349.11	342.84
19	Vulcain_C3H8	2068	351.68	347.40
20	Vulcain_C3H8	2250	351.68	348.96
21	Vulcain_LH2_1T	1485	infeasible	infeasible
22	Vulcain_LH2_1T	1870	452.99	447.44
23	Vulcain_LH2_1T	1976	452.99	447.44
24	Vulcain_LH2_1T	2068	452.99	447.44
25	Vulcain_LH2_1T	2250	452.99	447.44
26	Vulcain_RP1_1T	1485	infeasible	infeasible
27	Vulcain_RP1_1T	1870	328.29	326.32
28	Vulcain_RP1_1T	1976	328.29	326.32
29	Vulcain_RP1_1T	2068	328.29	326.32
30	Vulcain_RP1_1T	2250	328.29	326.32
31	Vulcain_CH4_1T	1485	infeasible	infeasible
32	Vulcain_CH4_1T	1870	infeasible	infeasible
33	Vulcain_CH4_1T	1976	353.71	infeasible
34	Vulcain_CH4_1T	2068	359.12	353.12
35	Vulcain_CH4_1T	2250	360.47	357.21
36	Vulcain_C3H8_1T	1485	infeasible	infeasible
37	Vulcain_C3H8_1T	1870	340.90	infeasible
38	Vulcain_C3H8_1T	1976	346.96	341.58
39	Vulcain_C3H8_1T	2068	347.30	344.66
40	Vulcain_C3H8_1T	2250	347.31	344.66

Table 99: Results for Final Application - Case 1 (Continued)

Discrete Variable Combination	*Engine Cycle*	*Maximum Allowable Turbine Inlet Temperature (R)*	*Solution of Lower Bounding Space*	*Solution of True Continuous Space*
41	SSME	1485	infeasible	infeasible
42	SSME	1870	infeasible	infeasible
43	SSME	1976	468.24	463.35
44	SSME	2068	468.30	464.58
45	SSME	2250	468.30	464.58
46	Vulcain_LH2_NBP	1485	infeasible	infeasible
47	Vulcain_LH2_NBP	1870	infeasible	infeasible
48	Vulcain_LH2_NBP	1976	454.90	449.77
49	Vulcain_LH2_NBP	2068	454.90	449.77
50	Vulcain_LH2_NBP	2250	454.90	449.77
51	Vulcain_RP1_NBP	1485	infeasible	infeasible
52	Vulcain_RP1_NBP	1870	328.60	326.85
53	Vulcain_RP1_NBP	1976	328.60	326.85
54	Vulcain_RP1_NBP	2068	328.60	326.85
55	Vulcain_RP1_NBP	2250	328.60	326.85
56	Vulcain_CH4_NBP	1485	infeasible	infeasible
57	Vulcain_CH4_NBP	1870	infeasible	infeasible
58	Vulcain_CH4_NBP	1976	infeasible	infeasible
59	Vulcain_CH4_NBP	2068	357.98	350.99
60	Vulcain_CH4_NBP	2250	361.79	358.59
61	Vulcain_C3H8_NBP	1485	infeasible	infeasible
62	Vulcain_C3H8_NBP	1870	infeasible	infeasible
63	Vulcain_C3H8_NBP	1976	346.55	337.65
64	Vulcain_C3H8_NBP	2068	348.89	344.65
65	Vulcain_C3H8_NBP	2250	348.89	346.15
66	Vulcain_LH2_1T_NBP	1485	infeasible	infeasible
67	Vulcain_LH2_1T_NBP	1870	448.26	442.18
68	Vulcain_LH2_1T_NBP	1976	448.26	442.18
69	Vulcain_LH2_1T_NBP	2068	448.26	442.18
70	Vulcain_LH2_1T_NBP	2250	448.26	442.18
71	Vulcain_RP1_1T_NBP	1485	infeasible	infeasible
72	Vulcain_RP1_1T_NBP	1870	324.61	322.94
73	Vulcain_RP1_1T_NBP	1976	324.61	322.94
74	Vulcain_RP1_1T_NBP	2068	324.61	322.94
75	Vulcain_RP1_1T_NBP	2250	324.61	322.94
76	Vulcain_CH4_1T_NBP	1485	infeasible	infeasible
77	Vulcain_CH4_1T_NBP	1870	infeasible	infeasible
78	Vulcain_CH4_1T_NBP	1976	346.44	infeasible
79	Vulcain_CH4_1T_NBP	2068	356.09	350.57
80	Vulcain_CH4_1T_NBP	2250	357.63	354.46
81	Vulcain_C3H8_1T_NBP	1485	infeasible	infeasible
82	Vulcain_C3H8_1T_NBP	1870	338.20	333.55
83	Vulcain_C3H8_1T_NBP	1976	344.09	338.69
84	Vulcain_C3H8_1T_NBP	2068	344.43	341.87
85	Vulcain_C3H8_1T_NBP	2250	344.45	341.87
86	SSME_1PB_1T	1485	infeasible	infeasible
87	SSME_1PB_1T	1870	441.57	infeasible
88	SSME_1PB_1T	1976	466.01	462.22
89	SSME_1PB_1T	2068	466.01	462.50
90	SSME_1PB_1T	2250	466.01	462.50
91	RD180_RP1	1485	367.81	355.53
92	RD180_RP1	1870	369.23	362.83
93	RD180_RP1	1976	369.23	362.92
94	RD180_RP1	2068	369.23	362.92
95	RD180_RP1	2250	369.23	362.92
96	RD180_CH4	1485	377.05	372.22
97	RD180_CH4	1870	381.48	377.46

Table 99: Results for Final Application - Case 1 (Continued)

Discrete Variable Combination	*Engine Cycle*	*Maximum Allowable Turbine Inlet Temperature (R)*	*Solution of Lower Bounding Space*	*Solution of True Continuous Space*
98	RD180_CH4	1976	381.48	377.46
99	RD180_CH4	2068	381.48	377.46
100	RD180_CH4	2250	381.48	377.46
101	RD180_C3H8	1485	367.14	362.92
102	RD180_C3H8	1870	370.35	367.47
103	RD180_C3H8	1976	370.35	367.47
104	RD180_C3H8	2068	370.35	367.47
105	RD180_C3H8	2250	370.35	367.47
106	RLX	1485	474.23	472.88
107	RLX	1870	474.23	472.88
108	RLX	1976	474.23	472.88
109	RLX	2068	474.23	472.88
110	RLX	2250	474.23	472.88

Table 100: Results for Final Application - Case 2

Discrete Variable Combination	*Engine Cycle*	*Maximum Allowable Turbine Inlet Temperature (R)*	*Solution of Lower Bounding Space*	*Solution of True Continuous Space*
1	Vulcain_LH2	1485	infeasible	infeasible
2	Vulcain_LH2	1870	infeasible	infeasible
3	Vulcain_LH2	1976	infeasible	infeasible
4	Vulcain_LH2	2068	461.97	452.13
5	Vulcain_LH2	2250	461.97	453.20
6	Vulcain_RP1	1485	infeasible	infeasible
7	Vulcain_RP1	1870	332.01	329.74
8	Vulcain_RP1	1976	332.01	329.74
9	Vulcain_RP1	2068	332.01	329.74
10	Vulcain_RP1	2250	332.01	329.74
11	Vulcain_CH4	1485	infeasible	infeasible
12	Vulcain_CH4	1870	infeasible	infeasible
13	Vulcain_CH4	1976	infeasible	infeasible
14	Vulcain_CH4	2068	360.52	354.59
15	Vulcain_CH4	2250	364.56	360.88
16	Vulcain_C3H8	1485	infeasible	infeasible
17	Vulcain_C3H8	1870	infeasible	infeasible
18	Vulcain_C3H8	1976	349.21	342.62
19	Vulcain_C3H8	2068	351.76	347.18
20	Vulcain_C3H8	2250	351.76	348.62
21	Vulcain_LH2_1T	1485	infeasible	infeasible
22	Vulcain_LH2_1T	1870	453.12	446.78
23	Vulcain_LH2_1T	1976	453.12	446.80
24	Vulcain_LH2_1T	2068	453.12	446.80
25	Vulcain_LH2_1T	2250	453.12	446.80
26	Vulcain_RP1_1T	1485	infeasible	infeasible
27	Vulcain_RP1_1T	1870	328.38	326.14
28	Vulcain_RP1_1T	1976	328.38	326.14
29	Vulcain_RP1_1T	2068	328.38	326.14
30	Vulcain_RP1_1T	2250	328.38	326.14
31	Vulcain_CH4_1T	1485	infeasible	infeasible
32	Vulcain_CH4_1T	1870	infeasible	infeasible
33	Vulcain_CH4_1T	1976	352.30	infeasible
34	Vulcain_CH4_1T	2068	359.25	353.03
35	Vulcain_CH4_1T	2250	360.45	356.86
36	Vulcain_C3H8_1T	1485	infeasible	infeasible
37	Vulcain_C3H8_1T	1870	338.27	infeasible
38	Vulcain_C3H8_1T	1976	347.00	341.09
39	Vulcain_C3H8_1T	2068	347.39	344.32
40	Vulcain_C3H8_1T	2250	347.39	344.32
41	SSME	1485	infeasible	infeasible
42	SSME	1870	infeasible	infeasible
43	SSME	1976	468.18	462.41
44	SSME	2068	468.19	464.03
45	SSME	2250	468.19	464.03
46	Vulcain_LH2_NBP	1485	infeasible	infeasible
47	Vulcain_LH2_NBP	1870	infeasible	infeasible
48	Vulcain_LH2_NBP	1976	454.83	449.26
49	Vulcain_LH2_NBP	2068	454.83	449.26
50	Vulcain_LH2_NBP	2250	454.84	449.26
51	Vulcain_RP1_NBP	1485	infeasible	infeasible
52	Vulcain_RP1_NBP	1870	328.92	326.67
53	Vulcain_RP1_NBP	1976	328.92	326.67
54	Vulcain_RP1_NBP	2068	328.92	326.67
55	Vulcain_RP1_NBP	2250	328.92	326.67
56	Vulcain_CH4_NBP	1485	infeasible	infeasible
57	Vulcain_CH4_NBP	1870	infeasible	infeasible

Table 100: Results for Final Application - Case 2 (Continued)

Discrete Variable Combination	*Engine Cycle*	*Maximum Allowable Turbine Inlet Temperature (R)*	*Solution of Lower Bounding Space*	*Solution of True Continuous Space*
58	Vulcain_CH4_NBP	1976	infeasible	infeasible
59	Vulcain_CH4_NBP	2068	357.99	352.38
60	Vulcain_CH4_NBP	2250	361.90	357.49
61	Vulcain_C3H8_NBP	1485	infeasible	infeasible
62	Vulcain_C3H8_NBP	1870	infeasible	infeasible
63	Vulcain_C3H8_NBP	1976	346.67	340.46
64	Vulcain_C3H8_NBP	2068	348.85	344.47
65	Vulcain_C3H8_NBP	2250	348.88	345.84
66	Vulcain_LH2_1T_NBP	1485	infeasible	infeasible
67	Vulcain_LH2_1T_NBP	1870	448.21	441.67
68	Vulcain_LH2_1T_NBP	1976	448.21	441.68
69	Vulcain_LH2_1T_NBP	2068	448.21	441.68
70	Vulcain_LH2_1T_NBP	2250	448.21	441.68
71	Vulcain_RP1_1T_NBP	1485	infeasible	infeasible
72	Vulcain_RP1_1T_NBP	1870	324.94	322.73
73	Vulcain_RP1_1T_NBP	1976	324.94	322.73
74	Vulcain_RP1_1T_NBP	2068	324.94	322.73
75	Vulcain_RP1_1T_NBP	2250	324.94	322.73
76	Vulcain_CH4_1T_NBP	1485	infeasible	infeasible
77	Vulcain_CH4_1T_NBP	1870	infeasible	infeasible
78	Vulcain_CH4_1T_NBP	1976	344.94	infeasible
79	Vulcain_CH4_1T_NBP	2068	356.19	349.97
80	Vulcain_CH4_1T_NBP	2250	357.62	354.12
81	Vulcain_C3H8_1T_NBP	1485	infeasible	infeasible
82	Vulcain_C3H8_1T_NBP	1870	336.62	333.38
83	Vulcain_C3H8_1T_NBP	1976	344.13	338.27
84	Vulcain_C3H8_1T_NBP	2068	344.53	341.55
85	Vulcain_C3H8_1T_NBP	2250	344.53	341.55
86	SSME_1PB_1T	1485	infeasible	infeasible
87	SSME_1PB_1T	1870	infeasible	infeasible
88	SSME_1PB_1T	1976	466.00	461.16
89	SSME_1PB_1T	2068	466.00	461.68
90	SSME_1PB_1T	2250	466.00	461.68
91	RD180_RP1	1485	367.09	355.26
92	RD180_RP1	1870	368.52	362.61
93	RD180_RP1	1976	368.52	362.65
94	RD180_RP1	2068	368.52	362.24
95	RD180_RP1	2250	368.52	362.24
96	RD180_CH4	1485	376.35	371.66
97	RD180_CH4	1870	380.81	376.97
98	RD180_CH4	1976	380.81	376.97
99	RD180_CH4	2068	380.81	376.97
100	RD180_CH4	2250	380.81	376.97
101	RD180_C3H8	1485	366.48	362.53
102	RD180_C3H8	1870	369.95	367.26
103	RD180_C3H8	1976	369.95	367.26
104	RD180_C3H8	2068	369.95	367.26
105	RD180_C3H8	2250	369.95	367.26
106	RLX	1485	infeasible	infeasible
107	RLX	1870	infeasible	infeasible
108	RLX	1976	infeasible	infeasible
109	RLX	2068	infeasible	infeasible
110	RLX	2250	infeasible	infeasible

Table 101: Results for Final Application - Case 3

Discrete Variable Combination	*Engine Cycle*	*Maximum Allowable Turbine Inlet Temperature (R)*	*Solution of Lower Bounding Space*	*Solution of True Continuous Space*
1	Vulcain_LH2	1485	infeasible	infeasible
2	Vulcain_LH2	1870	infeasible	infeasible
3	Vulcain_LH2	1976	infeasible	infeasible
4	Vulcain_LH2	2068	infeasible	infeasible
5	Vulcain_LH2	2250	infeasible	infeasible
6	Vulcain_RP1	1485	infeasible	infeasible
7	Vulcain_RP1	1870	332.07	329.74
8	Vulcain_RP1	1976	332.07	329.74
9	Vulcain_RP1	2068	332.07	329.74
10	Vulcain_RP1	2250	332.07	329.74
11	Vulcain_CH4	1485	infeasible	infeasible
12	Vulcain_CH4	1870	infeasible	infeasible
13	Vulcain_CH4	1976	infeasible	infeasible
14	Vulcain_CH4	2068	360.22	354.59
15	Vulcain_CH4	2250	364.69	360.88
16	Vulcain_C3H8	1485	infeasible	infeasible
17	Vulcain_C3H8	1870	infeasible	infeasible
18	Vulcain_C3H8	1976	348.76	342.62
19	Vulcain_C3H8	2068	351.79	347.18
20	Vulcain_C3H8	2250	351.80	348.61
21	Vulcain_LH2_1T	1485	infeasible	infeasible
22	Vulcain_LH2_1T	1870	infeasible	infeasible
23	Vulcain_LH2_1T	1976	infeasible	infeasible
24	Vulcain_LH2_1T	2068	infeasible	infeasible
25	Vulcain_LH2_1T	2250	infeasible	infeasible
26	Vulcain_RP1_1T	1485	infeasible	infeasible
27	Vulcain_RP1_1T	1870	328.45	326.14
28	Vulcain_RP1_1T	1976	328.45	326.14
29	Vulcain_RP1_1T	2068	328.45	326.14
30	Vulcain_RP1_1T	2250	328.45	326.14
31	Vulcain_CH4_1T	1485	infeasible	infeasible
32	Vulcain_CH4_1T	1870	infeasible	infeasible
33	Vulcain_CH4_1T	1976	353.74	infeasible
34	Vulcain_CH4_1T	2068	359.07	353.03
35	Vulcain_CH4_1T	2250	360.48	356.87
36	Vulcain_C3H8_1T	1485	infeasible	infeasible
37	Vulcain_C3H8_1T	1870	340.40	infeasible
38	Vulcain_C3H8_1T	1976	346.92	341.09
39	Vulcain_C3H8_1T	2068	347.42	344.32
40	Vulcain_C3H8_1T	2250	347.43	344.33
41	SSME	1485	infeasible	infeasible
42	SSME	1870	infeasible	infeasible
43	SSME	1976	infeasible	infeasible
44	SSME	2068	infeasible	infeasible
45	SSME	2250	infeasible	infeasible
46	Vulcain_LH2_NBP	1485	infeasible	infeasible
47	Vulcain_LH2_NBP	1870	infeasible	infeasible
48	Vulcain_LH2_NBP	1976	infeasible	infeasible
49	Vulcain_LH2_NBP	2068	infeasible	infeasible
50	Vulcain_LH2_NBP	2250	infeasible	infeasible
51	Vulcain_RP1_NBP	1485	infeasible	infeasible
52	Vulcain_RP1_NBP	1870	328.97	326.67
53	Vulcain_RP1_NBP	1976	328.98	326.67
54	Vulcain_RP1_NBP	2068	328.98	326.67
55	Vulcain_RP1_NBP	2250	328.98	326.67
56	Vulcain_CH4_NBP	1485	infeasible	infeasible
57	Vulcain_CH4_NBP	1870	infeasible	infeasible

Table 101: Results for Final Application - Case 3 (Continued)

Discrete Variable Combination	*Engine Cycle*	*Maximum Allowable Turbine Inlet Temperature (R)*	*Solution of Lower Bounding Space*	*Solution of True Continuous Space*
58	Vulcain_CH4_NBP	1976	infeasible	infeasible
59	Vulcain_CH4_NBP	2068	357.95	352.38
60	Vulcain_CH4_NBP	2250	361.92	357.49
61	Vulcain_C3H8_NBP	1485	infeasible	infeasible
62	Vulcain_C3H8_NBP	1870	infeasible	infeasible
63	Vulcain_C3H8_NBP	1976	346.29	340.43
64	Vulcain_C3H8_NBP	2068	348.93	344.69
65	Vulcain_C3H8_NBP	2250	348.93	345.84
66	Vulcain_LH2_1T_NBP	1485	infeasible	infeasible
67	Vulcain_LH2_1T_NBP	1870	448.27	infeasible
68	Vulcain_LH2_1T_NBP	1976	448.27	infeasible
69	Vulcain_LH2_1T_NBP	2068	448.27	infeasible
70	Vulcain_LH2_1T_NBP	2250	448.27	infeasible
71	Vulcain_RP1_1T_NBP	1485	infeasible	infeasible
72	Vulcain_RP1_1T_NBP	1870	325.00	322.73
73	Vulcain_RP1_1T_NBP	1976	325.00	322.73
74	Vulcain_RP1_1T_NBP	2068	325.00	322.73
75	Vulcain_RP1_1T_NBP	2250	325.00	322.73
76	Vulcain_CH4_1T_NBP	1485	infeasible	infeasible
77	Vulcain_CH4_1T_NBP	1870	infeasible	infeasible
78	Vulcain_CH4_1T_NBP	1976	350.89	infeasible
79	Vulcain_CH4_1T_NBP	2068	355.96	349.97
80	Vulcain_CH4_1T_NBP	2250	357.64	354.12
81	Vulcain_C3H8_1T_NBP	1485	infeasible	infeasible
82	Vulcain_C3H8_1T_NBP	1870	337.70	333.38
83	Vulcain_C3H8_1T_NBP	1976	344.06	338.31
84	Vulcain_C3H8_1T_NBP	2068	344.56	341.55
85	Vulcain_C3H8_1T_NBP	2250	344.56	341.55
86	SSME_1PB_1T	1485	infeasible	infeasible
87	SSME_1PB_1T	1870	infeasible	infeasible
88	SSME_1PB_1T	1976	466.03	461.02
89	SSME_1PB_1T	2068	466.03	461.69
90	SSME_1PB_1T	2250	466.03	461.69
91	RD180_RP1	1485	367.28	355.26
92	RD180_RP1	1870	368.54	362.61
93	RD180_RP1	1976	368.54	362.65
94	RD180_RP1	2068	368.54	362.65
95	RD180_RP1	2250	368.54	362.65
96	RD180_CH4	1485	376.35	371.66
97	RD180_CH4	1870	380.83	376.91
98	RD180_CH4	1976	380.83	376.91
99	RD180_CH4	2068	380.83	376.91
100	RD180_CH4	2250	380.83	376.91
101	RD180_C3H8	1485	366.45	362.53
102	RD180_C3H8	1870	370.03	367.26
103	RD180_C3H8	1976	370.03	367.26
104	RD180_C3H8	2068	370.03	367.26
105	RD180_C3H8	2250	370.03	367.26
106	RLX	1485	infeasible	infeasible
107	RLX	1870	infeasible	infeasible
108	RLX	1976	infeasible	infeasible
109	RLX	2068	infeasible	infeasible
110	RLX	2250	infeasible	infeasible

Table 102: Results for Final Application - Case 4

Discrete Variable Combination	*Engine Cycle*	*Maximum Allowable Turbine Inlet Temperature (R)*	*Solution of Lower Bounding Space*	*Solution of True Continuous Space*
1	Vulcain_LH2	1485	infeasible	infeasible
2	Vulcain_LH2	1870	infeasible	infeasible
3	Vulcain_LH2	1976	infeasible	infeasible
4	Vulcain_LH2	2068	infeasible	infeasible
5	Vulcain_LH2	2250	infeasible	infeasible
6	Vulcain_RP1	1485	infeasible	infeasible
7	Vulcain_RP1	1870	308.38	302.24
8	Vulcain_RP1	1976	infeasible	infeasible
9	Vulcain_RP1	2068	infeasible	infeasible
10	Vulcain_RP1	2250	infeasible	infeasible
11	Vulcain_CH4	1485	infeasible	infeasible
12	Vulcain_CH4	1870	infeasible	infeasible
13	Vulcain_CH4	1976	infeasible	infeasible
14	Vulcain_CH4	2068	infeasible	infeasible
15	Vulcain_CH4	2250	infeasible	infeasible
16	Vulcain_C3H8	1485	infeasible	infeasible
17	Vulcain_C3H8	1870	infeasible	infeasible
18	Vulcain_C3H8	1976	infeasible	infeasible
19	Vulcain_C3H8	2068	infeasible	infeasible
20	Vulcain_C3H8	2250	infeasible	infeasible
21	Vulcain_LH2_1T	1485	infeasible	infeasible
22	Vulcain_LH2_1T	1870	infeasible	infeasible
23	Vulcain_LH2_1T	1976	infeasible	infeasible
24	Vulcain_LH2_1T	2068	infeasible	infeasible
25	Vulcain_LH2_1T	2250	infeasible	infeasible
26	Vulcain_RP1_1T	1485	infeasible	infeasible
27	Vulcain_RP1_1T	1870	318.51	314.24
28	Vulcain_RP1_1T	1976	infeasible	infeasible
29	Vulcain_RP1_1T	2068	infeasible	infeasible
30	Vulcain_RP1_1T	2250	infeasible	infeasible
31	Vulcain_CH4_1T	1485	infeasible	infeasible
32	Vulcain_CH4_1T	1870	infeasible	infeasible
33	Vulcain_CH4_1T	1976	infeasible	infeasible
34	Vulcain_CH4_1T	2068	infeasible	infeasible
35	Vulcain_CH4_1T	2250	infeasible	infeasible
36	Vulcain_C3H8_1T	1485	infeasible	infeasible
37	Vulcain_C3H8_1T	1870	infeasible	infeasible
38	Vulcain_C3H8_1T	1976	infeasible	infeasible
39	Vulcain_C3H8_1T	2068	infeasible	infeasible
40	Vulcain_C3H8_1T	2250	infeasible	infeasible
41	SSME	1485	infeasible	infeasible
42	SSME	1870	infeasible	infeasible
43	SSME	1976	infeasible	infeasible
44	SSME	2068	infeasible	infeasible
45	SSME	2250	infeasible	infeasible
46	Vulcain_LH2_NBP	1485	infeasible	infeasible
47	Vulcain_LH2_NBP	1870	infeasible	infeasible
48	Vulcain_LH2_NBP	1976	infeasible	infeasible
49	Vulcain_LH2_NBP	2068	infeasible	infeasible
50	Vulcain_LH2_NBP	2250	infeasible	infeasible
51	Vulcain_RP1_NBP	1485	infeasible	infeasible
52	Vulcain_RP1_NBP	1870	317.35	313.14
53	Vulcain_RP1_NBP	1976	infeasible	infeasible
54	Vulcain_RP1_NBP	2068	infeasible	infeasible
55	Vulcain_RP1_NBP	2250	infeasible	infeasible
56	Vulcain_CH4_NBP	1485	infeasible	infeasible
57	Vulcain_CH4_NBP	1870	infeasible	infeasible

Table 102: Results for Final Application - Case 4 (Continued)

Discrete Variable Combination	*Engine Cycle*	*Maximum Allowable Turbine Inlet Temperature (R)*	*Solution of Lower Bounding Space*	*Solution of True Continuous Space*
58	Vulcain_CH4_NBP	1976	infeasible	infeasible
59	Vulcain_CH4_NBP	2068	infeasible	infeasible
60	Vulcain_CH4_NBP	2250	infeasible	infeasible
61	Vulcain_C3H8_NBP	1485	infeasible	infeasible
62	Vulcain_C3H8_NBP	1870	infeasible	infeasible
63	Vulcain_C3H8_NBP	1976	infeasible	infeasible
64	Vulcain_C3H8_NBP	2068	infeasible	infeasible
65	Vulcain_C3H8_NBP	2250	infeasible	infeasible
66	Vulcain_LH2_1T_NBP	1485	infeasible	infeasible
67	Vulcain_LH2_1T_NBP	1870	infeasible	infeasible
68	Vulcain_LH2_1T_NBP	1976	infeasible	infeasible
69	Vulcain_LH2_1T_NBP	2068	infeasible	infeasible
70	Vulcain_LH2_1T_NBP	2250	infeasible	infeasible
71	Vulcain_RP1_1T_NBP	1485	infeasible	infeasible
72	Vulcain_RP1_1T_NBP	1870	320.74	314.77
73	Vulcain_RP1_1T_NBP	1976	infeasible	infeasible
74	Vulcain_RP1_1T_NBP	2068	infeasible	infeasible
75	Vulcain_RP1_1T_NBP	2250	infeasible	infeasible
76	Vulcain_CH4_1T_NBP	1485	infeasible	infeasible
77	Vulcain_CH4_1T_NBP	1870	infeasible	infeasible
78	Vulcain_CH4_1T_NBP	1976	infeasible	infeasible
79	Vulcain_CH4_1T_NBP	2068	infeasible	infeasible
80	Vulcain_CH4_1T_NBP	2250	infeasible	infeasible
81	Vulcain_C3H8_1T_NBP	1485	infeasible	infeasible
82	Vulcain_C3H8_1T_NBP	1870	infeasible	infeasible
83	Vulcain_C3H8_1T_NBP	1976	infeasible	infeasible
84	Vulcain_C3H8_1T_NBP	2068	infeasible	infeasible
85	Vulcain_C3H8_1T_NBP	2250	infeasible	infeasible
86	SSME_1PB_1T	1485	infeasible	infeasible
87	SSME_1PB_1T	1870	infeasible	infeasible
88	SSME_1PB_1T	1976	infeasible	infeasible
89	SSME_1PB_1T	2068	infeasible	infeasible
90	SSME_1PB_1T	2250	infeasible	infeasible
91	RD180_RP1	1485	367.28	355.26
92	RD180_RP1	1870	368.40	358.99
93	RD180_RP1	1976	infeasible	infeasible
94	RD180_RP1	2068	infeasible	infeasible
95	RD180_RP1	2250	infeasible	infeasible
96	RD180_CH4	1485	376.36	371.64
97	RD180_CH4	1870	infeasible	infeasible
98	RD180_CH4	1976	infeasible	infeasible
99	RD180_CH4	2068	infeasible	infeasible
100	RD180_CH4	2250	infeasible	infeasible
101	RD180_C3H8	1485	366.34	361.70
102	RD180_C3H8	1870	infeasible	infeasible
103	RD180_C3H8	1976	infeasible	infeasible
104	RD180_C3H8	2068	infeasible	infeasible
105	RD180_C3H8	2250	infeasible	infeasible
106	RLX	1485	infeasible	infeasible
107	RLX	1870	infeasible	infeasible
108	RLX	1976	infeasible	infeasible
109	RLX	2068	infeasible	infeasible
110	RLX	2250	infeasible	infeasible

Table 103: Results for Final Application - Case 5

Discrete Variable Combination	*Engine Cycle*	*Maximum Allowable Turbine Inlet Temperature (R)*	*Solution of Lower Bounding Space*	*Solution of True Continuous Space*
1	Vulcain_LH2	1485	infeasible	infeasible
2	Vulcain_LH2	1870	infeasible	infeasible
3	Vulcain_LH2	1976	infeasible	infeasible
4	Vulcain_LH2	2068	infeasible	infeasible
5	Vulcain_LH2	2250	infeasible	infeasible
6	Vulcain_RP1	1485	infeasible	infeasible
7	Vulcain_RP1	1870	infeasible	infeasible
8	Vulcain_RP1	1976	infeasible	infeasible
9	Vulcain_RP1	2068	infeasible	infeasible
10	Vulcain_RP1	2250	infeasible	infeasible
11	Vulcain_CH4	1485	infeasible	infeasible
12	Vulcain_CH4	1870	infeasible	infeasible
13	Vulcain_CH4	1976	infeasible	infeasible
14	Vulcain_CH4	2068	infeasible	infeasible
15	Vulcain_CH4	2250	infeasible	infeasible
16	Vulcain_C3H8	1485	infeasible	infeasible
17	Vulcain_C3H8	1870	infeasible	infeasible
18	Vulcain_C3H8	1976	infeasible	infeasible
19	Vulcain_C3H8	2068	infeasible	infeasible
20	Vulcain_C3H8	2250	infeasible	infeasible
21	Vulcain_LH2_1T	1485	infeasible	infeasible
22	Vulcain_LH2_1T	1870	infeasible	infeasible
23	Vulcain_LH2_1T	1976	infeasible	infeasible
24	Vulcain_LH2_1T	2068	infeasible	infeasible
25	Vulcain_LH2_1T	2250	infeasible	infeasible
26	Vulcain_RP1_1T	1485	infeasible	infeasible
27	Vulcain_RP1_1T	1870	infeasible	infeasible
28	Vulcain_RP1_1T	1976	infeasible	infeasible
29	Vulcain_RP1_1T	2068	infeasible	infeasible
30	Vulcain_RP1_1T	2250	infeasible	infeasible
31	Vulcain_CH4_1T	1485	infeasible	infeasible
32	Vulcain_CH4_1T	1870	infeasible	infeasible
33	Vulcain_CH4_1T	1976	infeasible	infeasible
34	Vulcain_CH4_1T	2068	infeasible	infeasible
35	Vulcain_CH4_1T	2250	infeasible	infeasible
36	Vulcain_C3H8_1T	1485	infeasible	infeasible
37	Vulcain_C3H8_1T	1870	infeasible	infeasible
38	Vulcain_C3H8_1T	1976	infeasible	infeasible
39	Vulcain_C3H8_1T	2068	infeasible	infeasible
40	Vulcain_C3H8_1T	2250	infeasible	infeasible
41	SSME	1485	infeasible	infeasible
42	SSME	1870	infeasible	infeasible
43	SSME	1976	infeasible	infeasible
44	SSME	2068	infeasible	infeasible
45	SSME	2250	infeasible	infeasible
46	Vulcain_LH2_NBP	1485	infeasible	infeasible
47	Vulcain_LH2_NBP	1870	infeasible	infeasible
48	Vulcain_LH2_NBP	1976	infeasible	infeasible
49	Vulcain_LH2_NBP	2068	infeasible	infeasible
50	Vulcain_LH2_NBP	2250	infeasible	infeasible
51	Vulcain_RP1_NBP	1485	infeasible	infeasible
52	Vulcain_RP1_NBP	1870	infeasible	infeasible
53	Vulcain_RP1_NBP	1976	infeasible	infeasible
54	Vulcain_RP1_NBP	2068	infeasible	infeasible
55	Vulcain_RP1_NBP	2250	infeasible	infeasible
56	Vulcain_CH4_NBP	1485	infeasible	infeasible
57	Vulcain_CH4_NBP	1870	infeasible	infeasible

Table 103: Results for Final Application - Case 5 (Continued)

Discrete Variable Combination	*Engine Cycle*	*Maximum Allowable Turbine Inlet Temperature (R)*	*Solution of Lower Bounding Space*	*Solution of True Continuous Space*
58	Vulcain_CH4_NBP	1976	infeasible	infeasible
59	Vulcain_CH4_NBP	2068	infeasible	infeasible
60	Vulcain_CH4_NBP	2250	infeasible	infeasible
61	Vulcain_C3H8_NBP	1485	infeasible	infeasible
62	Vulcain_C3H8_NBP	1870	infeasible	infeasible
63	Vulcain_C3H8_NBP	1976	infeasible	infeasible
64	Vulcain_C3H8_NBP	2068	infeasible	infeasible
65	Vulcain_C3H8_NBP	2250	infeasible	infeasible
66	Vulcain_LH2_1T_NBP	1485	infeasible	infeasible
67	Vulcain_LH2_1T_NBP	1870	infeasible	infeasible
68	Vulcain_LH2_1T_NBP	1976	infeasible	infeasible
69	Vulcain_LH2_1T_NBP	2068	infeasible	infeasible
70	Vulcain_LH2_1T_NBP	2250	infeasible	infeasible
71	Vulcain_RP1_1T_NBP	1485	infeasible	infeasible
72	Vulcain_RP1_1T_NBP	1870	295.63	infeasible
73	Vulcain_RP1_1T_NBP	1976	infeasible	infeasible
74	Vulcain_RP1_1T_NBP	2068	infeasible	infeasible
75	Vulcain_RP1_1T_NBP	2250	infeasible	infeasible
76	Vulcain_CH4_1T_NBP	1485	infeasible	infeasible
77	Vulcain_CH4_1T_NBP	1870	infeasible	infeasible
78	Vulcain_CH4_1T_NBP	1976	infeasible	infeasible
79	Vulcain_CH4_1T_NBP	2068	infeasible	infeasible
80	Vulcain_CH4_1T_NBP	2250	infeasible	infeasible
81	Vulcain_C3H8_1T_NBP	1485	infeasible	infeasible
82	Vulcain_C3H8_1T_NBP	1870	infeasible	infeasible
83	Vulcain_C3H8_1T_NBP	1976	infeasible	infeasible
84	Vulcain_C3H8_1T_NBP	2068	infeasible	infeasible
85	Vulcain_C3H8_1T_NBP	2250	infeasible	infeasible
86	SSME_1PB_1T	1485	infeasible	infeasible
87	SSME_1PB_1T	1870	infeasible	infeasible
88	SSME_1PB_1T	1976	infeasible	infeasible
89	SSME_1PB_1T	2068	infeasible	infeasible
90	SSME_1PB_1T	2250	infeasible	infeasible
91	RD180_RP1	1485	367.27	355.18
92	RD180_RP1	1870	infeasible	infeasible
93	RD180_RP1	1976	infeasible	infeasible
94	RD180_RP1	2068	infeasible	infeasible
95	RD180_RP1	2250	infeasible	infeasible
96	RD180_CH4	1485	354.80	infeasible
97	RD180_CH4	1870	infeasible	infeasible
98	RD180_CH4	1976	infeasible	infeasible
99	RD180_CH4	2068	infeasible	infeasible
100	RD180_CH4	2250	infeasible	infeasible
101	RD180_C3H8	1485	infeasible	infeasible
102	RD180_C3H8	1870	infeasible	infeasible
103	RD180_C3H8	1976	infeasible	infeasible
104	RD180_C3H8	2068	infeasible	infeasible
105	RD180_C3H8	2250	infeasible	infeasible
106	RLX	1485	infeasible	infeasible
107	RLX	1870	infeasible	infeasible
108	RLX	1976	infeasible	infeasible
109	RLX	2068	infeasible	infeasible
110	RLX	2250	infeasible	infeasible

Table 104: Results for Final Application - Case 6

Discrete Variable Combination	*Engine Cycle*	*Maximum Allowable Turbine Inlet Temperature (R)*	*Solution of Lower Bounding Space*	*Solution of True Continuous Space*
1	Vulcain_LH2	1485	infeasible	infeasible
2	Vulcain_LH2	1870	infeasible	infeasible
3	Vulcain_LH2	1976	infeasible	infeasible
4	Vulcain_LH2	2068	814.60	821.28
5	Vulcain_LH2	2250	849.90	859.98
6	Vulcain_RP1	1485	infeasible	infeasible
7	Vulcain_RP1	1870	infeasible	infeasible
8	Vulcain_RP1	1976	infeasible	infeasible
9	Vulcain_RP1	2068	infeasible	infeasible
10	Vulcain_RP1	2250	infeasible	infeasible
11	Vulcain_CH4	1485	infeasible	infeasible
12	Vulcain_CH4	1870	infeasible	infeasible
13	Vulcain_CH4	1976	infeasible	infeasible
14	Vulcain_CH4	2068	infeasible	infeasible
15	Vulcain_CH4	2250	infeasible	infeasible
16	Vulcain_C3H8	1485	infeasible	infeasible
17	Vulcain_C3H8	1870	infeasible	infeasible
18	Vulcain_C3H8	1976	infeasible	infeasible
19	Vulcain_C3H8	2068	infeasible	infeasible
20	Vulcain_C3H8	2250	infeasible	infeasible
21	Vulcain_LH2_1T	1485	infeasible	infeasible
22	Vulcain_LH2_1T	1870	733.94	737.63
23	Vulcain_LH2_1T	1976	771.59	776.34
24	Vulcain_LH2_1T	2068	810.27	815.16
25	Vulcain_LH2_1T	2250	848.77	853.97
26	Vulcain_RP1_1T	1485	infeasible	infeasible
27	Vulcain_RP1_1T	1870	infeasible	infeasible
28	Vulcain_RP1_1T	1976	infeasible	infeasible
29	Vulcain_RP1_1T	2068	infeasible	infeasible
30	Vulcain_RP1_1T	2250	infeasible	infeasible
31	Vulcain_CH4_1T	1485	infeasible	infeasible
32	Vulcain_CH4_1T	1870	infeasible	infeasible
33	Vulcain_CH4_1T	1976	infeasible	infeasible
34	Vulcain_CH4_1T	2068	infeasible	infeasible
35	Vulcain_CH4_1T	2250	infeasible	infeasible
36	Vulcain_C3H8_1T	1485	infeasible	infeasible
37	Vulcain_C3H8_1T	1870	infeasible	infeasible
38	Vulcain_C3H8_1T	1976	infeasible	infeasible
39	Vulcain_C3H8_1T	2068	infeasible	infeasible
40	Vulcain_C3H8_1T	2250	infeasible	infeasible
41	SSME	1485	infeasible	infeasible
42	SSME	1870	infeasible	infeasible
43	SSME	1976	821.04	825.81
44	SSME	2068	860.60	867.35
45	SSME	2250	901.47	908.66
46	Vulcain_LH2_NBP	1485	infeasible	infeasible
47	Vulcain_LH2_NBP	1870	infeasible	infeasible
48	Vulcain_LH2_NBP	1976	769.41	772.93
49	Vulcain_LH2_NBP	2068	807.16	811.93
50	Vulcain_LH2_NBP	2250	844.99	850.18
51	Vulcain_RP1_NBP	1485	infeasible	infeasible
52	Vulcain_RP1_NBP	1870	infeasible	infeasible
53	Vulcain_RP1_NBP	1976	infeasible	infeasible
54	Vulcain_RP1_NBP	2068	infeasible	infeasible
55	Vulcain_RP1_NBP	2250	infeasible	infeasible
56	Vulcain_CH4_NBP	1485	infeasible	infeasible
57	Vulcain_CH4_NBP	1870	infeasible	infeasible

Table 104: Results for Final Application - Case 6 (Continued)

Discrete Variable Combination	*Engine Cycle*	*Maximum Allowable Turbine Inlet Temperature (R)*	*Solution of Lower Bounding Space*	*Solution of True Continuous Space*
58	Vulcain_CH4_NBP	1976	infeasible	infeasible
59	Vulcain_CH4_NBP	2068	infeasible	infeasible
60	Vulcain_CH4_NBP	2250	infeasible	infeasible
61	Vulcain_C3H8_NBP	1485	infeasible	infeasible
62	Vulcain_C3H8_NBP	1870	infeasible	infeasible
63	Vulcain_C3H8_NBP	1976	infeasible	infeasible
64	Vulcain_C3H8_NBP	2068	infeasible	infeasible
65	Vulcain_C3H8_NBP	2250	infeasible	infeasible
66	Vulcain_LH2_1T_NBP	1485	infeasible	infeasible
67	Vulcain_LH2_1T_NBP	1870	723.27	729.46
68	Vulcain_LH2_1T_NBP	1976	760.67	767.56
69	Vulcain_LH2_1T_NBP	2068	798.17	805.93
70	Vulcain_LH2_1T_NBP	2250	836.18	844.31
71	Vulcain_RP1_1T_NBP	1485	infeasible	infeasible
72	Vulcain_RP1_1T_NBP	1870	infeasible	infeasible
73	Vulcain_RP1_1T_NBP	1976	infeasible	infeasible
74	Vulcain_RP1_1T_NBP	2068	infeasible	infeasible
75	Vulcain_RP1_1T_NBP	2250	infeasible	infeasible
76	Vulcain_CH4_1T_NBP	1485	infeasible	infeasible
77	Vulcain_CH4_1T_NBP	1870	infeasible	infeasible
78	Vulcain_CH4_1T_NBP	1976	infeasible	infeasible
79	Vulcain_CH4_1T_NBP	2068	infeasible	infeasible
80	Vulcain_CH4_1T_NBP	2250	infeasible	infeasible
81	Vulcain_C3H8_1T_NBP	1485	infeasible	infeasible
82	Vulcain_C3H8_1T_NBP	1870	infeasible	infeasible
83	Vulcain_C3H8_1T_NBP	1976	infeasible	infeasible
84	Vulcain_C3H8_1T_NBP	2068	infeasible	infeasible
85	Vulcain_C3H8_1T_NBP	2250	infeasible	infeasible
86	SSME_1PB_1T	1485	infeasible	infeasible
87	SSME_1PB_1T	1870	infeasible	infeasible
88	SSME_1PB_1T	1976	787.30	790.51
89	SSME_1PB_1T	2068	825.36	830.12
90	SSME_1PB_1T	2250	863.98	869.72
91	RD180_RP1	1485	infeasible	infeasible
92	RD180_RP1	1870	infeasible	infeasible
93	RD180_RP1	1976	infeasible	infeasible
94	RD180_RP1	2068	infeasible	infeasible
95	RD180_RP1	2250	infeasible	infeasible
96	RD180_CH4	1485	infeasible	infeasible
97	RD180_CH4	1870	infeasible	infeasible
98	RD180_CH4	1976	infeasible	infeasible
99	RD180_CH4	2068	infeasible	infeasible
100	RD180_CH4	2250	infeasible	infeasible
101	RD180_C3H8	1485	infeasible	infeasible
102	RD180_C3H8	1870	infeasible	infeasible
103	RD180_C3H8	1976	infeasible	infeasible
104	RD180_C3H8	2068	infeasible	infeasible
105	RD180_C3H8	2250	infeasible	infeasible
106	RLX	1485	infeasible	infeasible
107	RLX	1870	infeasible	infeasible
108	RLX	1976	infeasible	infeasible
109	RLX	2068	infeasible	infeasible
110	RLX	2250	infeasible	infeasible

REFERENCES

1. Turchi, P., Propulsion Techniques: Action and Reaction, America Institute of Aeronautics and Astronautics, Reston, Virginia, 1998.

2. Dewey, A. P., Robert Goddard: Space Pioneer, Little, Brown and Company, Boston, 1962.

3. Lehman, M., Robert H. Goddard: Pioneer of Space Research, Da Capo Press, New York, 1988.

4. Pendray, G. E. and Goddard, E. C., The Papers of Robert H. Goddard, McGraw-Hill, New York, 1970.

5. Goddard Space Flight Center Website, Photo credited to Mrs. Robert Goddard, http://www.gsfc.nasa.gov, 2003.

6. Encyclopedia Astronautica, http://www.astronautix.com/index.html, 2003.

7. Lt Gen Thomas S. Moorman, Jr., "DoD Space Launch Modernization Plan", Briefing to the National Security Industrial Association (NSIA), June 8, 1994.

8. St. Germain, B., Olds, J. R., McIntire, J., Nelson, D., Weglian, J., Ledsinger, L., "*Starsaber:* A Small Payload-Class TSTO Vehicle Concept Utilizing Rocket-Based Combined Cycle Propulsion," AIAA 2001-3516, 37th AIAA/ASME/SAE/ASEE Joint Propulsion Conference And Exhibit, Salt Lake City, Utah, July 8-11, 2001.

9. Bradford, J., Olds, J. R., Bechtel, R., Cormier, T., Messitt, D., "Exploration of the Design Space for the *ABLV-GT* SSTO Reusable Launch Vehicle," AIAA 2000-5136, AIAA Space 2000 Conference and Exposition, Long Beach, CA, September 19-21, 2000.

10. Olds, J. R., Bradford, J., Charania, A., Ledsinger, L., McCormick, D., Sorensen, K., "*Hyperion*: An SSTO Vision Vehicle Concept Utilizing Rocket-Based Combined Cycle Propulsion," AIAA 99-4944, 9th International Space Planes and Hypersonic Systems and Technologies Conference, Norfolk, VA, November 1-5, 1999.

11. Olds, J. R. and Bellini, P. X., "Argus, a Highly Reusable SSTO Rocket-Based Combined Cycle Launch Vehicle with Maglifter Launch Assist," AIAA 98-1557, AIAA 8th International Space Planes and Hypersonic Systems and Technologies Conference, Norfolk, VA, April 1998.

12. Boeing Corporation, "Boeing: Rocketdyne: RS-84 Engine," http://www.boeing.com/defense-space/space/propul/RS84.html, 2003.

13. Raymer, D., Aircraft Design: A Conceptual Approach, American Institute of Aeronautics and Astronautics, Reston, Virginia, 1999.

14. Olds, J. R., "Multidisciplinary Design Techniques Applied to Conceptual Aerospace Vehicle Design", Ph.D. Dissertation, North Carolina State University, Raleigh, NC, June 1993.

15. Way, D. W., and Olds, J. R., "SCORES: Web-Based Rocket Propulsion Analysis Tool for Space Transportation System Design," AIAA 99-2353, 35th AIAA/ASME/SAE/ASEE Joint Propulsion Conference, Los Angeles, CA, June 20-24, 1999.

16. Cormier, T. A., "PHATCAT: Power-Head and Thrust Chamber Analysis Tool," Master's Degree Thesis, Georgia Institute of Technology, July 27, 2001.

17. Way, D. W. and Olds, J. R., "SCORES: Developing an Object-Oriented Rocket Propulsion Analysis Tool," AIAA 98-3227, 34th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit, Cleveland, OH, July 12 - 15, 1998.

18. "System Design Specification for the ROCETS (Rocket Engine Transient Simulation) System," Pratt & Whitney FR-20284, prepared for NASA MSFC, August 24, 1990.

19. Wang, T. S., et al, "Unified Navier-Stokes Flowfield and Performance Analysis of Liquid Rocket Engines," Journal of Propulsion and Power, September 1993.

20. Bradford, J., "SCORES-II: Design Tool for Liquid Rocket Engine Analysis," AIAA 2002-3990, 37th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit, Indianapolis, IN, July 8 - 10, 2002.

21. Vanderplaats, G., Numerical Optimization Techniques for Engineering Design, Vanderplaats Research & Development, Inc., Colorado Springs, CO, 1998.

22. Floudas, C. A., Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications, Oxford University Press, New York, NY, 1995.

23. Watkins, D., Fundamentals of Matrix Computations, John Wiley & Sons, New York, 1991.

24. Zoutendijk, G., "Method of Feasible Directions." Elsevier, Amsterdam, 1960.

25. ADS Users Manual, Version 3.0, VMA Engineering, Colorado Springs, CO, 1987.

26. DOT Users Manual, Version 4.2, Vanderplaats Research & Development, Inc., Colorado Springs, CO, 1995.

27. Vanderplaats, G., CONMIN User's Manual, NASA Technical Memorandum X-62282, 1978.

28. Dantzig, G. B., "Programming in a Linear Structure." Comptroller, USAF, Washington, D.C., 1948.

29. Dantzig, G. B., Linear Programming and Extensions, Princeton University Press, Princeton, N.J., 1963.

30. Wolfe, P., "The Product Form of the Simplex Method," Report P-3075, The RAND Corporation, Santa Monica, California, 1965.

31. Kelley, J. E., "The Cutting Plane Method for Solving Convex Programs," *J. SIAM*, 8:702-712, 1960.

32. Boot, J. C., Quadratic Programming: Studies in Mathematical Managerial Economics, Vol. 2, North-Holland, 1977.

33. Hadley, G., Nonlinear and Dynamic Programming, Addision-Wesley, New York, 1964.

34. Ganish, B, et al., "An Approach to Optimal Positioning of a New Product," *Management Science*, 29:1277, 1983.

35. Hoang, H., "Topological Optimization of Networks: A Nonlinear Mixed Integer Model Employing Generalized Benders Decomposition," *IEEE Trans. Automatic Control*, AC-27:164, 1982.

36. Aggarwal, A., and Floudas, C. A., "Synthesis of Heat Integrated Nonsharp Distillation Sequences," *Comp. & Chem. Eng.*, 16:89, 1992

37. Birewar, D. B. and Grossmann, I. E., "Incorporating Scheduling in the Optimal Design of Multiproduct Batch Plants," *Comp. & Chem. Eng.*, 13:141, 1989.

38. Ciric, A. R. and Floudas, C. A., "A Retrofit Approach of Heat Exchanger Networks," *Comp. & Chem. Eng.*, 13(6):703, 1989.

39. Ciric, A. R. and Floudas, C. A., “A Mixed-Integer Nonlinear Programming Model for Retrofitting Heat Exchanger Networks,” *I&EC Res.*, 29:239, 1990.

40. Faqir, N. M. and Karimi, I. A., “Design of Multipurpose Batch Plants with Multiple Production Routes,” In *Proceedings FOCAPD ’89*, Snowmass, Colorado, 1990.

41. Fletcher, R., et. al., “Flexible Retrofit Design of Multiproduct Batch Plants,” *Comp. & Chem. Eng.*, 15(12):843, 1991.

42. Floudas, C. A. and Ciric, A. R., “Strategies for Overcoming Uncertainties in Heat Exchanger Network Synthesis,” *Comp. & Chem. Eng.*, 13(10):1133, 1989.

43. Kalitventzeff, B. and Marechal, F., “The Management of a Utility Network,” *Process Systems Engineering*, PSE ’88, Sydney, Australia, 1988.

44. Kocis, G. R. and Grossmann, I. E., “Global Optimization of Nonconvex MINLP Problems in Process Synthesis,” *I&EC Res.*, 27(8):1407, 1988.

45. Kokossis, A. C. and Floudas, C. A., “Optimal Synthesis of Isothermal Reactor-Separator-Recycle Systems,” *Chem. Eng. Sci.* 46:1361, 1991.

46. Kravanja, Z. and Grossmann, I. E., “PROSYN – An MINLP Process Synthesizer,” *Comp. & Chem. Eng.*, 14:1363, 1990.

47. Paplexandri, K. and Pistikopoulos, E. N., “An MINLP Retrofit Approach for Improving the Flexibility of Heat Exchanger Networks,” *Ann. Oper. Res.*, 42:119, 1993.

48. Paules IV, G. E. and Floudas, C. A., “Synthesis of Flexible Distillation Sequences for Multi-period Operation,” *Comp. & Chem. Eng.*, 12(4):267, 1988.

49. Sahinidis, N. V. and Grossmann, I. E., “Convergence Properties of Generalized Benders Decomposition,” *Comp. & Chem. Eng.*, 15(7):481, 1991.

50. Viswanathan, J. and Grossmann, I. E., “A Combined Penalty Function and Outer Approximation for MINLP Optimization,” *Comp. & Chem. Eng.*, 14(7):769, 1990.

51. Yee, T. F. and Grossmann, I. E., “Simultaneous Optimization Model for Heat Integration – II. Heat Exchanger Network Synthesis,” *Comp. & Chem. Eng.*, 14(10):1165, 1990.

52. Land, A. H., Doig, A. G., “An Automatic Method for Solving Discrete Programming Problems”, Econometrica, Vol. 28, pp. 497, 1960.

53. Dakin, R. J., “A Tree Search Algorithm for Mixed Integer Programming Problems”, Computer Journal, Vol. 8, pp. 250, 1965.

54. Beale, E. M. L. and Forrest, J. J. H., “Global Optimization Using Special Ordered Sets,” *Ann. of Discrete Math.*, 5:201, 1977.

55. Beale, E. M. L., “Branch and Bound Methods for Mathematical Programming Systems,” *Ann. of Discrete Math.*, 5: 201, 1979

56. Horst, R. and Tuy, H, Global Optimization: Deterministic Approaches, Springer-Verlag, Berlin, 1996.

57. Martin, R. K. and Schrage, L., “Subset Coefficient Reduction Cuts of 0 – 1 Mixed Integer Programming,” *Oper. Res.*, 33:505, 1985.

58. Nemhauser, G. L. and Wolsey, L. A., Integer and Combinatorial Optimization, J. Wiley, New York, 1988.

59. Parker, R. G. and Rardin, R. L., Discrete Optimization, Academic Press, New York, 1988.

60. Schrijver, A., Theory of Linear and Integer Programming, Wiley-Interscience Series in Discrete Mathematics and Optimization, J. Wiley, New York, 1986.

61. Tomlin, J. A.,”An Improved Branch and Bound Method for Integer Programming,” *Oper. Res.*, 19:1070, 1971.

62. Geoffrion, A. M., “Generalized Benders Decomposition”, *Journal of Optimization Theory and Applications*, Vol. 10, pp. 237, 1972.

63. Bender, J. F., “Partitioning Procedures for Solving Mixed-Variable Programming Problems,” *Numer. Math.*, 4:238, 1962.

64. Flippo, O. E., et. al., “Duality and Decomposition in General Mathematical Programming,” Econometric Institute, Report 8747/B, Erasmus University Rotterdam, 1987.

65. Floudas, C. A., et. al., “Global Optimum Search for Nonconvex NLP and MINLP Problems,” *Comp. & Chem. Eng.*, 13(10):1117, 1989.

66. Paules IV, G. E. and Floudas, C. A., “APROS: Algorithmic Development Methodology for Discrete-Continuous Optimization Problems,” *Oper. Res.*, 37(6):902, 1989.

67. Duran, M. A., and Grossmann, I. E., “An Outer Approximation Algorithm for a Class of Mixed-Integer Nonlinear Programs”, Mathematical Programming, Vol. 36, pp. 307, 1986.

68. Leyffer, S., “Deterministic Methods for Mixed Integer Nonlinear Programming,” PhD Dissertation, University of Dundee, Dundee, Scotland, UK, Dec. 1993.

69. Duran, M. A., and Grossmann, I. E., “A Mixed-Integer Nonlinear Programming Algorithm for Process Systems Synthesis,” *AIChE Journal*, 32(4):592, 1986.

70. Fletcher, R. and Leyffer, S., “Solving Mixed Integer Nonlinear Programs by Outer Approximation,” *Math. Progr.*, 66(3):327, 1994.

71. Kocis, G. R. and Grossmann, I. E., “Relaxation Strategy for the Structural Optimization of Process Flow Sheets,” *I&EC Res.*, 26(9):1869, 1987.

72. Ryoo, H. S., and Sahinidis, N. V., “A Branch and Reduce Approach to Global Optimization,” Journal of Global Optimization, Vol. 8, pp. 107-139, 1996.

73. Kesavan, P., and Barton, P., “Decomposition Algorithms for Nonconvex Mixed-Integer Nonlinear Programs,” AIChE Symposium Series, Vol. 96, pp. 458-461, 2000.

74. Floudas, C., Deterministic Global Optimization: Theory, Methods, and Applications, Kluwer Academic Publishers, London, 2000.

75. Adjiman, C. S. and Floudas, C. A., “Rigorous Convex Underestimators for General Twice-Differentiable Problems,” *J. of Global Opt.*, 9:23, 1996.

76. Androulakis, I. P., et. al., “αBB: A Global Optimization Method for General Constrained Nonconvex Problems,” *J. of Global Opt.*, 7:337, 1995.

77. Maranas, C. D. and Floudas, C. A., “Global Minimum Potential Energy Conformations of Small Molecules,” *J. of Global Opt.*, 4:135, 1994.

78. Adjiman, C. S., et. al., “A Global Optimization Method, αBB, for General Twice-Differentiable NLPs – I. Theoretical Advances,” *Comp. & Chem. Eng.*, 22(9):1137, 1998.

79. Gordon, S., and McBride, B. J., “Computer Program for Calculation of Complex Chemical Equilibrium Compositions and Applications”, NASA Reference Publication 1311, 1994.

80. Turns, S., An Introduction to Combustion: Concepts and Applications, McGraw Hill, New York, NY, 1996.

81. Bradford, J. E., “A Technique for Rapid Prediction of Aftbody Nozzle Performance for Hypersonic Launch Vehicle Design”, Ph.D. Dissertation, Georgia Institute of Technology, Atlanta, GA, June 2001.

82. Sutton, G. P., Rocket Propulsion Elements: An Introduction to the Engineering of Rockets, 6th Edition, John Wiley & Sons, Inc, New York, 1992.

83. Manski, D., Goertz, D., Saβnick, H., Hulka, J., Goracke, B., Levack, D., “Cycles for Earth-to-Orbit Propulsion,” Journal of Propulsion and Power, Vol. 14, No. 5, pp. 588-604, Sept.-Oct. 1998.

84. Huzel, D., and Huang, D., Modern Engineering for Design of Liquid-Propellant Rocket Engines, American Institute of Aeronautics and Astronautics, Inc., Washington, DC, 1992.

85. Humble, R., Henry, G., Larson, W., Space Propulsion Analysis and Design, McGraw Hill, New York, 1995.

86. Jennings, T., Fentress, S., Peery, S., Minick, A., "Development and Test of an Advanced Expander Combustor," AIAA 1999-2599, AIAA/ASME/SAE/ASEE 35th Joint Propulsion Conference and Exhibit, Los Angeles, CA, June 20 – 24, 1999.

87. Jennings, T., Fentress, S., Lyda, R., "Fabrication and Test of an Advanced Expander Combustor," AIAA 2000-3776, AIAA/ASME/SAE/ASEE 36th Joint Propulsion Conference and Exhibit, Huntsville, AL, July 16 – 19, 2000.

88. Peery, S. and Minick, A., "Design and Development of an Advanced Expander Combustor," AIAA 1998-3675, AIAA/ASME/SAE/ASEE 34th Joint Propulsion Conference and Exhibit, Cleveland, OH, July 13 – 15, 1998.

89. Parsley, R. and Crocker, A., “The RL200: Incorporating a Balanced Approach for Reusable Propulsion Safety,” AIAA 2000-3287, AIAA/ASME/SAE/ASEE 36th Joint Propulsion Conference and Exhibit, Huntsville, AL, July 16 – 19, 2000.

90. Kiraly, J., editor, “Pratt & Whitney Directions: A Publication for All Employees,” East Hartford, CT, Summer 2002.

91. Goertz, C., “A Modular Method for the Analysis of Liquid Rocket Engine Cycles”, AIAA 95-2966, 1995.

92. Cengel, Y., and Boles, M., Thermodynamics: An Engineering Approach, McGraw Hill, New York, 1994.

93. Stangeland, J., “Turbopumps for Liquid Rocket Engines,” SAE SP-924, 9th Cliff Garrett Turbomachinery Award Lecture, Rocketdyne Division, April 1992.

94. NASA SP-8109, "Liquid Rocket Engine Centrifugal Flow Turbopumps," NASA Space Vehicle Design Criteria, Washington D.C., December, 1973.

95. Bissell, W. R., "Rocket Engine Turbopump Tutorial," Rocketdyne-RI/RD 85-245, USAF Contract F33657-82-C-0346 for Analysis of Foreign Chemical Rocket Propulsion Systems, Rockwell International, Rocketdyne Division, 1985.

96. NASA SP-8107, "Turbopump Systems for Liquid Rocket Engines," NASA Space Vehicle Design Criteria, Washington D.C., August, 1974.

97. NASA SP-8110, "Liquid Rocket Engine Turbines," NASA Space Vehicle Design Criteria, Washington D.C., January, 1974.

98. NASA SP-8081, "Liquid Propellant Gas Generators," NASA Space Vehicle Design Criteria, Washington D.C., March, 1972.

99. NASA SP-8087, "Liquid Rocket Engine Fluid-Cooled Combustion Chambers," NASA Space Vehicle Design Criteria, Washington D.C., April, 1972.

100. Zucrow, M. J., and Hoffman, J., Gas Dynamic: Volume I, Wiley, New York, 1976.

101. SUPERTRAPP: NIST Hydrocarbon Mixture Database 4, Version 3.0, National Institute of Standards and Technology, Gaithersburg, MD, 1999.

102. Stull, D. R., and Prophet, H., "JANAF Thermochemical Tables", 2nd Edition, NDRS-NBS 37, National Bureau of Standards, Washington, 1971.

103. REFPROP: Reference Fluid Thermodynamic and Transport Properties, NIST Standard Reference Database 23, Version 7.1, Beta version 02/25/03, National Institute of Standards and Technology, Gaithersburg, MD, 2003.

104. Younglove, B.A., "Thermophysical Properties of Fluids. I. Argon, Ethylene, Parahydrogen, Nitrogen, Nitrogen Trifluoride, and Oxygen," *J. Phys. Chem. Ref. Data*, Vol. 11, Suppl. 1, pp. 1-11, 1982.

105. Handbook of Aviation Fuel Properties (Third Printing), Prepared by the Coordinating Research Council, Inc., Society of Automotive Engineers, Inc., May 1988.

106. Isakowitz, S. J., Hopkins, Jr., J. P., Hopkins, J. B., International Reference Guide to Space Launch Systems, 3rd Edition, American Institute of Aeronautics and Astronautics, Reston, VA, 1999.

107. NAFCOM99: NASA/Air Force Cost Model 99, Documentation and User's Manual, Version 6.0, Prepared by SAIC under Contract Number: NAS8-40431, Huntsville, AL, November 1999.

108. Mavris, D. N, and Bandte, O., "Economic Uncertainty Assessment Using A Combined Design of Experiments/Monte Carlo Simulation Approach with Application to an HSCT," 17th Annual Conference of the International Society of Parametric Analysts, San Diego, CA., May 1995.

109. Tai, J. C., Mavris, D. N., Schrage, D. P., "Application of a Response Surface Method to the Design of Tipjet Driven Stopped Rotor/Wing Concepts," American Institute of Aeronautics and Astronautics, 1995.

110. Mason, R., Gunst, R., Hess, J., Statistical Design and Analysis of Experiments, John Wiley & Sons, New York, 1989.

111. JMP Statistics and Graph Guide, Version 5.0.1, SAS Institute, Inc., Cary, NC, 1995.

112. Davis, L, and Steenstrup, M., "Genetic Algorithms and Simulated Annealing: An Overview," Genetic Algorithms and Simulated Annealing, Lawrence Davis, Morgan Kauffman Publishers, Los Altos, CA, 1987.

113. Huyer, W., and Neumaier, A., "Global Optimization by Multilevel Coordinate Search," *Journal of Global Optimization*, Vol. 14, pp. 131-355, 1996.

114. Malone, B. and Papay, M., "ModelCenter: An Integration Environment for Simulation Based Design," *Simulation Interoperability Workshop*, Orlando, FL, March 1999.

115. Fausett, L., Fundamentals of Neural Networks, Prentice Hall, New York, 1994.

116. Dernuth, H. and Beale, M., Matlab Neural Network Toolbox, The Math Works Inc, 1992.

VITA

Brad David St. Germain was born on November 18, 1975 in New Orleans, Louisiana. He moved to Mandeville, Louisiana and attended grade school at Mandeville Elementary and Mandeville Junior High School. After graduating from Mandeville High School in May of 1993, he enrolled at Louisiana State University. In May of 1998, he graduated with honors from LSU with a degree in Mechanical Engineering. The following fall he enrolled at The Georgia Institute of Technology to pursue graduate studies in Aerospace Engineering. Brad received his Master's Degree in Aerospace Engineering from Georgia Tech in December of 1999 and continued toward his Doctoral degree in the same field.

While at Georgia Tech, Brad worked with Dr. John Olds as a member of the Space Systems Design Lab. Here he studied reusable launch vehicle design with a concentration in propulsion and was a team leader for several advanced vehicle concepts. During his graduate studies, Brad worked summers at the NASA Langley Research Center in Hampton, Virginia; the NASA Marshall Space Flight Center in Huntsville, Alabama; and Pratt & Whitney in West Palm Beach, Florida.

Brad is an alumni member of the Lambda Chi Alpha fraternity, Upsilon Zeta chapter, and is a member of the American Institute of Aeronautics and Astronautics.