A SIMULATION FRAMEWORK FOR THE ANALYSIS OF REUSABLE LAUNCH VEHICLE OPERATIONS AND MAINTENANCE

A Thesis
Presented to
The Academic Faculty

By

Patrick Daniel Dees

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Daniel Guggenheim School of Aerospace Engineering

Georgia Institute of Technology

December 2012

A SIMULATION FRAMEWORK FOR THE ANALYSIS OF REUSABLE LAUNCH VEHICLE OPERATIONS AND MAINTENANCE

Approved by:

Professor Dimitri N. Mavris, Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Vitali Volovoi
School of Aerospace Engineering
Georgia Institute of Technology

Mr. Stephen Edwards
School of Aerospace Engineering
Georgia Institute of Technology

Date Approved: July 9, 2012

ACKNOWLEDGEMENTS

There are many people in my life currently who I owe a debt of gratitude to from their help during my thesis process. First, to my committee, none of this would have been possible without your faith in my abilities and for that I thank you whole-heartedly.

Mr. Edwards, your continued guidance kept me from chasing down plenty of wild reusable launch geese while this project was in its infancy. Meeting with you always gave me new ideas to pursue, many of which have appear in this final work.

Dr. Mavris, your words of encouragement and advice challenge me every day to work harder and smarter. The amount of work you put into running ASDL is truly monumental, and I would not be the man I am today without you taking a chance on a physics undergrad from a small, strange school in Sarasota, Florida.

Dr. Volovoi, your attention to detail and thorough knowledge of this subject impresses me with every interaction we have. You understand the issues currently faced by the great movers and shakers in the world to a level which I aspire to and hope to achieve one day.

To Dr. Olds, Michael Kelly, and the staff of SEI, consulting with you has done more good for this project than I could have initially guessed and I thank you for that. Dr. Olds' appearance in my proposal seems quite serendipitous at this point, as his suggestions which at the time seemed radical I now see as wise and informed.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

AFRL……………………………………………………..……Air Force Research Laboratory

ALCCA…………………………………………………Aircraft Life Cycle Cost Analysis

ASDL…………………………………………………...Aerospace Systems Design Laboratory

CABAM…………………………………………….Cost And Business Analysis Module

CER……………..……………………………………………Cost-estimating relationship

CFD……………..…………………………………………Computational Fluid Dynamics

DES………………………………………………………….Discrete Event Simulation

EDL…………………………………………………………..Entry, Descent, and Landing

FAA…………...…………………………………………Federal Aviation Administration

GEM-FLO…………………………………Generic Model for Future Launch Operations

HRST…………………………………………...Highly Reusable Space Transportation

ISS…………………………………………………………..International Space Station

ITAR……………………………………………International Trade in Arms Restrictions

KSC…………………………………………………………..Kennedy Space Center

LOV………………………………………………………..…………….Loss of Vehicle

LCC…………………………………………………………………….Life-Cycle Cost

LLEGO……………………………………Launch & Landing Effects Ground Operations

LP…………………………………………………………………….Linear Programming

MMH……………………………………………………………Maintenance Man-Hours

NAFCOM……………………...…………………………NASA Air Force Cost Model

NASA……………………………………National Aeronautics and Space Administration

O&M……………..…………………………………………Operations & Maintenance

OR…………………..…………………………………………………Operations Research

RLV…………………..……………………………………………Reusable Launch Vehicle

ROM…………………..………………………………………..Rough order-of-magnitude

SME…………………...……………………………………………..Subject Matter Expert

STS……………………..……………………………………Space Transportation System

TFU…………………………………………………………………Theoretical First Unit

TPS……………………………………………………………Thermal Protection System

VAFB…………………………………………………………Vandenburg Air Force Base

SUMMARY

Every year humanity reaches farther and faster than ever previously thought possible. We build planes bigger, computers smaller, and materials stronger than ever before. The methods for developing the complex systems that enable our rapid growth and achievements have similarly grown in complexity and fidelity in order to maintain this progress. In some cases the best course of action is to build upon current constructions, such as the addition of new technologies into existing aircraft. However; recently engineers have instead sought performance progress by an overhaul of the entire conceptual architecture.

The current paradigm shift in engineering design is incorporating more and more information on the implications of vehicle configuration into conceptual design, taking advantage of the design freedom present within the conceptual stage to predict and address long-term feasibility and viability concerns. The complex and powerful tools this shift has inspired continue to enable valuable forecasting which goes above and beyond designing only for performance, to focus on designing for business. In pursuit of this new level of operational excellence, a design concept must not only be able to perform its intended mission, but also be constructed in the most intelligent manner possible by characterizing the implication of every detail of the design so that its continued operation can be achieved with the utmost efficiency.

Methods developed previously have used simple regressions based on gross vehicle weight or other high-level design characteristics and extensive databases of maintenance information from which to draw comparison. But what about when there is no such database? For the emerging commercial spaceflight industry, their vehicles and operational structure must be designed with utmost care, both to increase revenue and decrease overhead costs. However; information on the Shuttle, the only successful launch vehicle campaign to date, is mostly scattered and available only at highly aggregated levels.

The effect of performing missions on a vehicle's component subsystems is a subject which has received huge amounts of attention, and fully matured safety and reliability tools relating a vehicle's design to its long-term performance take many forms. In many cases these operations prediction tools are capable of meaningful analysis of vehicles with no historical precedent, such as a commercial reusable launch vehicle. The human role in keeping everything in an aerospace architecture running smoothly, however, has received far less attention.

Everything from air conditioners to cars to reusable launch vehicles to the International Space Station eventually requires maintenance. Each of these systems require technicians with specific skills relevant to that system, and in the case of very complex systems many different kinds of skillsets are required. During the Shuttle program operational costs ballooned due to lack of information and proper planning, but in the current commercial shift in spaceflight the emerging companies wish to approximate airline levels of operational efficiency in order to minimize their costs. One way in which this can be achieved is the optimization of a maintenance workforce so that the allocation of their skills most effectively achieves vehicle turnaround.

Over the course of this document, the paradigm shift in engineering design is expanded upon and used as justification for the inclusion of operations into the conceptual design of reusable launch vehicles. The issues currently facing those entities developing reusable launch vehicles is discussed, leading to the definition of fundamental research questions and the final aim of this work. The document then continues into describing the development of an experimental frame capable of variable fidelity investigation into the operations and maintenance of a reusable launch vehicle campaign, by comparing and contrasting a few of the methods and practices currently in use for modeling the operations of complex systems. Next, the structure of an efficiently high-fidelity model constructed to investigate the human factor in operations is described which employs stochastic methods and the lessons learned from a literature review. The tool constructed is an integrated series of Python codes allowing for a high level of customization and expansion, by using code-writing code to change the structure of events taking place within simulation to whatever degree the simulator requires. The actions taking place within simulation are similarly drawn from a literature review, relying upon historical precedent and data whenever available.

Finally, competing methods for finding the optimal maintenance skillset distribution are presented and compared by their ability to converge upon a skillset distribution which achieves reusable fleet operations in the most effective manner. This is accomplished by running a series of experiments designed to explore the possible operations & maintenance schemes available, by varying the quantities in which maintenance resources, the technicians, are available for doing work on the subsystems making up a reusable launch vehicle. The first manner in which a maintenance workforce could be put together is to always have the maximum number of people needed to perform maintenance on one vehicle or as an extension enough to work on multiple vehicles. Another manner would be to have fewer than the maximum, in which case the overall efficiency of maintenance on that subsystem would be decreased, however the utilization of the technicians performing maintenance on that subsystem would be increased. Combinations between these two workforces are also possible, balancing workforce population with operational efficiency. In particular, the synthesis of a number of such schemes is of interest, as it is in the combination of several possible scenarios which is of particular interest. By sacrificing efficiency in some areas in order to achieve higher fleet operational efficiency, an intelligently allocated maintenance workforce skillset distribution is converged upon using simple overall evaluation criteria and the grid search method.

For the optimization methods formulated for simulation, performance is compared to a baseline operations study using the maximum workforce for every subsystem. By having the maximum number of technicians available for each vehicle, maintenance is assumed to proceed at maximum efficiency, eliminating waiting time and minimizing necessary maintenance man-hours. The fleet-level performance metrics of annual flight rate, maintenance man-hours per vehicle, and total workforce are then compared with every alternate skillset distribution's performance. A common first round of simulation is used for both methods, composed of many experimental runs varying the numbers of available technicians on each subsystem to represent combinations of the operational schemes outlined previously. Each optimization method then chooses particular skillset distributions from among this set according to its preferences, and centers the technician availability ranges for the next round on those values it finds optimal.

In order to evaluate the varying operational architectures represented, workforce skillset distributions are judged according to their impact on fleet operations as a commercial entity would. In order to stay in business, a commercial entity doing reusable launch vehicle launches will seek to maximize its revenue by having as many launches as possible, using its workforce in the most efficient manner possible, minimizing the necessary maintenance man-hours while simultaneously minimizing the workforce to cut out as many costs as possible. To these companies, which are becoming more prevalent with every year, incorporating the maintenance on their vehicles into their design is not only practical, it is required by the FAA, which has for over a decade declared that just like an airline, space launch companies must have a plan for their maintenance.

As the commercial space sector expands in the coming years, operations analysis will prove invaluable. The return on invested time and effort from performing the kind of maintenance optimization studies presented in this work during the conceptual design of an aerospace system is huge, and continues to grow with every year the system is in operation. The studies do not need to be very complex in order to provide meaningful insight either, showing that intelligently implementing knowledge of a system's design during its development pays off in the long run. As shown in this work, taking advantage of the design freedom present within the conceptual design phase of a reusable launch vehicle to intelligently design each interacting mechanical and human part results in a system of systems which performs more efficiently and longer than those lacking operations analysis.

CHAPTER 1

INTRODUCTION

During conceptual design of a complex system, overall performance is the central concern of development. Whether the system is a mechanical assembly line, aircraft, or a reusable launch vehicle (RLV), its functional efficiency initially overshadows all other considerations. In the case of an aircraft or RLV, engineers explore multitudes of conceptual alternatives varying the size and shape of the vehicle to custom-tailor its physical configuration to suit a mission profile [18]. At this point of design the vehicle is unformed so its characteristics can be changed at very little cost. As development continues however, the ability to change vehicle characteristics decreases rapidly as the cost 'sunk' into design increases. Currently, a paradigm shift in engineering design is taking advantage of this fact rather than being victim to it, by bringing more and more computer-based analyses into conceptual design to increase a system's long-term efficiency.

In the second stage – preliminary design – the vehicle's configuration is frozen while physical and computerized testing can take place, which continues the downward trend in design freedom and upward trend in sunk cost [18]. Here more fine details such as materials used take precedence, however still for the purposes of achieving a desired mission profile. Continuing with the example of a RLV, these considerations will include choosing a thermal protection system (TPS), a combination of different ceramic and metallic materials for handling the extreme thermal loads present within the launch and entry, descent, and landing (EDL) portions of its mission scenario. At the end of the preliminary design phase, iterative analyses converge upon a vehicle whose size, shape, and subsystem configuration make it feasible for a mission profile. It is at this point in a vehicle's development that a major decision must be made – whether the huge cost of manufacturing the vehicle is worth its potential revenue [18]. The paradigm shift shows its merit here by providing more information to the decision-maker than has ever before been available, where in the past the best performing vehicle was rarely ever the most cost-effective [8].

Before making the final decision on a system's manufacture, the viability of the system must also be investigated. As an example of feasibility versus viability concerns, a RLV whose TPS utilizes thin sheets of Nickel super-alloys will have a reduced weight versus utilizing ceramics, a performance advantage, however at the cost of an increase in possibility of fatigue. Over many years of operation, the RLV's operating cost due to TPS maintenance will result in a Life Cycle Cost (LCC) greater than that of a heavier conceptual alternative, which had not been considered during conceptual design because it decreased performance. It is in this manner that a conceptual alternative which was found to be feasible for a mission scenario may not be viable in the long-term [23]. This is of particular concern for aerospace vehicles, as only ~5% of the cost to produce the end product is involved with the design process, while the other 95% is from to the developed system's construction [23], and its operations & maintenance (O&M) cost will comprise 60-80% of its LCC [15]. Drawing from the example of the Shuttle program, the RLV will spend the majority of its time undergoing maintenance of some sort [13]. Optimizing operations performance from the conceptual design phase on is crucial in communicating the ramifications of design decisions, as bad decisions are paid for and good ones are reaped in the much longer LCC of any product [54]. Specifically, poor planning can result in increased inventory holding costs, training costs, maintenance costs, support equipment costs, crew time costs, and operations costs [10].

Recently, the entities developing space vehicles have moved from strictly governmental agencies like the National Aeronautics and Space Administration (NASA) and the European Space Agency (ESA), to the military and commercial sectors. In particular these entities are pursuing RLVs because although they do require a higher development cost, when maintained properly they have a lower overall cost [49]. In particular, the turnaround time of a RLV must be quick enough to support the eventual increase in launch demand, while also being inexpensive enough to justify a higher development cost [34]. The military is interested in developing a quick-response sub-orbital vehicle for fast deployment of troops across the world and to establish space superiority for the U.S.A., an effort which has been in development through the Air Force Research Lab (AFRL) for many years [49],[43],[55],[39]. Commercial entities such are also developing RLVs, such as the Sierra Nevada Corporation's Dream Chaser [50], the SpaceX Dragon capsule [53], and Virgin Galactic's SpaceShipOne and White Knight vehicles [58], for commercial launch missions ranging from space tourism to ISS resupply missions. These entities have different goals for the vehicle; therefore the designs chosen for each will be different. However; the issue of long-term O&M affects each of these entities, and is of central concern to the commercial sector [49]. In particular, lowering the turnaround time of a RLV achieves the military's goal of a quickly deployable system, while also maximizing flight rate which is the source of revenue for a commercial entity. Looking further into the future, by considering the effects of O&M in the preliminary design of a RLV its LCC can be greatly reduced. In the case of a commercial entity, this increase in profits can result in driving down ticket prices, which will eventually translate to driving down the cost-per-kilogram to orbit, enabling common access to space.

In the present day with the advent of commercial spaceflight companies, the need for proper operations management has been recognized by the highest authorities. With the enactment of the Commercial Space Launch Act of 1998, the Federal Aviation Administration (FAA) was given statutory authority to regulate reentry and RLV activities, requiring a maintenance plan be “systematically formulated in the early conceptual design phase of the program to minimize problems during the operational phase.” [26] In the past, the logistical concerns of RLVs were not considered [13], however moving into the future they become centrally important. Therefore, not only is a system of modeling RLV O&M necessary from an economic long-term viability standpoint, but also from a legal standpoint.

When considering the lengths of time performing design on an aerospace vehicle and its use and the rewards which can be reaped from bringing as much information as possible into the design phase, it is obvious that logistical concerns should be included to eliminate unrealistic expectations and point to where improvements can be made in a design. In adding this layer to design, money can be much more wisely spent where it can make the biggest difference and ultimately lead to a product which is more efficient overall [43]. In this manner the incorporation of more information into the conceptual design process is shifting the traditional notion of ‘design for performance’ to ‘design for business’ [8].

Modeling as a Design Tool

Once a system in development has finished preliminary design, other crucial matters must be resolved, namely quantifying economic metrics such as theoretical first unit (TFU) and LCC. Due to the inherently stochastic nature of economic analysis, every alternative will then have risk introduced into the design. Tools such as the NASA-Air Force Cost Model (NAFCOM) and Aircraft Life Cycle Cost Analysis (ALCCA) utilize cost-estimating relationships (CER) based upon data from systems currently constructed and operating in order to construct estimates on TFU and maintenance costs. Since these relationships are built upon existing data however, they cannot be assumed to accurately predict the economic implications of the current state-of-the-art [40]. For example many CERs use gross system weight to make estimates, and may not be able to account for improvements which lower weight but increase long-term maintenance costs. In addition, consultation with subject matter experts (SME) have been utilized in the past [43],[55] to gather important information on system behavior, using the Delphi method [31] to standardize the qualitative inputs.

In general the method of regressing historical data to produce mathematical relationships between system characteristics and performance estimates is called modeling. Although models are very useful for the initial stages of design of an aerospace vehicle, where there is no historical precedent there can be no model. This is a fundamental issue with the design of RLVs. The only existing historical data is for one system, the Space Transportation System (STS), and although it has been in operation for a long time most of its performance data is not publicly available, and that which is available exists at highly aggregated levels [5].

There are three fundamental methods by which performance data can be gathered. The first is physical experimentation, where the system or a scale model of the system is built and tested. After conceptual design of a vehicle, physical tests are done either by computational fluid dynamics (CFD) calculations or by constructing a scale model of the vehicle to place in a wind tunnel for aerodynamics tests. In this case as long as the experimental apparatus is constructed as close to specifications as possible and any measuring apparatuses are calibrated correctly, then the results gathered are of the greatest fidelity and may be regarded as true. When many design configurations are in competition however, physical experimentation quickly becomes time-intensive and expensive, and in general engineering design desires to know as much as possible about the expected behavior of a concept before it exists in physical form [23]. To mitigate this issue, results from physical experiments were aggregated into databases from which overall trends between configurations and performance could be gleaned. These mathematical relationships or models capture overall trends but at the cost of fidelity. The monetary and time benefits usually outweigh the loss of accuracy however, and so models are widely used during conceptual design to use data gathered previously to decide on a system configuration. When the data required for modeling is either too scarce or unavailable, another option can be used: simulation.

Ultimately a model is a mathematical construction dependent upon the sources of variation (wing area, wing sweep, fuselage shape, propulsion choice, etc.) within the system which characterize the effect of each source of variation on the system's performance. The relationship between each source of variation and the system's performance results from the interactions with underlying physical laws a physical representation of the system would be subject to. Simulation takes this abstraction one step further, by applying a set of rules and relationships to approximate the dynamic behavior of the system with the goal of gaining insight on the system. In doing so fidelity is lost, however a higher level of abstraction allows the simulator to better define the model's behavior and prove properties of the system by manipulating the abstract model rules while addressing the problem at the right level of complexity, balancing time and required levels of fidelity [59]. These rules, when defined without the use of guiding models, must be defined in a robust and scale-able manner so that they may be easily modified.

When abstracting the behavior of a complicated system to the point at which simulation operates, it becomes necessary to verify and validate both the simulation constructed and the results it is producing. Validation is the process of determining how well the constructed simulation and associated data are accurate representations of the real world, while verification is the process of determining how well the simulation and its associated data accurately captures the developer’s conceptual description and specifications [22]. Without extensive data by which to compare with simulation outputs, the simulation cannot be truly validated. Although there is a modicum of data available publicly, it is at such a high level that many possible configurations of the internal data of the simulation could reproduce it. Verification analysis however can come from a qualitative understanding of the real-world interactions captured within the simulation, and so by reproducing those interactions as faithfully as possible the simulation can be considered verified. Without validation however, the simulation’s results cannot be considered representative of a real-world system.

RLV Operations Modeling Problem Definition

Five fundamental characteristics must be captured by whatever method of analysis is chosen. The first characteristic is that since the vehicles undergoing maintenance are reusable, this is a looping process which cannot be accurately captured by a single equation. At the very least, a recursive mathematical basis is required. Therefore, the method must allow for repetition and for separate entities within simulation to be on separate repeating paths. Secondly, there are entities constantly entering and leaving the loop. Examples of such would be loss of vehicle (LOV) at launch or EDL, and any parts which require replacements along with their replacement parts. For a model of high fidelity, each of these parts would need to be tracked for damage on each mission and subsequent cumulative failure modes. Other tools have spent years in development to achieve high fidelity in these areas of analysis [30] however this is out of the scope of this work. Therefore, the only entities entering or leaving the loop will be vehicles, based on the historical examples of the Challenger and Discovery accidents. Third, each vehicle and maintenance site involved has the potential for differing characteristics. Examples of this would be alternative mission profiles for the vehicles, and number of available technicians or subsystem specialty for the maintenance sites. The fourth consideration is that since the method is intended to capture the behavior of a real-world maintenance program which varies day to day due to technicians working faster or slower on specific tasks, then the method will likely be stochastic. This requires the inclusion and definition of uncertainty in the maintenance task list, and repetition of individual cases to gather statistical data. The fifth and final consideration has been touched upon already, that there is little historical data to validate the simulation [16]. In order to mitigate this validation issue, the method must be verified as much as possible so that the relationships between variables can be considered accurate qualitatively.

In such a system there are a multitude of potential sources of variation, and the more included the more accurate a model or simulation can be. Increasing the number of sources of variation will however adversely affect simulation time, and so a balance must be struck between accuracy and execution time [27]. Therefore, only those variables which are considered to have the largest effect on the major drivers of simulation metrics, and require the least amount of hard data or are based upon the bits of data known, will be included. By properly identifying these sources of variation and meaningful ranges they may be varied through, the fundamental behavior of this complex system can be uncovered [27].

To predict these sources of variation, an understanding of the interactions taking place within the system of interest must first be set down. For RLV O&M, vehicles are launched from some facility, perform a mission, return to a landing facility, and then undergo maintenance on each of their subsystems before they can be cleared to perform another mission. In order to represent this properly within a model, this maintenance cycle must be accurately captured. Maintenance on any aerospace vehicle can be assumed to proceed in roughly the same fashion. After a mission is performed regular maintenance is performed, and after a certain number of missions the vehicle undergoes scheduled maintenance which takes an in-depth look at the vehicle's state. In practice, the skills utilized by technicians working on a vehicle can be applied to several of the subsystems comprising it, however for the purposes of simplicity these skills are assumed to be mutually exclusive. Launch and landing sites may be at the same location or not, as the maintenance sites may also be located at either facility or elsewhere. In the case of maintenance sites placed at some distance from either launch or landing facilities, disassembly and/or integration facilities may be required. In order to capture the behavior of this complicated interconnected system of RLV O&M, as many of the potential procedural paths must be represented, and their calculable effects modeled to the greatest possible degree, which leads to the first research question of this study:

Research Question 1: What is the proper modeling method for capturing RLV O&M?

CHAPTER 2

LITERATURE REVIEW – PHILISOPHICAL BACKGROUND AND STATE OF THE ART

Previous research done on operations modeling, prediction, and optimization, has led to the inception of Operations Research (OR) as a practical application of state-of-the-art modeling practices. At its basis, OR seeks to optimize operations schemes by answering complicated decision-making problems whose solution requires addressing three major questions. The first is: What are the design alternatives? By constructing a model of the system under consideration, the possible modes of solution must be identified. For the purposes of this study, the answer to the first question would be in the sources of variation identified previously. In particular, by optimally allocating a maintenance crew to the regular upkeep of a fleet of RLVs (assuming mutually exclusive skillsets), it is expected that the overall effort spent performing that upkeep will be minimized. However to prove this hypothesis, modeling methods of sufficient power and fidelity must be utilized in order to answer the first research question. The second question is: Under what restrictions is the decision made? Any decision-maker reviewing the results of operations analysis must be basing their decision on some quantitative or qualitative metric. To answer the second question, the requirements of entities performing these campaigns much be taken into account. For a commercial RLV company, the total MMH per flight spent on a vehicle should be minimized so that the maximum number of flights per year can take place, generating the most revenue possible. The third and final question is: What is an appropriate objective criterion for evaluating the alternatives? In a situation with multiple metrics under consideration, particular decisions may produce a false optimum, where certain metrics are optimized while others are left non-optimal. In cases such as these, a proper way to compare and compromise these solutions is needed to give the decision-maker the most information possible.

The development of OR has also been spurred by the mistakes made by NASA. Although presently operational models can learn from many more years of data and practices, the specifics of performing maintenance on a system as complicated as a launch vehicle was simply not known. In general, the larger the vehicle the larger maintenance will take for that vehicle [43], however before the inception of the more powerful and descriptive tools developed from OR, the Saturn launch vehicle concept was originally justified in military studies for 100 flights/year [32] without much detailed information on how the support for this flight rate could be achieved. Further down the line, the original Shuttle operational concept was planned to achieve 40 launches/year from Kennedy Space Center (KSC) and 20 launches from Vanderburg Air Force Base (VAFB), relying upon a 2-week turnaround time [16]. It was not until the Shuttle had entered physical model testing that the original predictions on turnaround time were first challenged [16].During its operation however, the turnaround time would average to 88 days [15]. Due to this large gap in predictive ability, much effort was placed into the development of operational prediction models, each attempting to provide fidelity higher than previous models while incorporating the lessons learned from performing maintenance on the Shuttle.

As development in space continues into the future, the importance of considering the upkeep of complicated systems continues to gain credence. In particular, the International Space Station requires the longest logistical pipeline that has even been needed by a program developed by the U.S. [13]. There have even been reports of issues providing the correct amount of crew provisions [11]. Across the board there is a need for intelligent incorporation of operational concerns into the design of space systems, as unlike on the surface, once up into vacuum there are no corner stores with replacement or supplemental items when in a pinch.

Past & Present RLV Operations Models

As the prediction of RLV O&M is an important topic to many major companies in pursuit of spaceflight operations, there have been many tools constructed both by private and governmental entities over the years. What follows is a brief overview of several tools which have been developed in the past and continue to be used today.

The use of discrete event simulation (DES) to model the Space Shuttle actually began in 1970 before the Shuttle was approved for development, and those initial efforts suffered from the lack of an established baseline as there was no existing system from which to draw a good comparison [20]. In 1981 another simulation model was developed, showing for the first time that the original predictions of the Shuttle's flight rate were overly optimistic. It too suffered from lack of precedent and similarly relied heavily upon comparison to existing systems.

Later in 1997, Vision Spaceport was developed for the Highly Reusable Space Transportation (HRST) study by the Spaceport Synergy Team [52]. Due to the problem posed to the team during development, this tool is capable of predicting the effect of future concepts on operations, however there was a lack of relationships defined between functions for simulation modeling. Next, the Space Shuttle Ground Processing Simulation was developed at Kennedy Space Center (KSC), using all the information NASA had, which made it ideal for analyzing the Shuttle, however it had difficulty translating its results to other concepts [20]. It was so useful however that it was expanded around 2002 and is now known as the Generic Model for Future Launch Operations (GEM-FLO). Although this tool continues to be useful to NASA and its operations, it is only an upper-level view of RLV O&M, to such a point it was not deemed useful for the Air Force's RLV development [43]. To answer this need the Air Force Research Laboratory (AFRL) commissioned Boeing to conduct a study which would yield highly detailed operational analysis, resulting in the construction of the Space Operations Vehicle Operable Configurations Study (SOV-OCS) [43]. This study was so successful that it is subject to International Trades in Arms Restrictions (ITAR); however it is a static study which cannot incorporate changes in the maintenance workforce.

Another model which has undergone several stages was started by Dr. John Olds while he was still with the Aerospace Systems Design Laboratory (ASDL), called the Cost And Business Analysis Module (CABAM) [8]. Unlike previous studies, CABAM was based on fiscal units instead of labor metrics, and was capable of producing cost assessments for the entire life cycle of new launch vehicle concepts. It was extended later in 2009 by the team at SpaceWorks Enterprises, Inc. and renamed DESCARTES/Hyperport [34]. The finished model includes data gathered from several

NASA centers on not only the actions to be taken during maintenance, but also numbers of technicians normally allocated to these tasks. It is capable of approximating Shuttle operations, representing many different kinds of launch vehicles, and assessing the impact of future technology integration.

Finally, the most recent model found in a literature review is another model made at KSC, called the Launch & Landing Effects Ground Operations (LLEGO), developed around 2010. LLEGO is a big improvement on GEM-FLO, as it can model many different launch vehicle variants, and can calculate high and mid-level economic metrics for generic launch vehicle concepts.

Although each of these tools are very useful within their scope, most existing models are not robust enough or are too mired in detail and technical barriers to be useful [43]. In particular, these previous studies were only able to do rough order of magnitude (ROM) estimates for recurring costs, not detailed analysis [4], and don't account for variance, instead being based on expected values [5]. There is therefore a need for conducting a study which focuses where these tools have not: fitting a maintenance workforce skillset distribution to maximize operational efficiency, and incorporating variation so that the conclusions gleaned from such analysis are robust. By considering the efforts made in the past, and the potential for improvement inherent in answering this problem, the second research question of this work was formulated.

Research Question 2: *How can the skillset of a RLV maintenance workforce be optimized?*

CHAPTER 3

THEORY AND FORMULATION

In this chapter the overall methodology of the study performed will be presented and each portion justified. By its definition the third and final research question of this study will be addressed:

Research Question 3: How can RLV O&M be effectively captured by a model?

It is divided into three sections: describing the genesis of the study's methodology, presenting and discussing the assumptions and limitations inherent in the study, and finally defining the methodology for this study. During the genesis section I will present those sources which sharpened the first formulated research question into an achievable set of experiments, and how they lead to the research questions the remainder of this work has been dedicated to answer. In the assumptions and limitations section the issues with putting together experiments of this type are presented, along with any work-arounds identified either via a literature search or presented as a fundamental assumption of this work. Finally, the methodology for the study will be defined, including defining an overall evaluation criterion which will be applied to the results coming out of each experimental frame.

Genesis of the Method

During initial formulation of the work presented here, the possibilities of experimentation with RLV O&M allowed contained a multitude of potential paths. In particular, possible paths for investigation included, but were not limited to: combining limited data with failure models of components similar to those used on the Shuttle to predict failure rates of components associated with a subsystem (specifically the TPS), combining trajectory optimization software such as POST with current and future TPS materials to predict failure rates, investigating the potential benefits of a distributed maintenance workforce across sites across the country, the effect of increasing fleet size on a set maintenance workforce, and the optimization of a maintenance workforce skillset.

Since many other previous efforts have focused on the maintenance of a vehicle according to its subsystem configuration and the potential trades which can be done with the components of each, and that these studies have in many cases had access to sources not personally available [16],[43], performing a study relating vehicle configuration to maintenance characteristics was removed from the list of potential studies. Although initially intriguing, the effect of a distributed maintenance workforce was also eliminated from the list after performing a literature review and finding several sources [13],[17] which have found previously that as a RLV program continues, a centralized maintenance scheme was ultimately the most efficient, as it minimized the operational cost of keeping facilities up and running, and prevented costly delays in the maintenance cycle [13].

Investigating the issue of increasing fleet size with a given maintenance skillset distribution was initially considered to be the most intriguing of those subjects left, however it was ultimately eliminated for the purposes of relevancy. The entities which would be most interested in making RLV O&M its most efficient would be commercial ones, as governmental and military groups have the benefit of funds being provided for them, in contrast to commercial entities which have to generate their funds via launches to generate revenue. Commercial entities are just now beginning RLV campaigns, and those which are still attempting to get a single vehicle operating properly. It will likely be many years before any of these companies will have more than a handful of vehicles, and so maintenance skillset optimization for a small fleet was chosen as the path of investigation for this effort because it is currently relevant. Several studies have already been done attempting to do skillset optimization [38],[12], however these relied heavily on comparison with the currently existing B-2 and its maintenance workforce. Additionally, since several models are already performing very detailed analyses on vehicle configuration, studying the optimization of a maintenance skillset represents a relatively open field for development, and is considered to be an important field to explore [38].

Assumptions & Limitations

The first and most obvious limitation on the work presented is the lack of historical data by which to base experiments on and validate their results by, a problem which has plagued most other RLV O&M efforts [5]. Most of the assumptions present in this section are a direct result of this fact. A high-fidelity model of RLV O&M would require information on the relationship between vehicle components and their failure modes. These failure modes could then be related to a mission profile in order to calculate part failure on a mission-to-mission basis. In order to service these failed parts, maintenance technicians would repair or replace these parts, which to model would require information on the numbers of technicians required to perform maintenance on each of a RLV's subsystems, which is not available. In addition, the required amount of time to finish these maintenance tasks is not available [3], and by extension neither is data on the variation present within these task times, which would be a great asset to this study. Furthermore, the effect of having fewer or less available technicians on task times is also not known for RLVs.

In order to mitigate this lack of information, the work presented here has strived to use any and all historical data and relationships available, and to use as few assumptions as possible. When data is unavailable, the assumptions used are clearly defined and justified to the best possible degree. In this manner, while the experiments performed cannot be validated at the present time, they will be verifiable. As the issue of RLV O&M becomes more and more important and commercial entities performing RLV O&M compile data, the method constructed here will be equally applicable.

The first big assumption is that *RLVs undergo the same maintenance cycle as any aerospace vehicle*. This is to say that for any mission, a RLV will depart from some launch site, perform a mission, and return to some landing facility, where it will enter maintenance. Maintenance is completed at one or several centralized locations. Upon completing maintenance, the vehicle embarks on another mission and the cycle repeats.

Maintenance Task List

The second assumption of this work is that the *maintenance of a RLV can be represented by a task list comprising 16 subsystems*. In this work the maintenance of a single RLV is divided into work on: Avionics, Communications, Crew systems, Electrical & Wiring, Engines, Environmental Controls, Flight Controls, Hydraulics, Landing & Recovery, Navigation, Pneumatics, Propellant Management, Software, Structures, Thermal Protection System (TPS), and Tracking. Work on any of these subsystems is then subdivided into a set of tasks required for the subsystem to be properly checked out and maintained after each flight. Both the subsystems included and the tasks performed to maintain each of them come from the FAA's Guide to Commercial Launch Vehicle Operations & Maintenance [26]. There are many variants of maintenance task lists which have been used in previous work, however the task list taken from the FAA's guide is assumed here to be representative of a minimum task set, as they are listed in the guide because they have a direct impact on the safety of an RLV. A commercial entity would not only wish to cut maintenance costs by reducing their workforce while keeping them working efficiently to minimize the MMH spent, but also by performing the minimum amount of maintenance required to keep the vehicle operating safely. For this reason, the FAA guide task list was chosen to represent a bare-bones maintenance task list.

Maintenance Architecture

The third assumption is that the *tasks performed during maintenance of a RLV have different levels of complexity*, resulting in shorter or longer completion times. The first justification for this assumption is purely qualitative: tasks on any subsystem will have higher or lower importance related to its continued safe functioning. Those tasks with higher importance will be under the most scrutiny and thus will take a longer amount of time. The second justification for this assumption is from the scattered amount of information available. In the literature found which contained some high-level aggregated maintenance information [16], [43], although the tasks represented are consistent they do show variation in the amount of time required for maintaining their representative subsystems. For the purposes of this study, the line between abstraction and fidelity will be placed by defining each maintenance task identified from review of the FAA guide to requiring 1, 2, or 3 days to complete. To define these completion times, all documents containing information on tasks whose description approximates those defined in the FAA document were consulted. The full task list and the required time for completion for each are contained in appendix A. Most of the included tasks are set from considering multiple sources, however a few representative examples of this analysis are traced to specific entries within "Space Shuttle Operations and Infrastructure: A Systems Analysis of Design Root Causes and Effects." McCleskey, C. M. April 2005. NASA/TP - 2005-211519. Task #301 included in the resource shows SSME inspection done by Rocketdyne to take 24 hours, and so the inspection of engines has been similarly set to 24 hours (3 days) in this work's task list. Wheel inspection & removal (tasks #340,341) take close to 8 hours each, and so the inspections in the landing & recovery subsystem are each set to 8 hours (1 day). The environmental purge recorded (task #382) is 16 hours, and so the Atmosphere related task in the Environmental system is set to 16 hours (2 days). Finally, GPS troubleshooting (task #1036) is set to 8 hours, and so the GPS task in the Navigation subsystem is set to 8 hours (1 day).

Just as the maintenance tasks to be performed have varying levels of complexity, the numbers of technicians allocated to maintenance on subsystems will present its own complexities in communicating effectively what tasks need to be performed, by whom, and at what pace. This leads to the next assumption that *by allocating more technicians to maintenance on a subsystem, the total amount of time that maintenance will take is decreased*. There is a limit however; as previous studies on maintenance workforce have shown that increasing the workforce will only help up to a certain point [39], which leads to the next assumption: *there is a maximum number of technicians that can work on a subsystem at a time*. Without the example of previous models, this assumption can also make sense via a qualitative example: maintenance of the TPS subsystem of a RLV. While technicians are repairing or replacing tiles underneath the Shuttle, there is a physical limit to the amount of space within which they have to work. In order to increase the rate at which tile maintenance is done more technicians can be placed in the same area, but at some point there will be no more physical room for them to be placed. At this point, you have a 'too many cooks in the kitchen' situation, where placing more technicians on the task may actually hamper progress.

In order to reduce model complexity, it will also be assumed that *maintenance on any subsystem requires the use of a specific set of skills, which is mutually exclusive amongst subsystems*. This assumption comes from the complexity of subsystems making up a RLV and the fact that they perform a variety of tasks, requiring at the very least separate tools to maintain. In addition, this assumption removes the problem of characterizing how technicians allocated to subsystems could interact with one another.

Another limitation of this study resulting from lack of available information is the effect of allotting fewer technicians to a task on the task's required time for completion. In general, it is expected that fewer technicians will result in an increase of required time. Two models were considered to answer this limitation, a linear and reciprocal model. There are a few issues with a linear model, first that the intercept will be nonsense, as the amount of time 0 technicians can perform any task would theoretically be infinite. Secondly, assuming that by allocating the maximum number of technicians to a task the most efficient maintenance occurs resulting in the lowest amount of time yields a single point by which to base the model on, however a linear model requires at least two to define the slope. An assumption can be made here about the slope, however the alternate model was found to be superior.

The second (reciprocal) model is a better fit for several reasons. In software engineering, there is an effect known as Brooks' Law which in general states that at a certain point, including another person into the completion of a communal task will actually increase the amount of time to complete [42]. The phenomena is justified by stating that in any project requiring communal involvement, as the number of associated people increases the communications pathways required for efficiently working on the task becomes more and more complicated, resulting in diminishing returns as people are added onto a project. So according to Brooks' Law which is based on observation, adding more people onto a project like a maintenance task will initially have a large decreasing effect on the completion time. With each person however, the increase in efficiency is reduced, until reaching an inflection point where more people will cause an increase in the completion time. In conjunction with the previous assumption that there is a maximum number of technicians which can physically work on a subsystem at a time, and that a reciprocal model only requires one point for regression, the reciprocal model on task completion time is seen as a better descriptor for this unknown effect. The model is shown pictorially below in Figure 1, showing a rapid decrease as technicians are added until the maximum number of technicians is reached, after which there is no further decrease in task time. The selection of this model is the next assumption of this study: *the effect of reducing relevant workforce to the maintenance of a RLV subsystem's* *maintenance completion time follows a reciprocal trend.* The reciprocal model will be used whenever applicable; however there will be one example problem which uses the linear model.

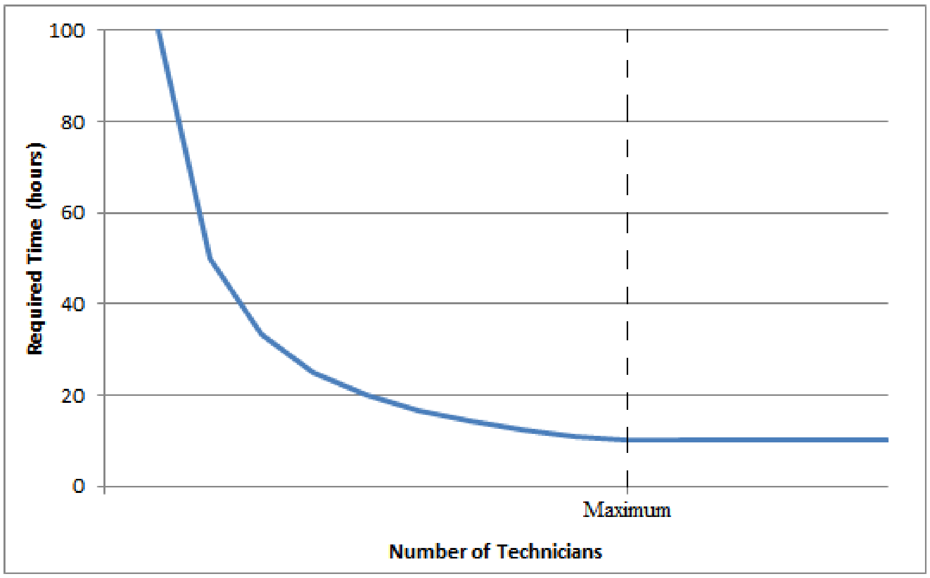

Figure 1: Reciprocal task time model

The fundamental equation for this model is shown below in Equation 1.

$$\mathit{Time\ with\ fewer\ technicians} = \frac{\mathit{Max.\ Technicians\ Allowed} * \mathit{Time\ assuming\ max.\ Technicians}}{\mathit{Technicians\ Available}}$$

Equation 1: Task completion time model equation

Although now the expected amount of time for a maintenance task has been modeled as a function of the number of available technicians, one of the fundamental aims of this work is to incorporate variation into the design so that the results gleaned can be robust. To that end, the amount of time a task will actually take must change from one execution to another, requiring the inclusion of probabilistic methods, which will now be addressed.

The log-Normal Distribution

The log-normal distribution has a direct application to RLV operations modeling, as the time for completion of individual maintenance tasks roughly follows a log-normal distribution, as there is less chance of the task taking less time than the average amount of time than the chance of the task taking longer than expected. This is due to the log-Normal distribution having tail behavior that is slower than exponential, allowing for data with a 'heavy' tail [28]. Some documents compiling task completion times have concluded that task time variation does in fact follow a log-Normal distribution [3], although this is based on only 29 of the 243 flights of the Shuttle program. Other modeling efforts have used a triangular distribution [55] with a 'fat' right tail, however NASA and military documents [37],[25] have chosen the log-Normal distribution to describe maintenance task completion times and the inter-arrival times of maintenance events [27].

Statistically speaking, the log-Normal distribution has positive skew (a fatter right tail) so more of the distribution lies to the right of the mean. It arises when the logarithm of a random variable is normally distributed, or the distribution of the random variable X when log(X) follows a Normal distribution with mean μ and variance σ^2. The probability density function (PDF) of the log-normal distribution is

$$f(x) = \frac{1}{x\sqrt{2\pi\sigma^2}} e^{-\frac{(lnx-\mu)^2}{2\sigma^2}}$$

Equation 2: log-Normal PDF Equation

$$E\{x\} = e^{\mu+\frac{\sigma^2}{2}}$$

$$V\{x\} = \left(e^{\sigma^2} - 1\right)e^{2\mu+\sigma^2}$$

Equation 3: Mean and Variance of log-Normal Distribution

Because the log-Normal distribution is used so widely in describing task times and has been empirically observed, it will be used for each of the maintenance tasks performed.

The assumptions which have just been presented are the result of an extended literature search which has sought to find the best justification and precedent possible. When taken together, they constitute the answer to research question 2, and are summarized here.

Answering Research Question 3: Maintenance of a RLV can be represented with the following 8 assumptions:

1. RLVs undergo the same maintenance cycle as any aerospace vehicle
2. RLV maintenance can be represented as composed of 16 subsystems
3. Tasks performed during maintenance have different levels of complexity
4. Allocating more technicians reducing the necessary maintenance time
5. There is a maximum number of technicians which can work on a RLV at a time
6. Maintenance on a subsystem requires unique skills

7. Changing the allotted number of technicians for a subsystem has a reciprocal effect on maintenance time

8. The variation in a maintenance task's completion time follows a log-Normal distribution

Methodology of Study

There are several types of O&M schemes which are of interest to this study. Each has strengths and weaknesses for the overall scheme and it is the proper combination of these schemes which it is expected will produce an optimal maintenance skillset distribution. The first scheme is to perform maintenance on one vehicle at a time, allocating the maximum number of technicians to work on each of the subsystems. This has the advantage of utilizing the skills of each subsystem’s workforce to its maximum; however it maximizes the number of technicians kept on hand. In Figure 2 below, this sort of scheme is shown notionally with a subset of the total number of subsystems. In each of the following figures, green represents maintenance done on Vehicle 1, blue represents maintenance done on Vehicle 2, and orange represents maintenance done on Vehicle 3.

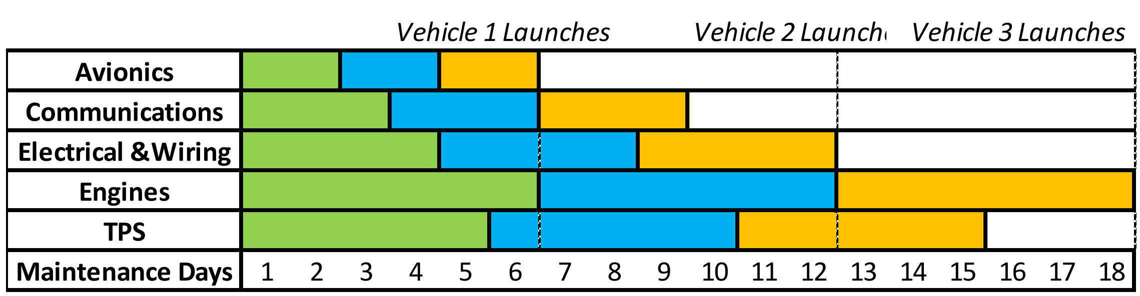

Figure 2: One at a time maintenance with maximum technicians

As can be seen, one subsystem (Engines) in particular drives the total amount of time required for satisfactory maintenance. An improvement on this scheme would be to reduce the number of technicians allotted to lower-time subsystems (Avionics, Communications), which would reduce the efficiency of maintenance on those subsystems and increase the time spent on them, but would reduce the total number of technicians required.

An example of the reduced workforce scheme is shown below in Figure 3. The lower-time subsystems (Avionics, Communications) are taking longer than previously to complete due to fewer available technicians, and all others are still at their most efficient. As can be seen below, the higher time subsystems are still driving the launch rate.

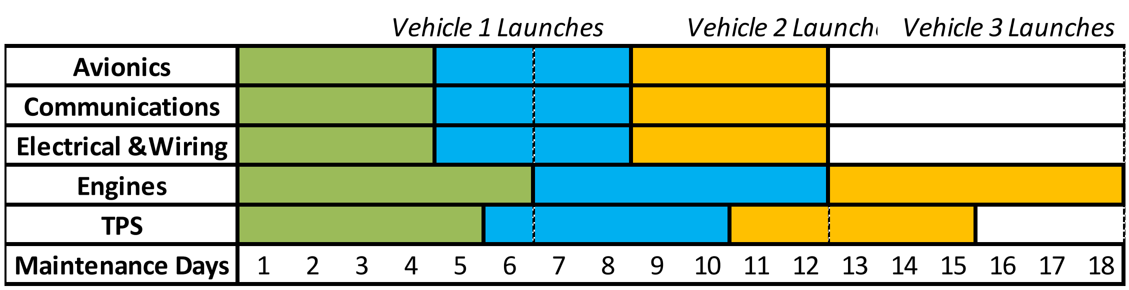

Figure 3: One vehicle at a time with reduced workforce

To improve upon this scheme further, the benefit gained from having a smaller workforce can be redirected into allotting more technicians to those subsystems which drive launch rate. In this manner, multiple vehicles may be worked on simultaneously, depicted below in Figure 4. In rows where multiple colors are present multiple vehicle maintenance is being represented.

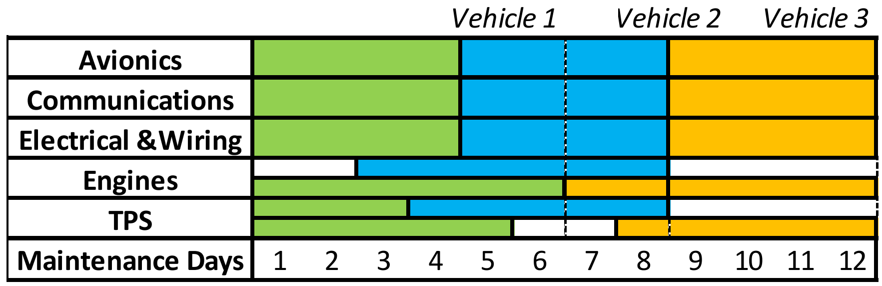

Figure 4: Multiple vehicles at a time with optimized workforce

By optimizing the maintenance workforce by allocating fewer technicians to those subsystems which require the least amount of time and do not drive launch rate, and allocating more to those which require the most amount of time so that multiple vehicle maintenance can take place, the launch rate can be improved upon while potentially reducing the overall workforce required. The highest-time subsystem (Engines) is still driving the launch rate of individual vehicles; however by performing overlapping maintenance this issue can be mitigated.

Overall Evaluation Criterion

In order to answer the second research question of this study, several competing methods for finding an optimal skillset distribution for RLV O&M will be presented and compared by the results they produce. As they are presented, the methods will grow in complexity and power, finally resulting in a justification for the use of discrete event simulation. For the final simulation constructed, a couple methods of optimization will be presented. To compare the results coming out of the competing optimization methods, a common basis is needed. To do so, an overall evaluation criterion (OEC) will be applied to the results coming out of each method in the simulation. An OEC is a method for solving multi-objective problems, by converting the original multiple objectives into a single goal represented by a single equation, upon which minimization or maximization is sought [48]. The resulting solution coming from an OEC does not always yield a solution which is optimal in each of its components; however it is the most efficient solution because its use yields a solution which performs the best across all categories [56].

To define the components of the OEC, the optimal technician skillset allocation distribution will be considered as a commercial entity would. In particular, a commercial entity would strive to achieve two things: maximize revenue and minimize costs to maximize profit. To maximize revenue, a commercial RLV company would want to maximize their flight rate, and so the maximum achievable flight rate will be included as a factor in the OEC. Two factors go into minimizing costs. The first is operational costs, the cost of ‘keeping the lights on’, and the second is personnel costs, or the cost of compensating the maintenance workforce for their work. To minimize the first, a commercial entity would seek to minimize the MMH spent on a vehicle after every flight, and so the average MMH/Flight/Vehicle will be included in the OEC. Secondly, by minimizing the necessary workforce to achieve maximization of flight rate and minimization of MMH/Flight/Vehicle, a commercial entity can drive down its personnel cost, and so the number of technicians allotted to maintenance on each subsystem will also be included in the OEC. In concert, the metrics defined above are placed together into the OEC shown in Equation 4 below. In finding the proper experimental frame and method within that frame, the maximum value of the OEC will be sought.

$$OEC = \frac{Flight\ Rate}{Max.Flight\ Rate} + \frac{Min.MMH\ per\ Flight}{MMH\ per\ Flight} + \frac{Max.Technicians}{\#\ Technicians}$$

Equation 4: Overall Evaluation Criterion

CHAPTER 3

EXPERIMENTAL DESIGN ALTERNATIVES

Before deciding on a platform upon which to perform experiments with O&M schemes, several alternatives must be investigated. Operations research (OR) has been going on for over half a century, beginning in World War II with Leonid Kantorovich using linear programming (LP) for logistics planning to predict expenditures and maximize enemy losses [36]. Once the war was over, his methods were widely used in industry for daily planning. As the needs of industry grew more complex, so did the methods used grow in complexity. The computational ability of computers similarly grew, eventually leading to the formulation of simulation. In the present day, simulations are the most widely used tool for complex OR problems.

The reason simulation is used so much is three-fold. First, simpler methods may not have the capability of solving the problem. This situation occurs when the effects a modeler is attempting to capture cannot be represented inside a LP formulation, such as the reciprocal dependency between technician availability and maintenance time presented in Figure 1. For many cases however, linear models are sufficient to perform rough order-of-magnitude (ROM) studies. Secondly, the problem may not have a closed-form solution. This situation can occur when a system a modeler is examining is a repetitive cycle of relationships, such as the maintenance cycle an aerospace vehicle undergoes over the course of its life-cycle. In this case, the results from previous loops affect the parameters describing the maintenance system, which cannot be captured by simple methods. Finally, the problem itself may be dynamic. This point can also be illustrated by the previous example, however it has another example. Over the course of an aerospace vehicle fleet campaign, individual vehicles or maintenance sites can change in time. For the vehicle these changes can take the form of replacing the TPS materials on a RLV, changing mission profiles, integrating new technologies, or retiring individual vehicles. For the maintenance sites the availability of technicians with certain skills can fluctuate, or new practices or tools can emerge which increase the efficiency of individual tasks. Any of these changes will fundamentally alter the way in which vehicles and maintenance interact, and without mechanisms to capture changes in time, simple methods cannot capture the effects of complicated behaviors.

The construction and execution of a simulation is very useful for the examination of alternatives. Within a computer-based simulation model, the speed at which analysis can be accomplished is much higher than waiting for a physical system to operate. Simulations are also much easier to manipulate than physical systems, providing a framework for testing the desirability of system modifications. Simulation-generated data often can provide sufficiently accurate estimates on the performance of alternatives under consideration, allowing the operator to sharpen their understanding of the system as a whole. Predicting the performance characteristics of a system before it is a physical entity is very useful for aerospace vehicles as described previously in the introduction, as by bringing more information into the conceptual design phase via simulation, the risk associated with multitudes of design alternatives may be calculated and compared.

In the conceptual construction of a simulation, several steps become important for making the simulation generic enough to capture many possible alternatives [54]. First, the domain of interest must be selected so that the objectives of the study may be adequately represented. If the simulation is too general, its construction and execution time may render the simulation inefficient, and so the modeler must find a balance between fidelity and efficiency. For the problem at hand, O&M of RLVs is selected as the domain of interest.

Second in a simulation's conceptual construction, the processes and interactions of the model as entities flow through it must be identified. Another balance must be struck here between model usefulness and abstraction similar to the first step. In this case though, if the simulation is found to be insufficiently descriptive, the addition of details may increase fidelity. Walking the line between fidelity and abstraction here requires identifying those factors within RLV O&M which can be captured using the limited amount of data available. An example of such would be defining a default set of maintenance tasks a RLV will undergo in one cycle versus linking the individual components comprising the vehicle to tasks required for each of them. Another example would be either to define a default length of time the RLV is on mission according to a sampling of mission profiles, or to perform complex calculations requiring a detailed breakdown of the aerodynamic characteristics of the vehicle, its propulsion, and the propulsion system's components, both for launch and EDL. Finally, the length of pre-flight operations can similarly be set to a default value from historical data, or could similarly be calculated with complicated analysis which would require extensive managerial and logistical information. For the purposes of this study, historical precedent will be used wherever possible to increase computational efficiency.

Third in conceptual construction, the constructs that make up the system must be characterized in the context of their interactions. In RLV O&M, these constructs would be the vehicles undergoing and maintenance sites performing maintenance. As discussed previously, the general characteristics of RLV O&M are very similar to the maintenance of any aerospace vehicle. What specifically happens then is subject to the assumptions of the modeler attempting to balance abstraction and fidelity.

Once the previous steps have completed, the system can then be represented by computer code which strikes the balances mentioned above. This step also requires selection of a simulation method, which will be expanded upon shortly. As a final step and to address the concerns of abstraction and fidelity previously discussed, the modeler must find any and all existing information about the behavior of the system to be simulated in order to base simulation in reality as much as possible.

What follows is an overall review of some modeling & simulation methods used for investigating operations. Each method is very good within its own domain; however each require much more quantitative information about the system under consideration than is available. Although none of these models can satisfy the requirements posed in the previous section, elements of each are present within the method ultimately chosen: Discrete Event Simulation (DES).

Linear Programming

The simplest method in OR is LP, which is fundamentally a collection of mathematical modeling techniques designed to optimize the usage of limited resources. Although simple, LP models form the basis for more complicated models, and allow the characterization of steady-state or reduced forms of complex problems. Its basic assumption is that an objective function representing the goal of the modeler and constraints on achieving that goal can be expressed as linear functions of decision variables representing the entities which affect the realization of that goal. The decision variables are the entities within the model which are controllable inputs to the system, and may be either an equation or inequality. The objective function or 'goal' of system effectiveness can then achieve optimality with limitations imposed by the constraints by varying the decision variables. Solving a LP problem thus requires finding the set of decision variable values which satisfy the objective function in the best possible manner.

Each decision variable is represented in its most general form asx_i, so that the objective criterion is the minimization or maximization of some function

$$z = c_1x_1 + c_2x_2 + c_3x_3 + \cdots + c_nx_n = \sum_{i=1}^{n} c_ix_i$$

Equation 5: LP Objective function definition

Where the c_i are problem-dependent constants. Resource limitations may sometimes restrict the values of the x_i , which can be represented as

$$a_1x_1 + a_2x_2 + a_3x_3 + \cdots + a_nx_n = \sum_{i=1}^{n} a_nx_n \leq b$$

Equation 6: LP constraint definition

Where b quantizes the resource shortage. There are two types of resource restrictions: the first has all a_n positive and represents a resource usage maximum; the second has both positive and negative a_n , which states that the difference in the value of those decision variables must satisfy the constraint. In addition, since the decision variables are to represent physical entities, non-negativity restrictions are placed on each:

$$x_i \geq 0$$

Equation 7: LP non-negativity constraint

In many operations problems, the amount by which any quantity goes over or under a certain value is also of interest. For these cases slack and surplus variables are introduced into the formulation. A slack variable is used for constraints of Equation 6's form, $\leq$ to some constant. A slack represents the amount by which the available amount of a resource exceeds its usage, which is of great interest when the variable in question is representing the items in an inventory. A surplus variable is used in cases where the constraint is $\geq$ some constant, representing the excess over a minimum requirement. By utilizing these two types of variables, the inequalities of Equation 5and Equation 6 can be made into equations. Defining s_i as either slack or surplus variables, Equation 6 can be re-formulated as

$$a_1x_1 + a_2x_2 + a_3x_3 + \cdots + a_nx_n + s_1 = b \quad (Slack)$$
$$a_1x_1 + a_2x_2 + a_3x_3 + \cdots + a_nx_n - s_1 = b \quad (Surplus)$$
$$s_1 \geq 0$$

Equation 8: Definition of LP slack & surplus variables

Using this formulation, the inequalities have been transformed into equalities with extra constraints. In addition, the inclusion of slack and surplus variables changes the objective criterion to the minimization or maximization of the slack and/or surplus variables.

Any set of decision variable values satisfying both Equation 5 and Equation 6 is called a feasible solution; however the real interest is in identifying the optimum feasible solution which yields the maximum system effectiveness. In the case of a LP model with simple objective and resource constraints, the set of feasible solutions is infinite, and so efficient procedures for identifying the optimum feasible solution are required.

Simplex Method

The simplex method can be summarized as an algorithm for identifying the corner or extreme points of a solution space. As a first step toward constructing the simplex method, the general model must be converted to standard LP form, which utilizes the slack and surplus variables introduced in the previous section. From this conversion, the LP problem exhibits a basic solution which comprises all the corner spaces of the solution space. This method is so useful for solving LP problems that it has been used from its inception in the 1940's to the present day [56],[24].

Converting into standard LP form has three steps. First, all the constraints must be equations with non-negative right hand side, non-negativity restrictions excluded. Because the value of z (Equation 5) may be negative, each decision variable is moved to the right-hand-side. Secondly, all variables must be non-negative. In cases where a variable must have the potential for negative values, a substitution is required. For any such variable, the substitution

$$x_j = x_j^+ - x_j^-, \quad x_j^+, x_j^- \geq 0$$

Equation 9: Non-Negativity Conservation

Is used to conserve non-negativity. Third, the objective function must be a maximization or minimization requirement. In cases where the problem calls for one while computational efficiency favors the other, a sign substitution may be made, as the maximization of a function $f(x_{1\to n})$ is equal to the minimization of $-f(x_{1\to n})$.

Once the above procedure is carried out, determination of basic solutions can proceed. The standard LP form includes *m* simultaneous linear equations or constraints in *n* unknowns or variables ($m < n$). In order to determine the corner points of this solution space, the *n* variables are divided into two groups: *n-m* variables which are set to 0; and the remaining variables are set by solving the resulting equations. If the reduced set of variables yields a unique feasible solution, they comprise a basic solution, and are called basic variables, while the zeroed variables are non-basic. In the case where there are few decision variables, this is equivalent to finding the constraint equation's intercept on that variable's axis. By iterating through each such set, the corner points of the problem at hand are identified. Based on the definition of the simplex method, the maximum number of possible basic solutions for *m* equations in *n* unknowns is

$$\binom{n}{m} = \frac{n!}{m!\,(n-m)!}$$

Equation 10: Number of possible solutions for LP problem

Which is a far cry from the potentially infinite set of feasible solutions. It is in this manner that the simplex method reduces the set of feasible solutions to investigate to the value of Equation 10 for the problem at hand.

In order to increase the efficiency of this process further, the simplex method provides an algorithm for moving closer to the optimal feasible solution with every iteration of analysis. In the first iteration, a basic solution is found which may or may not be optimal, and will have some number of decision variables non-basic. On the next iteration however, some basic variables will become non-basic, and vice versa. The basic variable which is zeroed is called the leaving variable, and the non-basic which becomes non-zero is the entering variable. In order to choose which will be which, the direction of greatest improvement in the objective criterion is chosen, which is the decision variable with the largest (in the case of a maximization problem) non-negative coefficient. The value of the entering variable is chosen by finding the ratio of each constraint equation's right-hand-side value to the entering variable's left-hand-side coefficient. The minimum non-negative value of these ratios represents an intercept of constraint equations, or a corner point of the solution space. At this point, the solution found may still not be optimal, and so Gauss-Jordan row operations [56],[24],[19] are performed in order to move from this corner point to one which is at least more optimal than the one found by the latest iteration.

Altogether, the simplex algorithm is composed of 4 steps, subject to two conditions. The first, optimality, states that the entering variable in a maximization (minimization) problem is the non-basic variable having the most negative (positive) coefficient in the objective criterion equation. The second, feasibility, states that for maximization and minimization problems, the leaving variable is the basic variable associated with the smallest non-negative ratio. In the first step, a basic feasible solution is found by zeroing decision variables. Second, an entering variable is selected using the optimality condition. If none satisfy the condition, the current solution is the most optimal. Third, a leaving variable is chosen via the feasibility condition. Fourth, a new basic solution is found using Gauss-Jordan row operations, and the process repeats at the second step.

Inherent in the construction of a LP model are two properties: proportionality and additivity. The first; proportionality, requires that the contribution of each design variable in both the objective function and constraints to be directly proportional to the value of the variable. This property limits LP models to capturing effects which can be represented via a linear equation. LP methods cannot, for example, be used to investigate the effect of the reciprocal model on the relationship between available maintenance technicians and maintenance time. The second; additivity, requires that the total contribution of all variables and constraints be a direct sum. In cases where the objective function or constraints may have cross-effects or recursive ones, such as an increase in flight rate of a RLV fleet resulting from trading personnel allocated to shorter-length maintenance subsystems to a longer-length subsystem or working with a reduced number of technicians on an individual vehicle because more are being used in the maintenance of another vehicle.

In cases where either proportionality or additivity is violated, then more complex methods are needed. In addition, the characterization of an optimal feasible solution requires that all c_n and a_n are constants known in advance. If there is any uncertainty in any of these values, then more complex methods are needed to compensate. Therefore the algorithm for moving beyond LP would be to check that the relationships of decision variables, objectives, and constraints satisfy proportionality and additivity, and that all problem-specific coefficients are known.

RLV O&M LP Model

Due to the assumptions presented in the previous chapter, the ability of a LP model in capturing RLV O&M is very limited. Due to the proportionality property, a LP model is limited to the linear model of maintenance times. In addition, as the MMH/Flight is a non-linear function as multiple vehicles are incorporated, a LP model can only represent maintenance on one vehicle. An objective function for a RLV O&M LP problem could be defined

$$minimize\ MMH = \sum_{i=1}^{16} -\frac{Lowest\ time\ subsystem\ i}{Maximum\ techs\ for\ subsystem\ i} * x_i$$

Equation 11: RLV O&M LP model objective function

Where the x_i represent the number of available technicians for work on subsystem i. Factors in the objective function are all negative because by increasing the x_i associated with a subsystem, overall MMH decreases which drives the solution toward global minimum.

For the LP model all 16 subsystems identified in the previous chapter are included, with the expected amount of time for individual tasks comprising maintenance on that subsystem added together to give an average time for maintenance on that subsystem. These average values are shown below in Table 1, whose values can be found by adding together the number of days required for each subsystem and multiplying that number by 8, which assumes 1 8-hour maintenance shift per day. The values in the table comprise the 'lowest time for subsystem *i*' variable in Equation 11, and the 'Maximum techs for subsystem *i*' variable is assumed to be 15 in each case.

Table 1: Expected times for maintenance actions by subsystem

Maintenance time (Hours)

Avionics	64	**Landing & Recovery**	88
Communications	104	**Navigation**	72
Crew	32	**Pneumatic**	104
Electrical & Wiring	152	**Propellant Management**	48
Engines	120	**Software**	88
Environmental	72	**Structures**	88
Flight Controls	104	**TPS**	72
Hydraulics	112	**Tracking**	80

Constraints on each x_i restrict its value to be non-negative, and between 1 & 15, the maximum being the assumed maximum capable of performing maintenance on a subsystem, or $0 \leq x_i \leq 15 \;\; i = 1,2,\dots,16$.

Due to the simplicity of the model up to this point of its construction, it is obvious that the solution which minimizes the MMH is to maximize all x_i, or $x_i = 15 \;\; i = 1,2,\dots,16$, and this result is found via the simplex method, resulting in a minimum MMH of 1400 hours. This is due to the fact that the model does not represent any point of diminishing returns from allotting more technicians to the maintenance of any subsystem. More interesting behavior can however be found by including another constraint. In particular, the method present in the discussion with Figure 3 can be investigated, in which the technicians allotted to maintenance on lower-length subsystems are reduced since the higher-length subsystems are driving flight rate. Since the proportionality for each subsystem is common (scaled to the maximum of 15 technicians), the number of technicians allotted to each subsystem can be scaled by the subsystem with the longest necessary time. According to the task list defined previously, this subsystem is Electrical & Wiring, at requiring 19 days to complete. Therefore, an additional constraint can be added to the LP model of the form

$$x_i \leq \frac{Lowest\ time\ subsystem\ i}{Lowest\ time\ Electrical\ \&\ Wiring}$$

Which has the effect of trading individual subsystem maintenance efficiency for a reduced technician workforce. After solving this problem via the simplex method, the technician availability levels for each subsystem get as close as it can to the constraint above, resulting in the distribution shown below in Table 2.

Table 2: Improved LP model skillset distribution

Available Technicians			
Avionics	6	**Landing & Recovery**	9
Communications	10	**Navigation**	7
Crew	3	**Pneumatic**	10
Electrical & Wiring	15	**Propellant Management**	5
Engines	12	**Software**	9
Environmental	7	**Structures**	9
Flight Controls	10	**TPS**	7
Hydraulics	11	**Tracking**	8

Using the distribution above, the MMH required increases to 1912 hours, which is a 37% increase in MMH over the previous optimal solution maximizing the technicians available for maintenance on each subsystem, however with a 58% reduction in workforce. Here is where this simple model shows its merit: even with an oversimplified version of the system under consideration, with a few assumptions about how an optimal maintenance workforce would be constructed the method produces a skillset distribution which can do simple trades between workforce and performance efficiency. The computer codes used for both these calculations are included in the appendices.

Conclusions

Instead of using a modeling method, the tool must be a simulation of RLV O&M. The reasoning here is three-fold: the methods do not have the capability of solving the problem, the problem does not have a closed-form solution, and the problem itself is dynamic. In the case of RLV O&M, all three of these cases are true in addition to another motivating factor: lack of historical precedent. In addition, the information which is available is scattered and usually not very informative. An example of such would be the average amount of time spent on the STS for O&M (88 days [15]) because there is no breakdown to individual task times or variance. In addition, as models previously used for predicting RLV O&M have normally used expected values instead of allowing for variation, a simulation which can incorporate uncertainty should be used in order to produce a solution which is robust. Such a solution would incorporate the variability inherent in RLV O&M and therefore would perform optimally with varying conditions. For all these reasons, the search for an experimental frame moves to simulation. There is still a chance that a simple method such as that shown above could find roughly the same optimal settings a more involved technique could, and so the technician availability levels found here will be investigated again when the full experimental frame is constructed.

Monte Carlo Simulation

A simple example of simulation is the Monte Carlo method, which tends to be used when it is infeasible to compute and exact result with a deterministic algorithm [29]. Essentially, the method uses random sampling in order to estimate the output of an experiment. Its inception and practice is thanks to improvements in computing technology allowing a multitude of computations to happen every second. As an example application, consider the task of approximating the value of π. To construct this problem, define a unit square within an xy-plane with corners at the origin and (1, 1), and then inscribe a quarter-circle centered at the origin of radius 1. The ratio of the two areas is then $\pi/4$. To proceed, the shape is randomly populated with a large number of points. By taking the ratio of the number of points which fall in the quarter-circle over those which fall outside but still within the square then multiplying by 4, the method yields an estimate of π. After a sufficiently large number of points have been placed in the area, the approximation can come very close to the value of π.

Without the requirements of proportionality and additivity, the Monte Carlo method is able to incorporate the reciprocal model of task times and the non-linear effect of multiple vehicle maintenance. The Monte Carlo method is also useful because of the fact that it can incorporate variation in its calculations. As a specific example, to improve upon the LP model constructed in the previous section, the number of hours required for maintenance on a specific subsystem can be allowed to vary within a particular range so that the inputs passed into the simulation the stochastic effects expected of the RLV O&M system. By incorporating this variation, results coming out of Monte Carlo can be more robust, however at a price. In stochastic systems, unless the optimum set of input variables is assumed to be one point, which in a stochastic system is likely to change from one run to the next, then it may not be possible to find a global minimum [60]. In addition, finding all minima to find global minimum is typically very difficult, and in some situations may be impossible.

Monte Carlo Method Model

The Monte Carlo simulation constructed over many repetitions attempts to find an optimum maintenance workforce which minimizes required workforce with the greatest possible flight rate. The scenario of this simulation is as such: a number of RLVs enter maintenance simultaneously, and the allocated workforce works on them assuming 1 8-hour shift per day, 5 days per week, and 52 weeks per year. If a particular subsystem has been allocated enough technicians to work on multiple vehicles, then the first vehicle is allotted up to the maximum (always set to 15) number of technicians and any subsequent vehicles are allotted the difference from the maximum. Each of the subsystems represented by the Monte Carlo model has the same time to complete as in the LP model, represented by ‘Lowest Hours’ in the equation below, however in this case the time to complete will be reciprocally related to the number of available technicians. The governing equation is

$$\mathit{Time\ to\ complete\ maintenance\ subsystem\ i} = \frac{\mathit{Lowest\ Hours} * \mathit{Maximum\ techs.}}{\mathit{Available\ techs\ subsystem\ i}}$$

Equation 12: Reciprocal completion time model for Monte Carlo

After cycling through each vehicle for a particular subsystem, if the original allocation allowed for multiple vehicle maintenance, then the maximum time spent amongst the vehicles represents how far into the year maintenance on that subsystem has required. If only single vehicle maintenance occurs for that subsystem (original allocation for that subsystem of less than the maximum of 15), then the sum of required times represents how far into the year maintenance on that subsystem has required. The scenario is shown pictorially below in Figure 5.

Figure 5: Comparison of single and multiple vehicle maintenance

<table>
<tr><th></th><th>Maintenance Days</th><th>1</th><th>2</th><th>3</th><th>4</th></tr>
<tr><td>Single Vehicle</td><td>Subsystem 1</td><td colspan="2">Vehicle 1</td><td colspan="2">Vehicle 2</td></tr>
<tr><td rowspan="2">Multiple Vehicle
Max Speed</td><td rowspan="2">Subsystem 1</td><td colspan="2">Vehicle 1</td><td colspan="2"></td></tr>
<tr><td colspan="2">Vehicle 2</td><td colspan="2"></td></tr>
<tr><td rowspan="2">Multiple Vehicle
Not Max</td><td rowspan="2">Subsystem 1</td><td colspan="2">Vehicle 1</td><td colspan="2"></td></tr>
<tr><td colspan="3">Vehicle 2</td><td></td></tr>
</table>

For Subsystem 1 shown notionally above, maintenance at maximum speed requires 2 days for a single vehicle. When allotted enough technicians for single-vehicle maintenance, the total amount of time required for 2 vehicles to complete maintenance on Subsystem 1 is 4 days, as shown in the 'Single Vehicle' row. However; if the subsystem was allotted enough technicians to perform multiple-vehicle maintenance at maximum speed, then maintenance on that subsystem is completed for all vehicles in only 2 days, as shown in the 'Multiple Vehicle Max Speed' row. Furthermore, if the subsystem is allotted more than enough technicians for work on one subsystem, but not enough to perform maintenance on subsequent vehicles at maximum speed, the vehicles which are worked on by the reduced leftover workforce will be completed later than those with maximum technicians applied. In this case, Subsystem 1 has been maintained across all vehicles after the maximum amount of time has elapsed, in this notional case 3 days into the year. Once maintenance on all subsystems have been addressed in this fashion, the subsystem with the longest required maintenance is that which drives the flight rate, and so the subsystem with the maximum required hours is selected, and divided by the total available hours per year (8 hours per day * 5 days per week * 52 weeks per year = 2080 available hours) to calculate a flight rate associated with that maintenance distribution.

To generate inputs for the Monte Carlo model, each of the 16 subsystems are on each run allotted an integer number of available technicians from 1 to 15x the number of vehicles so as to allow for multiple vehicle maintenance. After a randomly generated distribution is analyzed, its sum and flight rate are recorded. Initially, the best flight rate possible is set to 0 and the workforce distribution sum is set to the absolute maximum (16*15*number of vehicles). If on a certain run the randomly generated workforce distribution is able to achieve a higher flight rate than the current maximum with a lower workforce distribution sum, it is recorded as the new best. The simulation code then repeats the analysis until 1,000,000 runs have completed since the last best distribution has been found.

To characterize the results coming out of the Monte Carlo model, several representative cases are shown below. From the general formulation of the scenario this method is attempting to capture, it is expected that a randomly generated optimal case would allot fewer technicians to lower-time subsystems like Crew and Propellant Management, while allotting more to higher-time subsystems like Electrical & Wiring and Engines. Over 10 completed runs, the optimal settings which the Monte Carlo simulation has found without any variation in the amount of required time for a subsystem are shown below in Table 3.

As stated earlier, Monte Carlo methods can also incorporate variation, which can be simply incorporated into the simulation already constructed by allowing the maintenance required for maintenance on a subsystem to vary within a certain range. The range chosen to exhibit optimization on a stochastic maintenance system is +10%. On each repetition of the simulation, the time for each subsystem is probabilistically calculated using the base values established previously, then adding up to 10% more time to the total. Over 10 completed runs, the optimal settings which the Monte Carlo simulation has found including variation are shown below in Table 4.

Table 3: Monte Carlo model sample runs

	Sample Runs											
	1	2	3	4	5	6	7	8	9	10	μ	σ
Avionics	6	6	5	8	7	3	8	6	5	14	7	3
Communications	20	14	20	23	7	14	10	5	11	22	15	6
Crew	4	6	2	20	3	8	12	7	13	5	8	5
Electrical & Wiring	12	12	9	13	9	12	24	7	15	10	12	4
Engines	12	7	7	20	15	12	27	9	11	5	13	6
Environmental	30	10	9	14	13	10	13	7	19	11	14	6
Flight Controls	9	12	20	14	26	6	25	6	23	3	14	8
Hydraulics	15	15	6	12	7	5	9	25	11	7	11	6
Landing & Recovery	8	6	6	15	6	8	8	10	13	7	9	3
Navigation	13	9	5	12	11	8	10	7	9	12	10	2
Pneumatic	14	24	10	12	9	18	21	25	10	7	15	6
Propellant Management	5	4	3	17	13	2	10	17	7	2	8	6
Software	21	13	5	10	5	5	9	6	9	3	9	5
Structures	6	9	7	9	8	10	20	4	21	5	10	6
TPS	5	6	8	18	24	9	7	3	7	3	9	6
Tracking	7	21	9	9	8	12	7	6	7	12	10	4
Workforce Sum	187	174	131	226	171	142	220	150	191	128	172	33
Flight Rate	4.7	4	3.7	5.8	3.9	2.9	5.6	2.9	5.4	2	4	1
Found on run (x10,000)	96	16	105	2.3	21	20	20	109	130	37		
OEC	3.01	3.14	4.02	2.68	3.18	3.66	2.72	3.48	3.03	3.94		

In the above table, the randomly generated workforce skillset distribution for a run is shown in the top 16 rows. The average and standard deviation of each subsystem technician availability is shown in the rightmost columns. The bottom 4 rows of numbers in the above table show the sum of technicians, the flight rate the above distribution resulted in, and on what iteration of the code the 'optimal' distribution shown was found. Finally, at the bottom is a row evaluating the distribution according to its performance as compared with the baseline study performed in DES. These numbers will be important later when Monte Carlo and LP are compared with the results coming from DES.

Table 4: Monte Carlo model sample runs with variation

	Sample Runs with Variation											
	1	2	3	4	5	6	7	8	9	10	μ	σ
Avionics	7	4	5	18	8	6	18	5	10	12	9	5
Communications	9	13	12	6	12	7	10	13	9	15	11	3
Crew	8	16	12	4	4	5	2	8	23	11	9	6
Electrical & Wiring	12	24	13	8	13	12	10	9	14	14	13	4
Engines	9	11	27	6	7	11	26	4	8	14	12	8
Environmental	9	4	6	9	6	6	5	5	6	9	7	2
Flight Controls	11	9	9	12	11	13	20	9	19	10	12	4
Hydraulics	11	6	21	9	9	19	6	15	7	13	12	5
Landing & Recovery	11	5	8	6	5	23	7	3	10	19	10	6
Navigation	18	9	8	12	10	6	4	5	4	8	8	4
Pneumatic	8	8	10	7	6	7	9	6	7	11	8	2
Propellant Management	4	12	7	4	19	14	3	10	6	7	9	5
Software	15	14	8	6	8	9	4	3	10	23	10	6
Structures	11	7	19	30	19	14	9	7	12	26	15	8
TPS	5	4	9	4	4	6	3	8	11	8	6	3
Tracking	20	10	18	5	10	14	5	11	4	9	11	5
Workforce Sum	168	156	192	146	151	172	141	121	160	209	162	24
Flight Rate	4.5	3.5	5.1	3.2	3.6	4.2	2.8	2.2	3.3	5.9	4	1
Found on run (x10,000)	27	153	150	20	10	45	41	75	23	46		
OEC	3.29	3.41	2.99	3.59	3.52	3.19	3.67	4.18	3.31	2.86		

Conclusions

Although the Monte Carlo method is limited in the sense that it uses random input to find a result, which does not guarantee finding optimal settings for a given evaluation criterion in a stochastic system, it is very useful in performing a large number of experiments quickly. It is more important than ever to walk the line between abstraction and fidelity, as with every bit of information added into the simulation computational time is increased. Overall, the Monte Carlo method is very useful for getting an idea of how inputs relate to outputs in a complex system, however without extra supporting codes or nearly limitless computational time it is ultimately inefficient at arriving at optimal values. By random chance it may however land upon a workforce skillset distribution which can be found by more involved methods, and so the sample runs with the highest OEC values in the tables above will be considered alongside the results coming out of simulation.

Discrete Event Simulation

The final method considered is Discrete Event Simulation (DES), which has been the standard tool for evaluating operational scenarios for many years [5] because of its formulation as a sequential series of interconnected events. A properly constructed DES model can capture very complicated and time-dependent relationships because of this formulation, allowing it to capture the effects of future technologies and hybrid launch systems which have had very little physical use so far [7]. In fact, the use of DES to model Shuttle operations began as early as 1970, before the Shuttle program was even approved for development [47].

In DES, the phenomena of interest change value or state at discrete points in time. Similar to network models [56], DES is primarily composed of actions (nodes) and flow paths (arcs) which represent the behavior of a complicated system. Unlike a network model, DES specifically represents the operation of the simulated system as a chronological series of events. One of DES' attractive features is the fact that during simulation, the only points of time analyzed are those in which a discrete action or flow is taking place. The fundamental assumption here is that although time is continuous, only a finite number of events can occur in a given period [59].

Each event that occurs causes the simulation to move from one system state to another, each of which is a collection of variables necessary to describe the system at a particular time [27]. Any one state would hold values for the amount of MMH spent on each subsystem of each vehicle both in total and since the last flight took place, the number of vehicles and maintenance sites, the mission status of vehicles, and so on. In particular if the only state variable considered was which maintenance site was working on which vehicle's subsystem, the number of possible states assuming only true or false for values would be $\left[\left(2^{\#Vehicles}\right)^{\#Subsystems}\right]^{\#Maintenance\ Sites}$, which with only 2 vehicles each with 16 subsystems and only 1 maintenance site, is close to 4.3 billion possible states. Therefore the DES methodology used should not be state-based.

There are seven basic concepts DES embodies which are of particular use for RLV simulation: work, resources, routing, buffers, scheduling, sequencing, and performance [28]. Work denotes the entities moving through the system, such as customers arriving at a business or RLVs requiring maintenance. Resources are those entities which can provide services to the work, such as the site which takes in a RLV for maintenance. With every batch of work there is a route by which the work gathers the resources required for completion, and the order in which the work will be done. Buffers are those entities which hold work while required and in-use resources complete their current work. These buffers may have an infinite or finite capacity, the latter requiring specific rules for their behavior when full. Scheduling takes the concept of a buffer and relates it to the real-world time the simulation is emulating, usually consisting of times at which resources become available. Finally, sequencing contains information dictating the order in which resources handle work waiting in buffers. This may be a first-come-first-served order, or a hierarchical process, such as a RLV which must be launched sooner than others waiting in a queue for maintenance.

When considered together, the concepts important to this work: time delay, number waiting, resource utilization, and entity throughput comprise a queuing model [56], [19][27], which may be either closed or open-loop. In an open-loop system, work arrives from outside the system at a rate independent of the state of the system. However, when one has control over work arrival times it is a closed-loop system. In the case of RLV O&M, the queuing model is of the closed-loop variety, as the arrival of RLVs either from mission completion or manufacture is known.

There are two major forms of DES: the process-interaction and event-scheduling approaches [56]. The process-interaction approach provides a process for each entity in a system, essentially including the passage of time occurring during a process. Instead of focusing on the times events are started and finished, the process-interaction approach places more emphasis on the role of queue formation when resources are in use. The event-scheduling approach on the other hand comes from the fact that every discrete-event system has a collection of state variables that change values as time elapses. Each time one of these values changes it is called an event. It is primarily concerned with the timing of events and how they interact with one another through the use of a queuing system. The event-scheduling approach has several intrinsic properties. The first is that an event is executed if and only if a state change occurs, such as a vehicle being launched only after it has completed maintenance on each of its subsystems. The second is that simulated time remains constant while an event is being executed, which is especially useful for handling multiple vehicles and maintenance sites. Using this property, all events that would happen at a particular time in simulation can be handled sequentially in practice, while all still happening simultaneously from the perspective of simulation time. Third is the concept of a compound event, which is a recipe for executing a sequence of actions all at the same time such as gathering completion times of a subsystem's tasks. Fourth, DES consists of a sequence of instances of compound events of possibly different types ordered according to their scheduled execution times, which may be randomly generated. This property allows for each of the compound events which describe how work (vehicles) will utilize resources (maintenance sites and technicians) to interact in a meaningful manner.

Altogether, the event-scheduling approach lends itself best to the construction of an effective schedule for a simulated system in which the interactions between agents are considered to be instantaneous, which works well for RLV O&M simulation as the 'schedule' can take the form of technician availability. If there are only enough technicians allotted to a particular subsystem's maintenance for one vehicle, then the site will operate off of a ‘first-come, first-served’ basis. However, if there are more than enough for one vehicle (up to enough for several vehicles) then the allotment (schedule) could be described as ‘first-come, first-served-best’, due to the relationship between technician availability and task time described previously.

Software packages

Since DES has been in wide use for important operations problems, several computer programs have been built specifically to capture its architecture. Some available packages are: Arena from Rockwell Automation [44], Vensim from Ventana Systems, Inc. [57], and the open-source SimPy [51], which is a collection of libraries written in python. Arena is widely in use presently for the construction of RLV O&M models [43],[39], however it requires purchase. Vensim is also in use for DES, and while setting up a simulation is fairly easy, personal experiences with the program have been plagued with difficulties retrieving important information back out of the program. In addition, the license previously accessible has lapsed. Finally, SimPy is attractive because of its $0 price tag, however part of this line of research has been to personally understand all the underlying processes happening during DES.

The primary motivations for beginning this line of research & development is to increase the effectiveness of preliminary design by incorporating operations concerns into the design of an aerospace system as early as possible, with the most information possible. As outlined in the introduction, the construction of such a tool would be capable of optimizing on both the feasibility and long-term viability of a system before detailed design considerations came into play. In particular, the Aerospace Systems Design Laboratory (ASDL) at Georgia Tech has begun construction of an integrated RLV sizing

& synthesis (S&S) code with the potential of backing from the Air Force Research Laboratory (AFRL) with this very intent. Other portions of the S&S code include aerodynamics, propulsion, and other primary concerns for the feasibility of a RLV.

The tool constructed as part of this research has been coded in the Python programming language in order to facilitate integration using an open-source integration framework called Open Multi-Disciplinary Analysis & Optimization (openMDAO), and for that reason the tool which is constructed utilizing the lessons learned from a literature review is also be coded in Python. The reasoning for using openMDAO comes from the want of general accessibility. Open-source software is by its definition able to be run and modified by any user who acquires it, which is an attractive feature for programmers and end-users alike. The only expenditure associated with such a tool is in the acquisition of the code itself, without requiring special costly operating software such as ModelCenter. In addition due to uncertainty in the final requirements for the tool, the tool built must be scale-able in order to facilitate the inclusion of multiple vehicles and/or maintenance sites with varying characteristics. Because of these concerns, the tool built must be highly customizable, and so it has been built from the ground up, requiring a fair bit of development and debugging time.

After reviewing the methods of LP, Monte Carlo, and DES, and considering the long historical precedent of using DES for modeling RLV O&M, it has been selected for final consideration.

Answering Research Question 1: DES is the proper method for capturing RLV O&M, both in its power and by the precedent of its use.

Using the DES framework, a simulation of RLV O&M has been made, as will be expanded upon in the next section. Once the experimental frame is constructed, interesting O&M scheme combinations can be explored, which will ultimately lead to answering the second research question.

CHAPTER 4

WORKFORCE OPTIMIZATION EXPERIMENTS

The overarching design of experiments (DOE) code begins by loading two Excel sheets of run inputs. The first sheet loaded contains information pertinent to the case study being performed. In particular, the case study on workforce utilization loads in, for each of the 16 subsystems included, the maximum number of technicians available and the maximum number of technicians that can work on a single vehicle's subsystem at a time. The second document loaded is for general inputs which are

- Campaign years to simulate, defaulted to 20
- Maximum launches per year, defaulted to 12
- *Vehicle characteristics*
    - Initialization (Spawn) time, default is 1
    - Mission type
        - Satellite deployment
        - ISS resupply
        - Lunar mission
        - Extended on-orbit
    - Vehicle identification number, starting with 0
- *Maintenance Site Characteristics*
    - Initialization (Spawn) time, default is 1
    - Technicians available
    - Distance to integration center in miles

- Road Rating of 1-5
- Specialty which can be 'None', one, or multiple subsystems
- Close time, default is 0

The number of technicians available for each subsystem gathered from the case study spreadsheet are written to the corresponding maintenance site entries in the inputs spreadsheet. The maximum number of technicians which can work on a particular subsystem at once are also written to the corresponding spreadsheet containing pertinent data on that subsystem. Information contained within each subsystem spreadsheet:

- Name of subsystem
- Maximum number of technicians which can work at once
- List of tasks to be performed
- Descriptions of each task to be performed
- Mean and variance of the log-Normal distribution applied to the time required for each task

The tasks allotted to each subsystem come from the FAA’s Guide to Commercial Reusable Launch Vehicle Operations and Maintenance [26]. Each is set to a 1, 2, or 3-day setting according to best guess according to the description of the task and its associated complexity. The settings for each log-Normal distribution follow in Table 5.

Table 5: Task time log-normal distribution settings

	Mean	**Variance**		
1-Day	2.15	0.2	→	8 hours average
2-Day	2.8	0.1	→	16 hours average
3-Day	3.3	0.1	→	24 hours average

Once all the inputs for a single run have been set, the operations simulation itself is run. The simulation begins by defining class variables for instantiating vehicles and maintenance sites. For a vehicle, the class variables are:

- Total number of technicians currently in use
- System Check variable which tracks each subsystem, whether each have been cleared for the next launch, and how many hours have been spent on this cycle for maintenance. This is defaulted to each system being cleared for launch at initiation or True
- Variables for each subsystem simulated tracking the total number of maintenance hours applied
- TPS_Materials variable for potentially applying altering maintenance practices for different materials
- Time_Initialized which controls when the vehicle will be instantiated
- Default_Mission which controls how long missions will last
- Return_Time variable which tracks when the vehicle will return from its latest mission
- Num_Flights, the total number of flights the vehicle has performed
- TotalMMH, the total maintenance man hours from all subsystems

- VID, the vehicle identification number
- Maint_Site, a Maintenance Site variable which tracks which maintenance sites are working on which subsystem of that vehicle

For a maintenance site, the class variables are:

- Available, a boolean. Starting value is True so that it can begin work immediately.
- Techs, the maximum number of technicians available
- Techs In Use, which tracks the current utilization of the maintenance site
- WorkingOn, an array which keeps track of all the vehicles a maintenance site is working on, the subsystem for that vehicle it is working on, when work will be completed, and how many technicians are in use for the work
- Int_Dist, the distance from the maintenance site to the integration center
- Road_Rating, a 1-5 integer which along with Int_Dist dictates how much time is required for transportation to and from the integration center to the maintenance site
  - 1 assumes interstate travel, averaging 70 mph
  - 3 assumes highway travel, averaging 50 mph
  - 5 assumes country/urban travel, averaging 30 mph
- TimeInUse, the total number of maintenance man-hours this site has performed
- Specialty, which dictates which subsystems the site can work on. This can be 'None' so that the site can work on any subsystem, or any number of subsystems

Once the classes are set up, a code-writing module is run. This module takes in the input data set by the DOE code into the Inputs spreadsheet and translates it into python code which can be used by the simulation code. The code-writing module begins by sequentially loading each subsystem spreadsheet, and recording the maximum number of technicians which can work on that subsystem. Further detail on the subsystems module will follow shortly, for now the only information needed is that the name of the subsystem, along with the task names, descriptions, and statistical moments are taken from each subsystem spreadsheet and written to code. For in-depth descriptions of each subsystem and its tasks please refer to Appendix A. Once the subsystems module has been written, then vehicle and maintenance site input characteristics, number of years to simulate, and maximum number of launches per year are read from the Inputs spreadsheet and similarly written to an inputs module to be accessed by the simulation code.

Once the code-writing module has completed, the simulation loads the campaign years from the freshly written inputs module. The number of years dictates the total time of simulation, by assuming one 8-hour shift, 5 days per week, over the 52 weeks of a single year, which has been used in other analyses [6] basing their operational assumptions on Shuttle practices. The input characteristics of vehicles and maintenance sites are also read in and kept in separate arrays, MaintSite_Set and Vehicle_Set. Set here by the user is the frequency by which the simulation state will be recorded by a DataCollect module is set. Next, the DataCollect module is run to prepare an output file for the run. Preparing the output file does the tasks of creating a spreadsheet for housing the data for the run in progress, and records the input data from the two arrays of input characteristics to an inputs sheet.

Once preparation of the output file has completed, arrays to hold instantiated vehicle and maintenance site objects are initialized, aptly named 'Vehicles' and 'MaintSites'. At this point, a loop is started which will keep the remaining logic looping while the current time of simulation is less than the total time of simulation. An overall flowchart for what follow is shown below in Figure 6.

At the beginning of every loop, the input characteristic arrays (MaintSite_Set, Vechicle_Set) are checked to see if any element within them is to be spawned at that timestep. The default start time is 1, so any vehicles or maintenance sites which have a default start time will begin directly at the beginning of simulation simultaneously.

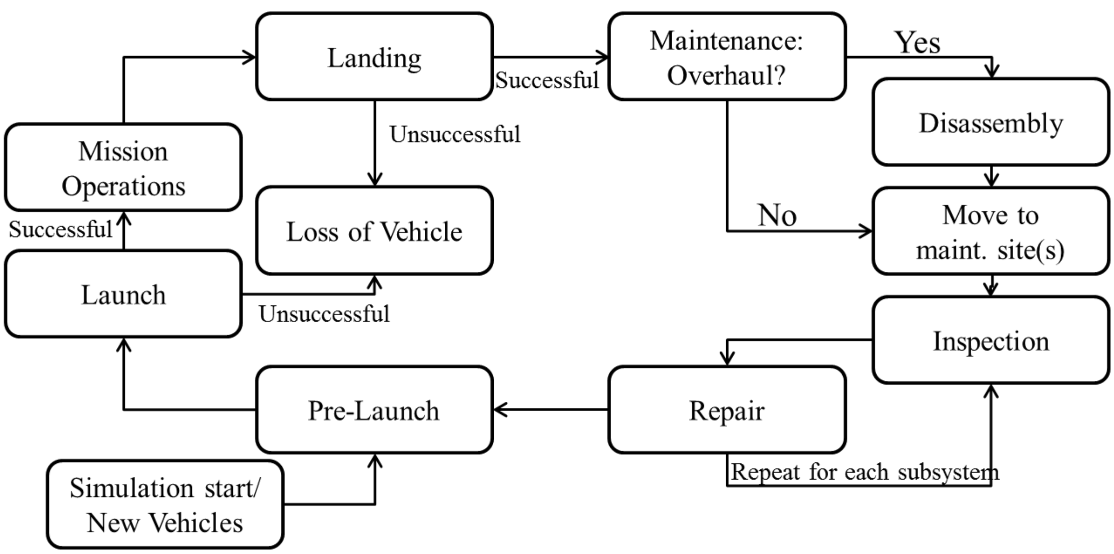

Figure 6: Simulation Events Flowchart

Next, all maintenance site resources are examined to find if the current time step is when work will be completed, thus requiring that resource to be released. If this condition is met, the maintenance site variables affected are:

- TechsInUse has the number of technicians which have just finished work (stored within the WorkingOn variable) subtracted from it
- Available is set to True
- The entry in the WorkingOn variable that was tracking this particular job is removed from the array

Vehicle characteristics affected are:

- Techs_In_Use variable has the number of technicians which have just finished work (stored within the MaintSite WorkingOn variable) subtracted from it
- The System_Check entry for the subsystem tracking completed maintenance is set to True
- Maint_Site entry tracking the work completed is removed from the array

Next, all vehicles are cycled through to determine what actions are required for them. The first check is to see whether any vehicles are returning from mission during this timestep. If the Return_Time variable matches the current time, then the vehicle lands by calling a 'Landing' function. In order to call this function however, the vehicle must exceed a 1/235 chance of breaking up during re-entry, which is based on STS history. If the vehicle does not pass this random chance, it is 'destroyed' by removing it from simulation. If the vehicle does pass the random chance, the Landing function cycles through the Vehicle's System_Check variable, setting each of the subsystems to 'False' -

meaning that maintenance is now required on each. Each entry in System_Check which has been tracking the MMH applied since the last launch are added to the aggregate variable for each subsystem, and then zeroed. Finally, the Return_Time variable is set to zero, meaning that it current has no return time as it is grounded. Other than these effects, the vehicles are essentially memoryless. Individual subsystems do not degrade over time and repeated launches.

After considering landing vehicles, vehicles requiring maintenance are examined. For a single vehicle, the System_Check variable is cycled through to first find whether it is currently undergoing work by searching through every maintenance site's WorkingOn variable to find if any maintenance site is specifically working on this vehicle, and on this particular vehicle's particular subsystem. If an entry is found, then the subsystem is passed over. During this operation, the availability of each maintenance site is also recorded so that further looping can be skipped if all maintenance sites are completely in use. Next, logic calls check to be certain that there are maintenance sites available, the subsystem in question is not being worked on by any of them, and that the subsystem has not already had work completed. Assuming all of these conditions are met, then the Subsystems module is called specifically for the subsystem under consideration.

The Subsystems module, written at the beginning of simulation code execution, dictates how the maintenance tasks required by vehicles and maintenance sites performing the tasks interact with one another. Each subsystem represented on a vehicle is a function call within the Subsystems module, each sharing many similarities. First, for an individual subsystem, the maximum number of people which can work on the subsystem is set. Next, the function searches through the total list of maintenance sites first to find an available site, then checks to see if an available site can work on the subsystem currently requiring work by referencing that maintenance site's Specialty variable. If a maintenance site meets both requirements, its number of technicians in use is examined to determine if it is already at capacity. If it is, then that site's availability is set to False. If it isn't, then calculation can proceed.

First, the transportation time to and from the maintenance site is calculated by referencing the distance from the integration center (Int_Dist) and the Road_Rating variables. Next, each of the tasks associated with the particular subsystem are assigned a number of required hours by randomly pulling from a log-Normal distribution whose moments are determined by each Subsystem workbook referenced in the code-writing phase. Once the tasks have been calculated, the code looks to see if there is an expansion module present for that subsystem. The potential for expansion modules allows for increasing the fidelity of any subsystem's maintenance work easily and without requiring any changes to the existing code. Without an expansion module, nothing happens here. Next, the number of technicians currently available is referenced again, in this case to compare with the maximum number of technicians which can work on the subsystem at a time. If the technicians available variable is equal to or greater than the maximum, nothing happens. However, if there are fewer available, then a reciprocal dependency on the amount of time required for subsystem work completion is applied. The dependency assumes that fewer technicians available will cause the time required to rise very rapidly, following the reciprocal dependency in Equation 9.

Once the total amount of hours has been set with this condition, maintenance site variables affected are:

- TechsInUse has the number of technicians beginning work added
- WorkingOn array has an entry added which contains the vehicle's identification number, the subsystem it is beginning work on, the timestep work will be completed, and the number of technicians doing the work
- TimeInUse variable has the maintenance man hours required added

Vehicle variables affected are:

- Techs_In_Use variable has the number of technicians beginning work added
- Maint_Site array has an entry added containing the maintenance site's identification number and the subsystem it is doing work on
- The System_Check entry for the subsystem under consideration and TotalMMH variables have the total maintenance man hours added to them

Once the function call is completed, the Subsystems module returns back to the simulation code.

Once all possible maintenance actions have taken place, the vehicle is checked to see if it is ready for launch. It should be mentioned here that when a vehicle is first instantiated, its System_Check variable is defaulted to all True, so new vehicles will always skip through the code to this point, meaning that new vehicles go directly to launch. For all others, a vehicle's System_Check variable is examined for any entries which are False, indicating that maintenance work has not completed on that subsystem and so it is not cleared for launch. Concurrently, each subsystem is checked against all maintenance sites WorkingOn variable to ensure that maintenance is not ongoing somewhere (It certainly would not do to launch a RLV without its avionics system).

Assuming that both these conditions are met, the vehicle is ready for launch. Just like landing however, there is a 1/235 chance (according to STS history) that LOV will occur at launch. If the vehicle does not pass this chance, it is removed from simulation. Assuming it does pass, the Launch function is called to handle mission operations. The Launch function references the vehicle's Default_Mission variable to determine how many work hours will go by while the vehicle is performing its mission. Each of these numbers comes straight from NASA data and averages. For the Satellite mission, which is a 5-day mission, 40 work hours will go by. For an ISS mission, a 10-day mission, 80 hours will go by. Lunar missions were historically done over 12 days, resulting in 96 hours. Extended stay missions can last up to 15 days, resulting in 120 hours. Once this referencing has completed, the vehicle's Return_Time variable is set by adding its on-mission time to the current timestep, and the vehicles num_flights variable is increased by 1.

After Launch is called and returns, the DataCollect module is called once again to record the state of simulation so that it is recorded at every launch along with it scheduled recordings. At this time, the workbook created at beginning of simulation is loaded, and the current state of simulation is recorded onto the next empty line of the workbook, split between sheets which contain data on each separate vehicle and maintenance site. Every variable within the class definitions is recorded on these sheets, along with the timestep it represents. Once writing is completed, the workbook is saved and simulation continues.

At this point, all previous discussion relating to vehicle activities will loop so that all vehicles will have all potential actions handles simultaneously in simulation time.

Finally, the simulation chooses which timestep to advance to by creating an array Important_Times and adding all spawn times, work completion times, and vehicle return times which are greater than the current timestep. Next, if the minimum of these entries is greater than the timestep during which scheduled simulation state recording is to take place, the current time is changed to the timestep for recording. If not, then the current timestep CurrentTime is changed to the minimum of the Important_Times array, and then Important_Times is emptied. Once this action is completed, the loop starts back at the beginning with the new timestep. In the case of the timestep being the next recording, CurrentTime will not match with any other activities and so the only action which will happen is state recording and then the next simulation event will take precedence. Once the simulation has exhausted its set number of campaign years, the simulation state at completion is also recorded, and the operations simulation is completed.

Once one run of the simulation is completed, the DOE code once again takes precedence for recording of aggregate results for many runs of the simulation. First, the inputs module used by the last run of simulation is loaded for both the number and characteristics of vehicles and maintenance sites. Next, the run results workbook generated by the last run and an aggregate results workbook (if it exists, if not, create it) are loaded for referencing and writing, respectively. For each vehicle in simulation, the total number of flights, MMH for each subsystem, average MMH per launch of each subsystem, and total MMH are recorded. For each maintenance site, the total MMH performed is recorded. At this point, for memory conservation, the previous run's results file is deleted along with any compiled versions of the Inputs, Subsystems, and OperationsModel codes so that the next run can have different inputs. Finally, the entire process repeats using the next row of inputs from the DOE spreadsheet until all runs have completed and the aggregate results workbook is completed.

Definition of Scenario Experiments

To find the optimal technician skillset distribution, the most complete method would be to construct a suitably high-fidelity simulation and conduct a full-factorial exploration. However; when simulating maintenance on 16 subsystems, each having a number of levels equal to 15x the number of vehicles, this method is computationally inefficient. In place of investigating the effect of each individual subsystem's associated workforce individually varying, design of experiments (DOE) methods are employed. Using DOE, the maximum amount of information from a combination of experimental variables can be obtained in a reduced amount of runs by applying statistical formulae to the selection of experimental variable values [23]. In its application, DOE drastically cuts down the number of experimental runs from full-factorial to a much more computationally and information efficient set of runs. In particular, for the construction of DOE the statistical software JMP [33] is used to generate tables of values to run sequentially through simulation.

Even by utilizing DOE the need to uncover gross trends in the metrics of interest and then find specific optimal values in those trends would still requires extensive testing. Therefore, the grid-search method will be used to iteratively zero in the technician availability levels that maximize the OEC. Grid search, depicted below in Figure 7, is a method for cutting down on the number of runs necessary for a DOE investigation to arrive at optimal settings by iteratively reducing the ranges through which experimental variables vary [9]. Grid search was chosen not only for reducing the number of required runs to find optimality, but also because it tends to perform better than more complicated methods like genetic algorithms, with lower computational and set-up time cost [14]. In the first round of a grid search, representative values for each experimental variable are chosen which span the entire design space. To explore the O&M schemes discussed previously, the levels chosen are the maximum for performing maintenance on one (15 technicians) and two vehicles (2x15=30 technicians), along with maintenance at a less efficient pace on one vehicle (7 technicians), performing maintenance efficiently on one vehicle and less efficiently on a second (22 technicians), and finally having more technicians than are usable for maintaining two vehicles (35 technicians), for a total of five levels for each variable. Choosing the values as such allows the simultaneous exploration of single and multiple vehicle maintenance for each subsystem while also allowing for workforce/efficiency trades.

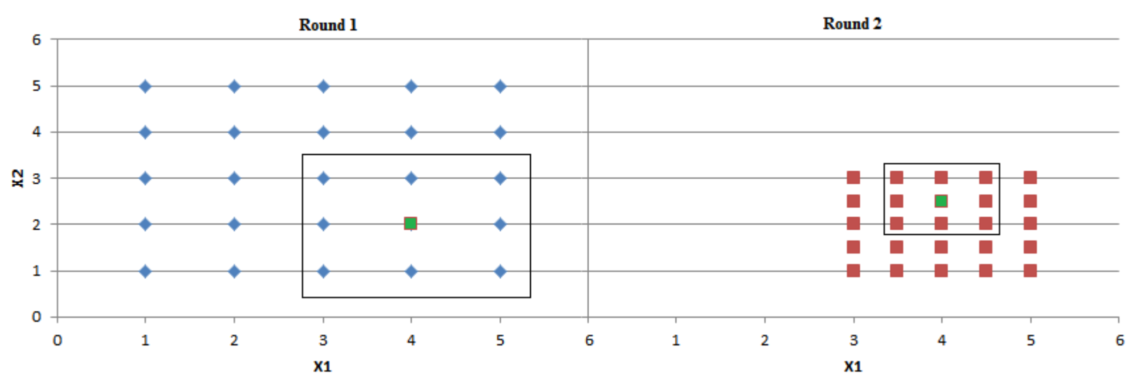

Figure 7: Grid search method depiction

After each round of grid search, the ranges for each experimental variable are reduced, and another DOE guided set of experiments take place within the reduced frame. In Figure 7 above, on the left is a representation of round 1 of a grid search method, and on the right is round 2. In the first round, the experiments of the design space of variables X1 & X2 each between values of one and five are performed. Once completed, the optimal settings for X1 and X2 are found by processing the outputs from experimentation. In the figure above, these settings would be X1 = 4, X2 = 2. In the second round, the experimental design space is contracted to those values around the optimal values in round 1. While compressed, the set of values still has the same number of points to explore, uncovering the optimal value of X2 = 2.5 which was not part of the original set of experiments. It is in this manner that each round of grid search closes in on more optimal values [1], by compressing the design space and including values not explored by previous rounds.

In subsequent rounds of grid-search optimization two methods for choosing the reduced set for the next round of experimentation will be explored and compared. The first attempts to optimize on each subsystem individually to achieve an overall optimum, assuming that fleet operations will be optimized in flight rate and MMH/flight if each comprising maintenance component has been optimized. For selection of optimality, the OEC defined previously in Equation 4 will be applied to each subsystem to find its optimal value via the maximum found from experimental runs. In order to characterize the performance of the system at those settings, an additional set of 100 runs are completed at the 'optimal' settings to generate μ & σ for the fleet flight rate and MMH/Flight/Vehicle metrics. The other points in the reduced set will range from the technician availability levels just below and above the optimal setting, with two settings added between the new maximum and minimum and the optimal central value as is depicted in Figure 7.

The second method attempts to optimize fleet operations by optimizing on the entire maintenance skillset distribution simultaneously by applying the OEC across each subsystem to find the distribution which achieves the greatest efficiency in flight rate and MMH/flight with the smallest total workforce. Once gathered, the average (μ) and standard deviation (σ) of the top 10% distributions are calculated. The average μ for each subsystem becomes the central point for the next round of experimentation, with $\mu \pm \sigma$ as the maximum and minimum values, and $\mu \pm \frac{1}{2}\sigma$ included to explore more settings. In both methods, as the number of available technicians is an integer, each optimized value is rounded to the nearest whole number. This restriction allows for a sharp cut-off of optimization rounds. If at some iteration the ranges for the next round are less than 1, then the optimal values have been found.

For each round of optimization and for each optimization method 1,500 runs are performed using a 'Custom Design' DOE from JMP.

Baseline Study

A crucial step for any analysis project is to first establish a baseline for comparison with final results. The baseline study consists of 2 identical vehicles with identical mission profiles (ISS rendezvous), and 1 maintenance site which is at the same location as the launch & landing site. Each run represents a 20 year campaign. Maintenance on each subsystem is assumed to require unique skills so that there is no overlap in maintenance technician utilization. Maintenance on each subsystem is allotted

30 technicians, and 15 is set as the maximum number of technicians which can work on a particular subsystem at a time. These settings are such that maintenance can always proceed at full speed no matter whether another vehicle is already undergoing maintenance.

Table 6: Baseline study technician availability levels

Avionics	30	**Landing & Recovery**	30
Communications	30	**Navigation**	30
Crew	30	**Pneumatic**	30
Electrical & Wiring	30	**Propellant Management**	30
Engines	30	**Software**	30
Environmental	30	**Structures**	30
Flight Controls	30	**TPS**	30
Hydraulics	30	**Tracking**	30

The overall results of the baseline study, shown below in Table 7 indicate that without any backup or lack of technicians with any subsystem, roughly 10 flights per year will be performed, with roughly 1,450 MMH spent on each vehicle after each flight.

Table 7: Baseline Study Overall Results

	Flight Rate	**Vehicle1 MMH/flight**	**Vehicle2 MMH/flight**	**MMH/Flight/Veh.**	**Workforce**
μ	10.5	1448	1449	1449	480
σ	0.01	3	4	3	

The values above are those which the two competing methods for optimization will be attempting to improve on, by simultaneously keeping the flight rate and

MMH/flight/Vehicle as close as possible to the baseline values while reducing the necessary workforce. As comparison, the results from each round and each method will be compared to their percentage increases or decreases from the values above. So for a method to be considered better than the other, it must converge on a maintenance workforce distribution which comes as close to possible to 0% deviation from the baseline values with the minimum workforce required. Over the next few sections the progression of grid search trials pursuing both methods of optimization will be presented and their results compared with baseline values.

Common Basis of Optimization: Round 0

Both methods of optimization have the same first round, as was outlined in the previous section concerning grid search. The levels are presented below in Table 8. Now after establishing the common basis for optimization so that the methods have no advantage over one another, the competing methods will be presented and explored.

Table 8: Round 1 Grid Search technician availability levels

	Technicians Available				
Avionics	5	15	22	30	35
Communications	5	15	22	30	35
Crew	5	15	22	30	35
Electrical & Wiring	5	15	22	30	35
Engines	5	15	22	30	35
Environmental	5	15	22	30	35
Flight Controls	5	15	22	30	35
Hydraulics	5	15	22	30	35
Landing & Recovery	5	15	22	30	35
Navigation	5	15	22	30	35
Pneumatic	5	15	22	30	35
Propellant Management	5	15	22	30	35
Software	5	15	22	30	35
Structures	5	15	22	30	35
TPS	5	15	22	30	35
Tracking	5	15	22	30	35

Experiment 1: Optimize subsystems individually

Once the 1,500 runs were completed using the common settings of round 0, the results coming from simulation were analyzed using the subsystem overall evaluation criterion (SOEC) shown below in Equation 13. Once optimal values were found, the next round of grid search was begun using these values and reduced variable ranges.

$$SOEC = \frac{Flight\ Rate}{Max.Flight\ Rate} + \frac{Min.MMH\ per\ Flight}{MMH\ per\ Flight} + \frac{Max.Techs.}{\#\ Technicians}$$

Equation 13: Subsystem Overall Evaluation Criterion

Round 1 Results

The results from applying the SOEC to the first round DOE study yield optimal subsystem technician availability levels which drastically reduce the number of technicians, however with a corresponding hit to flight rate and MMH. The settings from round 1 SOEC optimization are shown below in Table 9.

Table 9: Round 1 SOEC Optimization Technician Availability Levels

Avionics	5	**Landing & Recovery**	5
Communications	5	**Navigation**	5
Crew	5	**Pneumatic**	5
Electrical & Wiring	15	**Propellant Management**	5
Engines	5	**Software**	5
Environmental	5	**Structures**	5
Flight Controls	5	**TPS**	5
Hydraulics	5	**Tracking**	5

By running a further 100 cases at these levels, the metrics presented in Table 10 below were gleaned. From the gross reduction of available technicians across all subsystems, the flight rate is almost halved, while the MMM/Flight/Vehicle is over 250% of baseline values.

Table 10: Round 1 SOEC Optimization results

	Fleet Flight Rate	**Vehicle0 MMH/flight**	**Vehicle1 MMH/flight**	**MMH/Flight/Vehicle**
μ	5.5	4022	4005	4014
σ	0.03	31	37	19

Finally, comparing the results of one round of optimization with the baseline, it can be seen from Table 11 below that although there is an 80% reduction in workforce, there is a large corresponding increase in the MMH/Flight/Vehicle in comparison with the baseline.

Table 11: Round 1 Optimization Comparison with Baseline

Flight Rate (%)	MMH/Flight/Veh. (%)	Workforce (%)
-47.62	177.02	-81.25

Iterated SOEC Grid Search Results

In the following figures the path of SOEC optimization is shown graphically. The axes are percentage comparisons between the results coming out of simulation and the baseline study values for flight rate and MMH/Flight/Vehicle. The 1st round of optimization results from applying the SOEC to Round 0 in order to find optimal settings for technician availability on each subsystem individually. It is clearly visible in Figure 8 that in the first round of optimization, the SOEC has produced wildly off-optimal results, yielding almost a 50% reduction in flight rate and 180% increase in MMH/Flight/Vehicle as compared with the baseline. In order to view the subsequent rounds more easily, Round 1 is removed from Figure 9, which shows the progression of SOEC optimization toward the origin (0% difference from the baseline in flight rate and MMH/Flight/Vehicle).

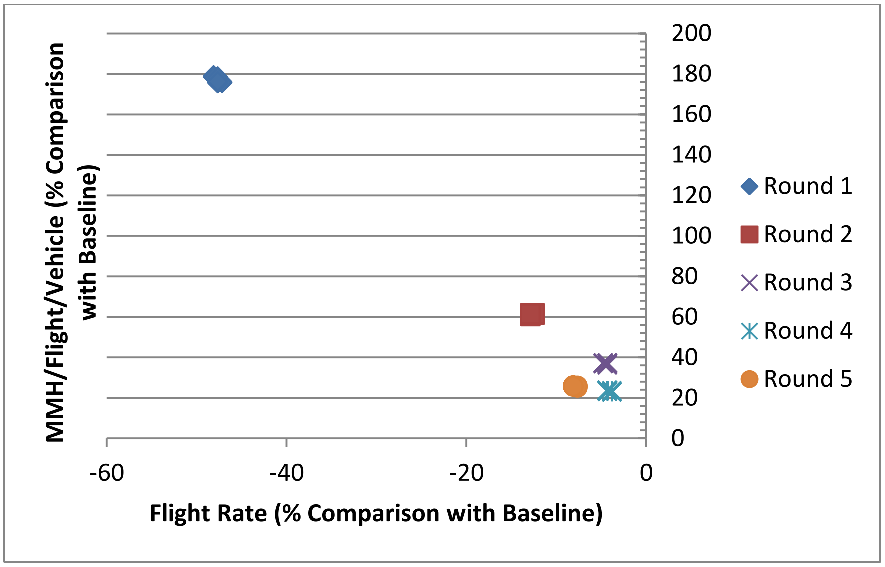

Figure 8: SOEC Round Results

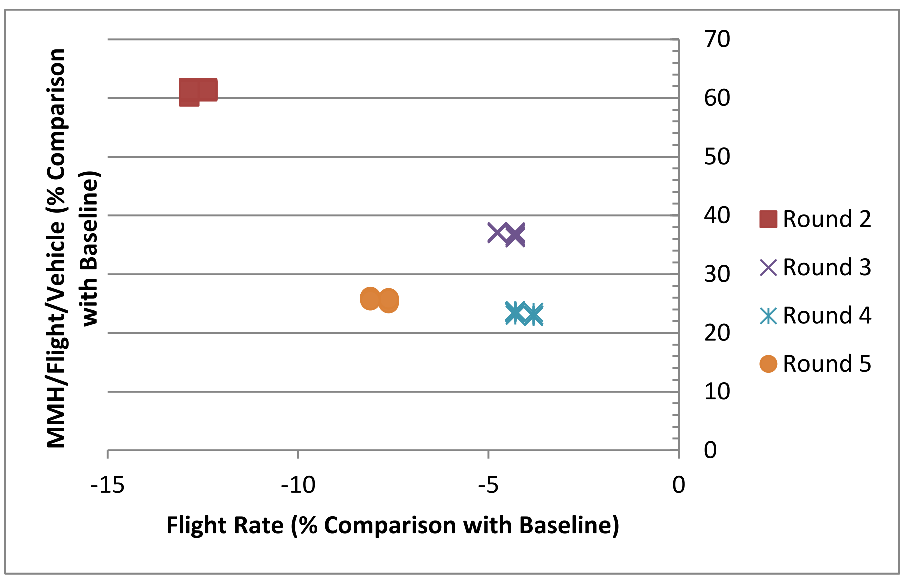

Figure 9: SOEC Round 2-5 Results

After 5 rounds the application of the SOEC finds optimal settings from assuming that each component subsystem's optimization will cause the optimization of fleet operations. Overall results for the iterative rounds of optimization are shown below in

Table 12. What is of interest for this version of optimization is that the numbers of technicians, shown in Table 13, are all at or below the number of technicians for work on only one vehicle. So the method of optimizing RLV O&M based upon the efficiencies of individual subsystem maintenance tends toward one vehicle-at-a-time maintenance, resulting here in a 60% reduction in workforce with an 8% reduction in flight rate and 26% increase in MMH/Flight/Vehicle.

An interesting characteristic of SOEC optimization is that due to working on each subsystem individually, the top 10% of cases for each subsystem never appear in concert using 1,500 runs. Another interesting feature of SOEC optimization is its tendency to rapidly work from a decidedly off-optimal skillset distribution (results from round 1) to optimal settings. This is shown pictorially in Figure 10 below, in the slope of % difference from baseline lines. This is of particular interest because of its effect on the second round of results. Due to the requirement that the skillset distribution must produce a minimum launch rate of 4 and the values used in the second round of SOEC optimization, < 2% of the DOE runs resulting from round 1 optimization meet this requirement and are discarded. It is from this result of SOEC optimization that the rapid increase of efficiency results.

Table 12: SOEC Optimization Round Results

		Round Results		Compare with Baseline		
		Flight Rate	**MMH/Flight/ Veh**	**Flight Rate**	**MMH/Flight/ Veh**	**Workforce**
1st Round	μ	5.5	4014	-47.62	177.02	-81.25
	σ	0.03	19			
2nd Round	μ	9.2	2338	-12.38	61.35	-69.17
	σ	0.02	4			
3rd Round	μ	10	1984	-4.76	36.92	-63.75
	σ	0.02	4			
4th Round	μ	10.1	1788	-3.81	23.4	-59.38
	σ	0.02	4			
5th Round	μ	9.7	1823	-7.62	25.81	-60
	σ	0.02	3			

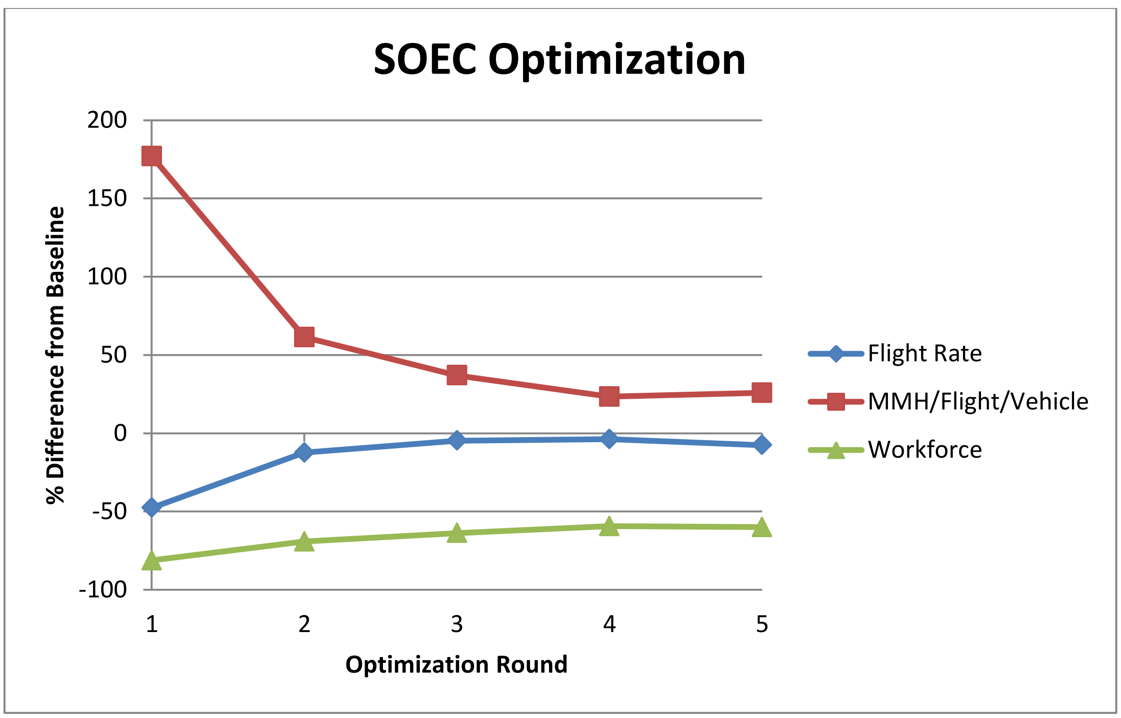

Figure 10: SOEC Optimization Evolution

As can be seen in Figure 10, round 1 is fairly off-optimal, as it has a very large difference from the baseline values of MMH/Flight/Vehicle and flight rate. Going from round 1 to round 2 however, due to the drastic reduction in cases to optimize on, only the most optimal are left which results in a large jump in similarity to the baseline. As the rounds continue, both these metrics get closer and closer to the baseline value much more slowly, ending within 8% of the original flight rate with a 60% reduction in workforce. The maintenance distribution which achieves this optimality is shown below in Table 13.

Table 13: Optimal Levels from SOEC Optimization

Avionics	12	**Landing & Recovery**	12
Communications	13	**Navigation**	13
Crew	13	**Pneumatic**	12
Electrical & Wiring	15	**Propellant Management**	10
Engines	10	**Software**	10
Environmental	11	**Structures**	11
Flight Controls	13	**TPS**	13
Hydraulics	14	**Tracking**	10

Via optimization with the SOEC, the optimal levels found have effectively traded off flight rate efficiency for a large reduction in required workforce. For further details on each round of SOEC optimization, consult Appendix B. By considering the effectiveness of the distribution as a whole new tradeoffs emerge, as will now be presented.

Experiment 2: Optimize subsystems simultaneously

As an alternative to optimizing on the performance of individual subsystems to find an optimum, the performance of a maintenance workforce could also be thought of as 'more than the sum of its parts.' This thinking inspired a second method of optimization, in which the entire distribution is optimized on simultaneously to achieve gross operational efficiency. In order to determine those distributions which are the most optimal, a gross overall evaluation criterion (GOEC) was applied to each distribution's simulated results, shown below in Equation 14.

$$GOEC = \frac{Flight\ Rate}{Max.Flight\ Rate} + \frac{Min.MMH\ per\ Flight}{MMH\ per\ Flight} + \frac{Sum(Max.Techs.)}{Sum(Subsys.Techs.)}$$

Equation 14: Gross Overall Evaluation Criterion

The big difference here is that the evaluation of each case run compares the total number of technicians with the baseline, rather than comparing on a subsystem-to-subsystem basis. This method similarly applies its optimization from the round 0 results, yielding the following results in round 1 of optimization.

Round 1 Results

After application of the GOEC to the results from round 0, the top 10% of performers were aggregated into the following levels shown below in Table 14. At first glance it can be seen that the GOEC produces a wide variety of availability levels across the subsystems, with a much larger standard deviation than the SOEC.

Table 14: Round 1 GOEC Optimization Technician Availability Levels

	μ	σ		μ	σ
Avionics	16	9	**Landing & Recovery**	18	8
Communications	19	8	**Navigation**	17	8
Crew	17	9	**Pneumatic**	19	8
Electrical & Wiring	20	7	**Propellant Management**	17	9
Engines	20	7	**Software**	20	7
Environmental	16	9	**Structures**	20	7
Flight Controls	19	8	**TPS**	16	8
Hydraulics	19	7	**Tracking**	19	8

The results from round 1 of GOEC optimization show that this method initially favors multiple vehicle simultaneous maintenance in contrast to the SOEC which found optimality in single vehicle maintenance. The overall performance metrics of the distributions within the top 10% of performers is shown in Table 15 below, and compared with the baseline in Table 16.

Table 15: Round 1 GOEC Optimization Results

.	**Vehicle0 MMH/flight**	**Vehicle1 MMH/flight**	**MMH/Flight/ Veh.**	**Flight Rate**
μ	1752	1754	1753	9.2
σ	110	106	107	0.6

In comparison with SOEC optimization, the first round of results only reduces the workforce by ½ as much, with a reduction close to 40%. However, the increased workforce results in a much smaller difference from the baseline in flight rate and

MMH/Flight/Vehicle. Specifically for MMH/Flight/Vehicle, GOEC optimization in 1 round performs 150% better.

Table 16: Round 1 GOEC Optimization Comparison with Baseline

Flight Rate (%)	MMH/Flight/Veh. (%)	Workforce (%)
-12.38	21.05	-39.17

Iterated GOEC Grid Search Results

In the following figures, the progression of GOEC optimization will be demonstrated by showing the performance improvement taking place in each round. The axes of each figure are percentage comparisons with the baseline study so that the MMH/Flight/Vehicle and flight rate of each experimental run can be compared on the same basis. The first round depicted below in Figure 11 results from choosing the top 10% of performers in Round 0, whose average and standard deviation are used for choosing the technician availability ranges used in the next round of optimization. The same process is used in all succeeding rounds of optimization.

In the 1st round, the top 10% of performers according to the GOEC have flight rates from 0-22% lower than the baseline study, and 5-40% higher MMH/Flight/Vehicle. However, after applying the GOEC in the first round, the technician availability combinations indicated have shrunk down the ranges to 0-8% reduction in flight rate, with 0-30% increase in MMH/Flight/Vehicle. As can be seen in Figure 12 below, the trend of moving skillset performance toward that of the baseline continues in the next round, as the flight rate is within 5%, and MMH/Flight/Vehicle is within 15% of the baseline.

As optimization continues, each successive round shrinks down the range of comparison with the baseline until optimal values are found. On from round 2, flight rate goes from within 4% to within 3% to within 2.5% until finally settling at 2-2.5% reduction in flight rate with optimal values. MMH/Flight/Vehicle goes from 10% to within 7% to within 6% until settling at 2-5% increase in MMH/Flight/Vehicle with optimal values. A main difference between the SOEC and GOEC methods is that the latter took 7 rounds to reach optimal settings, under the assumption that optimal settings are found once the standard deviation of the top 10% is < 1 (cannot add or subtract a portion of a technician).

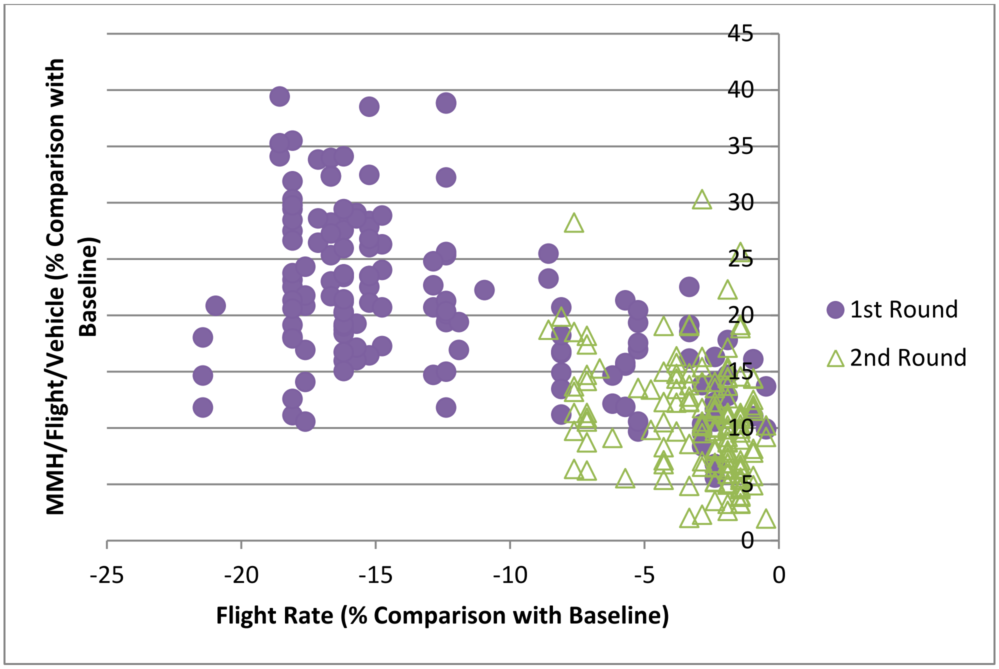

Figure 11: GOEC Optimization – Rounds 1 & 2

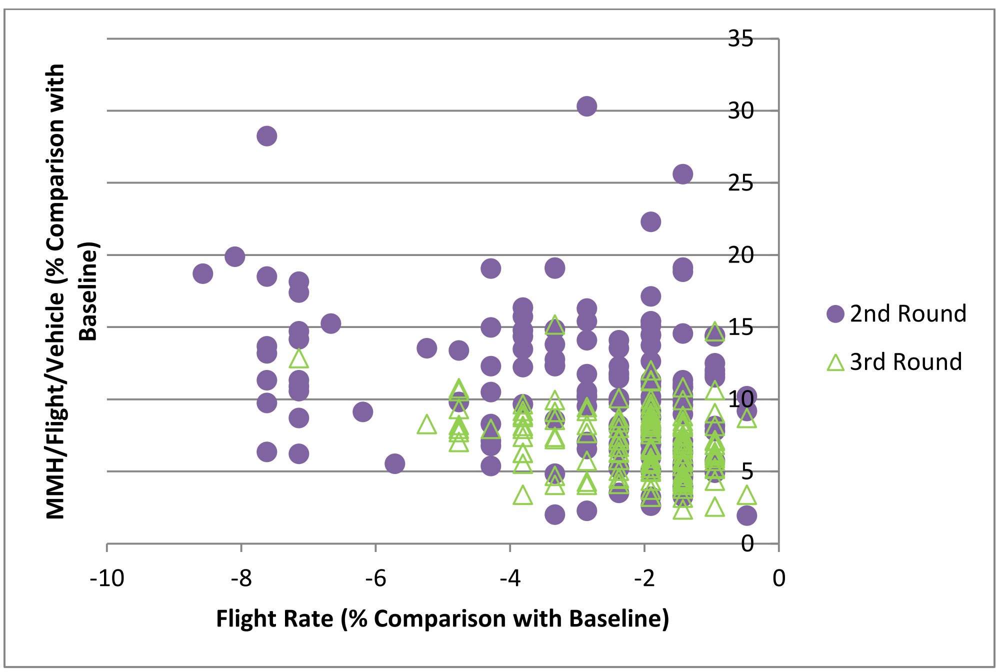

Figure 12: GOEC Optimization – Rounds 2 & 3

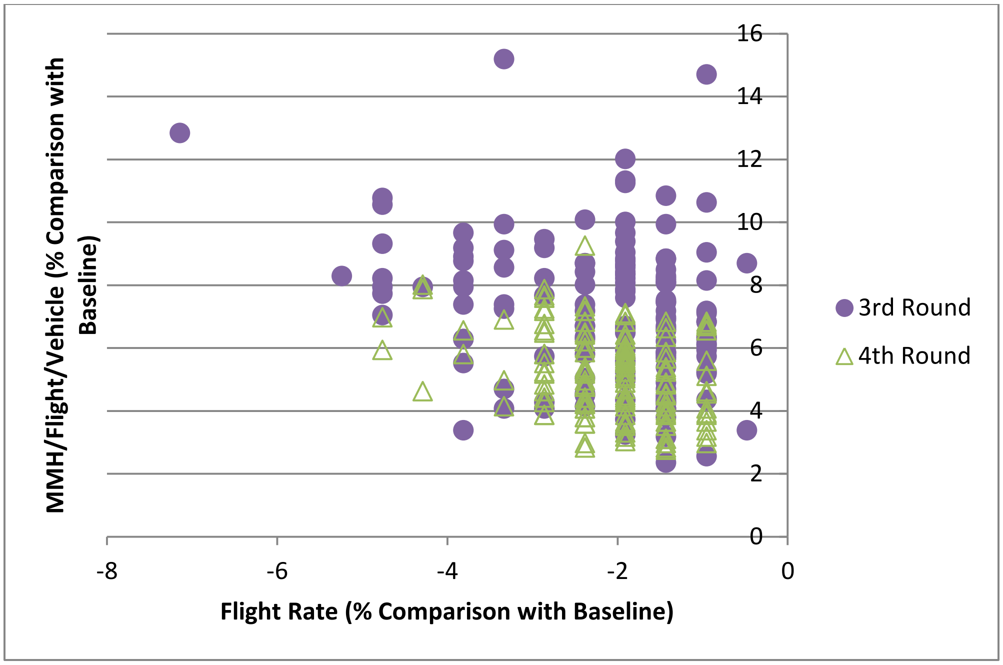

Figure 13: GOEC Optimization: Rounds 3 & 4

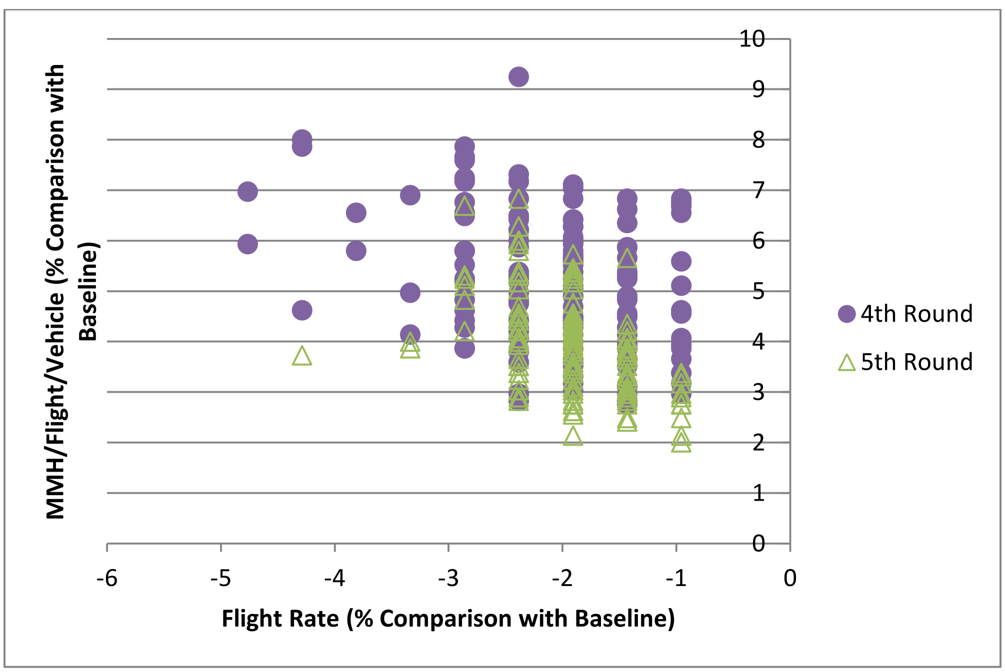

Figure 14: GOEC Optimization – Rounds 4 & 5

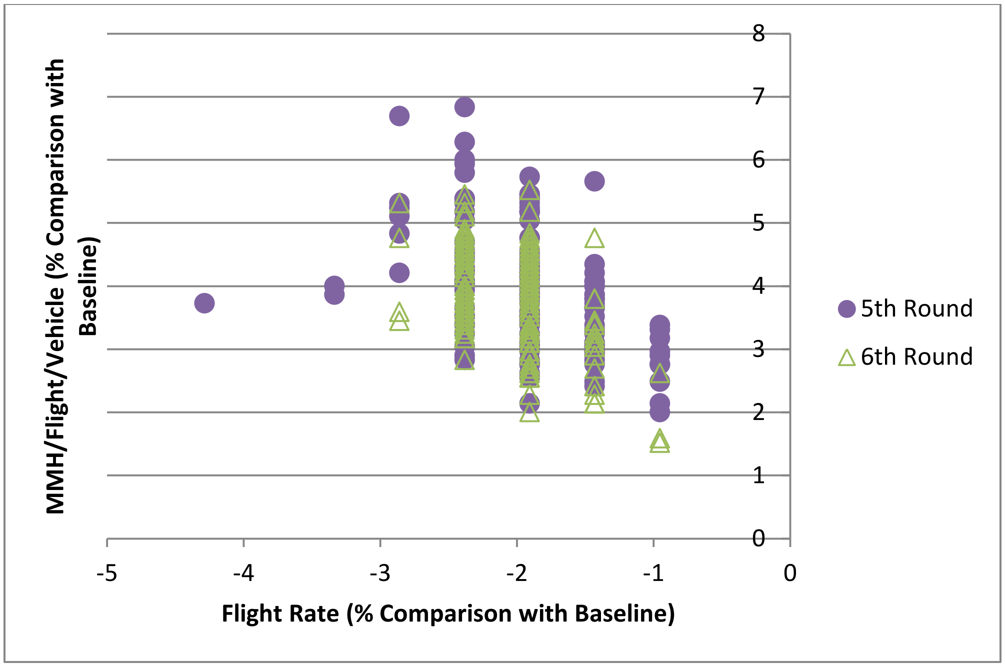

Figure 15: GOEC Optimization – Rounds 5 & 6

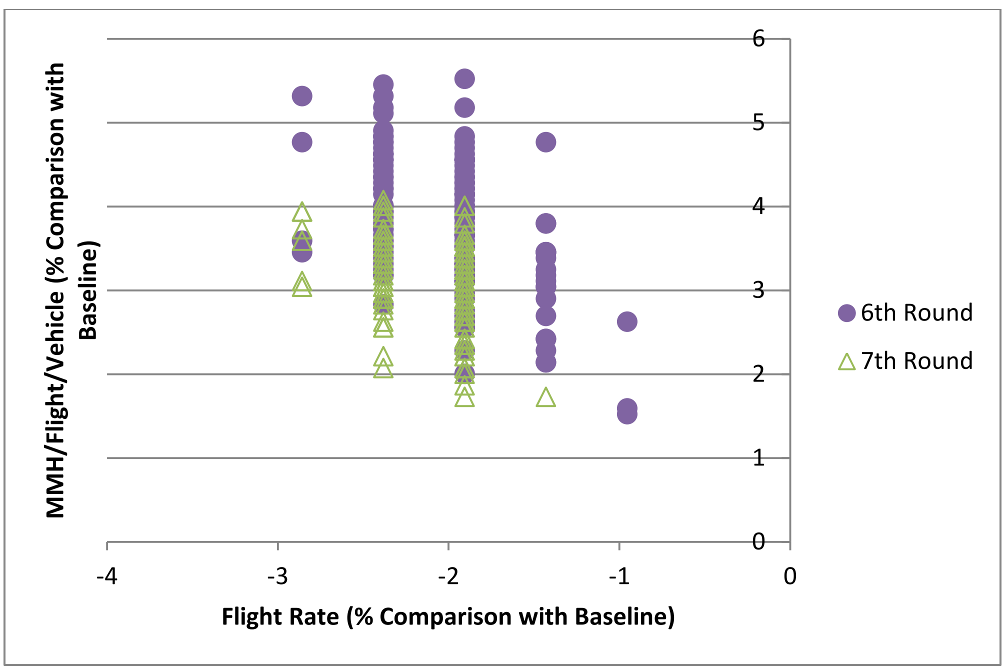

Figure 16: GOEC Optimization – Rounds 6 & 7

An interesting feature of optimizing on the distribution as a whole is that in successive rounds of optimization the flight rate is optimized fairly quickly, but by making further trades in technician allocation, the MMH/Flight/Vehicle continues to be reduced along with the required workforce. This is where the GOEC method shows its merit: by considering the performance of the skillset distribution as a whole rather as a sum of parts, technicians may be allocated where their skills will produce the greatest overall performance. The optimal levels converged upon are shown below in Table 18.

Table 17: GOEC Optimization Round Results

		Round Results		Compare with Baseline		
		Flight Rate	**MMH/Flight/Veh.**	**Flight Rate**	**MMH/Flight/Veh.**	**Workforce**
1st Round	μ	9.2	1754	-12.38	21.05	-39.17
	σ	0.6	107			
2nd Round	μ	10.2	1604	-2.86	10.7	-40.21
	σ	0.2	73			
3rd Round	μ	10.3	1551	-1.9	7.04	-42.71
	σ	0.1	33			
4th Round	μ	10.3	1524	-1.9	5.18	-43.33
	σ	0.1	19			
5th Round	μ	10.3	1506	-1.9	3.93	-43.75
	σ	0.1	14			
6th Round	μ	10.3	1503	-1.9	3.73	-44.38
	σ	0	12			
7th Round	μ	10.3	1494	-1.9	3.11	-44.38
	σ	0	7			

Table 18: Optimal Levels from GOEC Optimization

Avionics	15	**Landing & Recovery**	16
Communications	18	**Navigation**	15
Crew	13	**Pneumatic**	15
Electrical & Wiring	27	**Propellant Management**	14
Engines	19	**Software**	17
Environmental	15	**Structures**	17
Flight Controls	17	**TPS**	15
Hydraulics	17	**Tracking**	17

An added benefit to GOEC optimization is that in successive rounds it does not exhibit failed cases. Although in SOEC optimization the failed cases resulted in quicker convergence, it does so at a loss of statistical accuracy. In a stochastic system such as this, the more results there are to compile the more confidence one can have in their analysis. As shown in Figure 17 below, changes in comparison to the baseline do not change as rapidly as in SOEC optimization, and convergence on the optimum requires more rounds, however it results in a flight rate and MMH/Flight/Vehicle closer to the baseline.

On a qualitative basis, the technician availability levels which GOEC optimization converges on make logical sense. The Electrical & Wiring subsystem is by far the most time-intensive subsystem, and it has the most technicians allocated to it, such that it can work on 2 vehicles simultaneously with small loss of efficiency. The Crew subsystem by contrast is the least time-intensive and is allocated the fewest number of technicians, less than the maximum for one vehicle. This kind of allocation captures the effect in discussion with Figure 4, showing notionally the value of trading individual efficiency with overall efficiency. For further detail on GOEC optimization consult Appendix C.

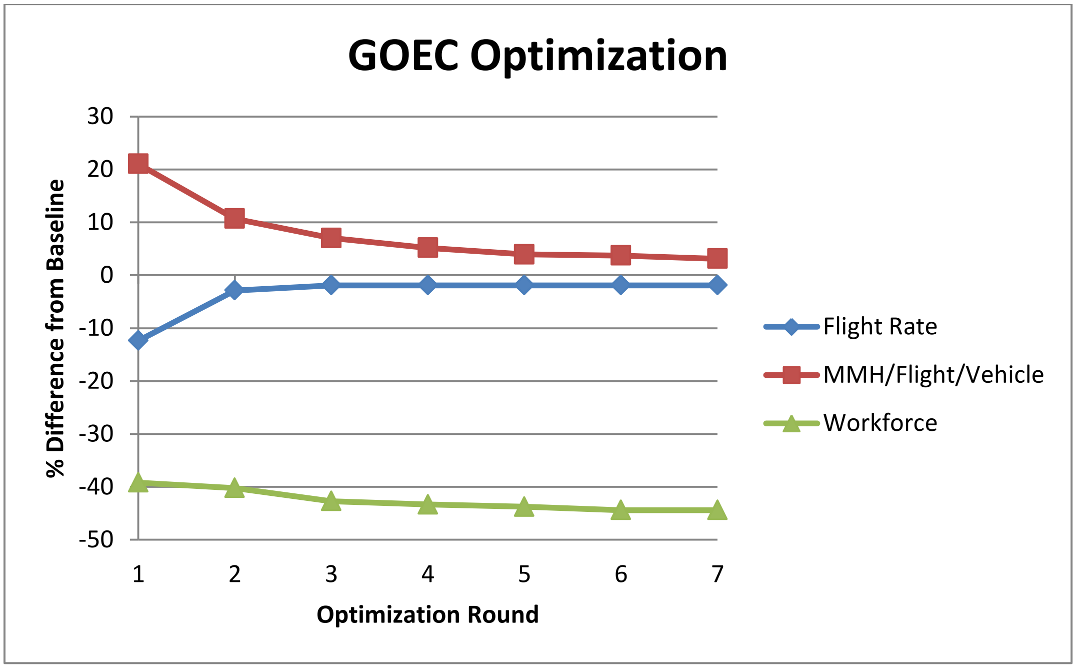

Figure 17: GOEC Optimization Evolution

Method Comparison

By comparing the results of each method of optimization side-by-side their relative strengths and weaknesses may be uncovered. To do so, workforce, and the MMH/Flight/Vehicle and flight rate metrics of interest will be considered separately. For workforce, both methods converge upon their optimal value before complete convergence, in that the percentage difference from the baseline does not change in those rounds. GOEC optimization however is already very close to its optimal value of workforce in the first round of optimization, while SOEC optimization takes 4 rounds to settle on a workforce sum. GOEC optimization's ability to hover around a certain sum comes from optimizing on the workforce as a whole: only those distributions which allocate technician skills more intelligently and with fewer technicians than before will improve on the previous round. SOEC optimization by comparison considers the skillsets one subsystem at a time, viewing the O&M scheme at a micro-level. Whereas the maintenance of individual subsystems is more efficient overall in terms of technician utilization, it does not consider potential trades between subsystems.

As the trends of MMH/Flight/Vehicle and flight rate are based upon the same data they will be considered together. As can be seen in Figure 19 & Figure 20 below, like workforce optimization, MMH/flight/Vehicle and flight rate optimization occurs very rapidly at first for SOEC optimization, in comparison with GOEC optimization which starts at a value very close to its optimum. However in contrast to workforce optimization, MMH/Flight/Vehicle and flight rate optimization diverges in the final round. This result is due to the method by which the SOEC achieves optimization. By taking the flight rate into account in its OEC, SOEC optimization does somewhat take overall efficiency into account, however not in the same manner as GOEC optimization. In the case of SOEC optimization, the inclusion of flight rate in the OEC produces the technician availability level which while optimizing the efficiency of maintenance on that subsystem, preferentially chooses the availability which minimizes effect on flight rate. This in effect will push availability toward the level which maximizes individual subsystem efficiency but minimizes changes in flight rate. By comparison, GOEC optimization simultaneously attempts to decrease overall availability levels but seeks out those levels which increase flight rate. Said in a more concise manner, SOEC optimization seeks out those technician availability levels with the highest gradient in subsystem efficiency and saddle points in flight rate. GOEC optimization in contrast seeks out the highest gradient levels in both subsystem efficiency and flight rate.

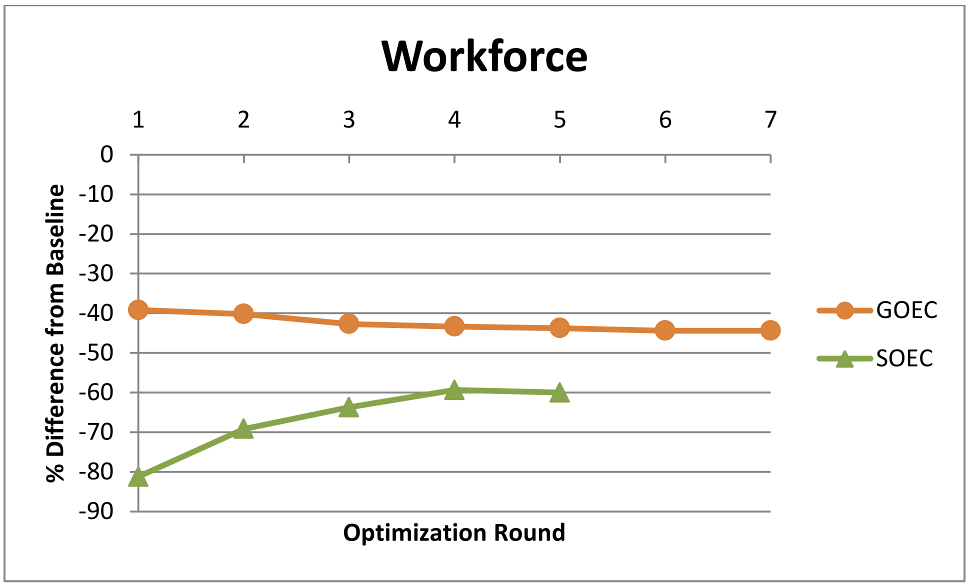

Figure 18: Comparison of workforce optimization

In addition, in SOEC's final round of optimization, each of the subsystems could vary ±1 technician, implying that there was still a decent amount of variation present within the technician availability levels. By comparison, in GOEC optimization's penultimate (6^{th}) round, only 5 subsystems still had any variation to explore. In this manner GOEC optimization achieves optimization by performing trades between technician availability levels in each subsystem until individual optimal levels are naturally converged upon, and continues until the optimum is found.

Both methods have their strengths and weaknesses. SOEC optimization for one quickly eliminated the off-optimal levels it had originally found (round 1 – round 2), came to convergence more quickly than GOEC optimization, and converged upon a solution with fewer required technicians than GOEC optimization. However, the elimination of points decreased the statistical confidence of results coming out of round 2 optimization, and convergence came to settings which performed worse than the competing method in fleet metrics (MMH/Flight/Vehicle, flight rate), without allowing for multiple vehicle maintenance. GOEC optimization on the other hand converges to values which perform very close to the baseline study with a 44% reduction in workforce, performs trades intelligently between the subsystem's technicians, and allows for multiple vehicle maintenance.

Ultimately the choice in optimization comes down to which is more important to the person performing the study. Both methods produce skillset allocation distributions with flight rates within 10% and MMH/Flight/Vehicle within 25% of the baseline study, which represents the performance which can be expected from always having enough technicians present to do maintenance in the most efficient manner on multiple vehicles. That said GOEC optimization performs better at aggregate levels, producing flight rates within 2% and MMH/Flight/Vehicle within 3% of the baseline, but with 20% more workforce than SOEC optimization. These comparative results are shown below in Figure 20.

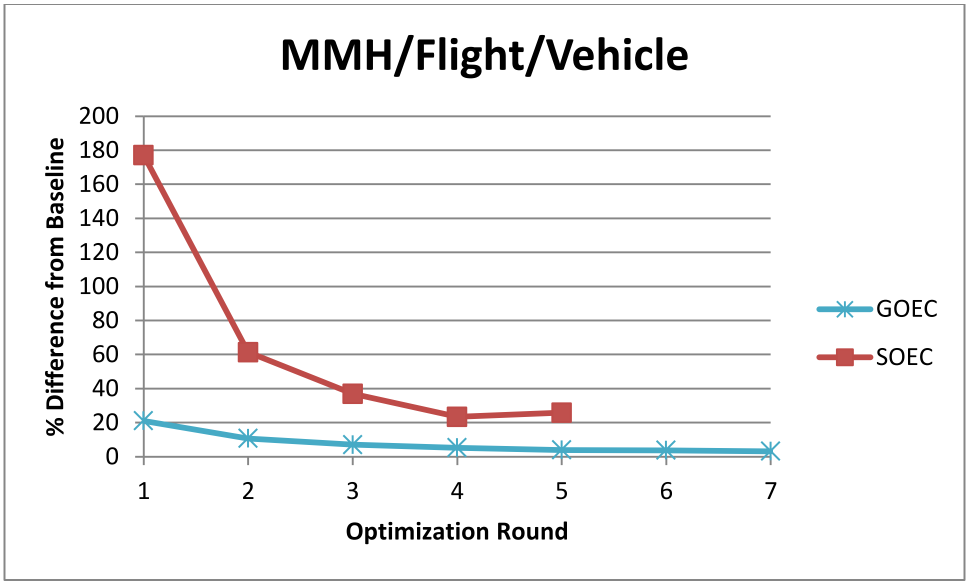

Figure 19: Comparison of MMH/Flight/Vehicle optimization

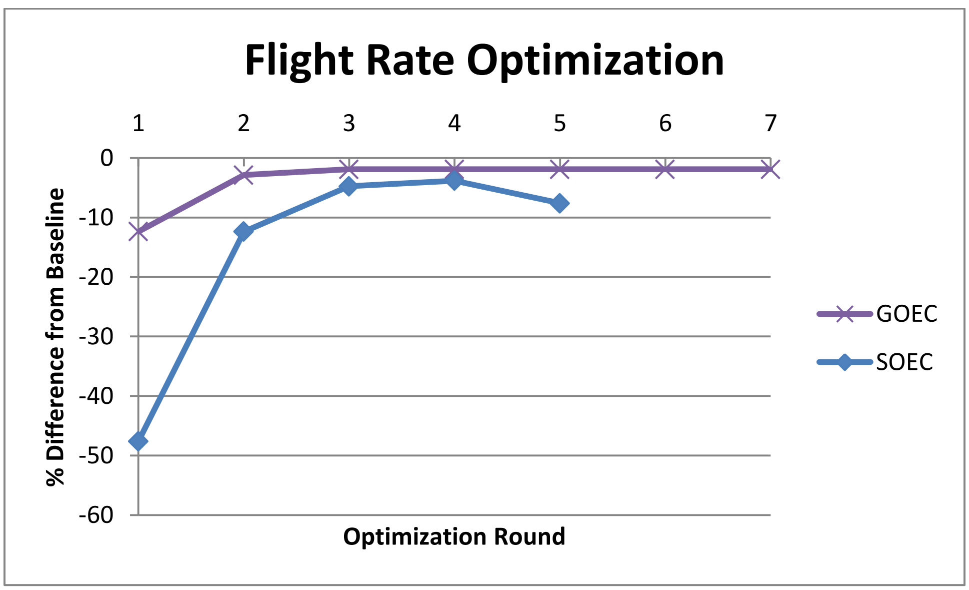

Figure 20: Comparison of flight rate optimization

Table 19: Optimization Technique Convergence Comparison

	Flight Rate (%)	MMH/Flight/Veh (%)	Workforce (%)
SOEC	-7.62	25.81	-60
GOEC	-1.9	3.11	-44.4

The choice of optimization technique then comes down to whether the fleet operating at maximum capability is the most important, or if it operating at maximum workforce efficiency is. For the emerging spaceflight companies, customers requiring launches are still building up their confidence in the commercial sector, resulting in yearly launch rates that can fluctuate quarter by quarter. To them, maximizing a flight rate is not quite as important as keeping a particular flight rate. However; keeping maintenance operating at a high efficiency is a boon. The fewer overall hours maintenance takes place reduces the operational cost of 'keeping the lights on' in maintenance facilities, and minimizing the technicians necessary for this work reduces the associated personnel cost. In conjunction with these concerns, and noting that the GOEC method arrives at its converged values by quickly achieving an optimal flight rate and then trading technicians across subsystem specializations to achieve that same rate while increasing efficiency and minimizing necessary workforce, the GOEC method is the clear winner.

In fact, GOEC optimization performs the best of all the tools and strategies employed within this work. To check that this is the case, the workforce skillset distributions resulting from the other methods presented are run through simulation 100 times each. First off, the LP model was considered. Although the LP model only considers single-vehicle maintenance, the single-vehicle preference for optimization shown by the SOEC method hints that this sort of optimization could yield good results. As shown below in Table 20, the LP model yields a skillset distribution which when input into simulation yields only a 7% reduction in flight rate, which is just barely better than the SOEC method. However, the MMH/Flight/Vehicle is 75% higher. Inspired by the performance yielded by such simple comparison, the distribution from the LP model was doubled (LP x2) and run again through simulation.

Table 20: Algorithm Comparisons with Simulation Baseline Study

	Flight Rate (%)	MMH/Flight/Veh. (%)	Workforce (%)
LP	-7.37	75.20	-71.25
LP x2	-3.60	10.77	-57.29
Monte Carlo	-33.09	122.44	-72.71
Monte Carlo w/ Variation	-58.02	142.70	-74.79
DES SOEC	-7.62	25.81	-60.00
DES GOEC	-1.90	3.11	-44.38

As can be seen above, the LP x2 model outperforms the LP model and the SOEC method, yielding only a 4% decrease in flight rate with an 11% increase in MMH/Flight/Vehicle. However, the SOEC method does still beat the LPx2 model in its workforce reduction, beating LPx2 by 3%. To choose which distribution coming from Monte Carlo, the run with the highest OEC value is selected. The OEC equation in this case is the GOEC equation. The Monte Carlo methods, both with and without variation, do not work as well, which is to be expected as their inputs are randomly generated. If the Monte Carlo simulations were allowed to run for a very long time it is possible that either could produce the distribution found by the simulation methods, however they could not do so dependably.

Overall, the DES GOEC method wins out by performing optimization in the most intelligent manner. Although the LP models were able to come close to its performance by adjusting the allocation of technicians to individual subsystems as per the strategy in

Figure 3, it was not able to account for multiple vehicle maintenance, which would allow it to employ the strategy in Figure 4. Although the case could be made for simply choosing the LP model distribution then scaling it to the number of vehicles present, the lack of uncertainty in the LP model implies that it is not robust to variation in the manner the GOEC distribution is, nor did it take into account the overall effect of the distribution in its formulation, which in the comparison between SOEC and GOEC methods it was seen that this feature is what defines GOEC's success. In conclusion, the use of an optimization method which takes into account the entire 16-variable design space on the 3-variable output space is superior, in its ability to find a global maximum, and its inclusion of variation, yielding a skillset distribution robust enough to handle multiple vehicle maintenance with a large reduction in the necessary workforce.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

Conclusions of Research

In this course of study, several fundamental research questions have been posed and now at the end may be answered completely. The first question required the selection of an experimental frame, i.e. a modeling method, which was both capable and efficient at capturing the effects and relationships present within RLV O&M.

Answering Research Question 1: What is the proper modeling method for capturing RLV O&M?

A literature review of previous efforts resulted in the selection of LP, Monte Carlo, and DES as the candidates, with the most historical precedent pointing toward DES. After constructing LP, Monte Carlo, and DES models of RLV O&M, DES is superior in its ability to model complex effects and its potential for expansion and customization.

After comparing the methods of LP, Monte Carlo, and DES, the power and precedent of DES make it the clear winner for performing simulation of RLV O&M. During optimization, the GOEC method emerged as the superior method for optimizing a DES model of RLV O&M due to the fact that it is a multi-objective optimization method. In the introduction of this work, the importance of bringing more information into the conceptual design phase of development was expanded upon because it ultimately parallels the selection of the GOEC method. Just like considering the viability of an aircraft while optimizing its feasibility improves the overall design by balancing the concerns of each area, so does considering the effect of the entire skillset distribution of technicians performing RLV O&M result in the greatest efficiency. After completing this analysis, the second research question can finally be answered.

Answering Research Question 2: How can the skillset of a RLV maintenance workforce be optimized?

Considering a maintenance workforce as more than the sum of its parts has resulted in a more intelligent skillset distribution. By optimizing on the entire distribution at once rather than one subsystem at a time, trades which are not available to the other method emerge which effectively balance the needs of a RLV O&M force. Finally, the third research question is also answered.

Answering Research Question 3: How can RLV O&M be effectively captured by a model?

The final research question is answered by making the following assumptions, each of which has been researched to find any existing precedent.

1. RLVs undergo the same maintenance cycle as any aerospace vehicle
2. RLV maintenance can be represented as composed of 16 subsystems
3. Tasks performed during maintenance have different levels of complexity
4. Allocating more technicians reducing the necessary maintenance time
5. There is a maximum number of technicians which can work on a RLV at a time

6. Maintenance on a subsystem requires unique skills

7. Changing the allotted number of technicians for a subsystem has a reciprocal effect on maintenance time

8. The variation in a maintenance task's completion time follows a log-Normal distribution

In performing this study, a generic method for optimizing a RLV O&M maintenance skillset distribution was shown which can be effective with the inclusion of more vehicles, more maintenance sites, and more maintenance detail. It is the hope of the author that as commercial entities move forward with their RLV campaigns, they will perform such analyses to make their ventures the most efficient, and move mankind amongst the stars.

Recommendations for Future Research

As with any simulation, increasing fidelity is of utmost concern. Specifically, the tasks required for each subsystem and the times associated with each of these tasks is very important if the simulation is to be used for future studies. To that end, the simulation has been coded with the potential for subsystem module extensions that require no changes to existing code. All one would have to do is code a module containing new tasks and times which can use vehicle and maintenance site characteristics, and have that code return an amount of time required. Another change which would improve fidelity on this front would be to distribute the subsystem spreadsheets themselves to commercial and government entities currently performing RLV maintenance, and to incorporate data found too late into this effort to include [16]. The code is currently designed to derive statistical moments from the columns associated to each task, so if these entities could log the times they actually spend on individual tasks, or even on the entire subsystem itself, then the fidelity of the simulation would be greater with each new entry.

Work which would require code changes include allowing complex interactions between maintenance work and technicians, and further probabilistic operational dependencies. The first would go hand in hand with the reciprocal relationship between task times and available workers. At present moment once the technician resource is released the remaining time on any work which started without the maximum number of technicians is not affected. In reality however, one can expect that if a team of technicians was split in two between maintaining two vehicles and work was completed on one, then their attention would be re-allocated so that work on the second vehicle could speed up. Along the same vein, one central assumption in the simulation is that maintenance skills are mutually exclusive which is not realistic. Further work on the simulation would allow for technicians to work on multiple subsystems, which would require further research and expert consultation. In addition to these two improvements, another modification having to do with vehicle to maintenance interactions would be including extra logic so that vehicle characteristics such as propulsion and TPS material choice will directly affect either the tasks done and/or the time they take to complete. For the second point, allowing for probabilistic launch window cancellations such as those employed in [5] would be helpful in producing a more robust workforce distribution.

APPENDIX A: TASK LISTS

The tasks associated with each subsystem come from the FAA's Guide to Commercial Reusable Launch Vehicle Operations and Maintenance [26], and while they are not assumed to be a complete list for any of the subsystems, the lists represent an ultimately universal minimum task list, as they are specifically required by the FAA due to their potential for environmental impact if not carried out after each launch. In this manner, the task lists to follow are considered to be representative of a commercial entity attempting to minimize maintenance costs by performing the least amount of maintenance possible. Task lists, descriptions, and assumed time means follow. Tasks are divided into three categories: 1-day, 2-day, and 3-day, assuming 1 8-hour shift per day. Their statistical moments are:

1-Day: $\mu = 2.15$, $\sigma^2 = 0.2 \rightarrow \sim 8$ hours average

2-Day: $\mu = 2.8$, $\sigma^2 = 0.1 \rightarrow \sim 16$ hours average

3-Day: $\mu = 3.3$, $\sigma^2 = 0.1 \rightarrow \sim 24$ hours average

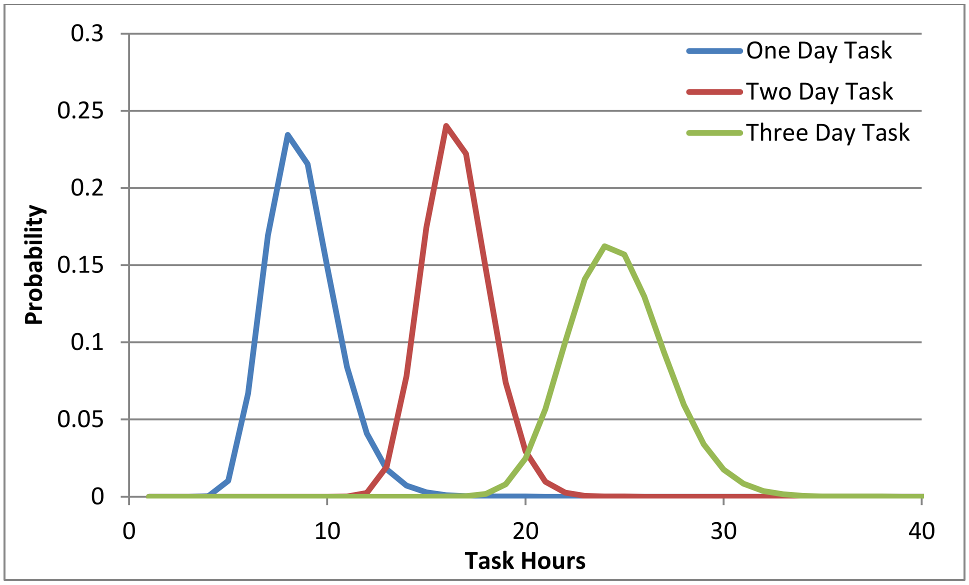

Figure 21: Task Time Levels Probability Density Functions

1. Avionics

   a. Inspection - Out-of-configuration Avionics conditions should be recognized and isolated upon activation – 1-day

   b. Maintenance check - Any software or hardware used to check avionics must itself be checked for errors – 1-day

   c. Intra-vehicle comms check - Communication between communications, guidance, navigation, environmental, and flight controls must be verified – 1-day

   d. Test redundancy safing - Avionics are designed to handle multiple failures through redundant hardware and software – 3-day

e. Test resource management - Software within avionics should give proper resource priority and deconfliction between different functions (time, data access, memory, display panels, etc) – 2-day

2. Communications

   a. Wiring inspection - All hardware within the communications subsystem must be inspected for damage – 2-day

   b. External inspection - All external transmitters, receivers, and antennae must be inspected for damage – 1-day

   c. Test main RLV comms - Information sent to the RLV must be received properly – 1-day

   d. Test comms from RLV - Transmissions from the RLV to SGLS must be sent correctly – 1-day

   e. Test backup RLV comms - Information sent to the RLV must be received properly – 1-day

   f. Test backup from RLV comms - Transmissions from the RLV to SGLS must be sent correctly – 1-day

   g. Link analysis - All internal comms must also be functional – 1-day

   h. End to end testing - Internal to external and vice versa must be functional – 2-day

   i. Repair - Any required repairs – 3-day

3. Crew Systems

   a. Inspection - Inspect cockpit equipment and crew restraint mechanisms for wear and damage – 1-day

b. Repair - Any equipment not meeting standards must be repaired or replaced – 3-day

4. Electrical & Wiring

a. Inspection - As much of the internal wiring of the RLV as possible must be inspected – 3-day

b. APU Inspection - Auxiliary Power Units must be tested and refilled with Hydrazine – 1-day

c. RTG Inspection - Radioisotope Thermoelectric Generators must be inspected for plutonium radiation leaks and refueled if necessary – 1-day

d. RAT Inspection - Ram Air Turbines must be inspected for damage and repaired – 1-day

e. Fuel Cell replacement - Any fuel cells used must be safely disposed of and replaced – 1-day

f. Solar cell inspection - Any Solar cells used for power generation must be inspected for damage – 1-day

g. Repairs - Any faulty wiring must be promptly replaced – 3-day

h. Replacement - Any faulty devices must be replaced – 3-day

i. Harness integrity - Post-repair inspection must ensure that all wiring harnesses removed during inspection & repair are replaced – 2-day

j. Testing - After maintenance, entire system must be checked for faults – 3-day

5. Engines

a. Safe Removal - SSMEs must be completely removed from STS for maintenance – 1-day

b. Mounting - Engines are then moved to their test facility and mounted for testing – 1-day

c. Venting - Engines must be vented of toxic fluids and gases for maintenance – 1-day

d. General Inspection - System testing and checkout of engines and thrusters to account for engine wear characteristics – 3-day

e. Inspect nozzles - Nozzles must be checked for cracks and fatigue – 1-day

f. Inspect feed lines - Feed lines must be checked for cracks and fatigue – 1-day

g. Inspect turbo pumps - Turbo-pump must be inspected for cracks and fatigue – 1-day

h. Inspect igniter - Igniter must be inspected for cracks and fatigue – 1-day

i. Repair - All damaged systems must be repaired or replaced – 3-day

j. Testing - Activation of engines should produce the expected amount of thrust at the expected burn rate – 3-day

6. Environmental

a. Atmosphere - Atmospheric controls must be inspected and tested for proper operation – 2-day

b. Water - Water treatment systems must be inspected and tested for proper operation – 1-day

c. Waste - Waste management systems must be inspected and tested for proper operation – 1-day

d. Redundancy - Test redundant environmental systems – 1-day

e. Suppression - Chemicals used for fire and explosion suppression must be checked for adequate pressure levels and freshness – 1-day

f. Repair - Any systems needing repair must receive it – 3-day

7. Flight Controls

a. Inspection - Flight control hardware including propulsive engines, reaction control jets, and aerodynamic control surfaces must be inspected for damage – 3-day

b. Testing - Control reactions and thrust vector alignments must be verified – 2-day

c. Software - All software components of flight control must be verified and maintained – 2-day

d. Repair - Any misalignments must be corrected – 3-day

e. Replacement - Any faulty control mechanisms must be replaced – 2-day

f. Post-Inspection - Flight controls should undergo a post-maintenance inspection to ensure proper operation – 1-day

8. Hydraulics

a. Clean joints - To prevent contamination, areas immediately adjacent to joints to be separated for maintenance should be cleaned before loosened – 1-day

b. Gimbal - Thrust vector control of SSMEs must be verified – 1-day

c. Valves - Control of propellant valves must be verified – 1-day

d. Orbiter Aerodynamics - Control of orbiter aero surfaces (elevons, body flap, rudder/speed brake) must be verified – 2-day

e. Retractor - Retraction of external tank/orbiter LOX/LH2 disconnect umbilicals must be verified – 1-day

f. Brakes - Main landing gear deployment verified – 1-day

g. Nose - Nose wheel steering must be verified – 1-day

h. Safing - Automatic safing procedure must be tested in case of low pressure situation – 1-day

i. Clamps - Any and all clamps or line blocks removed during repair must be inspected for proper reinstallation – 2-day

j. Replacement – 3-day

9. Landing & Recovery

a. Anti-Skid - Anti-Skid brakes and electrical power/pedal components must be inspected for wear and damage – 1-day

b. Autonomy - Autonomous landing equipment must be inspected for wear and damage – 1-day

c. Calibration - All components must be calibrated to flightworthiness standards – 2-day

d. Stowage - Inspect systems for landing/recovery gear stowage – 1-day

e. Tires - Check tires for wear – 1-day

f. Repair - Any faulty systems must be repaired or replaced – 3-day

g. Testing - After calibration, all systems must be tested again – 2-day

10. Navigation

   a. Test inertial measurement unit (IMU) - IMU must be functional for accurate navigation – 1-day

   b. Test global positioning system (GPS) - GPS must be functional for navigation – 1-day

   c. Calibrate IMU - Error inevitably crops up, account for this – 1-day

   d. Calibrate GPS - Error inevitably crops up, correct for this – 1-day

   e. Replace faulties - Any faulty hardware must be removed and replaced – 2-day

   f. Test system - After calibration, test entire system again – 3-day

11. Pneumatics

   a. Leaks - Check for leaks – 1-day

   b. Valves - Check for proper operation of shut-off valves – 1-day

   c. Contamination - Check for any contamination, and ensure that protection components are sound – 2-day

   d. Regulators - Ensure that temperature and pressure regulators are functioning properly – 1-day

   e. Pressure - Ensure that the proper pressure is held – 1-day

   f. Mounting - Ensure proper mounting of all units – 1-day

   g. Hoses - Inspect hoses for leaks and wear – 1-day

   h. Replacement - Any faulty components must be replaced – 3-day

   i. Testing - After inspection and repair, operational testing must ensure proper operation – 2-day

12. Propellant Management

   a. Inspection - Any and all feed lines, containers, and valves must be inspected for damage and leaks – 2-day

   b. Software - Onboard propellant management subsystems must remain within flightworthiness standards – 1-day

   c. Repair - Any faulty components must be repaired or replaced – 3-day

13. Software

   a. Test System - System software manage a computer's basic tasks – 2-day

   b. Test Utility - Utility software performs day-to-day tasks – 2-day

   c. Test Applications - Application software perform specialized RLV controls – 1-day

   d. Test PASS - Primary Avionics Software System must pass rigorous testing – 3-day

   e. Test BFS - Backup flight systems must pass rigorous testing – 3-day

14. Structures

   a. Inspect movables - All movable structures must pass integrity testing – 3-day

   b. Inspect plume - All structures within the plume impingement area must pass inspection and testing – 2-day

   c. Metals inspect - Metal-based structures must be inspected for damage - visual may be sufficient – 1-day

   d. Composites inspect - Composite materials require other techniques for structural testing – 2-day

e. Repair - Any structures not meeting damage tolerance tests must be replaced – 3-day

15. Thermal Protection System

    a. Inspection - Identify which tiles have been damaged by previous mission (sqft) – 1-day

    b. Tile removal - Removal of damaged tiles (hr/sqft) – 2-day

    c. Tile replacement - Replacing damaged tiles (hr/sqft) – 3-day

    d. New tile testing - Replacement tiles must be tested before vehicle operation to ensure proper operation – 3-day

16. Tracking

    a. Inspection - Inspect all hardware responsible for reporting the RLVs position for damage – 2-day

    b. Antenna - Test gearing and encoders on antenna dishes – 1-day

    c. Waveguide - Waveguide alignment must occur for proper tracking – 1-day

    d. Transponder - Transponders must be calibrated for any errors – 1-day

    e. Repair - Any faulty systems must be replaced – 3-day

    f. Testing - After calibration, all systems must be tested again – 2-day

APPENDIX B: SOEC OPTIMIZATION ROUND RESULTS

What follows are the round by round results of optimizing on a RLV maintenance workforce skillset distribution by considering the efficiency of each subsystem individually. As rounds progress, the average setting (denoted by μ) gets closer to its optimal value, which is in round 5 where the standard deviation σ goes to 0.

Round Levels

Table 21: SOEC Optimization Round Results

	1st Round		2nd Round		3rd Round		4th Round		5th Round	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
Avionics	5	4	9	2	11	2	13	1	12	0
Communications	5	4	9	2	11	2	13	1	13	0
Crew	5	4	9	2	11	2	13	1	13	0
Electrical_Wiring	15	4	15	2	15	2	15	1	15	0
Engines	5	4	9	2	11	2	11	1	10	0
Environmental	5	4	9	2	9	2	10	1	11	0
Flight_Controls	5	4	9	2	11	2	12	1	13	0
Hydraulics	5	4	9	2	11	2	13	1	14	0
Landing_Recovery	5	4	9	2	11	2	12	1	12	0
Navigation	5	4	9	2	11	2	13	1	13	0
Pneumatic	5	4	9	2	11	2	13	1	12	0
Propellant_Management	5	4	7	2	9	2	11	1	10	0
Software	5	4	9	2	11	2	11	1	10	0
Structures	5	4	9	2	11	2	11	1	11	0
TPS	5	4	9	2	11	2	13	1	13	0
Tracking	5	4	9	2	9	2	11	1	10	0
Flight Rate	5.5	0.03	9.2	0.02	10	0.02	10.1	0.02	9.7	0.02
MMH/Flight/Veh	4014	19	2338	4	1984	4	1788	4	1823	3

Table 22: SOEC Optimization Comparison Round Comparison with Baseline

	Flight Rate (%)	MMH/Flight/Veh (%)	Workforce (%)
1st Round	-47.62	177.02	-81.25
2nd Round	-12.38	61.35	-69.17
3rd Round	-4.76	36.92	-63.75
4th Round	-3.81	23.4	-59.38
5th Round	-7.62	25.81	-60

APPENDIX C: GOEC OPTIMIZATION ROUND RESULTS

What follows are the results coming from GOEC optimization, organized by the round in which they appear. As each round progresses, the technician availability levels get closer to their optimal values, taking 7 rounds to complete. At round 6 many of the subsystems have reached optimal values, and so are taken as constant in subsequent rounds.

Round Results

Table 23: GOEC Optimization Round Results

	1st Round		2nd Round		3rd Round		4th Round	
	μ	σ	μ	σ	μ	σ	μ	σ
Avionics	16	9	17	6	16	4	16	3
Communications	19	8	19	5	18	3	18	2
Crew	17	9	15	6	14	4	14	3
Electrical_Wiring	20	7	24	5	26	4	27	2
Engines	20	7	20	5	19	3	19	2
Environmental	16	9	16	5	15	3	15	2
Flight_Controls	19	8	18	5	17	3	17	2
Hydraulics	19	7	19	5	18	4	18	3
Landing_Recovery	18	8	17	5	16	3	16	2
Navigation	17	8	16	5	15	3	15	2
Pneumatic	19	8	18	5	18	4	17	3
Propellant_Management	17	9	16	6	15	4	14	3
Software	20	7	19	5	18	3	17	2
Structures	20	7	19	5	18	3	17	2
TPS	16	8	16	5	15	3	15	2
Tracking	19	8	18	5	17	3	17	2
Flight Rate	9.2	0.6	10.2	0.2	10.3	0.1	10.3	0.1
MMH/Flight/Veh	1754	107	1604	73	1551	33	1524	19

	5th Round		6th Round		7th Round	
	μ	σ	μ	σ	μ	σ
Avionics	16	2	15	1	15	0
Communications	18	1	18	0	18	0
Crew	14	2	13	1	13	0
Electrical_Wiring	27	1	27	0	27	0
Engines	19	1	19	0	19	0
Environmental	15	1	15	0	15	0
Flight_Controls	17	1	17	0	17	0
Hydraulics	17	2	17	1	17	0
Landing_Recovery	16	1	16	0	16	0
Navigation	15	1	15	0	15	0
Pneumatic	16	2	15	1	15	0
Propellant_Management	14	2	14	1	14	0
Software	17	1	17	0	17	0
Structures	17	1	17	0	17	0
TPS	15	1	15	0	15	0
Tracking	17	1	17	0	17	0
Flight Rate	10.3	0.1	10.3	0	10.3	0
MMH/Flight/Veh	1506	14	1503	12	1494	7

Table 24: GOEC Optimization Round Comparison with Baseline

	Flight Rate (%)	MMH/Flight/Veh (%)	Workforce (%)
1st Round	-12.38	21.05	-39.17
2nd Round	-2.86	10.7	-40.21
3rd Round	-1.9	7.04	-42.71
4th Round	-1.9	5.18	-43.33
5th Round	-1.9	3.93	-43.75
6th Round	-1.9	3.73	-44.38
7th Round	-1.9	3.11	-44.38

APPENDIX D: MODEL CODES

Linear Programming

```
from SimplexAlgorithm import simplex

#def simplex(f,contraints):

 # Sample Use:

"""print simplex([-52.4,-73.0,-83.4, -41.8],

[([1.5,1.0,2.4,1.0],200.0),

([1.0,5.0,1.0,3.5],800.0),

([1.5,3.0,3.5,1.0],500.0)])"""

#Subsystem times: (#Days)*(Hours/day)-> Hours

Avionics = -(8.0*8.0)/15.0

Communications = -(13.0*8.0)/15.0

Crew = -(4.0*8.0)/15.0

Electrical_Wiring = -(19.0*8.0)/15.0

Engines = -(15.0*8.0)/15.0

Environmental = -(9.0*8.0)/15.0

Flight_Controls = -(13.0*8.0)/15.0
```

```
Hydraulics = -(14.0*8.0)/15.0

Landing_Recovery = -(11.0*8.0)/15.0

Navigation = -(9.0*8.0)/15.0

Pneumatics = -(13.0*8.0)/15.0

Propellant_Management = -(6.0*8.0)/15.0

Software = -(11.0*8.0)/15.0

Structures = -(11.0*8.0)/15.0

TPS = -(9.0*8.0)/15.0

Tracking = -(10.0*8.0)/15.0


#xi are the number of technicians available for each subsystem

function =
[Avionics,Communications,Crew,Electrical_Wiring,Engines,Environmental,Flight_Contr
ols,Hydraulics,\


Landing_Recovery,Navigation,Pneumatics,Propellant_Management,Software,Structures,
TPS,Tracking,\

    1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1]
```

```
constraints =
[([1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0
.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
```

([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.

```
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0],15.0),\

([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0],15.0)]

answer = simplex(function,constraints)[0:16]

for i in range(len(answer)):

    answer[i] = round(answer[i],0)
```

```
print "The optimal skillset distribution is:"

print answer
```

Second Linear Programming

```
from SimplexAlgorithm import simplex

#def simplex(f,contraints):

 # Sample Use:

"""print simplex([-52.4,-73.0,-83.4, -41.8],

[([1.5,1.0,2.4,1.0],200.0),

([1.0,5.0,1.0,3.5],800.0),

([1.5,3.0,3.5,1.0],500.0)])"""

#Subsystem times: (#Days)*(Hours/day)-> Hours

Avionics = -(8.0*8.0)/15.0

Communications = -(13.0*8.0)/15.0

Crew = -(4.0*8.0)/15.0

Electrical_Wiring = -(19.0*8.0)/15.0

Engines = -(15.0*8.0)/15.0

Environmental = -(9.0*8.0)/15.0

Flight_Controls = -(13.0*8.0)/15.0
```

Hydraulics = -(14.0*8.0)/15.0

Landing_Recovery = -(11.0*8.0)/15.0

Navigation = -(9.0*8.0)/15.0

Pneumatics = -(13.0*8.0)/15.0

Propellant_Management = -(6.0*8.0)/15.0

Software = -(11.0*8.0)/15.0

Structures = -(11.0*8.0)/15.0

TPS = -(9.0*8.0)/15.0

Tracking = -(10.0*8.0)/15.0

function =

[Avionics,Communications,Crew,Electrical_Wiring,Engines,Environmental,Flight_Contr

ols,Hydraulics,\

Landing_Recovery,Navigation,Pneumatics,Propellant_Management,Software,Structures,

TPS,Tracking]

constraints = [([1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\

([1.0,0.0,0.0,-

Avionics/Electrical_Wiring,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],0.0),\

([0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\

```
([0.0,1.0,0.0,-
Communications/Electrical_Wiring,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],0.0),\
([0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,1.0,-
Crew/Electrical_Wiring,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],0.0),\
([0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,-
Engines/Electrical_Wiring,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],0.0),\
([0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,-
Environmental/Electrical_Wiring,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],0.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,-
Flight_Controls/Electrical_Wiring,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],0.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
([0.0,0.0,0.0,-
Hydraulics/Electrical_Wiring,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],0.0),\
([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\
```

([0.0,0.0,0.0,-

Landing_Recovery/Electrical_Wiring,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0],0.0),\

([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0],15.0),\

([0.0,0.0,0.0,-

Navigation/Electrical_Wiring,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0],0.0),\

([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0],15.0),\

([0.0,0.0,0.0,-

Pneumatics/Electrical_Wiring,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0],0.0),\

([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0],15.0),\

([0.0,0.0,0.0,-

Propellant_Management/Electrical_Wiring,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0],

0.0),\

([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0],15.0),\

([0.0,0.0,0.0,-

Software/Electrical_Wiring,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0],0.0),\

([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0],15.0),\

([0.0,0.0,0.0,-

Structures/Electrical_Wiring,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0],0.0),\

([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0],15.0),\

```
    ([0.0,0.0,0.0,-
TPS/Electrical_Wiring,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0],0.0),\

    ([0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0],15.0),\

    ([0.0,0.0,0.0,-
Tracking/Electrical_Wiring,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0],0.0)]

answer = simplex(function,constraints)[0:16]

for i in range(len(answer)):

    answer[i] = round(answer[i],0)

print "The optimal skillset distribution is:"

print answer
```

Monte Carlo

```
from random import randint

#Subsystem times: (#Days)*(Hours/day)-> Hours

Avionics = (8.0*8.0)*15.0

Communications = (13.0*8.0)*15.0

Crew = (4.0*8.0)*15.0

Electrical_Wiring = (19.0*8.0)*15.0
```

Engines = (15.0*8.0)*15.0

Environmental = (9.0*8.0)*15.0

Flight_Controls = (13.0*8.0)*15.0

Hydraulics = (14.0*8.0)*15.0

Landing_Recovery = (11.0*8.0)*15.0

Navigation = (9.0*8.0)*15.0

Pneumatics = (13.0*8.0)*15.0

Propellant_Management = (6.0*8.0)*15.0

Software = (11.0*8.0)*15.0

Structures = (11.0*8.0)*15.0

TPS = (9.0*8.0)*15.0

Tracking = (10.0*8.0)*15.0

Subsystems = [Avionics, Communications, Crew, Electrical_Wiring, Engines, Environmental,\

Flight_Controls, Hydraulics, Landing_Recovery, Navigation, Pneumatics, Propellant_Management,\

Software, Structures, TPS, Tracking]

```
all_vehicles_launched_best = 10000

num_vehicles = 2

max_ppl = 15*num_vehicles

distribution =
[max_ppl,max_ppl,max_ppl,max_ppl,max_ppl,max_ppl,max_ppl,max_ppl,\
    max_ppl,max_ppl,max_ppl,max_ppl,max_ppl,max_ppl,max_ppl,max_ppl]

best_run = 0

run = 0

print "Performing Monte-Carlo simulation of " + str(num_vehicles) + " vehicles
undergoing maintenance."

while True:

    MMH = 0

    flight_rate = 0

    people =
[randint(1,max_ppl),randint(1,max_ppl),randint(1,max_ppl),randint(1,max_ppl),\
        randint(1,max_ppl),randint(1,max_ppl),randint(1,max_ppl),randint(1,max_ppl),\
        randint(1,max_ppl),randint(1,max_ppl),randint(1,max_ppl),randint(1,max_ppl),\
        randint(1,max_ppl),randint(1,max_ppl),randint(1,max_ppl),randint(1,max_ppl)]
```

```
ppl_this_run =[]

vehicles = []

first = True

time_into_year = []

for q in range(len(Subsystems)):

    ppl_this_run.append(people[q])

    hit_max = False

    subsystem = []

    if people[q] <= 15:

        hit_max = True

    for r in range(num_vehicles):

        if people[q] >= 15:

            available = 15

            people[q] -= available

            subsystem.append(Subsystems[q]/available)

        elif people[q] == 0:

            subsystem.append(Subsystems[q]/last_available)
```

```
            else:
                available = people[q]
                last_available = available
                people[q] -= available
                subsystem.append(Subsystems[q]/available)
        if hit_max:
            time_into_year.append(sum(subsystem))
        else:
            time_into_year.append(max(subsystem))
    all_vehicles_launched = max(time_into_year)
    if sum(ppl_this_run) <= sum(distribution) and all_vehicles_launched <=
all_vehicles_launched_best:
        all_vehicles_launched_best = all_vehicles_launched
        distribution = ppl_this_run
        best_run = run
        print "New best found! - run #" + str(run)
        print "New best launch rate of " + str(round(2080.0/all_vehicles_launched_best,1))
+ \
```

```
        " flights/year with a total of " + str(sum(distribution)) + " technicians."
    run += 1
    if run - best_run > 1000000:
        print "Completed " + str(run) + " runs before completion."
        print "Optimum found on run #" + str(best_run)
        break
print distribution
print all_vehicles_launched_best
```

Monte Carlo with Variation

```
from random import randint
#Subsystem times: (#Days)*(Hours/day)-> Hours
all_vehicles_launched_best = 10000
num_vehicles = 2
max_ppl = 15*num_vehicles
distribution =
[max_ppl,max_ppl,max_ppl,max_ppl,max_ppl,max_ppl,max_ppl,max_ppl,\
    max_ppl,max_ppl,max_ppl,max_ppl,max_ppl,max_ppl,max_ppl,max_ppl]
```

```
best_run = 0

run = 0

print "Performing Monte-Carlo simulation of " + str(num_vehicles) + " vehicles undergoing maintenance."

while True:

    Avionics = (8.0*8.0)*15.0*(1+randint(1,10)/100.0)

    Communications = (13.0*8.0)*15.0*(1+randint(1,10)/100.0)

    Crew = (4.0*8.0)*15.0*(1+randint(1,10)/100.0)

    Electrical_Wiring = (19.0*8.0)*15.0*(1+randint(1,10)/100.0)

    Engines = (15.0*8.0)*15.0*(1+randint(1,10)/100.0)

    Environmental = (9.0*8.0)*15.0*(1+randint(1,10)/100.0)

    Flight_Controls = (13.0*8.0)*15.0*(1+randint(1,10)/100.0)

    Hydraulics = (14.0*8.0)*15.0*(1+randint(1,10)/100.0)

    Landing_Recovery = (11.0*8.0)*15.0*(1+randint(1,10)/100.0)

    Navigation = (9.0*8.0)*15.0*(1+randint(1,10)/100.0)

    Pneumatics = (13.0*8.0)*15.0*(1+randint(1,10)/100.0)

    Propellant_Management = (6.0*8.0)*15.0*(1+randint(1,10)/100.0)

    Software = (11.0*8.0)*15.0*(1+randint(1,10)/100.0)
```

```
Structures = (11.0*8.0)*15.0*(1+randint(1,10)/100.0)

TPS = (9.0*8.0)*15.0*(1+randint(1,10)/100.0)

Tracking = (10.0*8.0)*15.0*(1+randint(1,10)/100.0)

Subsystems = [Avionics, Communications, Crew, Electrical_Wiring, Engines,
Environmental,\

    Flight_Controls, Hydraulics, Landing_Recovery, Navigation, Pneumatics,
Propellant_Management,\

    Software, Structures, TPS, Tracking]

MMH = 0

flight_rate = 0

people =
[randint(1,max_ppl),randint(1,max_ppl),randint(1,max_ppl),randint(1,max_ppl),\

    randint(1,max_ppl),randint(1,max_ppl),randint(1,max_ppl),randint(1,max_ppl),\

    randint(1,max_ppl),randint(1,max_ppl),randint(1,max_ppl),randint(1,max_ppl),\

    randint(1,max_ppl),randint(1,max_ppl),randint(1,max_ppl),randint(1,max_ppl)]

ppl_this_run =[]

vehicles = []

first = True
```

```
time_into_year = []

MMH_vehicles = [0,0]

for q in range(len(Subsystems)):

    ppl_this_run.append(people[q])

    hit_max = False

    subsystem = []

    if people[q] <= 15:

        hit_max = True

    for r in range(num_vehicles):

        if people[q] >= 15:

            available = 15

            people[q] -= available

            subsystem.append(Subsystems[q]/available)

            MMH_vehicles[r] += Subsystems[q]/available

        elif people[q] == 0:

            subsystem.append(Subsystems[q]/last_available)

            MMH_vehicles[r] += Subsystems[q]/available
```

```
            else:
                available = people[q]
                last_available = available
                people[q] -= available
                subsystem.append(Subsystems[q]/available)
                MMH_vehicles[r] += Subsystems[q]/available
        if hit_max:
            time_into_year.append(sum(subsystem))
        else:
            time_into_year.append(max(subsystem))
    all_vehicles_launched = max(time_into_year)
    MMH_per_veh = sum(MMH_vehicles)/len(MMH_vehicles)
    if sum(ppl_this_run) <= sum(distribution) and all_vehicles_launched <=
all_vehicles_launched_best:
        all_vehicles_launched_best = all_vehicles_launched
        distribution = ppl_this_run
        MMH_per_veh_best = MMH_per_veh
        best_run = run
```

```
        print "New best found! - run #" + str(run)
        print "New best launch rate of " + str(round(2080.0/all_vehicles_launched_best,1)) + \
            " flights/year with a total of " + str(sum(distribution)) + " technicians."
    run += 1
    if run - best_run > 1000000:
        print "Completed " + str(run) + " runs before completion."
        print "Optimum found on run #" + str(best_run)
        break
print distribution
print all_vehicles_launched_best
print MMH_per_veh
```

REFERENCES

[1] “A Practical Guide to Support Vector Classification,” Hsu, C.W., Chang, C.C., Lin, C.J. (2010). National Taiwan University, Taipei 106, Taiwan.

[2] “An Integrated Modeling Tool for Sustainable Space Exploration,” Shull, S.A., Gralla, E.L., Armar, N., de Weck, O. (2006). 57th International Astronautical Congress 2006. IAC-06-D3.3.1.

[3] “Analysis of Shuttle Orbiter Reliability and Maintainability Data for Conceptual Studies” Morris, W.D., White, N.H. 1996. NASA Langley Research Center. AIAA 96-4245.

[4] “Cost Estimates of Near-Term, Fully-Reusable Space Access Systems,” Snead, J.M. Retrieved from: http://mikesnead.net/resources/spacefaring/tech_paper_cost_estimates_of_near-term_fully-reusable_space_access_systems.pdf

[5] “Defining Support Requirements During Conceptual Design of Reusable Launch Vehicles” Morris, W.D., White, N.H., Davis, W.T. (1995). NASA Langley Research Center. AIAA 95-3619.

[6] “Estimating Logistics Support of Reusable Launch Vehicles During Conceptual Design” Morris, W.D., White, N.H., Davis, W.T., Ebeling, C. E.. NASA Langley Research Center. 32nd Annual International Logistics Conference and Exposition. August 5-7, 1997.

[7] “Hybrid Launch System Vehicles, Ground Facilities and Operations Integration,” Rooney, B.D. (2006). AIAA 2006-7231.

[8] “Integrations of Cost Modeling and Business Simulation into Conceptual Launch Vehicle Design,” Lee, H., Olds, J.R. AIAA 97-3911

[9] "Iterated Grid Search Algorithm on Unimodal Criteria," Kim, J. (1997). Thesis: Virginia Polytechnic Institute and State University.

[10] "Logistics and Supply Chain Management – A Space Operations Enabler," Evans, W.A. (2006). AIAA 2006-5852.

[11] "Logistics Information Systems for Human Space Exploration: State of the Art and Emerging Technologies," Shull, S.A., Gralla, E.L., Silver, M., de Weck, O. (2006). AIAA 2006-5733.

[12] "Logistics Manpower Requirements for Ground Support of a Reusable Military Launch Vehicle," Michalski, S.C., Johnson, A.W. (2007). AIAA 2007-6273.

[13] "Logistics: An Integral Part of Cost-Efficient Space Operations" Montgomery, A.D. Space Mission Operations and Ground Data Systems - SpaceOps '96, Proceedings of the Fourth International Symposium held 16-20 September 1996 in Munich, Germany. Edited by T.-D. Guyenne. ESA SP-394. Paris: European Space Agency, 1996., p.1322

[14] "Maximum-Likelihood DOA Estimation by Data-Supported Grid Search," Stoica, P., Gershman, A.B.. IEEE Signal Processing Letters, Vol. 6, No. 10, October 1999.

[15] "Optimization of Operations Resources Via Discrete Event Simulation Modeling," B. Joshi, D. Morris, N. White, R. Unal. AIAA Paper 96-4181.

[16] "Space Shuttle Operations and Infrastructure: A Systems Analysis of Design Root Causes and Effects." McCleskey, C. M. April 2005. NASA/TP -2005-211519.

[17] "The Case for a Centralized Repair Depot for Space Shuttle" Enlow, R.D.. Kennedy Space Center. Space and society - Progress and promise; Proceedings of the Twenty-second Space Congress, Cocoa Beach, FL, April 23-26, 1985 (A86-34951 15-12). Cape Canaveral, FL, Canaveral Council of Technical Societies, 1985, p. 1-15 to 1-22.

[18] Anderson, J. D. Jr. (1999). Aircraft Performance and Design. McGraw-Hill. ISBN 978-0-07-001971-3.

[19] Carter, M. W., Price, C. C. (2001). Operations Research: A Practical Introduction. CRC Press, Boca Raton. ISBN 0-8493-2256-1

[20] Cates, G.R., Steele, M.J., Mollasghesemi, M., Rabadi, G., "Modeling the Space Shuttle," in E. Yucesan et al., (eds.), Proc. 2002 Winter Simulation Conference.

[21] Coad, P., Yourdon, E. (1990). Object Oriented Analysis. Prentice Hall. Englewood Cliffs, New Jersey. ISBN 0-13-629122-8.

[22] Department of Defense Documentation of Verification, Validation & Accreditation (VV&A) for Models and Simulations. Missile Defense Agency. 2008

[23] Dieter, G. E., Schmidt, Linda C. (2009). Engineering Design 4th Edition. McGraw-Hill Education. ISBN 978-007-126341-2.

[24] Ecker, J. G., Kupferschmid, M. (1998). Introduction to Operations Research. John Wiley & Sons, New York. ISBN 0-471-88445-6

[25] Electronic Reliability Design Handbook. Military Handbook. MIK-HDBK-338B. 1 October 1998.

[26] Federal Aviation Administration (2005), Guide to Commercial Reusable Launch Vehicle Operations and Maintenance. Office of Commercial Space Transportation, 800 Independence Avenue, Room 331, Washington, DC 20591

[27] Fishman, G. S. (1978). Principles of Discrete Event Simulation. John Wiley & Sons. New York, New York. ISBN 0-471-04395-8.

[28] Fishman, G.S. (2001), Discrete Event Simulation: Modeling, Programming, and Analysis. New York: Springer. ISBN 0-387-95160-1.

[29] Fishman, G. S. (1995). Monte Carlo: Concepts, Algorithms, and Applications. New York: Springer. ISBN 038794527X.

[30] Frank, M. V. (2008). Choosing Safety: A Guide To Using Probabilistic Risk Assessment and Decision Analysis in Complex, High-Consequence Systems. Resources for the Future. Washington D.C. ISB 978-1-933115-53-5.

[31] H. Murat Gunaydin, P. D. (1998). "The Delphi Method." Retrieved June 2012, from http://www.iyte.edu.tr/~muratgunaydin/delphi.htm.

[32] H.H. Koelle, Saturn System Study, Redstone Arsenal, AL, November 13, 1959, pp. 4–5, 21.

[33] JMP Website: JMP® for Design of Experiments: http://www.jmp.com/applications/doe/

[34] Kelly, M. J., Charania, A.C., Olds, J. R., DePasquale, D. (2009). Discrete Event Simulation of Reusable Launch Vehicle Development, Acquisition, and Ground Operations. AIAA 2009-6484.

[35] Kennedy Space Center, “Space Transportation Ground Processing Operations Modeling and Analysis: A Review of Tools and Techniques,” Edgar Zapata, Systems Engineering Office, NASA KSC, June 2003, p.6

[36] L.V. Kantorovich: A new method of solving some classes of extremal problems, Doklady Akad Sci USSR, 28, 1940, 211-214.

[37] Mean Time to Repair Predictions. NASA Technique AT-2. Public Lessons Learned Entry 0840. Retrieved from http://www.nasa.gov/offices/oce/llis/0840.html

[38] Michalski, S.C. (2007) Determining Logistics Ground Support Manpower Requirements for a Reusable Military Launch Vehicle. Thesis: Air Force Institute of Technology. AFIT/GLM/ENS/07-09.

[39] Molina, C.A. (2009) Reusable Launch Vehicle Design Implications for Regeneration Time. Thesis: Air Force Institute of Technology. AFIT/GLM/ENS/09-08.

[40] Morris, W.D., White, N.H., Davis, W.T. (1995). Defining Support Requirements During Conceptual Design of Reusable Launch Vehicles. AIAA 95-3619.

[41] NASA SP-407, Space Shuttle, ca. 1976; p. 4

[42] Pei Hsia, Chih-tung Hsu, David C. Kung. "Brooks' Law Revisited: A System Dynamics Approach," compsac, p. 370, Twenty-Third Annual International Computer Software and Applications Conference, 1999.

[43] Pope, J.T. III (2006). Discrete Event Simulation Model of the Ground Maintenance Operations Cycle of a Reusable Launch Vehicle. Thesis: Air Force Institute of Technology. AFIT/GLM/ENS/06-14.

[44] Rockwell Automation Website: Arena Simulation Software. Retrieved from: http://www.arenasimulation.com/Arena_Home.aspx

[45] Rooney, B. D., Hartong, A. (2004). A Discrete-Event Simulation of Turnaround Time and Manpower of Military RLVs. AIAA 2004-6111.

[46] Santovincenzo, A., L. Innocenti and M.O. van Pelt. "The Socrates Vehicle – ESTEC CDF Design". *AIAA 13th International Space Planes and Hypersonic Systems and Technologies.* Capua, Italy. June 2005. AIAA 2005-3341.

[47] Schlagheck, R. A. and J. K. Byers (1971). Simulating the Operations of the Reusable Shuttle Space Vehicle. 1971 Summer Computer Simulation Conference.

[48] Sengupta, J. K. (1982). Decision Models in Stochastic Programming: Operational Methods of Decision Making Under Uncertainty. Series Volume 7. Elsevier Science Publishing Co., Inc. New York, New York. ISBN 0-444-00667-2.

[49] Servidio, J.A. (2008). "Process Time Refinement for Reusable Launch Vehicle Regeneration Modeling." Thesis: Air Force Institute of Technology, Department of Operational Sciences. AFIT/GLM/ENS/08-11.

[50] Sierra Nevada Corp. Website: Space Exploration Systems. Retrieved from: http://sncspace.com/space_exploration.php

[51] SimPy Simulation Package Website. Retrieved from: http://simpy.sourceforge.net/

[52] Spaceport Synergy Team, "Catalog of Spaceport Elements with Functional Definition," NASA Highly Reusable Space Transportation (HRST) Study, October 1997.

[53] SpaceX Website: Interview with Elon Musk. Retrieved from: http://www.spacex.com/npc-luncheon-elon-musk.php

[54] Steele, M. J., Cates, G., Mollaghesimi, M., Rabadi, G. (2002), Generic Simulation Models of Reusable Launch Vehicles. Proceedings of the 2002 Winter Simulation Conference. Page 747-753.

[55] Stiegelmeier, B.A. (2006) A Discrete-Event Simulation Model for Evaluating Air Force Reusable Military Launvh Vehicle Prelaunch Operations. Thesis: Air Force Institute of Technology. AFIT/GLM/ENS/06-16.

[56] Taha, H. A. (1997). Operations Research: An Introduction 6th Edition. Prentice Hall, New Jersey. ISBN 0-13-272915-6.

[57] Ventana Systems, Inc. Website: Vensim Software. Retrieved from: http://www.vensim.com/software.html

[58] Virgin Galactic Website: Company History Overview. Retrieved from: http://www.virgingalactic.com/overview/

[59] Wainer, G. A. (2009). Discrete Event Modeling & Simulation: A Practitioner's Approach. CRC Press Boca Raton FL. ISBN 978-1-4200-5336-4.

[60] Zhigljavsky, A., Zilinskas, A. (2008) Stochastic Global Optimization. Springer Science+Business Media, LLC. New York, New York. ISBN 978-0-387-74022-5.

[61] Zimmermann, A. (2008). Stochastic Discrete Event Systems: Modeling, Evaluation, Applications. Springer Berlin. ISBN 978-3-540-74172-5.