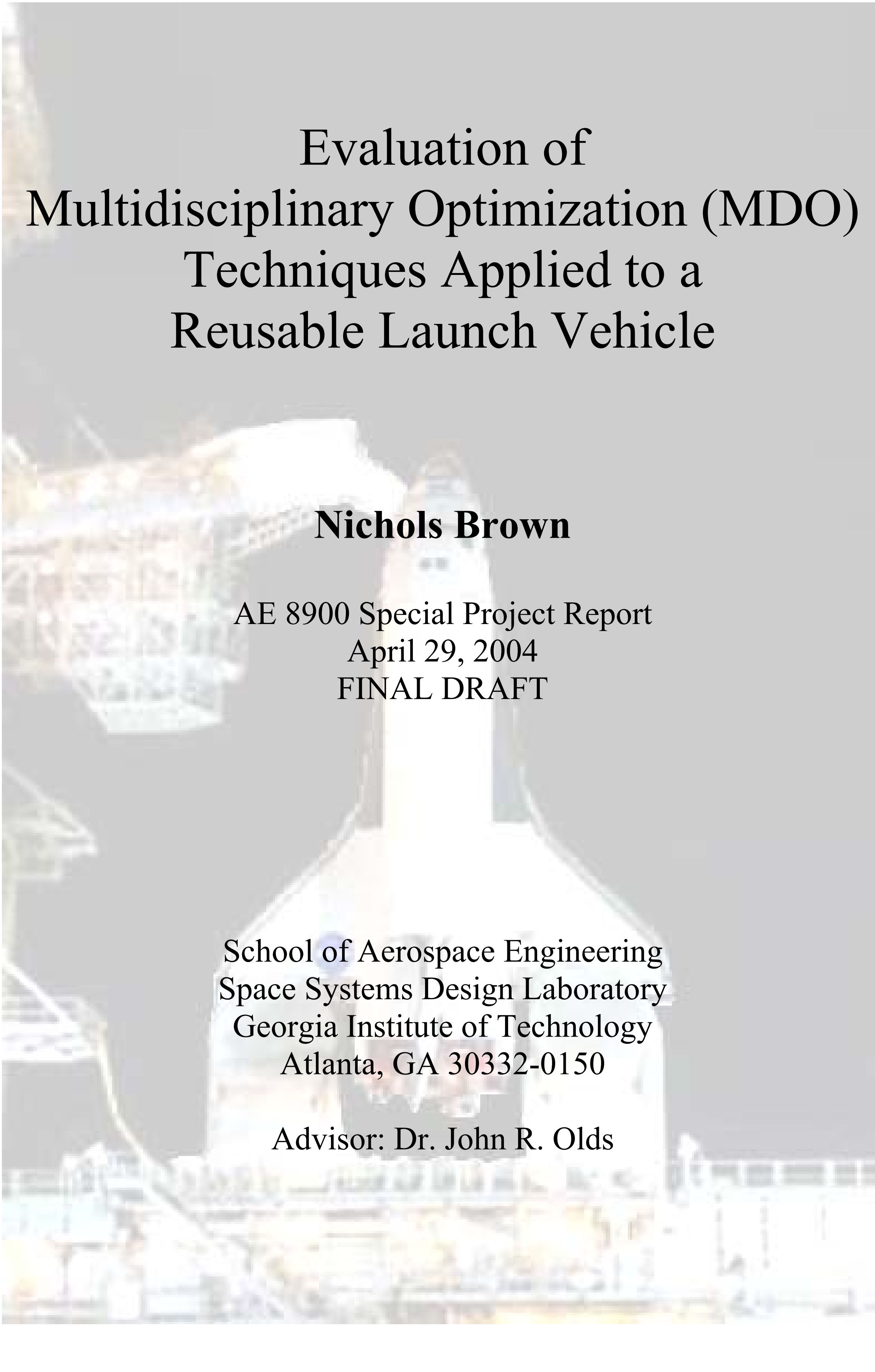

Evaluation of Multidisciplinary Optimization (MDO) Techniques Applied to a Reusable Launch Vehicle

Nichols Brown

AE 8900 Special Project Report
April 29, 2004
FINAL DRAFT

School of Aerospace Engineering
Space Systems Design Laboratory
Georgia Institute of Technology
Atlanta, GA 30332-0150

Advisor: Dr. John R. Olds

Evaluation of Multidisciplinary Optimization (MDO) Techniques Applied to a Reusable Launch Vehicle

Nichols Brown

AE 8900 Special Project Report
April 29, 2004

School of Aerospace Engineering
Space Systems Design Laboratory
Georgia Institute of Technology
Atlanta, GA 30332-0150

Advisor: Dr. John R. Olds

Abstract

Optimization of complex engineering systems has always been an integral part of design. Due to the size and complexity of aerospace systems the design of a whole system is broken down into multiple disciplines. Traditionally these disciplines have developed local design tools and computer codes (legacy codes) allowing them to perform optimization with respect to some aspect of their local discipline. Unfortunately, this approach can produce sub-optimal systems as the disciplines are not optimizing with respect to a consistent global objective. Multidisciplinary design optimization (MDO) techniques have been developed to allow for multidisciplinary systems to reach a global optimum. The industry accepted All-at-Once (AAO) technique has practical limitations and is confined to only small, conceptual level problems.

New multi-level MDO techniques have been proposed which may allow for the global optimization of the large, complex systems involved in higher levels of design. Three of the most promising multi-level MDO techniques, Bi-Level Integrated System Synthesis (BLISS), Collaborative Optimization (CO) and Modified Collaborative Optimization (MCO) are applied, evaluated and compared in this study.

The techniques were evaluated by applying them to the optimization of a next generation Reusable Launch Vehicle (RLV). The RLV model was composed of three loosely coupled disciplines, Propulsion, Performance, and Weights & Sizing, composed of stand-alone, legacy codes not originally intended for use in a collaborative environment.

Results from the multi-level MDO techniques will be verified through the use of the AAO approach and their benefits measured against the traditional approach where the multiple disciplines are converged using the fixed point iteration (FPI) process.

All the techniques applied will be compared against each other and rated qualitatively on such metrics as formulation and implementation difficulty, optimization deftness and convergence errors.

Table of Contents

Table of Figures

Table of Tables

Acronyms and Symbols

A	Area
AAO	All-at-Once
BLISS	Bi-Level Integrated System Synthesis
BTU	British Thermal Unit
c*	Characteristic exhaust velocity
CA	Disciplinary Contributing Analysis
c_F	Thrust coefficient
CO	Collaborative Optimization
DOE	Design of Experiments
DOF	Degree of Freedom
DSM	Design Structure Matrix
ETO	Earth to Orbit
FPI	Fixed Point Iteration
ft	Feet
g	Inequality constraint
h	Enthalpy / Equality constraint / Altitude
HRST	Highly Reusable Space Transportation
INC	Incomplete
Isp	Specific impulse
ISS	International Space Station
klb	Kilo pounds
KSC	Kennedy Space Center
LH2	Liquid Hydrogen
LOX	Liquid Oxygen
MCO	Modified Collaborative Optimization
MDA	Multidisciplinary Analysis
MDO	Multidisciplinary Design Optimization
MER	Mass Estimating Relationship
MoFD	Method of Feasible Directions
MR	Mass Ratio ($=W_{gross}/W_{insertion}$)
nmi	Nautical mile
OBD	Optimizer Based Decomposition
OMS	Orbital Maneuvering System
P	Power
p	Pressure
Perf	Performance

POST	Program to Optimize Simulated Trajectories
Prop	Propulsion
psia	Pounds per square inch absolute
r	Fuel to oxidizer ratio (propellant mixture ratio)
RCS	Reaction Control System
REDTOP	Rocket Engine Design Tool for Optimal Performance
RLV	Reusable Launch Vehicle
RSE	Response Surface Equation
RSM	Response Surface Model
S	Wing area
s	Seconds
SAND	Simultaneous Analysis and Design
SLP	Sequential Linear Programming
SQP	Sequential Quadratic Programming
SSTO	Single Stage to Orbit
T	Thrust
TRF	Technology Reduction Factor
W	Weight
w	Weighting factor (BLISS only)
W&S	Weights & Sizing
X	Design or input variable
Y	Output or behavior variable
ΔV	Change in velocity
ε	Nozzle expansion ratio
$\dot{\theta}$	Pitch angle rate
Φ	Design objective

Subscripts

c	combustor
e	exit
eng	engine
loc	local
o	optimized
ref	reference
req	required
sh	shared input in multiple CA's but not calculated by any (for BLISS)
SL	sea-level
sys	system

t	throat
vac	vacuum
veh	vehicle

Superscripts

*	Output passed to a CA (for BLISS)
^	Output from a CA to system (for BLISS)
pf	Performance (for CO and MCO)
pp	Propulsion (for CO and MCO)
t	Target from system optimizer (for CO and MCO)
ws	Weights & Sizing (for CO and MCO)

1 Introduction

Optimization of complex engineering systems has always been an integral part of design. Man has never created anything which he then didn't wonder how he could make better. This is true in the aerospace industry dating back to wind tunnel studies conducted by the Wright bothers to study wing shapes. Originally those that created aerospace vehicles were responsible for every aspect from wing shape to propulsion. As the size and complexity of aerospace systems grew, though, it became apparent that the design of such enormously complex problems would have to be broken down into disciplines with groups concentrating only on their own part of the whole.

While breaking apart the overall problem into different contributing analyses (CA's) made it humanly possible to design inhumanly complex systems, the ability for all designers to see how their specific changes would affect the overall goodness of the whole was lost. Communicating between disciplines became increasingly difficult and thus each discipline developed metrics to which to optimize their own individual part of the total system or vehicle. Unfortunately, no discipline is an island as every discipline will affect another and vice versa. If they don't agree as to what the coupling variable values should be across disciplines then the system is not converged or valid.

To solve this multidisciplinary analysis (MDA) problem engineers have traditionally used an iterative process known as Fixed Point Iteration (FPI). FPI, though, is just a method to converge the MDA problem and does not perform overall system optimization or the configuring of the whole system such as to reach an optimum according to one global design objective. The conventional design approach is to try several configurations and solve the MDA convergence problem for each using FPI. The best configuration according to the global variable is then selected.

Multidisciplinary optimization (MDO) is when, instead of just arriving at the best or a good vehicle via multiple trials, a methodology is applied which actively changes the design variables to find the configuration that produces the optimal system.

All-at-Once (AAO) the most basic and accepted MDO method is insufficient as it cannot be applied to large engineering problems. Recently multi-level MDO algorithms have been proposed with the expectation that they might efficiently solve large, complex

MDO problems. Their purported benefits are still under inspection as is the question of which one will produce the greatest benefits with the least amount of effort or cost.

Three multi-level MDO methods are evaluated in this study: Collaborative Optimization (CO) [1], its derivative Modified Collaborative Optimization (MCO)[2], and Bi-Level Integrated System Synthesis (BLISS)[3] which has multiple derivatives most notably BLISS-2000[4].

2 Study Objectives

Three main goals were explored during the course of this study:

1) Determine the benefits of MDO versus using multiple trials of iterative optimizers using FPI convergence.
2) Create a realistic test problem which will add to a growing field of research trying to evaluate novel MDO techniques: CO, MCO and BLISS.
3) Allow for across the board comparison of the MDO techniques by using the statistical method of blocking to remove external variability when comparing between techniques.

2.1 Benefits of MDO

The first goal is to understand the overall benefit of using an MDO method to optimize the RLV test problem versus a few trials converged using a fixed point iteration process (FPI). For this purpose several FPI models of the next generation RLV were created in order to offer some insight as to what is the best vehicle that could be designed without using MDO. The results achieved using FPI will be compared to the techniques mentioned in order to better understand the benefits derived from MDO.

FPI is not an MDO method it is merely a way to converge a multidisciplinary analysis (MDA) process. It does not perform any global optimization to find the optimum system configuration. In discipline a designer will try several configurations and converges each one using FPI. Then one of the configurations is deemed the best and selected for further study. Thus, FPI usually results in the best configuration for the options tested, but does guarantee that the true optimum will be found.

The argument for using FPI to test a limited number of configurations versus applying an MDO process are practical or “real world” in nature. FPI has been the method of choice in the aerospace industry. Thus there are already legacy tools and design practices developed by many individuals over time. Changing this structure would require a large initial investment. Also, the experts performing each analysis have been trained and are experienced in solving the problem as it is currently formulated for FPI.

Application of other methods may require new training and will take time to gain full acceptance.

Eight RLV analysis models were tested and converged using FPI during this study. The vehicle configuration producing the lowest test problem global objective, W_{dry}, was selected as the best FPI configuration. For the application any MDO to be worthwhile, the method must show improvement in the global objective or produce project time savings in order to offset its application cost.

2.2 Authenticity of Test Problem

It is intended that the test problem, the optimization of a next generation RLV, have enough realism that it can add to a growing body of work attempting to evaluate some of the most novel and promising MDO algorithms: CO, MCO and BLISS.[5,6,7,8] These MDO techniques were "crafted" as opposed to "rigorously derived."[9] They lack a general mathematical proof showing for which MDO problems they are suited. Thus, the algorithms require that they be validated via a statistically significant number of test cases of realistic, complex system applications.[9]

To ensure realism in the test problem, legacy tools were employed which are the same or very similar to those used in industry. Also, one of the tools selected, POST, while being the industry standard code is known to be very troublesome to work with. This will add the realism of the problem as in the "real world" one often has to work with the tools available and cannot just select tools, such as using all Microsoft spreadsheets, that are usually well behaved.

There are some practical obstacles to the goal of reaching a high level of realism. First, since it is proposed that the AAO method be used to validate the newer, unproven MDO techniques. This limits test problems to those that can still be handled by AAO which does not scale well with the system size and complexity. Secondly, it is intended that the same developer create all the models applying each technique. The developer though has external constraints limiting the time he can spend on this project; if the problem is too large then the work may never be completed to some degree of satisfaction.

It is the author's belief that that a convincingly high degree of realism was achieved and this study does produce statistically significant data.

2.3 Comparison between CO, MCO & BLISS

There are some deficiencies in the currently accepted design methods used in industry, namely discipline optimization with FPI does not do system-level optimization and AAO cannot be applied to large, complex engineering problems. New multi-level MDO techniques, of which CO, MCO and BLISS are three of the most promising, are attempting to overcome some of these deficiencies. The work presented intends to some add insight as to which of the most novel techniques, CO, MCO or BLISS, showed the most promise when applied to the RLV test problem.

It is difficult to draw any conclusions between, CO, MCO and BLISS, using the current literature available.[5,6,7,8,11] Test applications that have produced varying degrees of success, but it is difficult to determine if this is due to differences in the algorithms or other factors. The statistical reasons making it difficult to compare between competing MDO algorithms are:

1) Variability in the level of success with which each of the three competing MDO techniques has been applied can be attributed to external variances.
2) There are still too few data points to draw any statistically significant conclusions.

In order to address the external variance issue, this study will use the statistical practice of "blocking" to "help eliminate the variation due to extraneous sources so that a more sensitive comparison can be made"[10] (page 567) between CO, MCO and BLISS. The blocking effect helps to "eliminate variation due to blocking variables" which in this case are developer or user capacity and difficulty of the test problem selected. Blocking is achieved in this study by having the same developer create all of the RLV models for each of the MDO techniques in question and by applying all of them to the same test problem.

While this study will not settle the question of which is better, CO, MCO or BLISS, as it is but one data point (see item 2 above), it is one of the very first attempts at addressing the external variance problem when trying to compare the benefit, validity and implementation cost of the new multi-level MDO techniques. To date the author is not aware of any other attempt.

3 Multi-level MDO Techniques

This section is an introduction to multi-level MDO techniques and discusses some of the possible benefits and disadvantages common to all the multi-level techniques applies in this study, CO, MCO and BLISS.

3.1 MDO Background

Aerospace system analysis and design is usually broken down into multiple disciplines due to their enormous complexity. This impedes MDO as the local-level disciplines lose knowledge of how their local design affects the system as a whole. AAO can be used to solve the MDO problem, but it takes away all design responsibilities from the discipline experts and AAO does not perform well in higher fidelity applications with a lot of design variables (See 8 AAO: All-at-Once pg 36). Multi-level MDO techniques attempt to correct this problem by allowing some form of local-level design optimization while adding a system-level optimizer guide the process to a globally optimized solution.

One of the greatest impediments in the acceptance of multi-level MDO techniques is the fact that they have to date been "crafted" as opposed to "rigorously derived."[9] They lack a general mathematical proof showing for which MDO problems they are suitable. Thus, the algorithms require that they be validated via a statistically significant number of test cases of realistic, complex system applications.[9] Program managers or designers are reluctant to implement the new techniques to a "real world", industry application as they cannot be confident that the multi-level MDO algorithm will indeed arrive at the global optimum. This creates a Catch-22 scenario for multi-level MDO techniques as the only way to gain acceptance is to show success in a statistically significant number of "real world", industry-sized test problems; paradoxically not many application attempts are made as program managers are not willing to risk their projects on an unproven method. This contradiction means that acceptance of any multi-level MDO technique may be a slow one.

If multi-level MDO techniques can be proven to work efficiently for a range of engineering problems, then they would provide several advantages over the presently accepted MDA and MDO methods, FPI and AAO.

3.2 MDO Possible Benefits

First, two-level MDO algorithms closely resemble the conventional work structure of the aerospace industry. In industry the local-level design is performed by discipline experts who have a high degree of design freedom within their own discipline. The system-level optimization is analogous to a chief engineer or program manager who is in charge of making sure that the disciplines work with each other to solve the overall system problem. At the same time the disciplines get to keep the local discipline design roles they are accustomed to. Thus it may be easier to implement and gain acceptance of a two-level MDO architecture than the AAO technique which removes all design responsibilities from the disciplines.

Next, it is expected that multi-level MDO techniques will perform better as the scale of the MDO problem grows. AAO is severely limited by the optimization deftness of the system-level optimizer, which is charged with the exceedingly difficult task of optimizing all design variables simultaneously subject to a series of constraints. Two-level MDO techniques split up the design and optimization responsibilities so that the system-level and local-level sub-problems share the load. This decreases the probability that one group or optimizer will fail because the problem given was too difficult to solve.

Lastly, many multi-level MDO techniques including the three investigated in this study, CO, MCO and BLISS, can potentially allow for parallel or simultaneous execution of the local-level disciplines. In parallel execution, the system-level optimizer will provide all the local-level disciplines a set of inputs which they can use to perform the local-level optimization. With this information disciplines can perform their analysis at the same time. This contrasts to a sequential execution where each discipline must go in order so that it can pass its outputs to other disciplines down the line. Parallel execution, if efficient, could result in large time savings for the entire optimization process. This savings is expected to be most significant for large problems where several of the disciplines involved take a lot of time and effort to execute their local-level sub-problem design and optimization.

3.3 MDO Possible Drawbacks

While novel multi-level MDO techniques may offer some significant benefits they often have some drawbacks which may offset any potential gains over the established FPI and AAO methods.

First, in order to perform global, system-level optimization while still allowing local-level optimization, new design variables are introduced to the problem. These new variables must be added to either the local or system-level optimizer. This increases the size and complexity of one or more of the optimization sub-problems.

Second, the system-level problem complexity may grow rapidly if a lot of coupling variables exist between the disciplines. This is because coupling variables are usually controlled or coordinated by the system optimizer.

Third, the need for coordinating the local-level sub-problem optimizations for the betterment of a global objective often results in changing the way the each discipline has traditionally been analyzed. The local-level objective may need to be changed to composite objective composed in some form of two or more variables or outputs. Also, this objective may keep changing with each system-level iteration. This can cause problems especially in “real world” applications where the computational analysis and optimization of the local-level sub-problems is performed using legacy codes with internal tolerances. If the source code of these legacy tools needs to be altered that may require a considerable amount of time. Also whenever codes are altered it increases the probability of encountering problems due to human errors. It would be ideal for any MDO technique to be able to use the legacy tools originally written for the FPI process without altering them.

Finally, the complexity and novel nature of multi-level MDO methods means that the successful application of these techniques is more susceptible to user inexperience. Multi-level MDO techniques are complex and the problem formulation and decomposition require careful planning. They are not as straight forward as FPI or AAO and may be very sensitive to the conditioning of optimization parameters such as optimization methods used (direct vs. indirect methods, SQP vs. MoFD, etc.), numerical gradient estimation methods and steps, normalization methods, and allowable tolerances.[8] A range of problems from model developer inexperience to difficulties when trying to take numerical gradients may cause unsuccessful results. There have been cases where attempts at MDO applications, including in RLV test problems[11], have been fruitless.

In an attempt to differentiate external variance in application results to those caused by differences in the MDO algorithms, this study uses the statistical method of blocking to remove unwanted external variance (see page 5).

3.4 MDO Conclusions

No multi-level MDO technique has been embraced by industry. Without a mathematical proof the only way to gain acceptance is through a statistically significant number of test applications. Paradoxically most are unwilling to bank their projects on an unproven method and thus MDO applications have been few and mostly relegated to academia. In order to determine which one(s) of the promising multi-level MDO techniques may best make the transition from theory to practice, a number of realistic test studies must be performed that use blocking effects to isolate the differences between techniques. Without blocking it is difficult to determine if an unsuccessful attempt at applying a MDO technique is due to the algorithm itself or other unrelated issues.

4 Test Problem

In order to evaluate the MDO techniques a test problem had to be selected on which to apply them. For the methodology evaluation goals of this study, though, the test problem is just a vessel. It is not of critical issue what problem was selected, just as long as it a realistic problem that resembles one that may be used in "real world" applications.

The test problem selected is the optimization of a next generation single-stage-to-orbit (SSTO) earth-to-orbit (ETO) RLV. Launching from Kennedy Space Center (KSC), the RLV must be capable of delivering a 25 klb payload to the International Space Station (orbiting at 220 nmi x 220 nmi x 51.6° inclination). To accomplish this goal a base vehicle configuration, ACRE-92, will be optimized via the use of three disciplines, Propulsion, Performance and Weights and Sizing.

The goal of this study is not to answer what a next generation RLV might look like or how much it might weigh but instead to evaluate how well the MDO techniques performed when applied to the problem.

4.1 Base Launch Vehicle Configuration

The ACRE-92 RLV[12] is an HRST-class SSTO designed and documented by SpaceWorks Engineering, Inc.[15] It is based upon a NASA Langley concept dubbed WB-003 Single Stage Shuttle.[13] Employing five LOX/LH2 main engines the baseline configuration is able of transporting about 25 klb, 70% of the payload capability of the current Space Shuttle (34.4 klb)[14], to the ISS. The five main propulsion engines will insert the RLV into ISS insertion orbit (50 nmi x 100 n mi x 51.6° inclination) and then let the orbital maneuvering system (OMS) raise the orbit to a circular orbit at 220 nmi altitude.

In order to increase the realism of this study, adjustments were made during the vehicle analysis to account for technological improvements expected for a next generation vehicle (see page 16).

The base vehicle characteristics and configuration are shown below.

Table 1: Concept Description ACRE-92.

Item	Characteristics
Configuration	Wing-body configuration Vertical take-off, un-powered horizontal landing
Reference Mission	~25 klb to ISS 15 ft dia. X 25 ft payload bay Cargo delivery or passanger delivery using cargo pod
Flight Performance	Human rated; can fly unpiloted Crew survivable abort capability No engine out capability to make mission Cross range capability of several hundred nmi

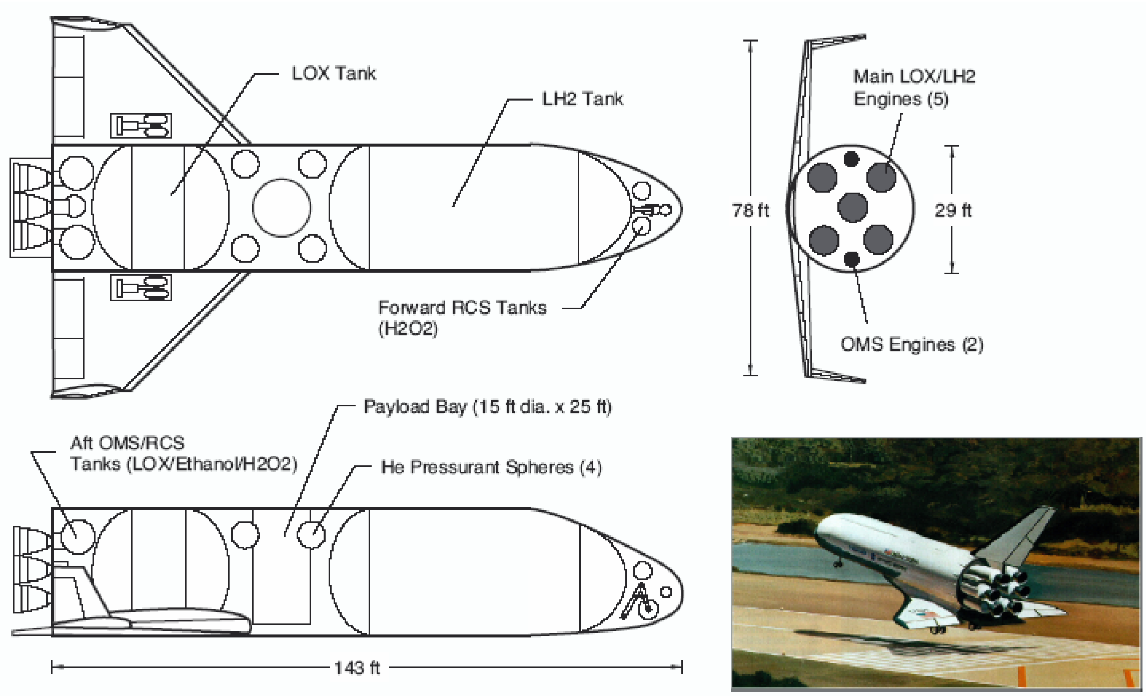

Figure 1: Vehicle Configuration ACRE-92.

The base vehicle was photographically scaled to meet sizing requirements of the test problem and its aerodynamic data served for the models created in this project.

5 Computational Tools

This section describes the legacy codes and other computational tools used for all MDO and MDA technique applications.

5.1 Propulsion Tool

Rocket Engine Design Tool for Optimal Performance (REDTOP) was used to analyze and predict the rocket engine propulsion system. Developed by SpaceWorks Engineering, Inc.[15] the code allows quick analysis of propulsion systems and is suitable for conceptual level design.

REDTOP models the rocket engine by first determining the chemical reactions occurring in the combustion chamber and then analyzing the expansion of the hot gasses as they travel through a convergent-divergent nozzle. The combustion is modeled adiabatically and at constant pressure. Additional accuracy in the prediction of the engine performance is reached through a built-in engine efficiency database. It will make performance corrections due to engine cycle type, nozzle flow losses, degree of reaction, and combustor efficiency.

To execute REDTOP the user must specify the propellant characteristics, chamber pressure, nozzle expansion ratio, mixture ratio and sizing criteria. Sizing could be executed to meet a required thrust at a given ambient pressure, desired mass flow rate, or to meet a specific engine throat area. REDTOP will then output general performance metrics, efficiencies used, exhaust characteristics, and species mole fractions at inlet, throat and exhaust.

A high-fidelity estimate of the engine thrust to weight (T/W_{eng}) for the propulsion system is not provided by REDTOP, but is required as an input when estimating the vehicle weight. Thus a low-fidelity estimate of T/W_{eng} was calculated based on historical data[16] of rocket engine power to weight ratio (P/W). This calculation is as follows:

$$P = \dot{m}(h_c - h_e)$$ **Equation 1: Engine Power**

$$W_{eng} = P/k$$ **Equation 2: Engine Weight**

$$T/W_{eng} = T_{SL}/W_{eng}$$ **Equation 3: Sea Level Thrust to Engine Weight**

The parameter "k" is a sizing parameter that varies with technology level. The historical average value of "k" based on data from 28 rocket engines is k = 520 BTU/s/lbf.[16] This number is therefore representative of state of the art technologies. A value of k = 600 BTU/s/lbf was used for this study, this represents the use of advanced technologies in the engines for a next generation RLV.

The propulsion inputs and outputs used for the course of this study are shown below.

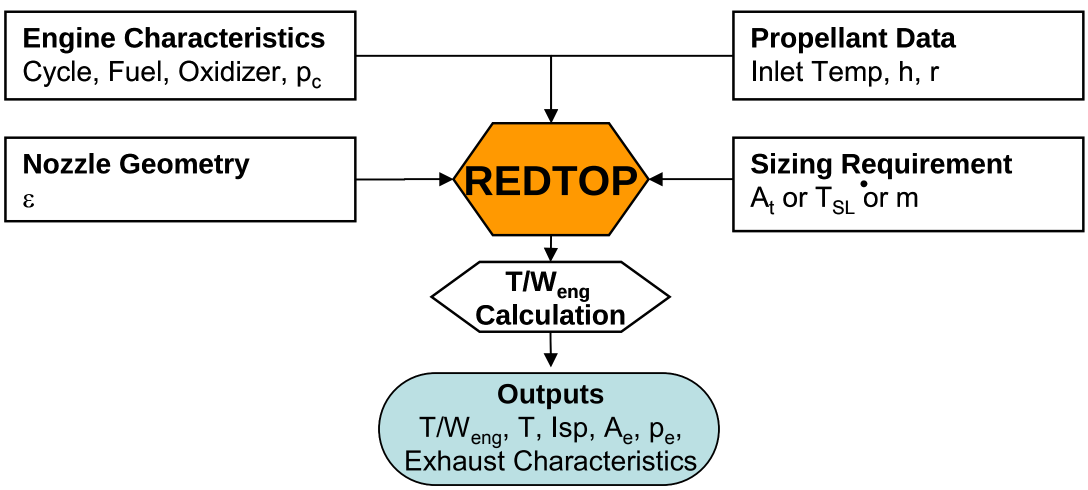

Figure 2: REDTOP Inputs and Outputs.

5.2 Performance Tool

Program to Optimize Simulated Trajectories (POST)[17] was used to simulate the trajectories of the RLV. Originally developed in 1970 by Lockheed Martin as a Space Shuttle Trajectory Optimization Program, POST is one of the most commonly used codes for trajectory analysis and optimization. It is a multi-phase simulation program which numerically integrates the equations of motion and provides the capability to target and optimize point mass trajectories for generalized vehicles and planets.

POST was selected not only due to the fact that it is often the tool of choice for trajectory optimization throughout the aerospace industry, but also because it is a fastidious code which will often present trouble during analysis. This increases the realism of the RLV problem as in "real world" or practical application of MDO methodologies one will often have to deal with codes that are difficult to use. Legacy tools like POST are very old by aerospace industry standards, but replacing them would incur large costs.

To execute POST the user must provide planetary and vehicle characteristics, define any trajectory constraints, select appropriate guidance algorithms to be used for each phase of the trajectory, and define the objective variable to be used for optimization. The inputs and outputs used for the course of this study are shown below.

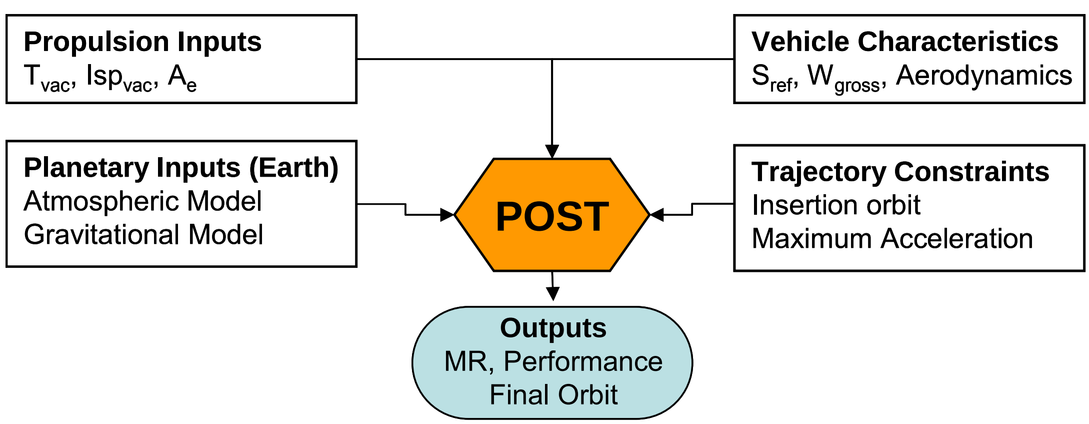

Figure 3: POST Inputs and Outputs.

While newer versions of POST can calculate 6 degree of freedom (DOF) trajectories, for this project a 3 DOF simulation was executed. POST is written in FORTRAN and has been compiled for use in batch mode for both the UNIX and PC platforms. Both platforms were used during the course of this study.

5.3 Weights & Sizing Tool

Weights & Sizing analysis is performed via a set of Space Shuttle derived mass estimating relationships (MER's) developed by NASA Langley[18]. The MER's are calculated using a Microsoft Excel spreadsheet created for this project. The MER's represent mass relationships using technology levels commensurate with the Space Shuttle, thus they must be altered in order to more realistically model a next generation RLV. The MER's are broken down by system and subsystem, thus technology reduction factors (TRF's) can be implemented to represent technological advancements (the majority in materials) that will allow for mass savings in next generation RLV's. Table 2 shows the reduction factors used for this project.

Table 2: Technology Reduction Factors.

Subsystem	TRF
Wing Group	20%
Tail Gourp	10%
Body Group	
Wbody	20%
Wsecondary	15%
Wbodyflap	20%
Wthrust	30%
Landing Gear	9%
Main Propulsion	25%
Electric Conversion and Distribution	18%
Avionics	50%
Environmental Control	10%

To execute the Weights & Sizing tool, one must provide a base vehicle to photographically scale, propulsion characteristics, performance requirements, a dry mass contingency (15% for this study), and TRF's. The Weights & Sizing tool will then calculate the vehicle's available mass ratio (MR_{avail}), W_{gross}, W_{dry} and create a weight breakdown by subsystem. The tool is designed to photographically scale the vehicle via a scale factor (SF) until the $MR_{avail} = MR_{req}$. The inputs and outputs used in Weights & Sizing are below.

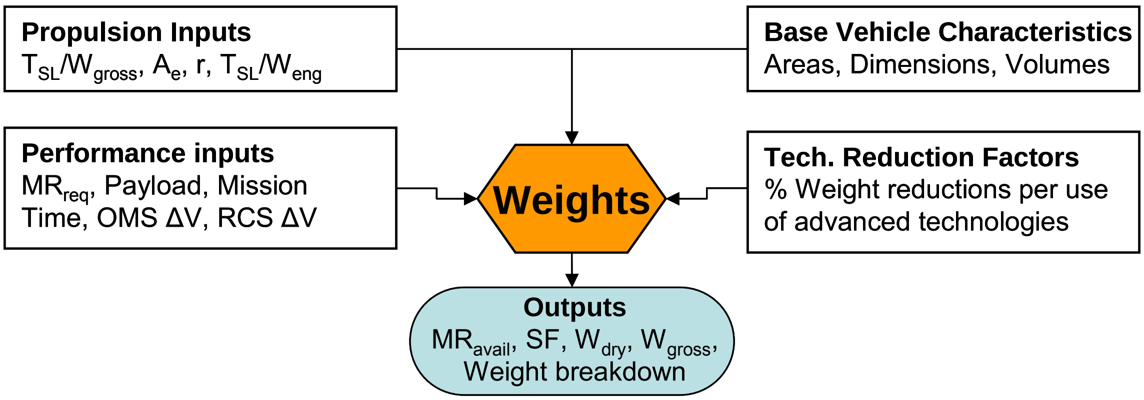

Figure 4: Weights & Sizing Inputs and Outputs.

5.4 Computational Framework

Phoenix Integration's ModelCenter[19] software was used to communicate and coordinate between tools. ModelCenter is a process integration and design optimization package. It was used to integrate all the discipline codes into a single interface without having to actually combine the source codes of each tool.

Framework integration is done by writing a wrapping script for each discipline that will automatically send inputs, execute and retrieve the outputs for any discipline codes. It allows for codes written in different languages and located in different computers within a network to be accessed from a single user interface. For this study REDTOP, written in JAVA, POST, written in FORTRAN, and the Microsoft Excel Weights & Sizing tool were all wrapped while Phoenix Integration's Analysis Server software allowed for each to be accessed using ModelCenter. Once the discipline codes are wrapped, one can easily link and transfer data between them. Also, ModelCenter allows for scripts to be easily written within the integrated frmaework and linked to the other codes.

Along with integration ModelCenter also provides optimization capabilities through the use of the DOT optimizer which is incorporated in the ModelCenter software package. DOT can perform gradient-based optimization through the use of both direct methods (SLP, MoFD, SQP) and unconstrained methods (Variable Metric and Conjugate Gradient).[20] It also records and organizes data for each iteration when performing an optimization routine; this greatly facilitates to understand model behavior and to fix any bugs. This data can then be saved for further analysis.

A screen shot showing the ModelCenter user interface is shown below.

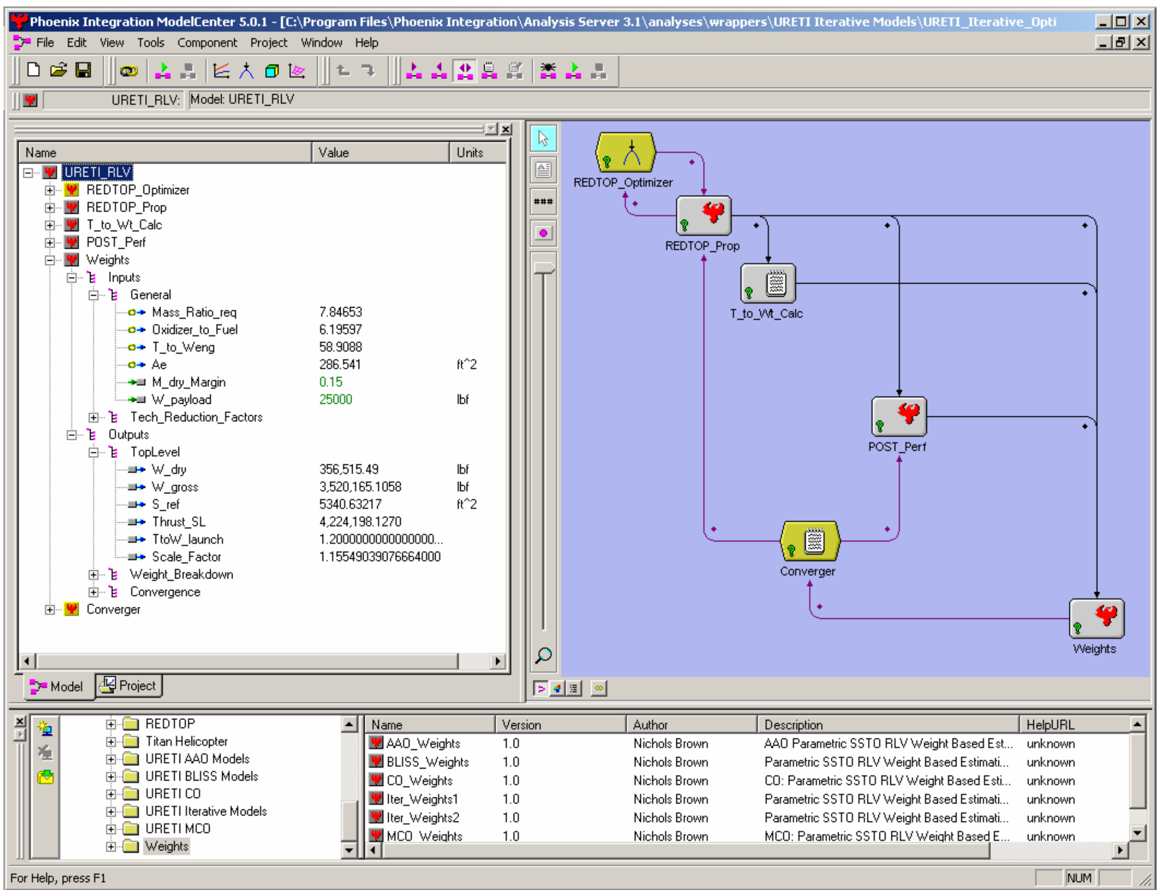

Figure 5: ModelCenter User Interface.

The use of ModelCenter or a program like it greatly facilitated the application and execution of the MDO techniques studied for this project. If no such software is available then it would have been necessary to manually run and exchange data between each discipline code. This would have been an extremely intensive procedure and would have greatly slowed down the creation and running of RLV models. The breadth of work shown in this report would not have been realizable without a computational framework tool.

5.5 Response Surface Modeling Tool

Pi Blue's ProbWorks [21] software package, specifically its Response Surface Equation (RSE) Generator component, was used to create response surface models (RSM's) of the discipline codes. ProbWorks components are coded in JAVA and can easily be incorporated into ModelCenter's computational frameworks.

The RSE Generator is capable of automatically creating different types of design of experiments (DOE) given a number of design variables (2nd Order Central Composite Design was used throughout this project) and then producing a wrapper containing all the RSE's of the response variables which make up an RSM.

RSM's were used in this study to create fast and accurate models of each of the discipline tools only when called for by an MDO technique.

6 Convergence Errors

Whenever a complex system is decomposed into discipline sub-problems one needs to make sure that any final configuration is converged. Convergence occurs when all disciplines or contributing analyses agree on the values for the coupling variables while meeting any constraints required by the test problem.

For example, Table 3 shows that the insertion orbit altitude ($h_{insertion}$) and nozzle exit area (A_e) are both converged with errors very close to zero or within problem requirement tolerances.

Table 3: Example of Converged Variables, $h_{insertion}$ & A_e.

Variable	Problem Requirement	System	Propulsion	Performance	Weights	Max Abs % Error
Intradisciplinary						
$h_{insertion}$	303805+/-100			303717		0.029%
Interdisciplinary						
A_e			281.56	281.56	281.56	0.000%

Red underline signifies the value used to normalize in the calculation of % error.

The RLV test problem requires that the vehicle be capable of reaching ISS insertion orbit (see page 11). To fulfill this requirement the orbit $h_{insertion}$ must reach 303805 ft within a tolerance of ±100 ft. Here the performance estimated value for $h_{insertion}$ is within acceptable tolerance. A_e does not have to match a value set by the test problem requirements, but for convergence the value of A_e that is used or calculated between all disciplines and levels must be the same.

In practical applications there will usually be some convergence error. Throughout the course of this study a universal error threshold was not required when attempting to converge the all the MDA and MDO techniques, but instead the user conditioned model's optimization parameters until it appeared convergence would not improved using the computational tools available.

This study differentiates between two types of convergence error: intradisciplinary and interdisciplinary.

Interadisciplinary convergence errors occur during internal convergence within a discipline in order to meet local optimization equality constraints (h_{loc}). For example, the performance legacy tool, POST, performs local optimization to maximize the MR_{req} but must do so while reaching ISS insertion orbit. In practical applications POST will meet the desired orbit within user defined tolerances. Tolerances were set as small as could be efficiently reached by the legacy tools to minimize intradisciplinary errors.

Interdisciplinary convergence errors are when the value for a coupling variable is not exactly the same between two disciplines.

7 FPI: Fixed Point Iteration

Fixed Point Optimization (FPI) or the Iterative procedure[20] is the traditional way of solving or converging a multidisciplinary analysis (MDA) problem. It is not, though, a multidisciplinary optimization (MDO) technique. While applying FPI will produce a converged model it does not attempt to optimize the model to a global objective as is the aim of MDO. In this procedure each discipline may perform local optimization on some aspect of their discipline, but no global system-level optimization is executed. Therefore, since only local optimization based on local objectives is perfomed FPI solutions are expected to produce sub-optimal RLV's with respect to a global objective.

This section will provide some background information about FPI and its relevance to this study; show the formulation when it is applied to the RLV problem, present the results gathered and discuss findings. There were two different FPI formulations applied, Option 1 and 2, which differed in which discipline owned design control over the fuel mixture ratio (r). Also, it was attempted to more accurately represent industry practice by having an expert set the design space used for the propulsion design variables.

7.1 FPI: Background

When aerospace systems grew too complex to be designed by one group or individual, the system analysis was broken down into multiple disciplines that dealt with only part of the problem. Unfortunately there is coupling between the disciplines and thus one discipline could not complete their design without inputs from another and vice versa. This creates a multidisciplinary analysis (MDA) problem where in order to design the vehicle each group must a first guess what the other discipline's input is going to be and then see how close their guess was to the actual value. The MDA problem is solved when all the inputs and outputs shared between disciplines converge at the same values. FPI has traditionally been the method of choice for solving or converging MDA problems.

In FPI each discipline optimizes their part of the entire system with respect to a local variable, for example the Structures group may try to make a plane as strong as possible while the Aerodynamics group will try to reduce drag. It is expected, though, that this approach will provide solutions that are sub-optimal in the system-level as it does not exploit multi-discipline interactions. For example if Aerodynamics makes wings that are really thin to reduce drag, this may negatively affect Structures as now the wings are weak and may break. Also, lost in the shuffle is the bottom line. The customer for such a system does not care about discipline level goals but is more interested in global metrics relating to cost or weight of the entire system.

FPI is the standard design practice used for solving MDA problems in the design of aerospace systems and vehicles. The creation of an FPI model, thus, will create a basis with which to judge the purported benefits of applying MDO to the design of a next generation RLV.

7.2 FPI: Formal Problem Statement

While in FPI there is no system-level optimization, there is local-level or disciplinary optimization. A general problem statement for each of the three disciplines involved in the RLV design, Propulsion, Performance, and Weights & Sizing, is shown in this section.

The FPI approach is unique in that a total of 8 different models that each provided a converged vehicle solution using FPI were created. The different models were a result of: 1) allowing mixture ratio (r) to be controlled by Propulsion (Option 1) or by Weights & Sizing (Option 2), 2) the local-level objective for the Propulsion discipline (Φ_{Prop}) was allowed to vary between Isp_{vac} and T_{SL}/W_{eng} and 3) two different design spaces were applied to the design variables for the Propulsion analysis.

Varying the control of mixture ratio (r) between the Propulsion and Weights & Sizing disciplines was performed in order to investigate how changing the ownership of shared input variables could affect the overall design of an RLV. Ideally, no matter who has ownership of a shared input variable, the overall vehicle design should not be affected. This is not true in FPI due to each discipline optimizing only at the local-level. In industry, the general practice is to give ownership of "r" to the Propulsion discipline.

Next, while maximizing Isp_{vac} is the most common Φ_{Prop} used in industry, it is not self-evident why the objective could not be to maximize T_{SL}/W_{eng}. Due to the strictly local nature of optimization in FPI changing Φ_{Prop} will result in different converged vehicle configurations.

Lastly, the design space was altered in order to give a better representation of what occurs in industry practice. The overall vehicle design was found to be very sensitive to the side constraints applied to the local design variables in the Propulsion analysis. In industry, this sensitivity is well known thus an expert is used to set the design space at an appropriate range for a specific system. This is fine, except that a new or radical design might be incorrectly influenced or constrained by the designer's experiences. It would be ideal for a design technique to allow for the widest design space possible for a given set of tools and let the tools determine the correct values. That way the design is ruled solely by physics and not by human bias.

For this study the two Propulsion design spaces were evaluated as shown below:

Expert Design Space: Side constraints are determined by a propulsion expert. This design space reflects the suggested values from Tim Kokan, space propulsion graduate specialist at Georgia Institutes of Technology's Space Systems Design Lab[22].

Large Design Space: Side constraints are maximized so as to not limit the optimization of the system to preconceived notions.

Table 4: Propulsion Design Spaces Investigated for FPI.

Expert Design Space	Large Design Space
1. $50 \le \varepsilon \le 90$	1. $30 \le \varepsilon \le 100$
2. $5 \le r \le 7$	2. $4 \le r \le 10$
3. $1500 \le p_c \le 3100$ (psia)	3. $200 \le p_c \le 3100$ (psia)

The 8 different FPI formulations investigated for this study were:

I. Option 1: Mixture ratio (r) controlled by Propulsion

A. Expert Design Space

i. $\Phi_{Prop} = T_{SL}/W_{eng}$ (1)

ii. $\Phi_{Prop} = Isp_{vac}$ (2, Industry Standard)

B. Large Design Space

i. $\Phi_{Prop} = T_{SL}/W_{eng}$ (3)

ii. $\Phi_{Prop} = Isp_{vac}$ (4)

II. Option 2: Mixture ratio (r) controlled by Performance

A. Expert Design Space

i. $\Phi_{Prop} = T_{SL}/W_{eng}$ (5)

ii. $\Phi_{Prop} = Isp_{vac}$ (6)

B. Large Design Space

i. $\Phi_{Prop} = T_{SL}/W_{eng}$ (7)

ii. $\Phi_{Prop} = Isp_{vac}$ (8)

The resulting configuration with the lowest W_{dry} was selected as the best FPI configuration. Each MDO technique application model was initialized using the best FPI configuration.

7.2.1 FPI: Propulsion Standard Form

Minimize:	Φ_{Prop}	$-(Isp_{vac})$ or $-(T_{SL}/W_{eng})$		
Subject to:	g	$p_e \geq 5$ psia		
	h	$T_{SL,avail} = T_{SL,req}$		
	Side	$4 \leq r \leq 10$	or	$5 \leq r \leq 7$
		$30 \leq \varepsilon \leq 100$	or	$50 \leq \varepsilon \leq 90$
		$200 \leq p_c \leq 3100$ psia	or	$1500 \leq p_c \leq 3100$ psia
By Changing:	$\mathbf{X}_{loc}$	ε, p_c, A_t, r	(r only for Option 1)	

7.2.2 FPI: Performance Standard Form

Minimize:	Φ_{Perf}	MR_{req}
Subject to:	h	$h_{insertion} = 303805$ ft $i_{insertion} = 51.6\,°$ $\gamma_{insertion} = 0\,°$
By Changing:	$\mathbf{X}_{loc}$	$\dot{\theta}_{Azimuth}$, $\dot{\theta}_{Pitch1}$, $\dot{\theta}_{Pitch2}$, $\dot{\theta}_{Pitch3}$, $\dot{\theta}_{Pitch4}$

7.2.3 FPI: Weights & Sizing Standard Form

Minimize:	$\Phi_{W\&S}$	W_{dry}
Subject to:	h Side	$MR_{avail} = MR_{req}$ $T_{SL}/W_{gross} \geq 1.2$
By Changing:	$\mathbf{X}_{loc}$	SF_{veh}, T_{SL}/W_{gross}, r (r only for Option 2)

7.3 FPI: Data Flow

In order to better understand the coupling of shared variables the design structure matrices (DSM's) for both FPI Option 1 and 2 are provided. Also provided are variable tables which help to quickly observe the important variables for each local-level discipline.

7.3.1 FPI: Design Structure Matrix

<u>Option 1: Mixture ratio (r) controlled by Propulsion</u>

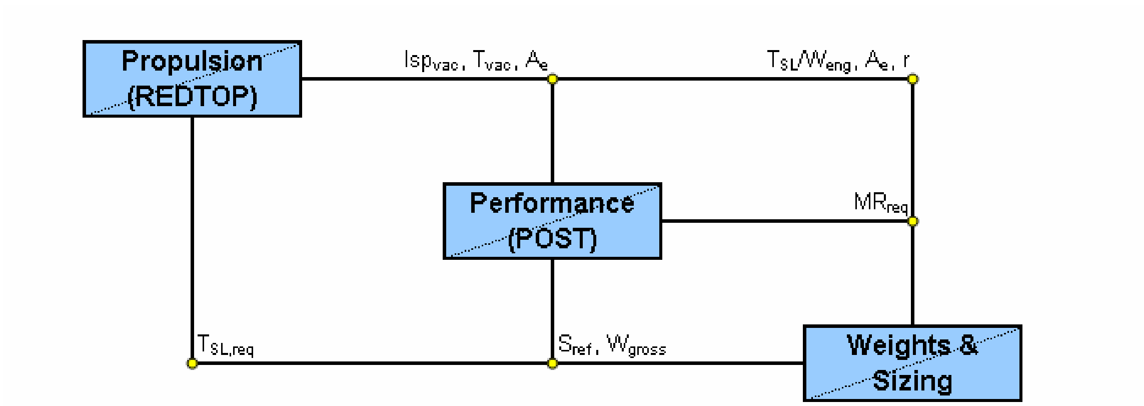

Figure 6: DSM for FPI Option 1.

<u>Option 2: Mixture ratio (r) controlled by Weights & Sizing</u>

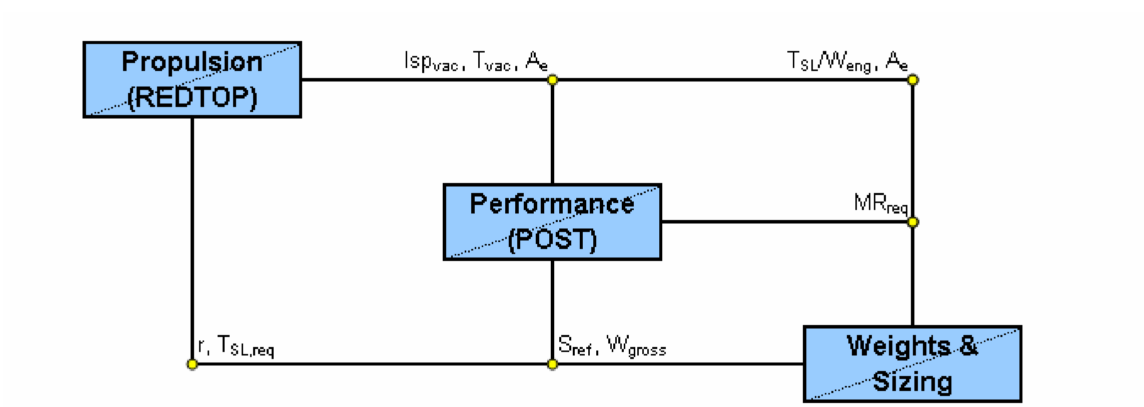

Figure 7: DSM for FPI Option 2.

Note that both FPI DSM's have feed forward and feed back loops. This means that tightly coupled disciplines or contributing analyses (CA's) must iterate between them several times. This could be costly if the CA's are expensive or time consuming. If the feed back loops are not needed then the disciplines can just be run one time in sequential order from left to right. If forward loops between disciplines are broken, then the CA's could execute in parallel or simultaneously.

7.3.2 FPI: Variable Table

<u>Option 1: Mixture ratio (r) controlled by Propulsion</u>

Table 5: Variable Table for FPI Option 1.

Contributing Analysis	Output Variables	Input Variables X_{sys}	Input Variables X_{loc}	Input Variables Y^*	Φ_{loc}	Local Constraints
1. Propulsion	1. Isp_{vac} 2. T_{vac} 3. A_e 4. T_{SL}/W_{eng} 5. r		r, ε, p_c, A_t	$T_{SL,req}$	1. T_{SL}/W_{eng} (max) 2. Isp_{vac} (max)	1. $p_e \geq 5$ (psia) 2. $T_{SL,req} = T_{SL,avail}$ 3. $30 \leq \varepsilon \leq 100$ 4. $4 \leq r \leq 10$ 5. $200 \leq p_c \leq 3100$ (psia)
2. Performance	1. MR_{req}		$\theta_{Azimuth}$, θ_{Pitch1}, θ_{Pitch2}, θ_{Pitch3}, θ_{Pitch4}	S_{ref}, A_e, W_{gross}, Isp_{vac}, T_{vac}	1. MR_{req} (min)	1. $h_{insertion} = 303805$ (ft) 2. $i_{insertion} = 51.6$ (deg) 3. $\gamma_{insertion} = 0$ (deg)
3. Weights	1. S_{ref} 2. W_{gross} 3. $T_{SL,req}$		SF_{veh}, T_{SL}/W_{gross}	r, MR_{avail}, A_e, T_{SL}/W_{eng}	1. W_{dry} (min)	1. $MR_{req} = MR_{avail}$ 2. $T_{SL}/W_{gross} \geq 1.2$

Option 2: Mixture ratio (r) controlled by Weights & Sizing

Table 6: Variable Table for FPI Option 2.

Contributing Analysis	Output Variables	Input Variables			Φ_{loc}	Local Constraints
		X_{sys}	X_{loc}	Y^*		
1. Propulsion	1. Isp_{vac} 2. T_{vac} 3. A_e 4. T_{SL}/W_e		ε, p_c, A_t	r, $T_{SL,req}$	1. T_{SL}/W_{eng} (max) 2. Isp_{vac} (max)	1. $p_e \geq 5$ (psia) 2. $T_{SL,req} = T_{SL,avail}$ 3. $30 \leq \varepsilon \leq 100$ 4. $4 \leq r \leq 10$ 5. $200 \leq p_c \leq 3100$ (psia)
2. Performance	1. MR_{req}		$\theta_{Azimuth}$, θ_{Pitch1}, θ_{Pitch2}, θ_{Pitch3}, θ_{Pitch4}	S_{ref}, A_e, W_{gross}, Isp_{vac}, T_{vac}	1. MR_{req} (min)	1. $h_{insertion} = 303805$ (ft) 2. $i_{insertion} = 51.6$ (deg) 3. $\gamma_{insertion} = 0$ (deg)
3. Weights	1. S_{ref} 2. W_{gross} 3. $T_{SL,req}$ 4. r		r, SF_{veh}, T_{SL}/W_{gross}	MR_{avail}, A_e, T_{SL}/W_{eng}	1. W_{dry} (min)	1. $MR_{req} = MR_{avail}$ 2. $T/W_{LO} \geq 1.2$

Note that both FPI variable tables have the X_{sys} empty. This shows that for FPI there is not a system optimizer. All variables are controlled at the local-level, X_{loc}.

7.4 FPI: Results

This section shows final configuration results for RLV's converged using the FPI method.

7.4.1 FPI: Configuration Results

<u>Option 1: Mixture ratio (r) controlled by Propulsion</u>

Table 7: Final Configuration Table for FPI Option 1.

	Expert Design Space		Large Design Space		Units
Propulsion Objective	T/We	Isp,vac	T/We	Isp,vac	
Direction	Maximize	Maximize	Maximize	Maximize	
T_{SL}/W_{eng}	60.12	58.91	73.79	58.98	
Isp_{vac}	441.5	446.2	387.4	447.3	s
c*	7,418	7,603	6,650	7,769	ft/s
Isp_{SL}	386.8	390.2	358.0	394.4	s
Cf_{vac}	1.874	1.856	1.816	1.824	
Cf_{SL}	1.642	1.623	1.678	1.614	
T_{SL}/W_{gross}	**1.20**	**1.20**	**1.20**	**1.20**	
ε	**50.0**	**50.0**	**30.0**	46.2	
r	**7.00**	6.20	**10.00**	5.43	
p_c	**3,100.0**	**3,100.0**	**3,100.0**	**3,100.0**	psia
p_e	5.557	**4.997**	11.12	**5.002**	psia
A_e	261.36	286.54	431.07	319.79	ft^2
A_t	5.23	5.73	14.37	6.92	ft^2
SF	1.102	1.155		1.258	
MR	8.01	7.85	10.34	7.79	
S_{ref}	4,853	5,341	8,718	6,328	ft^2
W_{gross}	3,262,594	3,520,165	9,258,826	4,203,727	lbf
W_{dry}	320,811	356,515	714,903	433,967	lbf

BOLD values represent active contraints

Option 2: Mixture ratio (r) controlled by Weights & Sizing

Table 8: Final Configuration Table for FPI Option 2.

	Expert Design Space		Large Design Space		Units
Propulsion Objective	T/We	Isp,vac	T/We	Isp,vac	
Direction	Maximize	Maximize	Maximize	Maximize	
T_{SL}/W_{eng}	60.12	59.06	73.56	65.32	
Isp_{vac}	441.5	443.1	387.3	399.8	s
c*	7,418	7,418	6,646	6,650	ft/s
Isp_{SL}	386.8	383.8	356.7	345.0	s
Cf_{vac}	1.874	1.884	1.816	1.894	
Cf_{SL}	1.642	1.631	1.673	1.635	
T_{SL}/W_{gross}	**1.20**	**1.20**	1.31	**1.20**	
ε	**50.0**	54.3	**30.0**	55.9	
r	**7.00**	**7.00**	**10.00**	**10.00**	
p_c	**3,100.0**	**3,100.0**	2,973.5	**3,100.0**	psia
p_e	5.557	**5.003**	10.68	**5.000**	psia
A_e	261.38	281.56	517.20	621.07	ft^2
A_t	5.23	5.19	17.24	11.10	ft^2
SF	1.102	1.096	1.500	1.335	
MR	8.01	7.96	10.14	9.78	
S_{ref}	4,854	4,801	8,994	7,132	ft^2
W_{gross}	3,262,698	3,212,690	9,723,742	6,892,809	lbf
W_{dry}	320,820	317,606	769,786	560,975	lbf

BOLD values represent active contraints

The column highlighted represents the best FPI model, selected with the desire to minimize the W_{dry} of the system.

7.4.2 FPI: Convergence Error

Convergence error analysis was done only on the best FPI model with respect to the global objective, W_{dry}. The best FPI model was [Option 2, Expert Design Space, $\Phi_{,Prop} = Isp_{vac}$].

Table 9: Convergence Error for best FPI Configuration.

Variable	Problem Requirement	System	Propulsion	Performance	Weights	Max Abs % Error
Intradisciplinary						
$\gamma_{insertion}$	0+/-0.001			-0.0001		N/A
$h_{insertion}$	303805+/-100			303717		0.029%
$i_{insertion}$	51.6+/-0.1			51.60		0.000%
MR	$MR_{avail} = MR_{req}$			7.964	7.964	0.000%
T_{SL}	$T_{SL,avail} = T_{SL,req}$		3855234		3855201	0.001%
Interdisciplinary						
A_e			281.56	281.56	281.56	0.000%
Isp_{vac}			443.081	443.081		0.000%
r			7.000		7.000	0.000%
S_{ref}				4801.2	4801.3	0.001%
T_{SL}/W_{eng}			59.063		59.063	0.000%
T_{vac}			4450810	4450810		0.000%
W_{gross}				3212668	3212695	0.001%

Red underline signifies the value used to normalize in the calculation of % error.

The intradisciplinary convergence error in the best FPI model is very low, never above 0.03%. This low error is thanks to the great time and effort spent to improve the precision of the legacy codes used in this study. The legacy codes were created with the FPI process in mind and are thus highly evolved to solve the FPI problem with the greatest efficiency and accuracy possible.

Also, the interdisciplinary convergence error is very low thanks to the fact that there are virtually no numerical errors in the FPI procedure as there are never any design space gradients taken.

Note that the “system” column is empty. This is due to the lack of a system-level optimizer when performing the FPI process.

7.5 FPI: Discussion & Qualitative Ratings

This section will address FPI optimization conclusions, lessons learned when implementing the method and provide qualitative ratings with discussion for each.

7.5.1 FPI: Optimization Conclusions

FPI is not an MDO method, thus it does not optimize to achieve a global objective. FPI, though, is often employed in industry to solve an MDA problem and then variable sweeps are used to manually find what the globally optimized configuration is. This is not ideal as iteration may have to be performed many times for the variable sweeps and the design may be limited by the preconceived notions of the designer.

Of the 8 different FPI models generated, the best solution was reached by the combination [Option 2, Expert Design Space, $\Phi_{Prop} = Isp_{vac}$] which converged the RLV problem at $W_{dry} \approx 317$ klb. This best combination, though, is <u>not</u> the industry standard practice ([Option 1, Expert Design Space, $\Phi_{Prop} = Isp_{vac}$] with $W_{dry} \approx 356$ klb) thus it can be surmised that alternating which discipline has ownership of critical shared variables can significantly affect the best resulting design converged using FPI.

7.5.2 FPI: Lessons Learned

Lessons learned in the implementation of the FPI process are few as FPI is the method for which all the legacy tools were designed. As expected, FPI models were easy straight forward to create and the tools behaved well when applied. Nevertheless, some lessons learned are:

1) The order in which CA's are excecuted is critical for the model to run smoothly. In this case the POST – Weights & Sizing loop, see Figure 6 on page 27, needed to be iterated last. If not the POST – Weights & Sizing loop would often converge at a

vehicle weight so high that the thrust provided by the Propulsion analysis was not enough to lift the RLV off the launch pad.

2) The use of relaxation to perform the vehicle system convergence was able to slightly increase the range of starting points from which the model could be initialized and still find a converged solution. Its use, though, could greatly increase convergence time.

7.5.3 FPI: Qualitative Ratings

It is very difficult to qualitatively assess the experience of implementing a given MDA or MDO technique without first comparing it to other the other techniques being evaluated. Therefore these ratings were assigned after all the techniques had been applied.

Table 10: Qualitative Ratings for FPI.

Criteria	Grade	Discussion
Implementation Difficulty	A	Virtually all legacy tools were created for an iterative environment. This gives the implementation of the FPI process a big advantage. One could run into more difficulty if codes cannot be readily modified to allow for ownership of internal design variables by different codes. Thus it could be difficult to vary the discipline that has control over a shared variable.
Total Execution Time	A	(5 to 20 minutes) Varied with the amount of relaxation used & initialization point.
Robustness	A	Model converged for a wide range of initialization points. Relaxation could be used to converged otherwise unstable sections of the model.
Formulation Difficulty	A	The formulation is straight forward and is usually evident from the legacy tools themselves.
Optimization Deftness	D	FPI will not, by itself find a globally optimized solution. It requires a human to make a sweep of possible design spaces or configurations to find the global optimum. Though multiple configurations were tried, the best FPI solution was still found to be a sub-optimal solution.

Convergence Error	A	There is no gradient taking. Convergence is limited solely on the convergence tolerance of the legacy codes used. These codes usually converge with very tight tolerances thanks to time invested in conditioning their optimization to the specific problem.
Issues		1) If control of coupling variables is not varied between disciplines then FPI may result in very poor results. In this study, when the industry standard of having Propulsion control the mixture ratio (r) and maximize Isp_{vac} was used; the FPI solution converged with a vehicle 15-20% heavier than the true optimum. 2) This is not an MDO technique thus vehicle optimization is dependent on human intervention. Varying optimization parameters (such as propulsion design space) can greatly affect the resulting vehicle configuration.
Unique Benefits		1) Makes use of the legacy code's original optimizer which has already been conditioned to best handle the discipline's local-level problem. 2) Virtually no legacy code modifications were needed.

8 AAO: All-at-Once

All-at-Once (AAO) is the most basic of MDO techniques and has wide industry acceptance although it is restricted to small design problems. It takes all local-level optimization away and gives control of all the design variables to one, system-level optimizer. It ensures that MDO is performed and a global objective is met by in essence doing away with all the disciplines (except with respect to straight forward analysis) and having only one designer control the entire vehicle system design.

This section will provide some background information about AAO and its relevance to this study; show the formulation when it is applied to the RLV problem, present the results gathered and discuss findings. The major drawback to AAO is that it does not scale well with increasing complexity. This became apparent during this study as it seemed that the system-level optimizer had some difficulty precisely reaching the most optimized design due to the large number of design variables. Therefore, there are two AAO results presented, one where all the local-level variables were transformed to the system-level design variables and one where two variables, T_{SL}/W_{gross} and p_c, were set as parameters at known "preferred" values.

8.1 AAO: Background

Aerospace systems are broken down into multiple disciplines that deal with only part of the problem. This makes MDO difficult as when each discipline is optimizing their part of the system to meet a local objective it becomes impossible to optimize the whole to reach a more desired global objective.

AAO is the most basic solution available to solve the global MDO problem. It moves all the local-level design variables and constraints away from the each discipline and gives them to a new system-level optimizer which is entrusted with optimizing the vehicle to meet a global objective. The disciplines remain but they are but a shell of their old selves as all design freedom is taken away and they are entrusted solely with doing analysis, no design. The system-level optimizer will vary all the design variables and passes inputs to each discipline so that they can perform local-level disciplinary analysis and design.

The great benefit of AAO is that all the design variables are controlled by a single user that can immediately see how changes to a particular part of the vehicle affect the vehicle as a whole. AAO with Optimizer Based Decomposition (OBD) breaks the feedback loops between CA's in a DSM. Thus whenever the system makes a change in the vehicle configuration, each discipline only needs to run one time. This may save total execution time versus FPI where tightly coupled CA's may have to iterate and converge before the rest of the analysis can continue.

While AAO is the most straight forward way to solve an MDO problem it has great drawbacks that makes it inapplicable to the detailed design of aerospace systems.

First, taking the design responsibilities completely away from the disciplines means that discipline experts are no longer involved in design. For decades people have specialized their studies and research amassing a great deal of knowledge related to the design of a particular discipline. The depth of knowledge that all the discipline experts hold together cannot possibly be recreated by a single human being. Thus, if all design control is given to a single system-level optimizer, one loses the knowledge that has been accumulated.

Secondly, the vehicle design was originally broken up into disciplines because the problem was so big and intricate that no single human or optimizer was able to handle the complexity. This is still true of aerospace systems and AAO cannot change that. Thus, unless simplified, the design of an aerospace system is too complex for a single user or system-level optimizer to handle. While AAO may be able to solve the MDO problem for conceptual level design (which has a limited number of design variables) it does not scale well with complexity and is not used during detailed design.

Despite limitations, if an MDO problem can be solved via AAO, there is a high degree of confidence that the final configuration produced will be the true global optimum. The resultant AAO configuration will be used to validate the solutions produced by applying CO, MCO and BLISS.

8.2 AAO: Formal Problem Statement

AAO does not have any local-level optimization, thus the MDO problem is strictly a system-level problem. AAO is composed of one system-level formulation which handles all the design variables for the entire RLV across all disciplines.

8.2.1 AAO: System Standard Form

Minimize:	Φ_{Sys}	W_{dry}
Subject to:	g	$p_e \geq 5$ psia
		$T_{SL}/W_{gross} \geq 1.2$
	h	$\gamma_{insertion} = 0°$
		$i_{insertion} = 51.6°$
		$h_{insertion} = 303805$ ft
		$MR_{req} = MR_{avail}$
		$S_{ref,guess} = S_{ref,actual}$
		$W_{gross,\ actual} = W_{gross,\ guess}$
		$T_{SL,avail} = T_{SL,req}$
	Side	$40.29 \leq \theta_{Azimuth} \leq 44.53$ °/s
		$-1.533 \leq \theta_{Pitch1} \leq -1.386$ °/s
		$0.04191 \leq \theta_{Pitch2} \leq 0.05431$ °/s
		$-0.2696 \leq \theta_{Pitch3} \leq -0.2440$ °/s
		$-0.1421 \leq \theta_{Pitch4} \leq -0.1285$ °/s
		$6.3 \leq r \leq 7.7$
		$45 \leq \varepsilon \leq 60$
		$4.705 \leq A_t \leq 5.750$ ft^2
		$2790 \leq p_c \leq 3100$ psia
		$0.9913 \leq SF \leq 1.21169$
		$1.2 \leq T_{SL}/W_{gross} \leq 1.5$
		$4368 \leq S_{ref,guess} \leq 5339$ ft^2
		$2936 \leq W_{gross,guess} \leq 3559$ klb
By Changing:	$\mathbf{X}_{sys}$	$\dot{\theta}_{Azimuth}$, $\dot{\theta}_{Pitch1}$, $\dot{\theta}_{Pitch2}$, $\dot{\theta}_{Pitch3}$, $\dot{\theta}_{Pitch4}$, r, ε, A_t, p_c, SF, T_{SL}/W_{gross}, $S_{ref,guess}$, $W_{gross,guess}$

8.3 AAO: Data Flow

In order to better understand the coupling of design variables the design structure matrix (DSM) for AAO is provided. Also provided is a variable table which helps to quickly observe the important variables for each local-level discipline.

8.3.1 AAO: Design Structure Matrix

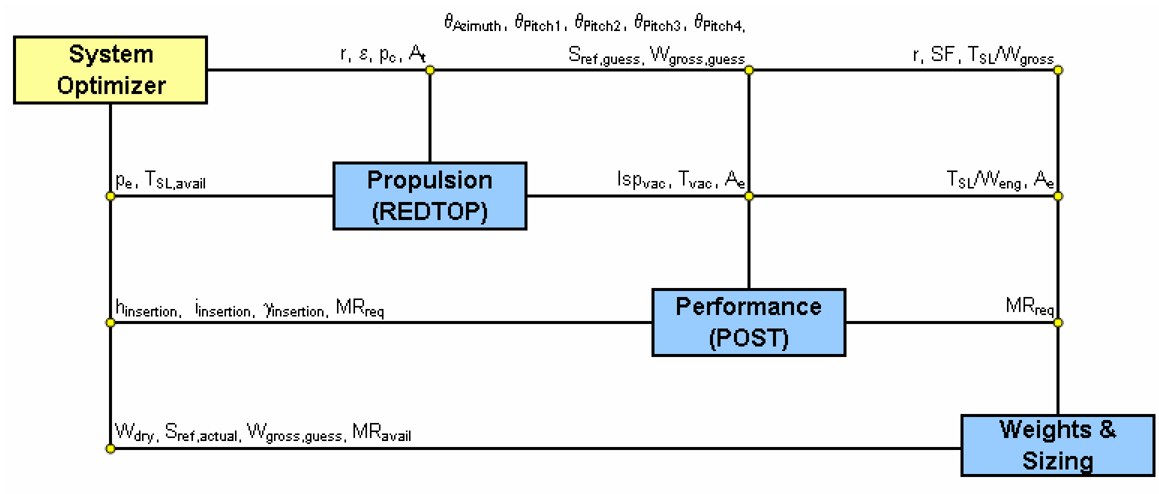

Figure 8: DSM for AAO with Partial OBD.

Note that backward loops between the CA's have been broken. Now all feedback from the disciplines goes directly to the system optimizer. This is achieved through Optimizer Based Decomposition.

OBD or Simultaneous Analysis and Design (SAND) is a method of breaking the feed-back loops in a DSM that has a system-level optimizer. Guess or intermediate copies of the feed-back coupling variables are created and these are used by the system-level optimizer as design variables. Also, compatibility constraints are added at the system-level which ensures that the guess or intermediate variables are equal to the actual values fed back by the CA's.

8.3.2 AAO: Variable Table

Table 11: Variable Table for AAO.

Contributing Analysis	Output Variables	Input Variables: X_{sys}	Input Variables: X_{loc}	Input Variables: Y^*	Φ_{loc}	Local Constraints
1. Propulsion	1. Isp_{vac} 2. T_{vac} 3. A_e 4. T_{SL}/W_{eng} 5. p_e 6. $T_{SL,avail}$	r, ε, p_c, A_t				1. $p_e \geq 5$ (psia) 2. $30 \leq \varepsilon \leq 100$ 3. $4 \leq r \leq 10$ 4. $200 \leq p_c \leq 3100$ (psia) 5. $T_{SL,avail} = T_{SL,req}$
2. Performance	1. MR_{req} 2. $h_{insertion}$ 3. $i_{insertion}$ 4. $\gamma_{insertion}$	$\theta_{Azimuth}$, θ_{Pitch1}, θ_{Pitch2}, θ_{Pitch3}, θ_{Pitch4}, $S_{ref,guess}$, $W_{gross,guess}$		A_e, Isp_{vac}, T_{vac}		1. $h_{insertion}$ = 303805 (ft) 2. $i_{insertion}$ = 51.6 (deg) 3. $\gamma_{insertion}$ = 0 (deg)
3. Weights	1. $S_{ref,actual}$ 2. $W_{gross,actual}$ 3. MR_{avail} 4. W_{dry}	SF, r, T_{SL}/W_{gross}		MR_{req}, A_e, T_{SL}/W_{eng}		1. $MR_{req} = MR_{avail}$ 2. $T_{SL}/W_{gross} \geq 1.2$ 3. $W_{gross,guess} = W_{gross,actual}$ 4. $S_{ref,guess} = S_{ref,actual}$

Note that the AAO variable table have the X_{loc} column empty. This shows that for AAO there is no local optimization. All variables are controlled by the system optimizer, X_{sys}.

8.4 AAO: Results

This section shows final configuration results for the RLV test problem solved using the AAO method and describes the convergence errors resulting from this method.

8.4.1 AAO: Configuration Results

As previously discussed on page 36, AAO does not scale well as many optimizers have trouble handling problems that are complex and have a lot of design variables. This scalability problem is not due to the theory of AAO, but to practical limitations with optimizers.

Scalability difficulty was encountered during the course of this study when the system optimizer tried to simultaneously optimize all of the vehicle's design variables. While the optimizer did a good job at approaching the optimal configuration, numerical errors and insufficient optimizer robustness did not allow for the optimum to be reached as precisely as desired. To fix this problem, two of the original design variables, T_{SL}/W_{gross} and p_c, were changed to system-level parameters at the constraint values that each seemed to be approaching. Thus the final configuration table has two columns:

1) AAO, Full: Indicates stringent application of the AAO procedure as described in theory. All of the design variables moved from the local-level to the system-level are handled as system design variables.
2) AAO, Reduced: Indicates the use of manual overrides to set p_c and T_{SL}/W_{gross} as parameters. All other variables are still handled as system design variables.

Table 12: Final Configuration Table for AAO.

	AAO, Full	AAO, Reduced	Units
System Objective	W_{dry}	W_{dry}	
Direction	Minimize	Minimize	
T_{SL}/W_{eng}	59.76	59.27	
Isp_{vac}	439.6	440.1	s
c*	7,291	7,282	ft/s
Isp_{SL}	380.3	378.4	s
Cf_{vac}	1.896	1.902	
Cf_{SL}	1.641	1.635	
T_{SL}/W_{gross}	**1.207**	**1.200**	
ε	55.2	57.5	
r	7.47	7.51	
p_c	**3,099.9**	**3,100.0**	psia
p_e	5.243	**4.999**	psia
A_e	279.84	290.13	ft^2
A_t	5.07	5.05	ft^2
SF	1.076	1.073	
MR	8.06	8.07	
S_{ref}	4,629	4,610	ft^2
W_{gross}	3,149,989	3,138,648	lbf
W_{dry}	306,413	305,101	lbf

BOLD values represent active contraints

The column highlighted represents the best AAO model, selected with the desire to minimize the W_{dry} of the system.

One can observe that when AAO, Full was used T_{SL}/W_{gross} seemed to approach its constrained minimum (1.2) while p_c approached its constrained maximum (3100 psia). In practical application, though, the optimizer was unable to exactly reach the constrained limits. Unfortunately, this is common in engineering models where the legacy tools have a lot of numerical errors and make it very hard to determine the system derivatives necessary for the system-level optimization process. This can be helped by reducing the number of design variables in the system optimizer. When the values of T_{SL}/W_{gross} and p_c are manually set at the limits and held constant (parameters), the optimizer had an easier time reaching the true optimum solution for the test RLV problem.

8.4.2 AAO: Convergence Error

AAO has no local-level optimization and the only type of convergence error is interdisciplinary error. Even the ISS insertion orbit requirement is now calculated by Performance but the variables affecting it and making sure the requirement is met are a responsibility of the system optimizer.

Compatibility error analysis was done on the best AAO model with respect to W_{dry}; AAO, Reduced.

Table 13: Convergence Error for AAO, Reduced Configuration.

Variable	Problem Requirement	System	Propulsion	Performance	Weights	Max Abs % Error
Interdisciplinary						
A_e			290.13	290.13	290.13	0.000%
A_t		5.0457	5.0457			0.000%
ε		57.501	57.501			0.000%
$\gamma_{insertion}$	0+/-0.001			-0.0009		N/A
$h_{insertion}$	303805+/-100			303706		0.033%
$i_{insertion}$	51.6+/-0.1			51.52		0.155%
Isp_{vac}			440.094	440.094		0.000%
MR				8.0774	8.0658	0.144%
r		7.506	7.506		7.506	0.000%
SF		1.07349914			1.07349914	0.000%
S_{ref}		4608.6		4608.6	4609.6	0.022%
T_{SL}			3765880		3766378	0.013%
T_{SL}/W_{eng}			59.269		59.269	0.000%
T_{vac}			4379612	4380389		0.018%
W_{gross}		3138881.02		3138881.02	3138648	0.007%

Red underline signifies the value used to normalize in the calculation of % error.

While the compatibility error of AAO is low, it is not as low as that reached using the FPI method (see Table 9 on page 32). This is a result of the larger size of the system-level optimization problem.

For this study optimization parameter conditioning was performed in order to minimize errors observed in the model. Due to internal tolerances within the legacy tools, though, a point was reached where seemingly errors could not be driven any lower with the available computational tools. It is possible that there is some set of optimization parameters that would have produced somewhat lower errors.

8.5 AAO: Discussion & Qualitative Ratings

This section will address AAO optimization conclusions, lessons learned when implementing the method and provide qualitative ratings with discussion for each.

8.5.1 AAO: Optimization Conclusions

AAO is not a multi-level MDO optimization technique since all the vehicle optimization is contained at the system-level. AAO, though, is the MDO method that is accepted in industry to solve conceptual level MDO problems. While in theory it will reach a globally optimized configuration, in practice limitations in system-level optimizers and legacy tools severely limit the level of MDO complexity that can be solved with the AAO method. This is not a fault with the AAO methodology, but simply "real world" limitations.

Indeed a precise convergence using the full application of AAO to the RLV test problem, which is still a conceptual level problem, was not reached with the tools and optimizers used for this study. Once the problem was simplified (see page 41), though, the true optimum was reached at $W_{dry} \approx 305$ klb. This is a 4% improvement over the best FPI configuration (see Table 8 page 31).

A 4% improvement in the global objective over the best FPI configuration is a noticeable but modest improvement in the design optimization of the RLV. AAO, as are most MDO optimization algorithms, are often very difficult to apply due to "real world" limitation present in the legacy codes and reconditioning of the optimizers used. Therefore one must weigh the predicted objective benefits of applying an MDO technique versus the cost of implementing it. In fairness to the MDO techniques, though, the best FPI solution was already an unconventional application of the FPI process. If FPI had only been applied as is most common in industry today, then the FPI method would have resulted in a vehicle with $W_{dry} \approx 356$ klb (see page 33). In this case MDO resulted in a 14% improvement over the industry standard FPI method.

8.5.2 AAO: Lessons Learned

Lessons learned in the implementation of the AAO process are as follows:

1) Most legacy tools will allow one to easily turn off all local optimization which turns the computational code into a strictly analysis tool. This was beneficial in the AAO application as it was unnecessary to alter the internal code of any of the legacy tools which may have been a time consuming endeavor.
2) Even for a small design space, POST would crash often. It could be shown that POST would work for most configurations within a given design range. Even so, it was often the case that a design configuration point would be called for during optimization that would cause POST to crash. Thus there were gaps or holes where POST would crash within a generally valid design space. Thus, after the optimization problem was initialized and started moved toward the optimum, it might hit a gap in the design space that would cause POST to crash and the optimization to fail. This problem was overcome by varying the initialization point until the optimizer happened to take a path that did not hit any gaps.

8.5.3 AAO: Qualitative Ratings

It is very difficult to qualitatively assess the experience of implementing a given MDA or MDO technique without first comparing it to other the other techniques to be evaluated. Therefore these ratings were assigned after all the techniques had been applied.

Table 14: Qualitative Ratings for AAO.

Criteria	Grade	Discussion
Implementation Difficulty	C	Often the internal optimizer can be switched off easily, facilitating implementation. Conditioning the system optimizer to simultaneously optimize all the design variables proved difficult. A problem slightly larger or more complex may be beyond the capability of most optimizers.
Total Execution Time	A-	(40 to 50 minutes)
Robustness	D	The optimizer would not converge if the design space was not relatively small, stable and almost centered around the eventual optimized values. Also, there were problems in that the legacy tools would crash in points within the design space. This is a problem with the legacy tool but shows some of the problems that can be encountered in "real world" applications.
Formulation Difficulty	A	The formulation is straight forward as one can just automatically move all the variables from the local-level to the system-level without having to add any variables.
Optimization Deftness	A-	A full application of AAO did not precisely converge at the true optimum due to numerical difficulties with the legacy codes that impaired the system-level optimizer. The true optimum was precisely obtained by reducing the size of the problem by setting two at the constrained minimum they were approaching. The difficulties encountered show the scalability problems with AAO even when applied to a relatively small, conceptual level problem.

Convergence A-Error	Simultaneous optimization of all variables by a single optimizer will generally not have compatibility error as low as FPI unless a long time used to carefully condition the optimizer. This is because the local-level optimizers are free to use the optimization method most suitable for their specific problem, while a system optimizer has to be general enough to optimize all parts simultaneously.
Issues	1) Numerical difficulties from the legacy codes may make it very difficulty to condition optimizer to converge tightly. 2) Lack of robustness in the legacy codes caused crashes even when in a region well within the code's capability.
Unique Benefits	It can be mathematically proven that in theory AAO should reach the true system-level optimum. Thus, if AAO can be successfully applied, there is a high degree of confidence in its result.

9 BLISS: Bi-Level Integrated System Synthesis

Bi-Level Integrated System Synthesis (BLISS) is a two-level optimization algorithm originally developed by Dr. Jaroslaw Sobieszczanski-Sobieski, et al., of NASA Langley. Being a multi-level optimization algorithm there is optimization in more than one level of the entire RLV design, as opposed to FPI which only has local optimization or AAO which only has system-level optimization.

This section will provide some background information about BLISS and its relevance to this study, show the formulation when it is applied to the RLV problem, present the results gathered and discuss findings.

9.1 BLISS: Background

BLISS, along with CO and MCO, is one of the most promising multi-level MDO techniques. The first version of BLISS[3] was developed in 1998 with the most recent derivative, BLISS-2000[4], introduced in 2002. BLISS is a two-level MDO technique with both local and system-level optimization. Only the BLISS-2000 algorithm will be applied for this study as it is anticipated by the BLISS developer, Dr. Sobieszczanski-Sobieski, to the best derivative published to date.

As with other multi-level MDO algorithms proposed, one of the greatest obstacle to the acceptance of BLISS is the fact that it was "crafted" as opposed to "rigorously derived."[9] Thus it requires studies applying it to realistic test problems, like the one presented here, to gain industry acceptance.

BLISS intends to allow for large MDO problems to be solved while at the same time minimizing the amount of changes that need to be made to current design practices. BLISS, like CO and MCO, is a two-level MDO technique and thus is believed to be well suited for the conventional disciplinary structure used in industry. It also lets experts have most of the control over the local discipline design taking advantage of their advanced knowledge. Since it does not add any new local-level design variables, CA's can be performed similarly to current practices developed for use with an FPI process in mind.

In order to be able to successfully coodinate between disciplines and arrive at a global optimum, BLISS needs to introduce new design variables to the system-level optimizer. BLISS uses weighting factors (w's) on local-level outputs to produce an overall system-level MDO solution. The system-level optimizer is now in charge of handling all the shared inputs or coupling variables for the system as well as the new weighting factors. These weighting factors are used to dynamically control each discipline's local-level objective. Due to the addition of new variables, which have the potential to complicate the system-level optimization, it is expected that BLISS, like CO and MCO, is best suited for MDO problems with low dimensionality coupling.

A feature added to BLISS in the BLISS-2000 derivative is the creation RSM's of all the CA's and applying the bi-level optimization scheme directly to the RSM's instead of the original legacy tools. Substituting for the original legacy codes is an attempt to avoid the numerical integration and stability problems that are often present when using troublesome legacy tools. The RSM's, since they are just curve fits, will provide clean consistent gradients for the system-level optimizer. They will also run much faster than the original legacy tools; providing execution time savings.

RSM's, though, require that the original CA's must be run in a design of experiments (DOE) to create the data to which the RSM's will be fitted. This could be a costly endeavor if there are a lot of local-level design variables. On the other hand, using parallel execution to distribute this process over a computer network means that, in theory, this could be done in the amount of time required to execute the most time consuming CA just once. If the RSM is not an accurate fit of the original tools, then additional DOE's may need to be run until the design space is small enough to bring RSE error within desired tolerance. Each time that the design space is decreased around the previous configuration, new RSM's are created and used to solve the MDO problem is considered a BLISS iteration.[4]

For this study a 2nd Order Central Composite DOE was used to create all the discipline RSM's. The number of variables to be fitted, resulting number of CA runs needed, and the total execution time called for by the DOE's needed in BLISS, are shown in Table 15. Note, execution of all CA's was performed on the same Pentium 4, 2.25 Ghz, 512 MB RAM machine.

Table 15: CCD Properties for BLISS.

Discipline / Tool	No. Variables	No. of Runs	Exceution Time (min)
Propulsion / REDTOP	6	45	40 - 60
Performance / POST	5	27	3 - 4
Weights & Sizing / Excel MER's	8	81	1 - 2

9.2 BLISS: Formal Problem Statement

BLISS-2000 has both system and local-level optimization.

9.2.1 BLISS: System Standard Form

Minimize:	Φ_{Sys}	W_{dry}	
Subject to:	h	$\mathbf{Y}^* = \mathbf{Y}\hat{}_o(\mathbf{X}_{sys})$	
By Changing:	$\mathbf{X}_{sys}$	$\mathbf{X}_{sh}$	r
		$\mathbf{Y}^*$	Isp_{vac}, T_{vac}, A_e, T_{SL}/W_{eng}, S_{ref}, W_{gross}, $T_{SL,req}$, MR_{req}
		w	w1[Isp_{vac}], w2[T_{vac}], w3[A_e], w4[T_{SL}/W_{eng}], w5[W_{dry}], w6[S_{ref}], w7[W_{gross}], w8[$T_{SL,Req}$]

Note: for w1…w8 the [ ]'s show the variable to which the weighting factor corresponds

9.2.2 BLISS: Propulsion Standard Form

Minimize:	$\mathbf{\Phi}_{Prop}$	$w1(Isp_{vac}) + w2(T_{vac}) + w3(A_e) + w4(T_{SL}/W_{eng})$
Subject to:	g	$p_e \geq 5$ psia
	h	$T_{SL,avail} = T_{SL,req}$
	Side	$4 \leq r \leq 10$
		$30 \leq \varepsilon \leq 100$
		$200 \leq p_c \leq 3100$ psia
Given as Parameter:	$\mathbf{X}_{sh}$	r
	Y*	$T_{SL,req}$
	w	w1, w2, w3, w4
Find:	$\mathbf{Y}^{\wedge}_{o}$	Isp_{vac}, T_{vac}, A_e, T_{SL}/W_{eng}
By Changing:	$\mathbf{X}_{loc}$	ε, p_c, A_t

9.2.3 BLISS: Performance Standard Form

Minimize:	Φ_{Perf}	MR_{req}
Subject to:	h	$h_{insertion} = 303805$ ft
		$i_{insertion} = 51.6°$
		$\gamma_{insertion} = 0°$
Given as Parameter:	**Y***	Isp_{vac}, T_{vac}, A_e, S_{ref}, W_{gross}
Find:	$\mathbf{Y}^{\wedge}_{o}$	MR_{req}
By Changing:	$\mathbf{X}_{loc}$	$\dot{\theta}_{Azimuth}$, $\dot{\theta}_{Pitch1}$, $\dot{\theta}_{Pitch2}$, $\dot{\theta}_{Pitch3}$, $\dot{\theta}_{Pitch4}$

9.2.4 BLISS: Weights and Sizing Standard Form

Minimize:	$\mathbf{\Phi}_{W\&S}$	$w5(W_{dry}) + w6(S_{ref}) + w7(W_{gross}) + w8(T_{SL,req})$
Subject to:	h	$MR_{avail} = MR_{req}$
	Side	$T_{SL}/W_{gross} \geq 1.2$
Given as Parameter:	$\mathbf{X}_{sh}$	r
	Y*	T_{SL}/W_{eng}, A_e, MR_{req}
	w	w5, w6, w7, w8
Find:	$\mathbf{Y}\hat{}_o$	W_{dry}, S_{ref}, W_{gross}, $T_{SL,req}$
By Changing:	$\mathbf{X}_{loc}$	SF, T_{SL}/W_{gross}

9.3 BLISS: Data Flow

In order to better understand the coupling of design variables the design structure matrix (DSM) for BLISS is provided. Also provided is a variable table, proposed by Dr. Sobieski of NASA Langley, which helps to quickly observe the important variables for each local-level discipline.

9.3.1 BLISS: Design Structure Matrix

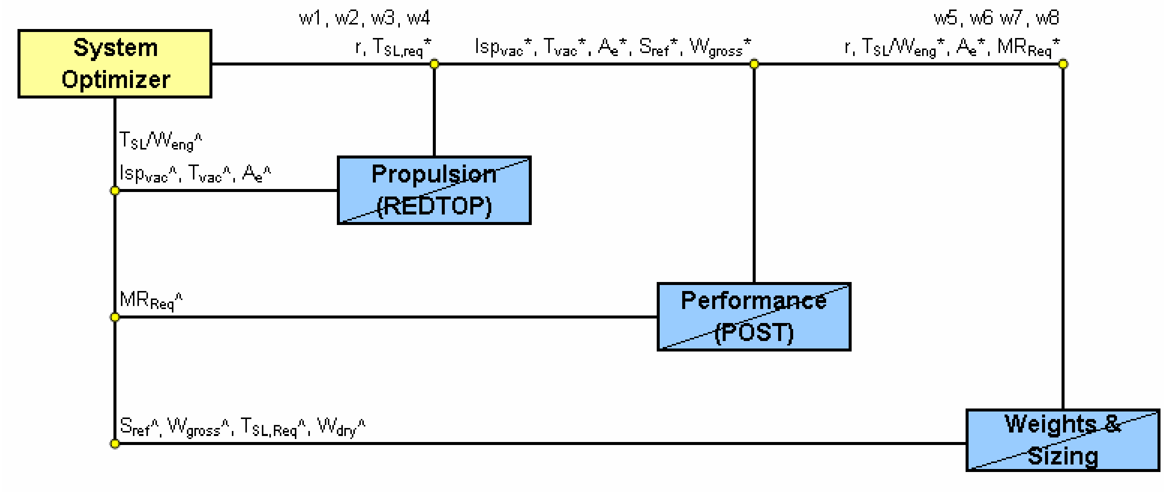

Figure 9: DSM for BLISS.

Note that forward and backward loops between the discipline CA's have been broken. This allows for parallel execution of the CA's.

9.3.2 BLISS: Variable Table

Table 16: Variable Table for BLISS.

Contributing Analysis	Output Variables	Input Variables X_{sh}	X_{loc}	Y^*	w	Φ_{loc}	Local Constraints
1. Propulsion	1. Isp_{vac} 2. T_{vac} 3. A_e 4. T_{SL}/W_{eng}	r	ε, p_c, A_t	$T_{SL,req}$	w1, w2, w3, w4	w1(Isp_{vac}) + w2(T_{vac}) + w3(A_e) + w4(T_{SL}/W_{eng})	1. $p_e \geq 5$ (psia) 2. $T_{SL,req} = T_{SL,avail}$ 3. $30 \leq \varepsilon \leq 100$ 4. $4 \leq r \leq 10$ 5. $200 \leq p_c \leq 3100$ (psia)
2. Performance	1. MR_{req}		$\theta_{Azimuth}$, θ_{Pitch1}, θ_{Pitch2}, θ_{Pitch3}, θ_{Pitch4}	Isp_{vac}, T_{vac}, A_e, S_{ref}, W_{gross},		(MR_{req})	1. $h_{insertion} = 303805$ (ft) 2. $i_{insertion} = 51.6$ (deg) 3. $\gamma_{insertion} = 0$ (deg)
3. Weights	1. S_{ref} 2. W_{gross} 3. $T_{SL,req}$ 4. W_{dry}	r	SF, T_{SL}/W_{gross}	T_{SL}/W_{eng}, A_e, MR_{Req}	w5, w6, w7, w8	w5(W_{dry}) + w6(S_{ref}) + w7(W_{gross}) + w8($T_{SL,req}$)	1. $MR_{req} = MR_{avail}$ 2. $T_{SL}/W_{gross} \geq 1.2$

9.4 BLISS: Results

This section shows the final configuration results for the RLV test problem solved using the BLISS-2000 method and describes the convergence errors resulting from this model.

9.4.1 BLISS: Configuration Results

As previously mentioned, the BLISS-2000 algorithm calls for RSM's to be created for each CA and then for the RSM's to be the ones actually used when solving the MDO problem.

The RSM's though do not include internal variables used within the disciplines but not passed on to others, thus, once the RSM's found the correct configuration, each of the tools was run one more time to get internal values and also as a quick check on the fidelity of the RSM fits used.

Table 17: Final Configuration Table for BLISS.

	BLISS-2000 RSM	BLISS-2000 Actual	Units
System Objective	W_{dry}	W_{dry}	
Direction	Minimize	Minimize	
T_{SL}/W_{eng}	59.52	59.14	
Isp_{vac}	440.8	441.9	s
c*		7,358	ft/s
Isp_{SL}		381.6	s
Cf_{vac}		1.892	
Cf_{SL}		1.634	
T_{SL}/W_{gross}	**1.200**	**1.200**	
ε	52.9	55.6	
r	7.22	7.22	
p_c		**3,100.0**	psia
p_e		**4.999**	psia
A_e	275.76	279.91	ft^2
A_t	5.22	5.03	ft^2
SF	1.079	1.079	
MR	8.03	8.03	
S_{ref}	4,659	4,659	ft^2
W_{gross}	3,124,072	3,124,062	lbf
W_{dry}	305,661	305,660	lbf

BOLD values represent active contraints

One can see that the RSM models created were of high fidelity. Also, the optimum W_{dry} calculated by the RSM's was virtually identical to the actual value output by the original legacy codes.

9.4.2 BLISS: Convergence Error

BLISS has both local and system-level optimization thus there is both intradisciplinary and interdisciplinary convergence error.

Convergence error analysis was performed using the actual values as outputted by the legacy tools, not those estimated via RSM's. This ensures that the errors due to RSE fits are accounted in the convergence error.

Table 18: Convergence Error for BLISS Configuration.

Variable	Problem Requirement	System	Propulsion	Performance	Weights	Max Abs % Error
Intradisciplinary						
$\gamma_{insertion}$	0+/-0.001			-0.0001		N/A
$h_{insertion}$	303805+/-100			303824		0.006%
$i_{insertion}$	51.6+/-0.1			51.60		0.000%
Interdisciplinary						
A_e		277.15	279.91	277.15	277.15	0.987%
Isp_{vac}		441.360	441.918	441.360		0.126%
MR		8.019		8.032	8.019	0.158%
r		7.225	7.225		7.225	0.000%
S_{ref}		4661.6		4661.6	4654.9	0.143%
T_{SL}		3748492	3748490			0.000%
T_{SL}/W_{eng}		60.57	59.14		60.57	2.430%
T_{vac}		4330789	4340595	4330789		0.226%
W_{gross}		3123575		3123575	3124313	0.024%

Red underline signifies the value used to normalize in the calculation of % error.

While the compatibility error of BLISS is not as low as that resulting from application of the FPI method (see Table 9 page 32) or AAO (see Table 13 page 43) it is still sufficiently low.

The acceptable level of this error is determined by the sensitivity of down-stream analysis to the specific variable. In this case the largest error was T_{SL}/W_{eng} which is used as an input in the Weights & Sizing analysis. Weights & Sizing, though, is not very sensitive the error in T_{SL}/W_{eng} and showed negligible effects when the T_{SL}/W_{eng} value was changed by 2.4%, the error observed.

For this study optimization parameter conditioning was performed in order to minimize errors observed in the model. Due to internal tolerances within the legacy tools and errors in the RSM fit, though, a point was reached where seemingly errors could not be driven any lower with the available computational tools. It is possible that there is some set of optimization parameters that would have produced somewhat lower errors.

9.5 BLISS: Discussion & Qualitative Ratings

This section will address BLISS optimization conclusions, lessons learned when implementing the method and provide qualitative ratings with discussion for each.

9.5.1 BLISS: Optimization Conclusions

BLISS is a multi-level MDO optimization technique and thus has not gained wide acceptance within industry. The work presented here, though, found that BLISS could successfully find the global optimum of the test RLV problem with similar convergence accuracy compared to the more accepted AAO. There is still to little data available, though, to claim BLISS a viable alternative to AAO but the observations made during this study were promising.

BLISS found the RLV's optimum at $W_{dry} \approx 305$ klb. This is virtually identical to the value found using AAO* (see Table 1 page 12). This is a 4% improvement over the best FPI configuration (see Table 8 page 31).

9.5.2 BLISS: Lessons Learned

Lessons learned in the implementation of the BLISS-2000 process are as follows:

1) There is a need to improve current framework integration tools so that future BLISS models can evaluate the parallel execution benefits of the algorithm. If simultaneous execution of all the CA runs needed for the DOE's were possible, this would lead to big time savings.

* A small 0.1% discrepancy between the two W_{dry} values resulting from AAO and BLISS can be accounted by POST internal tolerances. AAO's insertion orbit (303,706 ft) was near the tolerance minimum allowed by the user in POST (303,805±100 ft) while in the BLISS orbit was slightly above (303,824 ft) the required orbit. This is not a discrepancy between algorithms, it is entirely an intradisciplinary convergence error within performance.

2) BLISS showed a very significant implementation advantage in the RLV test problem due to the fact that the most troublesome legacy tool, POST, had only one coupling output variable (MR_{req}). This meant that the POST legacy tool be used completely unaltered without so much as having to add the weighting factors usually needed fo the system-level optimization process.
3) It was comparatively simple to make CA alterations needed to include the new composite objectives that included weighting factors. This is because, besides changing the local objective, analysis was performed in the same way as before. There were still the same local design variables and none were added.
4) The creation of RSM's could be facilitated through the use of software that automatically creates them within the framework tool. Such a tool, ProbWork's RSE Generator, was used in this project (page 19). This eliminated the need to transfer data between to statistical software to make the RSE's and avoided errors due to data transfer.
5) It is important to allow both negative and positive values for the weighting factors (w's). This allows for the system optimizer to choose if a local output should be minimized or maximized.
6) There is a system design space singularity when all w's equal zero. This is because all local outputs are multiplied by zero and the local objectives vanish. Good results were obtained when all but one w was initialized with a value of zero. This was circumvented by initializing the w's in the system-level optimizer to all zeroes except one. Then the variable that received an initial w value was varied to ensure that the true minimum was being reached. When this is done, it is highly unlikely that the system optimizer will encounter the <u>all</u> w = 0 singularity.

9.5.3 BLISS: Qualitative Ratings

It is very difficult to qualitatively assess the experience of implementing a given MDA or MDO technique without first comparing it to the other techniques to be evaluated. Therefore these ratings were assigned after all the techniques had been applied.

Table 19: Qualitative Ratings for BLISS.

Criteria	Grade	Discussion
Implementation Difficulty	B+	While use of composite local-level objectives may require the legacy codes to be modified, the changes are small and relatively easy to implement.
Total Execution Time	B+	(2 to 3 hours) (10 to 20 minutes possible) Due to software limitations, the benefits of parallel computing described in the BLISS literature were not realized. This reality is reflected in the 2 to 3 hour convergence time. If the capacity for parallel execution of the RSM's while solving the MDO problem was added the process would only see a marginal benefit due to the rapid execution of the RSM's. If the capacity for parallel execution were added during the data gathering part of the DOE, though, a very large time savings would incur. If this capability were available, then the convergence time would be about 10 to 20 minutes.
Robustness	B+	Converged adequately from given initialization point (best FPI configuration), but if the RSM's are a poor fit problems may incur. Also, does have vanishing local objective when all weighting factors (w's) are equal to zero.
Formulation Difficulty	B+	Not as straight forward as FPI or AAO but still changes are not very large. It does, though, require a separate formulation for each discipline and the system.
Optimization Deftness	A	May be able to handle more complex problems than AAO, but still may run into constraints if a lot of weighting variables are needed or there are a lot of shared variables. This was the only MDO technique which solved the "real world" RLV MDO problem with the accuracy and efficiency predicted in theory.

Convergence Error	B+	Letting local optimizers control local design variables improves convergence of those variables. There may be increased error, though, due to the RSM fit and parallel execution.
Issues		1) If one legacy tool takes a lot longer to create an RSM than others, then parallelization is of little help 2) Every CA should not be made into an RSM. Some codes may run quickly and do not have numerical problems. 3) It may take several BLISS iterations to converge and have a good RSE fit. In this study the RSM fit was good and the data points were accurate thus it only required two BLISS iterations to reach a tightly converged system with little error.
Unique Benefits		1) If a discipline has only one shared output, then that discipline's formulation is the same as FPI and the original legacy tool can be used unaltered. 2) Use of RSM's eliminates numerical problems present in the legacy codes. This allows for high accuracy when taking numerical derivatives. 3) May allow use of the legacy code's original optimizer which has already been conditioned to best handle the discipline's given problem.

10 CO: Collaborative Optimization

Collaborative Optimization (CO) is a two-level MDO algorithm originally developed by Dr. Robert Braun for his Ph.D. dissertation at Stanford University.[1] Being a multi-level optimization algorithm there is optimization in more than one level of the entire RLV design, as opposed to FPI which only has local optimization or AAO which only has system-level optimization.

This section will provide some background information about CO and its relevance to this study; show the formulation when it is applied to the RLV problem, present the results gathered and discuss findings.

10.1 CO: Background

CO, along with BLISS and MCO, is one of the most promising multi-level MDO techniques. CO was proposed in 1996 as a technique best suited for "large scale distributed design." Thus, like BLISS, it is expected to scale better than AAO as the size and complexity of an MDO problem increases.

As with other multi-level MDO algorithms proposed, one of the greatest impediments in the acceptance of CO is the fact that it was "crafted" as opposed to "rigorously derived."[9] Thus it requires studies applying it to realistic test problems, like the one presented here, to gain industry acceptance.

CO, like other multi-level MDO techniques, is expected to offer the benefits of removing large iteration loops, allowing the expert disciplinary design teams to have a large amount of design freedom and parallel execution of the disciplinary analysis. Also, like BLISS, CO's two-level structure is similar to conventional disciplinary structures currently used in industry.

In order to be able to coordinate between disciplines and arrive at an optimum MDO solution, CO creates copies of all the interdisciplinary coupling variables at the system-level. The system-level optimizer then uses these copies to send out design targets to each discipline. There may not exist sufficient local degrees of freedom to satisfy all the targets while meeting local constraints, therefore the local-level subspaces are allowed to depart from the targets but this departure is to be minimized. In theory, if there are enough local-level degrees of freedom, the variable targets and disciplinary values will converge or match. Of course, in practical applications there may be some error when trying to converge these values.

In CO, as in MCO, the system-level sends variable targets to any CA's that deal with the variable either as inputs or calculate their actual values as part of their analysis. For example, in the original FPI DSM (see page 27) the variable A_e was an output calculated by the Propulsion and an input used by both Performance and Weights & Sizing. In CO, the system would set a target value for A_e. Propulsion would optimize its input variables to match as closely as possible the A_e target along with any other discipline target (Isp_{vac}, T_{SL}, etc.). Both Performance and Weights & Sizing will optimize their input variables (including the local version of A_e) so as to match as closely as possible the A_e target along with their other discipline target (W_{gross}, MR, etc.).

In CO the local objectives (Φ_{Prop}, Φ_{Perf} and $\Phi_{W\&S}$) are formulated so as the local-level optimizer moves closer to matching their targets the local objective function decreases. Two different formulations for reaching this goal are were considered as follows:

$$\Phi_{loc} = \sum_{i=1}^{\#\text{targets}} \left(1 - \frac{X_i^{loc}}{X_i^t} \right)^2$$

Equation 4: Φ_{loc} formulation without normalization constant

$$\Phi_{loc} = \sum_{i=1}^{\#\text{targets}} \left(\frac{X_i^t - X_i^{loc}}{C_i} \right)^2$$

Equation 5: Φ_{loc} formulation with a normalization constant

C_i = user defined normalization constant for i^{th} coupling variable

While both equations above have similar local convergence in that Φ_{loc} will minimize to value of zero when all the targets are met, the normalization is a little different. Equation 4 divides the local value of the design variable (X^{loc}) by the given target value (X^t) when their ratio is 1 (they are equal) then the equation is zero. In Equation 5, the difference between X^{loc} and X^t calculated and constant (C) is used for normalization purposes. The value for C is set by the model developer during the initial problem formulation. The value for C_i should be something with the same order of magnitude as the anticipated value of X_i.

10.2 CO: Formal Problem Statement

CO has both system and local-level optimization.

10.2.1 CO: System Standard Form

Minimize:	Φ_{sys}	W_{dry}
Subject to:[*]	h	$\Phi_{Prop} + \Phi_{Perf} + \Phi_{W\&S} = 0$
By Changing:	$\mathbf{X}_{sys}$	$r^t, Isp^t_{vac}, T^t_{vac}, A^t_e, (T_{SL}/W_{eng})^t, S^t_{ref}, W^t_{gross}, T^t_{SL}, MR^t$

[*] Note: Often the system equality constraint (h) could separated for each discipline ($\Phi_{loc,i} = 0$) instead of the single constraint formulation used here ($\sum\Phi_{loc,i} = 0$). During the course of this study both formulations were applied. Both formulations are fine as they both accomplish the same goals. During this study both formulations were tried; the single formulation was found to give better converged results for this test problem.

10.2.2 CO: Propulsion Standard Form

Minimize: Φ_{Prop}

$$\left(\frac{r^{t}-r^{pp}}{6}\right)^2+\left(\frac{T_{SL}^{t}-T_{SL}^{pp}}{3500000}\right)^2+\left(\frac{Isp_{vac}^{t}-Isp_{vac}^{pp}}{400}\right)^2+\left(\frac{T_{vac}^{t}-T_{vac}^{pp}}{4000000}\right)^2+\left(\frac{A_{e}^{t}-A_{e}^{pp}}{250}\right)^2+\left(\frac{T_{SL}/W_{e}{}^{t}-T_{SL}/W_{e}{}^{pp}}{50}\right)^2$$

Subject to: g: $p_e \geq 5$ psia

Side: $4 \leq r \leq 10$

$30 \leq \varepsilon \leq 100$

$200 \leq p_c \leq 3100$ psia

Given as Target: $\mathbf{X}_{sys}$: $r^{t}, Isp_{vac}^{t}, T_{vac}^{t}, A_{e}^{t}, (T_{SL}/W_{eng})^{t}, T_{SL}^{t}$

Find: $\mathbf{Y}_{loc}$: $Isp_{vac}^{pp}, T_{vac}^{pp}, A_{e}^{pp}, (T_{SL}/W_{eng})^{pp}, T_{SL}^{pp}$

By Changing: $\mathbf{X}_{loc}$: r^{pp}, ε, p_c, A_t

10.2.3 CO: Performance Standard Form

Minimize: Φ_{Perf}

$$\left(1-\frac{Isp_{vac}^{pf}}{Isp_{vac}^{t}}\right)^2+\left(1-\frac{T_{vac}^{pf}}{T_{vac}^{t}}\right)^2+\left(1-\frac{A_{e}^{pf}}{A_{e}^{t}}\right)^2+\left(1-\frac{S_{ref}^{pf}}{S_{ref}^{t}}\right)^2+\left(1-\frac{W_{gross}^{pf}}{W_{gross}^{t}}\right)^2+\left(1-\frac{MR^{pf}}{MR^{t}}\right)^2$$

Subject to: h: $h_{insertion} = 303805$ ft

$i_{insertion} = 51.6\ °$

$\gamma_{insertion} = 0\ °$

Given as Target: $\mathbf{X}_{sys}$: $Isp_{vac}^{t}, T_{vac}^{t}, A_{e}^{t}, S_{ref}^{t}, W_{gross}^{t}, MR^{t}$

Find: $\mathbf{Y}_{loc}$: MR^{pf}

By Changing: $\mathbf{X}_{loc}$: $Isp_{vac}^{pf}, T_{vac}^{pf}, A_{e}^{pf}, S_{ref}^{pf}, W_{gross}^{pf}, \dot{\theta}_{Azimuth}, \dot{\theta}_{Pitch1}, \dot{\theta}_{Pitch2}, \dot{\theta}_{Pitch3}, \dot{\theta}_{Pitch4}$

10.2.4 CO: Weights and Sizing Standard Form

Minimize: $\Phi_{W\&S}$

$$\left(1-\frac{A_e^{ws}}{A_e^t}\right)^2+\left(1-\frac{T_{SL}/W_e^{\ ws}}{T_{SL}/W_e^{\ t}}\right)^2+\left(1-\frac{r^{ws}}{r^t}\right)^2+\left(1-\frac{S_{ref}^{ws}}{S_{ref}^t}\right)^2+\left(1-\frac{W_{gross}^{ws}}{W_{gross}^t}\right)^2+\left(1-\frac{T_{SL}^{ws}}{T_{SL}^t}\right)^2+\left(1-\frac{MR^{ws}}{MR^t}\right)^2$$

Subject to: Side $T_{SL}/W_{gross} \geq 1.2$

Given as Target: $\mathbf{X}_{sys}$ $r^t, A_e^t, (T_{SL}/W_{eng})^t, S_{ref}^t, W_{gross}^t, T_{SL}^t, MR^t$

Find: $\mathbf{Y}_{loc}$ $S_{ref}^{ws}, W_{gross}^{ws}, T_{SL}^{ws}, MR^{ws}$

By Changing: $\mathbf{X}_{loc}$ $r^{ws}, A_e^{ws}, T_{SL}/W_{eng}^{\ ws}, SF, T_{SL}/W_{gross}$

10.3 CO: Data Flow

In order to better understand the coupling of design variables the design structure matrix (DSM) for CO is provided. Also provided is a variable table which helps to quickly observe the important variables for each local-level discipline.

10.3.1 CO: Design Structure Matrix

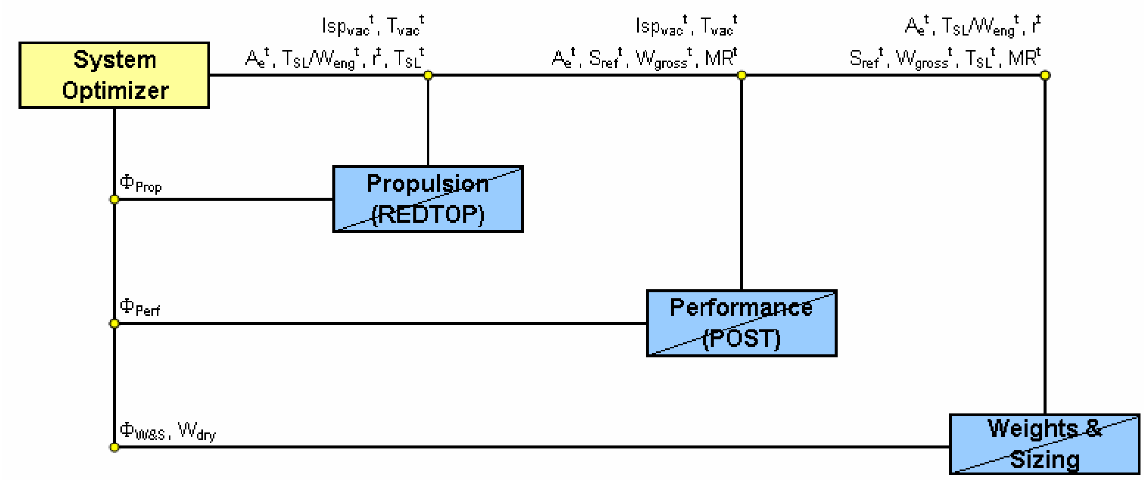

Figure 10: DSM for CO.

From the DSM, one can see that local versions (X^{pp}, X^{pf} or X^{ws}) of each target variable is not directly inputted to the DSM, but only the $\Phi_{loc,i}$ is looped back to the system-level. Thus, like in BLISS, parallel execution of the CA's is possible.

10.3.2 CO: Variable Table

Table 20: Variable Table for CO.

Contributing Analysis	Output Variables: To System	Output Variables: Y_{loc}	Input Variables: X_{loc}	Input Variables: X_{sys}	Φ_{loc}	Local Constraints
1. Propulsion	Φ_{Prop}	Isp_{vac}^{PP}, T_{vac}^{PP}, A_e^{PP}, T_{SL}/W_{eng}^{PP}, T_{SL}^{PP}	r^{PP}, ε, p_c, A_t	Isp_{vac}^t, T_{vac}^t, A_e^t, T_{SL}/W_{eng}^t, r^t, T_{SL}^t	$[(r^{PP}-r^t)/6]^2 + [(Isp_{vac}^{PP}-Isp_{vac}^t)/400]^2$ $+ [(T_{vac}^{PP}-T_{vac}^t)/4000000]^2$ $+ [(A_e^{PP}-A_e^t)/250]^2$ $+ [((T_{SL}/W_{eng})^{PP}-(T_{SL}/W_{eng})^t)/50]^2$ $+ [(T_{SL}^{PP}-T_{SL}^t)/3500000]^2$	1. $p_e \geq 5$ (psia) 2. $30 \leq \varepsilon \leq 100$ 3. $4 \leq r \leq 10$ 4. $200 \leq p_c \leq 3100$ (psia)
2. Performance	Φ_{Perf}	MR^{pf}	Isp_{vac}^{pf}, T_{vac}^{pf}, Ae^{pf}, S_{ref}^{pf}, W_{gros}^{pf}, $\theta_{Azimuth}$, θ_{Pitch1}, θ_{Pitch2}, θ_{Pitch3}, θ_{Pitch4}	Isp_{vac}^t, T_{vac}^t, A_e^t, S_{ref}^t, W_{gross}^t, MR^t	$[1-(Isp_{vac}^{pf}/Isp_{vac}^t)]^2$ $+ [1-(T_{vac}^{pf}/T_{vac}^t)]^2 + [1-(A_e^{pf}/A_e^t)]^2$ $+ [1-(S_{ref}^{pf}/S_{ref}^t)]^2$ $+ [1-(W_{gross}^{pf}/W_{gross}^t)]^2$ $+ [1-(MR^{pf}/MR^t)]^2$	1. $h_{insertion} = 50$ (nmi) 2. $i_{insertion} = 51.6$ (deg) 3. $\gamma_{insertion} = 0$ (deg)
3. Weights	$\Phi_{W\&S}$, W_{dry}	S_{ref}^{ws}, W_{gross}^{ws}, T_{SL}^{ws}, MR^{ws}	A_e^{ws}, T_{SL}/W_{eng}^{ws}, r^{ws}, SF, T_{SL}/W_{gross}	A_e^t, T_{SL}/W_{eng}^t, r^t, S_{ref}^t, W_{gross}^t, T_{SL}^t, MR^t	$[1-(A_e^{ws}/A_e^t)]^2$ $+ [1-((T_{SL}/W_{eng})^{ws}/(T_{SL}/W_{eng})^t)]^2$ $+ [1-(r^{ws}/r^t)]^2 + [1-(MR^{ws}/MR^t)]^2$ $+ [1-(S_{ref}^{ws}/S_{ref}^t)]^2$ $+ [1-(W_{gross}^{ws}/W_{gross}^t)]^2$ $+ [1-(T_{SL}^{ws}/T_{SL}^t)]^2$	1. $T_{SL}/W_{gross} \geq 1.2$

10.4 CO: Results

This section shows the final configuration results for the RLV test problem solved using the CO method and describes the convergence errors resulting from this model.

10.4.1 CO: Configuration Results

Table 21: Final Configuration Table for CO.

	CO	Units
System Objective	W_{dry}	
Direction	Minimize	
T_{SL}/W_{eng}	59.34	
Isp_{vac}	440.8	s
c*	7,334	ft/s
Isp_{SL}	380.5	s
Cf_{vac}	1.892	
Cf_{SL}	1.633	
T_{SL}/W_{gross}	**1.200**	
ε	54.6	
r	7.31	
p_c	3,036.2	psia
p_e	5.077	psia
A_e	280.06	ft^2
A_t	5.13	ft^2
SF	1.075	
MR	7.96	
S_{ref}	4,624	ft^2
W_{gross}	3,097,190	lbf
W_{dry}	303,663	lbf

BOLD values represent active contraints

One can see the MDO solution obtained through applying CO does not activate the constraints for p_c and p_e, this is different than MDO solutions obtained by applying AAO (see Table 12 page 42) and BLISS (see Table 1 page 55). Also of interest is that the value of the global objective obtained by applying CO, $W_{dry} \approx 303$ klb, is somewhat lower than the value arrived at both by AAO and BLISS, $W_{dry} \approx 305$ klb. At first one might think that CO did a slightly better job than the previous two, actually this lower value is due to large convergence error in the CO model.

10.4.2 CO: Convergence Error

In CO there is both local and system-level optimization; thus there is both intradisciplinary and interdisciplinary convergence error.

Table 22: Convergence Error for CA Configuration.

Variable	Problem Requirement	System	Propulsion	Performance	Weights	Max Abs % Error
Intradisciplinary						
$\gamma_{insertion}$	0+/-0.001			-0.0009		N/A
$h_{insertion}$	303805+/-100			303755		0.017%
$i_{insertion}$	51.6+/-0.1			51.60		0.000%
Interdisciplinary						
A_e		280.82	280.06	281.55	280.83	0.532%
Isp_{vac}		441.835	440.781	443.072		0.520%
MR		7.946		7.964	8.000	0.677%
r		7.288	7.309		7.257	0.710%
S_{ref}		4658.7		4799.4	4623.8	3.799%
T_{SL}		3686193	3740517		3716628	1.452%
T_{SL}/W_{eng}		59.96	59.37		59.89	0.985%
T_{vac}		4397268	4332952	4450220		2.706%
W_{gross}		3110988		3211326	3097190	3.685%

Red underline signifies the value used to normalize in the calculation of % error.

The convergence in CO is not as tight as that achieved with FPI, AAO nor BLISS. There are large interdisciplinary errors reaching almost 4%. More troubling is larger error in coupling variables to which the system is highly sensitive such as Isp_{vac}.

One of the well known causes for this loose convergence in CO is due to the sum-of-squares formulation of the Φ_{loc} used for this MDO technique. The use of a quadratic form for the formulation of Φ_{loc} means that the CA's local optimizers approach their desired solution (local values = targets), it loses its ability to compute gradients and cannot tell how to vary the design variables to minimize the local objective. As X^{loc} approaches X^t the gradient of Φ_{loc} approaches zero meaning that the Jacobian matrix built to calculate local gradients vanishes.

This vanishing Jacobian effect is analogous to a the behavior of a tension spring. When the difference between X^{loc} and X^t is high (left side of Figure 11) the spring (analogous to the local optimizer) is stretched and has a high tension driving the design variable values down to the Φ_{loc} minimum. Here there is a large Φ_{loc} gradient thus gradient optimizers can easily tell which direction to go in order to minimize the function. As the optimizer gets closer to the desired target values the difference between X^{loc} and X^t decreases (right side of Figure 11). Using the spring analogy one can see that as this happens the tension of the spring quickly drops and it is not able to drive the function to the minimum as well as before. Here there is a small Φ_{loc} gradient thus gradient optimizers have a harder time telling which direction to go in order to minimize the function. How well the optimizer finds the exact minimum depends on how robust the optimizer is and how much noise there is in the model.

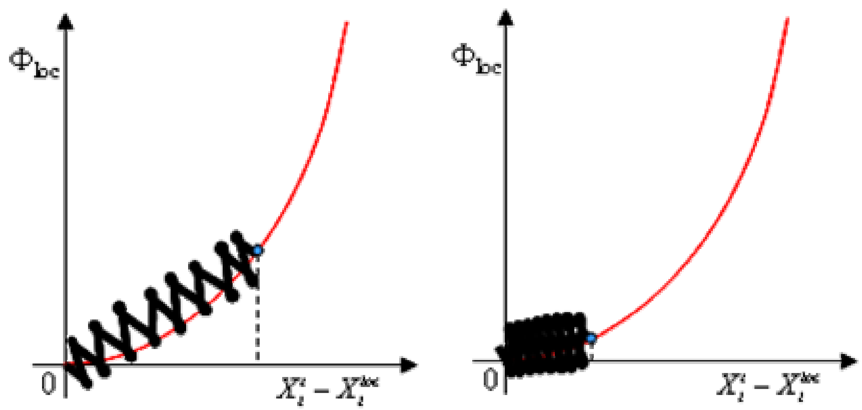

Figure 11: Spring Analogy of Singular Jacobian Phenomenon.

The vanishing Jacobian effect was noticed in the early days of CO. Later derivatives of CO, like Modified Collaborative Optimization (MCO), have attempted to avoid or ameliorate errors caused by this effect.

Besides the vanishing Jacobian effect, another phenomenon, an "opportune error direction effect", was observed and explains why the optimum value of W_{dry} obtained using CO was less than that obtained through the application of both AAO and BLISS.

The "opportune error direction effect" observed occurs when the system-level optimizer chooses the direction of local convergence errors with respect to variable targets such as to artificially benefit its global objective.

In CO each local-level optimization tries to match all its coupling variables to system targets. But while the target is the same for any given variable (for example Isp), each local optimization tries to match the targets independently. This independence may result in the CA's involved to all have a different value for their local version for the same coupling variable.

The system optimizer takes advantage that the local optimizers can have different errors with respect to given targets and chooses the most opportune error directions such as to benefit the system-level goal of minimizing W_{dry}.

Figure 12 below shows an example of the "opportune error direction effect" encountered during this study.*

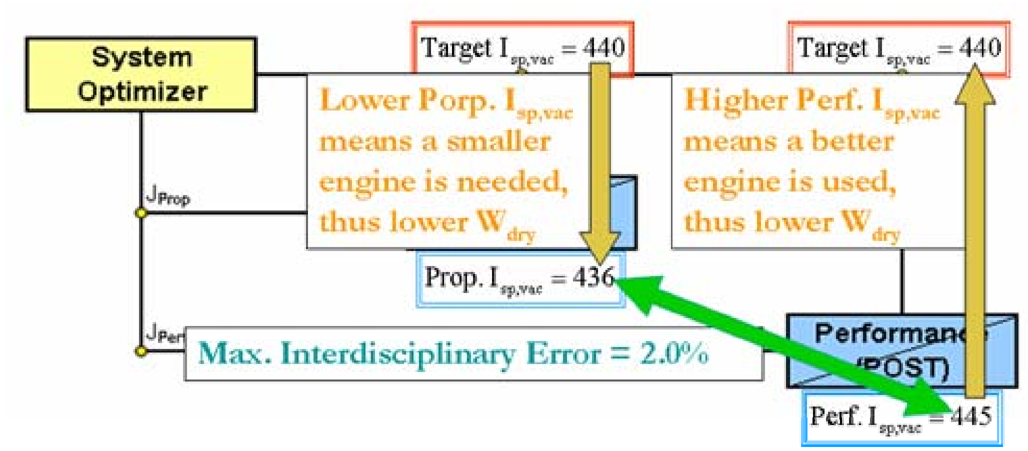

Figure 12: Example of Opportune Error Direction Effect, Isp_{vac}.

In the diagram above, Isp_{vac} is calculated by Propulsion and used by Performance to do its trajectory analysis. The system optimizer sends out a target value for Isp (440 in the example above). Because of the vanishing Jacobian effect, numerical errors, etc., there will be a convergence error between the system target and the value used by the local discipline. This system-discipline error is only 0.9% for Propulsion, (440-436)/440, and 1.1% for Performance, (445-440)/440. Since local convergence error is inevitable (due to internal tolerances, software limitations, etc.), the system optimizer takes advantage of them. It chooses the most opportune error directions to help with the minimization of W_{dry}. This means that propulsion will actually arrive at an Isp below the target and performance above. In reality, though, it means that the two disciplines have a

* The values for Isp_{vac} in Figure 12 are notional and while they show the tendencies observed in this study the exact values are not meant to be representative of those viewed during CO or MCO applications.

larger convergence error between them (2%) that what they had versus the target the local optimizers were trying to match.

The percent errors and directions observed with respect to their desired targets for the CO configuration can be observed in Figure 13 below.

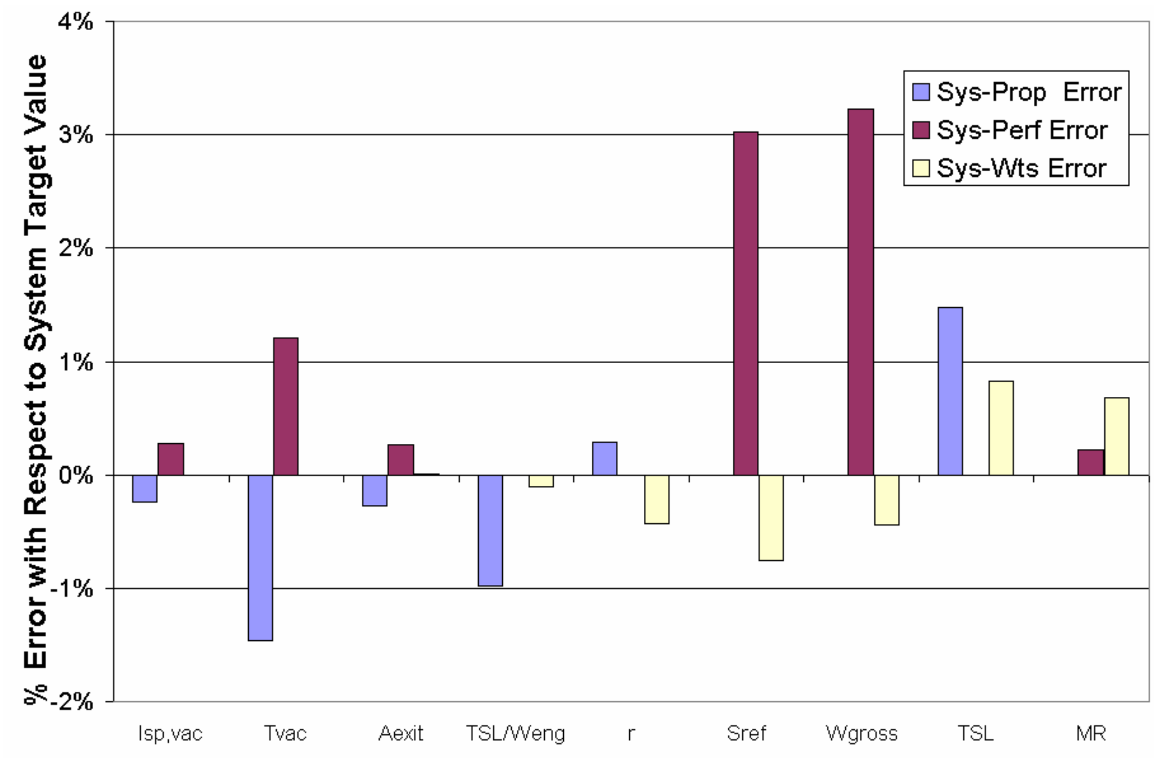

Figure 13: % Error and Direction with Respect to System Targets for CO Configuration.

For this study optimization parameter conditioning was performed in order to minimize errors observed in the model. Due to internal tolerances within the legacy tools and the vanishing Jacobian effect a point was reached where seemingly convergence errors could not be driven any lower with the available computational tools. It is possible that there is some set of optimization parameters that would have produced lower errors. More time was spent trying to drive down the convergence errors in CO than was used in

BLISS, yet the BLISS algorithm yielded smaller converged error. It also appeared that CO was more sensitive to optimizer parameter conditioning than BLISS.

10.5 CO: Discussion & Qualitative Ratings

This section will address CO optimization conclusions, lessons learned when implementing the method and provide qualitative ratings with discussion for each.

10.5.1 CO: Optimization Conclusions

CO is a multi-level MDO optimization technique and like other such techniques has not gained wide acceptance within industry. The results of this study found that due to the vanishing or singular Jacobian effect it was very difficult to very tightly converge the coupling variable local values to the system-level target values. Also, this study observed an "opportune error direction effect" which led to CO producing artificially optimistic values for W_{dry}.

One possible way to mitigate convergence problems is to reformulate the local objective functions such as to increase their smoothness and remove the Jacobian singularity. This is one of the goals of MCO, which reformulates the local objective functions. Of course another option that may result in reduced convergence errors is the use a different optimizer. In practice different optimizers will have different levels of success when trying to precisely match the given targets. Throughout this study all the MDO techniques evaluated, AAO, CO, BLISS, and MCO, used the either the optimizer built into the legacy tools or the DOT optimizer. This was done to remove external variance that may affect the performance of different MDO techniques. Consistent tools allowed this study to use the blocking effect to remove external variances when trying to make comparisons between MDO techniques.

Using RSM's to model each discipline of legacy code has also been suggested as a possible method to mitigate some of the convergence problems.[8] Like in BLISS-2000, RSM's may be used with CO to reduce execution time and lessen numerical integration problems. The CO algorithm does not call for the making of RSM's but had they been made the number of runs needed to create these RSE's would have been very comparable to those needed in BLISS (see Table 15 page 50). The number of variables to be fitted, resulting number of CA runs needed, and the total execution time called for by the DOE's using a Central Composite Design needed in CO, are shown in Table 23 below.

Table 23: CCD Properties for CO.

Discipline / Tool	No. Variables	No. of Runs
Propulsion / REDTOP	6	45
Performance / POST	6	45
Weights & Sizing / Excel MER's	7	47

Using RSM's should allow for less numerical error and much faster execution of the local-level discipline calculations when solving the MDO problem with the CO technique. This, though, assumes that the RSM fits points which are accurate. If convergence problems due to the vanishing Jacobian effect cannot be overcome within the CA's, then the RSM will be fitting to points which themselves have a lot of error.

10.5.2 CO: Lessons Learned

Lessons learned in the implementation of the CO process are as follows:

1) There is a need to improve current framework integration tools so that future CO models can evaluate the parallel execution benefits of the algorithm. Should it be attempted to create RSM's of the disciplines before executing the CO algorithm, the ability to simultaneously calculate all the CA points needed for the DOE would result in large time savings.
2) The modifications to the original legacy codes called for by CO can be hard to implement. CO changes local-level inputs that were originally given parameters into local-level design variables. This entails that besides optimizing to a different local objective the local-level optimizer must now vary an increased number of design variables. The addition of new local design variables increased the difficulty in

conditioning the local-level optimizers. Conditioning of the local optimizers so that they produced reasonably reliable, consistent results was challenging. More time was spent trying to condition the CO model than was spent on BLISS yet convergence never reached the accuracy shown by BLISS.

10.5.3 CO: Qualitative Ratings

It is very difficult to qualitatively assess the experience of implementing a given MDA or MDO technique without first comparing it to other the other techniques to be evaluated. Therefore these ratings were assigned after all the techniques had been applied.

Table 24: Qualitative Ratings for CO.

Criteria	Grade	Discussion
Implementation Difficulty	C+	Every local-level, discipline optimizer had to be reconditioned due to poor performance after changing the local objective and increasing the number of local variables. Reconditioning each local optimizer was very time consuming. Also, due to the vanishing Jacobian effect optimization parameter conditioning was even more difficult.
Total Execution Time	B-	(3 to 4 hours) (1 to 2 hours possible) Due to software limitations, the benefits of parallel computing described in the CO literature were not realized. If the capacity for parallel execution the CA's was added there would be a significant time savings.
Robustness	B	It is difficult to converge each local optimization problem due to the vanishing Jacobian. Acceptable local-level optimization is difficult to achieve over a wide range of inputs.
Formulation Difficulty	B-	Not as straight forward as FPI or AAO or BLISS. Changes required were greater than BLISS. Both BLISS, CO and MCO require a separate formulation for each discipline and the system.

Optimization Deftness	B	The "opportune error direction effect" creates an artificially optimistic answer. If there were zero error at the discipline level then this would not be an issue. If error is present, the system optimizer uses the local target convergence error to improve the global optimum.
Convergence Error	C+	The combination of the vanishing Jacobian and more difficult optimizer conditioning caused the method to be harder to converge than FPI, AAO or BLISS.
Issues		1) Discipline level convergence errors were never successfully reduced to a level where convergence using CO was comparable to that of AAO or BLISS. This was true even though more time was spent trying to reduce convergence errors than was spent for BLISS. 2) Assuming that perfect convergence is impossible in practical applications, CO will take advantage of local-level errors to provide an artificially optimistic optimum solution. This underestimating the system mass could prove more problematic to engineering programs than if the optimum had been slightly pessimistic.
Unique Benefits		None observed.

11 MCO: Modified Collaborative Optimization

Modified Collaborative Optimization (MCO) is a two-level MDO algorithm developed by Angel Victor DeMiguel and Walter Murray at Stanford University.[2]

This section will provide some background information about MCO and its relevance to this study; show the formulation when it is applied to the RLV problem, present the results gathered and discuss findings.

11.1 MCO: Background

MCO, is a derivative of CO, suggested in 1998 as way to "surmount some deep technical challenges"[2] present in the original formulation of CO. The developers of MCO noted the lack of a proof of convergence for CO and practical application difficulties due to the vanishing Jacobian effect.

The MCO architecture is very similar to that of CO, thus if MCO was shown to be efficient and overcome CO's vanishing Jacobian effect, it would have the same inputs and outputs as CO. Also, like CO or BLISS, MCO is most ideal for large problem with costly discipline analysis but relatively few coupling variables.

The first difference between MCO and CO is that MCO uses an "exact"[2] penalty function as the system-level objective. MCO uses the local-level sub-problem objective functions as penalty terms for the system-level objective function creating an unconstrained system-level problem. This contrasts CO which imposed constraints at the system-level to enforce that the local-level sub-problem objective functions reach their expected minimums at zero.

The second difference is that at the MCO does away with the quadratic form of the local objectives and replaces it with a format similar to linear programming called Individual Discipline Feasible (IDF) formulation. As in CO, during the system-level optimization the system may call for an infeasible set of targets. For these cases, the local-level disciplines will find it impossible to match their given targets. IDF solves this problem by the introduction of additional local-level variables and constraints. In IDF, if the set of targets was infeasible, elastic variables (s and t) are added to make up the difference between the local-level values and the given targets. The local-level objective is now to minimize the amount of corrections that must be performed using the new elastic variables. All variables in the IDF formulation are required to have positive values thus elastic variables must be introduce for both when the local variable is less than (s) or greater than (t) the target value. In the IDF formulation there can only be equality constraints, thus inequality disciplinary constraints in the CA's must be changed to equality constraints by adding another elastic variable.

11.2 MCO: Formal Problem Statement

MCO has both system and local-level optimization.

Note: normalization was used for all the equality constraints used in the local-level formulations but is not shown for the sake of brevity. The format shown is $X^{loc}+s_x-t_x=X^t$ but the normalized format used for the MCO application was $(X^{loc}/C)+s_x-t_x=(X^t/C)$. The normalization constants used for each variable are shown below.

Table 25: Normalization Constants for MCO Equality Constraints.

Variable	Norm. Constant
A_e	250
$h_{insertion}$	200,000
$i_{insertion}$	40
Isp_{vac}	400
MR	7
p_e	4
r	6
S_{ref}	4,000
T_{SL}	3,500,000
T_{SL}/W_{eng}	50
T_{vac}	4,000,000
W_{gross}	3,000,000

11.2.1 MCO: System Standard Form

Minimize: Φ_{sys} $W_{dry} + r_p*(\Phi_{Prop} + \Phi_{Perf} + \Phi_{W\&S})$

Subject to:

By Changing: $\mathbf{X}_{sys}$ $r^t, Isp^t_{vac}, T^t_{vac}, A^t_e, T_{SL}/W_{eng}{}^t, S^t_{ref}, W^t_{gross}, T^t_{SL}, MR^t$

Note: r_p is a penalty parameter that can be used by the developer to alter the acceptable level of error in the CA's.

11.2.2 MCO: Propulsion Standard Form

Minimize: $\mathbf{\Phi}_{Prop}$ $(s_r+t_r) + (s_{Ispv}+t_{Ispv}) + (s_{Tvac}+t_{Tvac}) + (s_{Ae}+t_{Ae}) + (s_{Tsl/Weng}+t_{Tsl/Weng}) + (s_{Tsl}+t_{Tsl})$

Subject to: h

$$p_e^{pp} - s_{pe} - 5\text{psia} = 0$$
$$r^{pp} + s_r - t_r = r^t$$
$$Isp^{pp}_{vac} + s_{Ispv} - t_{Ispv} = Isp^t_{vac}$$
$$T^{pp}_{vac} + s_{Tvac} - t_{Tvac} = T^t_{vac}$$
$$A^{pp}_e + s_{Ae} - t_{Ae} = A^t_e$$
$$(T_{SL}/W_{eng})^{pp} + s_{Tsl/Weng} - t_{Tsl/Weng} = (T_{SL}/W_{eng})^t$$
$$T^{pp}_{SL} + s_{Tsl} - t_{Tsl} = T^t_{SL}$$

Side

$$4 \le r \le 10$$
$$30 \le \varepsilon \le 100$$
$$200 \le p_c \le 3100 \text{ psia}$$
$$s_{pe}, s_r, t_r, s_{Ispv}, t_{Ispv}, s_{Tvac}, t_{Tvac}, s_{Ae}, t_{Ae}, s_{Tsl/Weng}, t_{Tsl/Weng}, s_{Tsl}, t_{Tsl} \ge 0$$

Given as Target: $\mathbf{X}_{sys}$ $r^t, Isp^t_{vac}, T^t_{vac}, A^t_e, (T_{SL}/W_{eng})^t, T^t_{SL}$

Find: $\mathbf{Y}_{loc}$ $Isp^{pp}_{vac}, T^{pp}_{vac}, A^{pp}_e, (T_{SL}/W_{eng})^{pp}, T^{pp}_{SL}$

By Changing: $\mathbf{X}_{loc}$ $r^{pp}, \varepsilon, p_c, A_t, s_{pe}, s_r, t_r, s_{Ispv}, t_{Ispv}, s_{Tvac}, t_{Tvac}, s_{Ae}, t_{Ae}, s_{Tsl/Weng}, t_{Tsl/Weng}, s_{Tsl}, t_{Tsl}$

11.2.3 MCO: Performance Standard Form

Minimize:	Φ_{Perf}	$(s_{Ispv}+t_{Ispv}) + (s_{Tvac}+t_{Tvac}) + (s_{Ae}+t_{Ae}) + (s_{Sref}+t_{Sref}) + (s_{Wgross}+t_{Wgross}) + (s_{MR}+t_{MR})$
Subject to:	h	$h_{insertion} = 303805$ ft $i_{insertion} = 51.6\ °$ $\gamma_{insertion} = 0\ °$ $Isp^{pf}_{vac} + s_{Ispv} - t_{Ispv} = Isp^{t}_{vac}$ $T^{pf}_{vac} + s_{Tvac} - t_{Tvac} = T^{t}_{vac}$ $A^{pf}_{e} + s_{Ae} - t_{Ae} = A^{t}_{e}$ $S^{pf}_{ref} + s_{Sref} - t_{Sref} = S^{t}_{ref}$ $W^{pf}_{gross} + s_{Wgross} - t_{Wgross} = W^{t}_{gross}$ $MR^{pf} + s_{MR} - t_{MR} = MR^{t}$ $s_{Ispv}, t_{Ispv}, s_{Tvac}, t_{Tvac}, s_{Ae}, t_{Ae}, s_{Sref}, t_{Sref},$ $s_{Wgross}, t_{Wgross}, s_{MR}, t_{MR} \geq 0$
Given as Target:	$\mathbf{X}_{sys}$	$Isp^{t}_{vac}, T^{t}_{vac}, A^{t}_{e}, S^{t}_{ref}, W^{t}_{gross}, MR^{t}$
Find:	$\mathbf{Y}_{loc}$	MR^{pf}
By Changing:	$\mathbf{X}_{loc}$	$Isp^{pf}_{vac}, T^{pf}_{vac}, A^{pf}_{e}, S^{pf}_{ref}, W^{pf}_{gross}, \theta_{Azimuth}, \theta_{Pitch1},$ $\theta_{Pitch2}, \theta_{Pitch3}, \theta_{Pitch4}, s_{Ispv}, t_{Ispv}, s_{Tvac}, t_{Tvac},$ $s_{Ae}, t_{Ae}, s_{Sref}, t_{Sref}, s_{Wgross}, t_{Wgross}, s_{MR}, t_{MR}$

11.2.4 MCO: Weights and Sizing Standard Form

Minimize:	$\mathbf{\Phi}_{W\&S}$	$(s_{Ae}+t_{Ae}) + (s_{Tsl/Weng}+t_{Tsl/Weng}) + (s_r+t_r) + (s_{Sref}+t_{Sref}) + (s_{Wgross}+t_{Wgross}) + (s_{Tsl}+t_{Tsl}) + (s_{MR}+t_{MR})$
Subject to:	h	$A_e^{ws} + s_{Ae} - t_{Ae} = A_e^t$ $(T_{SL}/W_{eng})^{ws} + s_{Tsl/Weng} - t_{Tsl/Weng} = (T_{SL}/W_{eng})^t$ $r^{ws} + s_r - t_r = r^t$ $S_{ref}^{ws} + s_{Sref} - t_{Sref} = S_{ref}^t$ $W_{gross}^{ws} + s_{Wgross} - t_{Wgross} = W_{gross}^t$ $T_{SL}^{ws} + s_{Tsl} - t_{Tsl} = T_{SL}^t$ $MR^{ws} + s_{MR} - t_{MR} = MR^t$
	Side	$T_{SL}/W_{gross} \geq 1.2$ $s_{Ae}, t_{Ae}, s_{Tsl/Weng}, t_{Tsl/Weng}, s_r, t_r, s_{Sref}, t_{Sref}, s_{Wgross}, t_{Wgross}, s_{Tsl}, t_{Tsl}, s_{MR}, t_{MR} \geq 0$
Given as Target:	$\mathbf{X}_{sys}$	$r^t, A_e^t, (T_{SL}/W_{eng})^t, S_{ref}^t, W_{gross}^t, T_{SL}^t, MR^t$
Find:	$\mathbf{Y}_{loc}$	$S_{ref}^{ws}, W_{gross}^{ws}, T_{SL}^{ws}, MR^{ws}$
By Changing:	$\mathbf{X}_{loc}$	$r^{ws}, A_e^{ws}, (T_{SL}/W_{eng})^{ws}, SF, T_{SL}/W_{gross}, s_{Ae}, t_{Ae}, s_{Tsl/Weng}, t_{Tsl/Weng}, s_r, t_r, s_{Sref}, t_{Sref}, s_{Wgross}, t_{Wgross}, s_{Tsl}, t_{Tsl}, s_{MR}, t_{MR}$

11.3 MCO: Data Flow

In order to better understand the coupling of design variables the design structure matrix (DSM) for MCO is provided. Also provided is a variable table which helps to quickly observe the important variables for each local-level discipline.

11.3.1 MCO: Design Structure Matrix

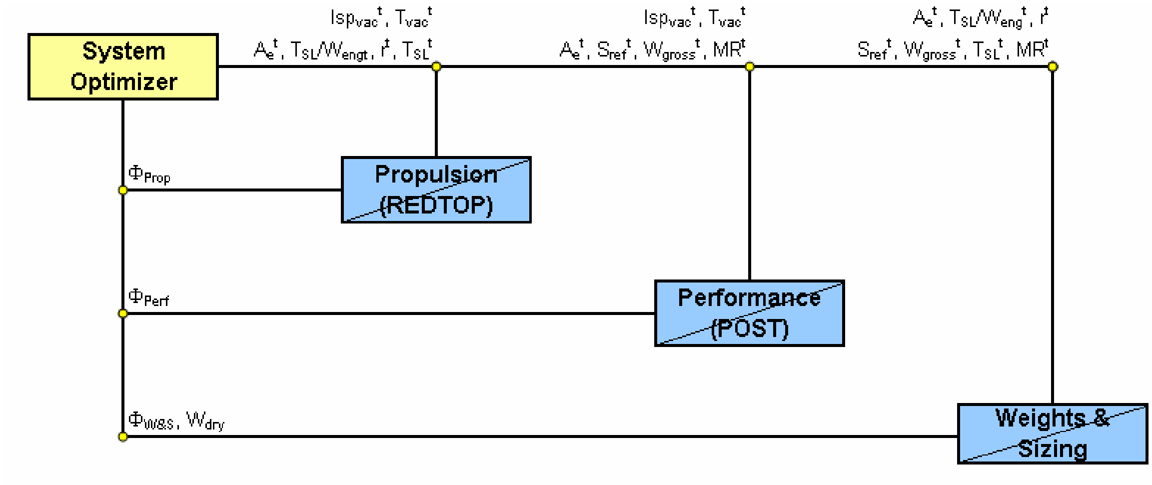

Figure 14: DSM for MCO.

The MCO DSM looks identical to the CO DSM. This is because MCO does not change the general architecture of CO but just reformulates the objective functions in both the system and local-level.

11.3.2 MCO: Variable Table

Table 26: Variable Table for MCO.

Contributing Analysis	Output Variables		Input Variables		Φ_{loc}	Local Constraints
	To System	Y_{loc}	X_{loc}	X_{sys}		
1. Propulsion	Φ_{Prop}	Isp_{vac}^{PP}, T_{vac}^{PP}, A_e^{PP}, T_{SL}/W_{eng}^{PP}, T_{SL}^{PP}	r^{PP}, ε, p_c, A_t, s_{pe}, s_r, t_r, s_{Ispv}, t_{Ispv}, s_{Tvac}, t_{Tvac}, s_{Ae}, t_{Ae}, $s_{Tsl/Weng}$, $t_{Tsl/Weng}$, s_{Tsl}, t_{Tsl}	Isp_{vac}^t, T_{vac}^t, A_e^t, T_{SL}/W_{eng}^t, r^t, T_{SL}^t	$(s_r+t_r) + (s_{Ispv}+t_{Ispv}) + (s_{Tvac}+t_{Tvac}) + (s_{Ae}+t_{Ae}) + (s_{Tsl/Weng}+t_{Tsl/Weng}) + (s_{Tsl}+t_{Tsl})$	1. $p_e^{PP}-s_{pe}-5(psia)=0$ 2. $r^{PP}+s_r-t_r=r^t$ 3. $Isp_{vac}^{PP}+s_{Ispv}-t_{Ispv}=Isp_{vac}^t$ 4. $T_{vac}^{PP}+s^{Tvac}-t_{Tvac}=T_{vac}^t$ 5. $A_e^{PP}+s_{Ae}-t_{Ae}=A_e^t$ 6. $T_{SL}/W_{eng}^{PP}+s_{Tsl/Weng}-t_{Tsl/Weng}=T_{SL}/W_{eng}^t$ 7. $T_{SL}^{PP}+s_{Tsl}-t_{Tsl}=T_{SL}^t$ 8. $30 \le \varepsilon \le 100$ 9. $4 \le r \le 10$ 10. $200 \le p_c \le 3100$ (psia) 11. All s & t ≥ 0
2. Performance	Φ_{Perf}	MR^{pf}	Isp_{vac}^{pf}, T_{vac}^{pf}, Ae^{pf}, S_{ref}^{pf}, W_{gross}^{pf}, $\theta_{Azimuth}$, θ_{Pitch1}, θ_{Pitch2}, θ_{Pitch3}, θ_{Pitch4}, s_{Ispv}, t_{Ispv}, s_{Tvac}, t_{Tvac}, s_{Ae}, t_{Ae}, s_{Sref}, t_{Sref}, s_{Wgross}, t_{Wgross}, s_{MR}, t_{MR}	Isp_{vac}^t, T_{vac}^t, A_e^t, S_{ref}^t, W_{gross}^t, MR^t	$(s_{Ispv}+t_{Ispv}) + (s_{Tvac}+t_{Tvac}) + (s_{Ae}+t_{Ae}) + (s_{Sref}+t_{Sref}) + (s_{Wgross}+t_{Wgross}) + (s_{MR}+t_{MR})$	1. $h_{insertion} = 50$ (nmi) 2. $i_{insertion} = 51.6$ (deg) 3. $g_{insertion} = 0$ (deg) 4. $Isp_{vac}^{pf}+s_{Ispv}-t_{Ispv}=Isp_{vac}^t$ 5. $T_{vac}^{pf}+s^{Tvac}-t_{Tvac}=T_{vac}^t$ 6. $A_e^{pf}+s_{Ae}-t_{Ae}=A_e^t$ 7. $S_{ref}^{pf}+s_{Sref}-t_{Sref}=S_{ref}^t$ 8. $W_{gross}^{pf}+s_{Wgross}-t_{Wgross}=W_{gross}^t$ 9. $MR^{PP}+s_{MR}-t_{MR}=MR^t$ 10. All s & t ≥ 0
3. Weights	$\Phi_{W\&S}$, W_{dry}	S_{ref}^{ws}, W_{gross}^{ws}, T_{SL}^{ws}, MR^{ws}	A_e^{ws}, T_{SL}/W_{eng}^{ws}, r^{ws}, SF, T_{SL}/W_{gross}, s_{Ae}, t_{Ae}, $s_{Tsl/Weng}$, $t_{Tsl/Weng}$, s_r, t_r, s_{Sref}, t_{Sref}, s_{Wgross}, t_{Wgross}, s_{Tsl}, t_{Tsl}, s_{MR}, t_{MR}	A_e^t, T_{SL}/W_{eng}^t, r^t, S_{ref}^t, W_{gross}^t, T_{SL}^t, MR^t	$(s_{Ae}+t_{Ae}) + (s_{Tsl/Weng}+t_{Tsl/Weng}) + (s_r+t_r) + (s_{Sref}+t_{Sref}) + (s_{Wgross}+t_{Wgross}) + (s_{Tsl}+t_{Tsl}) + (s_{MR}+t_{MR})$	1. $A_e^{ws}+s_{Ae}-t_{Ae}=A_e^t$ 2. $T_{SL}/W_{eng}^{ws}+s_{Tsl/Weng}-t_{Tsl/Weng}=T_{SL}/W_{engt}$ 3. $r^{ws}+s_r-t_r=r^t$ 4. $S_{ref}^{ws}+s_{Sref}-t_{Sref}=S_{ref}^t$ 5. $W_{gross}^{ws}+s_{Wgross}-t_{Wgross}=W_{gross}^t$ 6. $T_{SL}^{ws}+s_{Tsl}-t_{Tsl}=T_{SL}^t$ 7. $MR^{ws}+s_{MR}-t_{MR}=MR^t$ 8. $T_{SL}/W_{gross} \ge 1.2$ 9. All s & t ≥ 0

11.4 MCO: Results

The main differences between the CO and MCO algorithms are that MCO uses the IDF formulation for the local objective function and it removes the equality constraints at the system-level, replacing them with a penalty function. While the IDF formulation seemed to offer some advantages when applied to the CA's, the penalty function formulation appeared to give inconsistent results depending on the value of the penalty parameter (r_p).

The first MCO modification was to use the local-level objective functions as penalty terms for the system-level objective function. Previously CO had imposed equality constraints to ensure that the local-level penalty function were zero. This new penalty function formulation that appeared to cause problems in the RLV model using the MCO technique. The resulting configuration varied greatly with the selected value of the penalty parameter (r_p), see Figure 15.

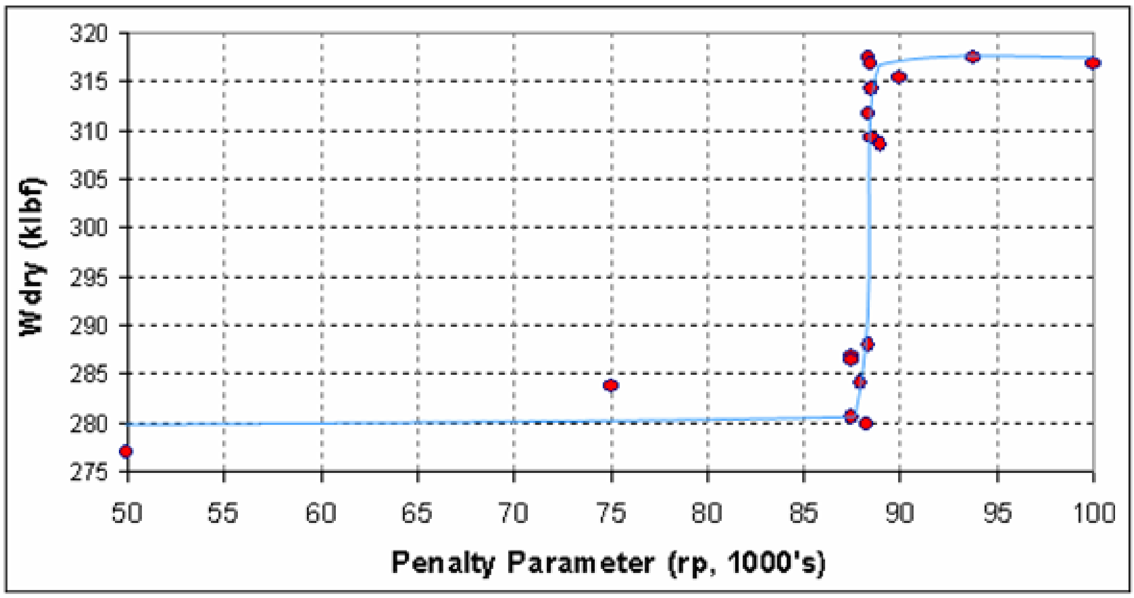

Figure 15: Resultant W_{dry} vs. Penalty Parameter (r_p).

Note: the bue line in Figure 15 is not a calculated regression line for the data plotted. It is hand drawn trend line to accentuate the general tendency of the data.

Figure 15 shows that the resulting value for W_{dry} using the MCO model had a step function relation with r_p. If r_p was too high, then the MCO model would conclude that it could not improve its initialization point (the best FPI configuration, $W_{dry} \approx 317$ klbf) and the optimizer would not move. If r_p was too low, then the MCO configuration would move from its initial point and drive the value of W_{dry} down to around $W_{dry} \approx 287$ klbf. These low results though had very poor convergences with errors often above 5%.

The value of W_{dry} jumps somewhere around 88,000 ≤ r_p ≤ 90,000. It was attempted to determine if there was actually a singular value of r_p that would produce a W_{dry} similar to the value found using AAO and BLISS ($W_{dry} \approx 305$ klbf). Unfortunately, isolating any single value of r_p that would produce the expected W_{dry} was not realizable as the there was not a smooth relationship between the value W_{dry} and the r_p in the 88,000 ≤ r_p ≤ 90,000 range.

Also, the MCO model was not consistent and identical MCO runs could produce very different results. Some of the MCO trials, shown in Figure 15, outputted resultant W_{dry} values somewhat close to the $W_{dry} \approx 305$ klbf expected (the closes run at 308.5 klbf). When it was attempted to verify these results by re-running the exact same trial, the second trial never reproduced the first results. Often times when a trial was re-tried, the second try resulted in the optimizer either never moving or falling down the $W_{dry} \approx 287$ klbf range.

The resultant inconsistencies observed in the MCO model made it impossible to draw any conclusions from the test trials run. Unless one already knew what the true optimum was, there would be no way to distinguish why any of the test trials run should be deemed worthier than another.

Despite problems due to the use of a penalty function in the system-level objective, the MCO algorithm was able to mitigate some of the local-level convergence problems present in CO due to the vanishing Jacobian effect.

MCO's second modification to CO is replacing the sum-of-squares formulation of the Φ_{loc} with the IDF formulation. This is proposed as a way to mitigate the vanishing Jacobian effect observed in the original CO algorithm. When the IDF formulation was applied to each individual CA, it appeared to provide tighter local-level convergence versus variable targets than the CO quadratic formulation. Test target sets were tried using both the CO and MCO formulation and it appeared that the new MCO was able to more closely match the desired target values. When tested independently, it appeared MCO's new IDF formulation was able to resolve some of the local-level convergence problems that had been accentuated in CO due to the vanishing Jacobian effect.

11.5 MCO: Discussion & Qualitative Ratings

This section will address CO optimization conclusions, lessons learned when implementing the method and provide qualitative ratings with discussion for each.

11.5.1 MCO: Optimization Conclusions

While MCO was able to ameliorate the vanishing Jacobian problems that CO encountered due the quadratic local-level objective formulation, inconsistencies in the overall model impeded a global optimum to be found.

While several test trials of the RLV model using the MCO technique were tried, never was any pattern identified that would allow the user to confidently determine what the MCO resultant optimum was. The resulting value for any trial was too erratically dependent of the penalty parameter used to allow the user to draw conclusions.

It is still possible, though, that the MCO formulation will provide consistent and clear results through further conditioning of optimization parameters. The amount of time and effort spent during this study trying to bring the MCO application to a successful conclusion was commensurate with the effort put forward for the other MDO techniques applied. Nevertheless, a solution using the MCO model was not found. It is possible that should more time be spent that MCO could come to a successful conclusion, but the amount of time required to do so seems to be greater than that needed for any of the other MDO techniques.

11.5.2 MCO: Lessons Learned

Lessons learned in the implementation of the MCO process are as follows:

1) The IDF formulation for the local-level objectives, while complicated, did seem to provide better convergence in the local-level over CO's original sum-of-squares formulation.
2) Optimization parameter conditioning for MCO proved to be the most difficult of all MDO techniques applied. There were a lot more local-level constraints and the value with which each variable was normalized could change convergence error at the local-level.
3) It was observed that the solution using MCO varied greatly with the value of the penalty parameter (r_p).

11.5.3 MCO: Qualitative Ratings

It is very difficult to qualitatively assess the experience of implementing a given MDA or MDO technique without first comparing it to other the other techniques to be evaluated. Therefore these ratings were assigned after all the techniques had been applied.

Table 27: Qualitative Ratings for MCO.

Criteria	Grade	Discussion
Implementation Difficulty	INC	MCO seemed to be more difficult to apply than any of the other MDO techniques simply by the fact that after a similar time spent for each, only was the MCO model not brought to a successful conclusion. Since, it was not successfully applied, though; it is difficult to tell exactly how hard it really is to implement.
Total Execution Time	C	(4 to 8 hours) Execution time varied with the value of the penalty parameter and other optimization parameters.

Robustness	INC	MCO was never brought to a successful conclusion, thus it is difficult to judge its robustness.
Formulation Difficulty	C-	MCO had the most complex formulation of all the MDO techniques applied. MCO required a separate formulation for each discipline and the system. There were a lot of additional variables and constraints at the local-level.
Optimization Deftness	INC	MCO was never brought to a successful conclusion, thus it is difficult to judge its optimization deftness.
Convergence Error	B	Convergence at the local-level was improved over CO, but it was not as tightly converged as results from FPI, AAO or BLISS.
Issues		1) There was a strong relationship between the resulting optimum value of the W_{dry} and the r_p used during a trial run. In the region where the W_{dry} jump the relation between W_{dry} and r_p was inconsistent. 2) Many MCO executions had to be performed to try and condition the model to provide consistent and convincingly converged solutions, a goal that was never reached. More time was spent trying to condition the MCO model than any other application except AAO yet reproducible results were never reached.
Unique Benefits		The IDF formulation for the local-level objectives did appear to provide better local-level convergence than the original CO formulation.

12 Conclusions

At the start of this study three objectives were listed:

1) Determine the benefits of MDO versus using multiple trials of iterative optimizers using FPI convergence.
2) Create a realistic test problem which will add to a growing field of research trying to evaluate novel MDO techniques: CO, MCO and BLISS.
3) Allow for across the board comparison of the MDO techniques by using the statistical method of blocking to remove external variability when comparing between techniques.

This section will describe some of the findings with respect to these goals and attempt to evaluate how successfully the goals were met.

12.1 Benefits of MDO

The results presented in this study found that RLV optimization employing MDO techniques did show some improvement over the results obtained using strictly the FPI process. This confirms that without human intervention the traditional method using FPI does not provide optimum results. While use of MDO showed only a modest improvement in the global objective, this benefit would probably increase for the large, complex problems for which the MDO methods were designs.

The best FPI model resulted in a vehicle dry weight (W_{dry}) of 317 klb, this was greater than the 305 klb value successfully determined with MDO via both the AAO and BLISS technique. This is only a 4% improvement in the global objective and it would have probably been possible to match the MDO results through the use of carpet plots and other techniques with FPI. For small, conceptual level problems where the CA's execute quickly and inexpensively, it would probably be faster and easier to perform parameter sweeps to arrived at a globally optimized vehicle than to apply an MDO methodology. This is because the large implementation cost of MDO techniques would probably not be offset by any subsequent reduction in the number of CA executions needed.

If the size of the MDO problem to be solved is small and the execution of the CA's involved is not very costly, then traditional methods using FPI will probably be the best choice. For these cases the benefits of MDO do not warrant the upfront cost of implementing the MDO technique. For problems of large size with costly CA's the implementation cost of MDO will be more likely to be offset by the efficiency benefits of MDO.

12.2 Authenticity of Test Problem

While there were limitations to the size and complexity of test problem that could be tried, it is believed that the next generation RLV problem did show enough "real world" characteristics to provide a realistic test problem.

The test problem tried could not be too large and complex for two reasons: 1) it was proposed that the AAO method be used to validate the results of the multi-level MDO techniques and 2) if the complexity of the test problem was too large then the time allowed for this project could easily have proven insufficient to allow for multiple applications to be realized. While using the AAO technique to solve and MDO problem is widely accepted, this technique is constrained to smaller, conceptual level problems. Thus test problems were limited in size to those for which AAO might still be applicable. The test problem used for this study seemed to be bordering the size and complexity that was still solvable using AAO. The AAO size constraint is unfortunate, but necessary so that results from the newer multi-level MDO techniques can be validated.

Despite size restrictions, the RLV test problem selected for this project did have a series of characteristics that increased its likeness to "real world" problems. First, the coupling disciplines were broken down along the conventional lines of Propulsion, Performance and Weights & Sizing. The disciplines kept their independence and were not merged into on single code or analysis. Secondly, technology reduction factors were added to the analysis and design of the RLV to provide a more realistic scenario of what might be encountered in the real world. Also, the base vehicle was a real concept studied both by NASA and in industry. Lastly and most importantly, is that widely used legacy codes were used for the design and analysis of each of the disciplines. This study used software like POST which, while problematic, is the industry standard code for trajectory analysis. It was of interest to see which algorithms would best handle using codes that were not always well behaved and might produce numerical errors.

12.3 Comparisons between CO, MCO and BLISS

Historical attempts at making comparisons between CO, MCO and BLISS have been ineffective because there have been too many external factors present to effectively isolate variance caused by differences in the algorithms and because there are too few test cases available with which to make any statistically significant conclusions.

It was known from the start that this study would be unable to conclude if CO, MCO or BLISS showed greater promise than any other simply because it is but one study from which it is impossible to draw any statistically significant conclusions.

On the other hand, this study was successful in using the Blocking Effect to reduce the number of external factors that could make it difficult to tell any variance between the multi-level MDO techniques. External factors were removed by:

1) having the same user apply all the MDO techniques evaluated
2) using the same analysis and optimization tools for all techniques
3) solving the same multi-disciplinary test problem

While the blocking method is necessary if external variances are to be removed it makes it even harder to collect a statistically significant number of data points. First, in most industry applications few are going to solve an MDO technique with one method and then try to solve the exact same problem again using another MDO technique. This would most likely be viewed as time wasted when the goal is to get an answer for the MDO problem; not to determine which technique solved it more efficiently. Second, since the same user must be used in order to block out variance due to user competency, a study like this can take a very long time. Implementation of some of the multi-level MDO techniques can be a time consuming process, fewer will have the amount of time necessary to make multiple applications. If it was hard to persuade people to try one application of an MDO technique, it will be even harder to persuade people to try several applications on the same problem.

Drawing on the experience of performing multiple applications of different MDO techniques on the same problem and verifying the results and benefits against the more traditional techniques, Table 28 shows a qualitative report card made for all the techniques applied. Grading is only accurate with relation to each other for this specific study.

Table 28: Qualitative Report Card, All Techniques.

Criteria	FPI	AAO	BLISS-2000	CO	MCO
Implementation Difficulty	A	C	B+	C+	Incomplete
Total Excecution Time	A	A-	B+	B-	C
Model Robustness	A	D	B+	B	Incomplete
Formulation Difficulty	A	A	B+	B-	C-
Optimization Deftness	D	A-	A	B	Incomplete
Convergence Error	A	A-	B+	C+	B

While a model of the next generation RLV was created for each of the candidate multi-level MDO techniques selected at the start of the study, the MCO model was never able to reach enough behavioral consistency for any conclusions to be made.

The report card above shows that for this test study FPI seems to win in most categories, this is to be expected as FPI has been the design method traditionally employed and the legacy tools used to make the models are the same or very similar to ones widely used. One can notice, though, that FPI has a very poor optimization deftness score. This reflects the fact that FPI will result in sub-optimal design configurations. This, depending on the problem, can be a critical deficiency.

For the MDO techniques, AAO and BLISS appear to be the best performers. This study applied the techniques to a conceptual, level problem due to known limitation of the AAO technique. The test problem was intentionally kept small due to a desire to validate the new multi-level techniques against the more accepted AAO technique. The size of this test problem may actually be approaching the limit of what can be successfully solved with AAO. It had to be helped a little bit by reducing the size of the problem before AAO converged at the true optimum.

BLISS, on the other hand, was able to locate at the true system-level optimum without any user guidance. Of the new multi-level MDO techniques, it was the one which required the fewest changes to be made from the traditional FPI approach. Also, the local-level alterations only consisted of changing the local objective to a composite objective and didn't introduce any new design variables at the local-level. In the case where there was only one local output, the original discipline formulation could be used unaltered. In the system-level, BLISS did introduce weighting factors, but the problem was still easily solved by the system optimizer.

While it is tempting to believe that BLISS is the most promising multi-level MDO technique applied, is but one data point. It is statistically impossible to declare BLISS the winner with any degree of certainty unless more studies employing the blocking effect are conducted and show similar results between the multi-level MDO methods.

References

1 Braun, R.D., *Collaborative Architecture for Large-Scale Distributed Design,* PhD Thesis, Stanford University, Sanford, CA, June 1996.

2 DeMiguel, A. V., Murray, W., *An Analysis of Collaborative Optimization Methods,* 8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Long Beach, CA, Sept. 2000, AIAA-2000-4720.

3 Sobieszanski-Sobieski, J., Agte, J, Sandusky, R., *Bi-Level Integrated System Synthesis,* 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, MO, Sept. 1998, AIAA 1998-4916.

4 Sobieszanski-Sobieski, J., Altus, T., Phillips, M., Sandusky, R., *Bi-Level Integrated System Synthesis (BLISS) for Concurrent and Distributed Processing,* 9th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Atlanta, GA, Sept. 2002, AIAA 2002-5409.

5 Alexandrov, N.M., *Comparative Properties of Collaborative Optimization and Other Approaches to MDO*, Institute for Computer Applications in Science and Engineering, NASA Langley Research Center, Hampton, VA, July 1999, NASA/CR-1999-209354, ICASE Report No. 99-24.

6 Alexandrov, N.M, Lewis, R.M., *Analytical and Computational Properties of Distributed Approaches to MDO*, 8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Long Beach, CA, Sept. 2000, AIAA-2000-4718.

7 Budianto, I., Olds, J., *A Collaborative Optimization Approach to Design and Deployment of a Space Based Infrared System Constellation*, 2000 IEEE Aerospace Conference, Big Sky, MT, March 2000, IEEE P335E.

8 Kroo, I., Manning, V., *Collaborative Optimization: Status and Directions*, 8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Long Beach, CA, Sept. 2000, AIAA-2000-4721.

9 Personal communication, J. Sobieski 10/28/2003.

10 Hayter, A.J., Probability and Statistics for Engineers and Scientists, 2nd Ed., Duxbury Thompson Learning, 2002, ISBN: 0-534-38669-5.

11 Cormier, T., Scott, A., Ledsinger, L., McCromick, D., Way, D., Olds, J., "Comparison of Collaborative Optimization to Conventional Design techniques for a Conceptual RLV", 8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Long Beach, CA, Sept. 2000, AIAA-2000-4885.

12 Olds, J., Crocker, A., Bradford, J., Charania, A.C., *Concept Overview and Model Operation: Reduced Order Simulation for Evaluating Technologies and Transport Architechtures (ROSETTA)*, SpaceWorks Engineering, Inc., Atlanta, GA, Apr. 2001, http://www.sei.aero/downloads/SEI_ACRE92_09Apr01.pdf (Last visited 4/26/2004).

13 MacConochie, I.O., *Characterization of Subsystems for a WB-003 Single Stage Shuttle*, NASA Langley Research Center, Hampton, VA, Feb. 2002, NASA/CR-2002-211249.

14 Isakowitz, S. J., Hopkins, J.P., Hopkins, J.B., International Refernce Guide to Space Launch Systems, 3rd Ed., AIAA, Reston, VA, 1999.

15 SpaceWorks Engineering, Inc., URL: http://www.sei.aero (Last visited 4/26/2004).

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

17 Powell, R.W., Striepe, A., Desai, P.N., Braun, R.D., Brauer, G.L., Cornick, D.E., Olson, D.W., Petersen, F.M., Stevenson, R., Program to Optimize Simulated Trajectories (POST): Volume II Utilization Manual Version 5.2, Oct. 1997.

18 Talay, T., "ME250 Launch Vehicle Design Class Notes", George Washington University, 1992.

19 Phoenix Integration, Inc., URL: http://www.phoenix-int.com.

20 Vanderplaats, G.N., Numerical Optimization Techniques for Engineering and Design, 3rdEd., McGraw-Hill Companies, 1984, ISBN: 0-944956-01-7, URL: http://www.vrand.com (Last visited 4/26/2004).

21 Pi Blue Software, Inc., URL: http://www.piblue.com (Last visited 4/26/2004).

22 Space Systems Design Lab, Georgia Institute of Technology, URL: http://www.ssdl.ae.gatech.edu (Last visited 4/26/2004).