

MULTIDISCIPLINARY ANALYSIS AND DESIGN OPTIMIZATION OF AN EFFICIENT SUPERSONIC AIR VEHICLE

Darcy L. Allison

Dissertation submitted to the faculty of the Virginia Polytechnic Institute and State University in
partial fulfillment of the requirements for the degree of

Doctor of Philosophy

In

Aerospace Engineering

Joseph A. Schetz, Co-Chair

Rakesh K. Kapania, Co-Chair

Robert A. Canfield

John A. Burns

April 25, 2013

Blacksburg, Virginia

Keywords: Aircraft design, Multidisciplinary design optimization, Supersonic military aircraft,
Design of computer experiments

© Darcy L. Allison, 2013

Multidisciplinary Analysis and Design Optimization of an Efficient Supersonic Air Vehicle

Darcy L. Allison

ABSTRACT

This work seeks to develop multidisciplinary design optimization (MDO) methods to find the optimal design of a particular aircraft called an Efficient Supersonic Air Vehicle (ESAV). This is a long-range military bomber type of aircraft that is to be designed for high speed (supersonic) flight and survivability. The design metric used to differentiate designs is minimization of the take-off gross weight.

The usefulness of MDO tools, rather than compartmentalized design practices, in the early stages of the design process is shown. These tools must be able to adequately analyze all pertinent physics, simultaneously and collectively, that are important to the aircraft of interest.

Low-fidelity and higher-fidelity ESAV MDO frameworks have been constructed. The analysis codes in the higher-fidelity framework were validated by comparison with the legacy B-58 supersonic bomber aircraft. The low-fidelity framework used a computationally expensive process that utilized a large design of computer experiments study to explore its design space. This resulted in identifying an optimal ESAV with an arrow wing planform. Specific challenges to designing an ESAV not addressed with the low-fidelity framework were addressed with the higher-fidelity framework. Specifically, models to characterize the effects of the low-observable ESAV characteristics were required. For example, the embedded engines necessitated a higher-fidelity propulsion model and engine exhaust-washed structures discipline. Low-observability requirements necessitated adding a radar cross section discipline.

A relatively less costly computational process utilizing successive NSGA-II optimization runs was used for the higher-fidelity MDO. This resulted in an optimal ESAV with a trapezoidal wing planform. The NSGA-II optimizer considered arrow wing planforms in early generations during the process, but these were later discarded in favor of the trapezoidal planform. Sensitivities around this optimal design were computed using the well-known ANOVA method to characterize the surrounding design space.

The lower and higher fidelity frameworks could not be combined in a mixed-fidelity optimization process because the low-fidelity was not faithful enough to the higher-fidelity analysis results. The low-fidelity optimum was found to be infeasible according to the higher-fidelity framework and *vice versa*. Therefore, the low-fidelity framework was not capable of guiding the higher-fidelity framework to the eventual trapezoidal planform optimum.

To Holly, all my children, and my parents.

Acknowledgements

I am grateful that I got to share an office with Craig Morris who made graduate school at Virginia Tech an enjoyable experience. I am also grateful for the guidance offered by my co-advisors listed on the title page, committee members, other professors, and the industry and government experts who affected this work.

I wish to specifically thank all those professionals in industry and government laboratories who answered emails and offered advice pertaining to their individual expertise. However, in doing this I wish to state that all material herein is my sole responsibility.

FLOPS	Linwood A. McCullers (Arnie)
NPSS	Timothy Terrell and Ryan Battelle
WATE++	Michael Tong and Doug Thurman
ZONAIR	Reggie Chang and P.C. Chen

Finally, I wish to thank the Air Force Research Laboratory at Wright-Patterson Air Force Base for their financial support and continued interest by Ed Alyanak, Ray Kolonay, and Dan Strong in my research progress.

This material is based on research sponsored by Air Force Research Laboratory under agreement number FA8650-09-2-3938. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Air Force Research Laboratory or the U.S. Government.

Table of Contents

1.1	Putting Design and Optimization Together	1
1.2	Multidisciplinary Design Optimization of an Efficient Supersonic Air Vehicle.....	6
1.3	The Beginnings of Optimization.....	11
1.4	Constrained Multi-Variable, Nonlinear Optimization	12
1.5	A Brief History of Aircraft Multidisciplinary Analysis, Design, and Optimization	15
2.1	MDO Environment	21
2.2	Parametric Geometry Model.....	22
3.1	Low-Fidelity Framework	35
3.1.1	Flight Optimization System.....	36
3.1.2	Aerodynamics Model	39
3.1.3	Engine Cycle Analysis FLOPS Module	42
3.1.4	Structures and Weights FLOPS Module	45
3.1.5	Take-off and Landing FLOPS Module.....	46
3.1.6	Mission Performance and Noise Footprint FLOPS Modules.....	48
3.1.7	Optimization Models	49
3.2	Low-Fidelity Design of Experiments and Optimization Problem	50
3.2.1	Multidisciplinary Design Optimization Problem Set-Up	51
3.2.2	Design of Experiments for Low-Fidelity ESAV MDO Framework	53
3.2.3	Optimization for Low-Fidelity ESAV MDO Framework	58

3.2.4	Reduced-Order Design Space.....	83
3.2.5	Low-Fidelity Optimization Conclusions	98
4.1	Selection Criteria for the Propulsion Model to be used in the ESAV MDO Framework	101
4.2	Candidate Propulsion Models for the ESAV MDO Framework	104
4.3	Detailed Propulsion Model Development.....	109
4.3.1	Choice of Propulsion Cycle Models.....	109
4.3.2	Turbojet Model	111
4.3.3	Low Bypass Turbofan and Variable-Cycle Models	121
4.4	Final Propulsion Model.....	122
4.4.1	Validation of NPSS/WATE++ Model.....	123
4.4.2	Summary of Final ESAV MDO Framework Propulsion Model Process.....	124
5.1	Expanded N ² Diagram	125
5.2	Propulsion	128
5.3	Aerodynamics	130
5.4	Engine Exhaust-Washed Structures	134
5.5	Radar Cross Section Estimation.....	137
5.6	Validation of Analysis Codes in Higher-Fidelity Framework by Comparison with the Convair B-58 Hustler Supersonic Bomber	141
5.7	Higher-Fidelity ESAV MDO Framework Overview.....	149
6.1	Practical ESAV MDO Framework in ModelCenter™	151

6.1.1	NPSS/WATE++ Propulsion Model in ModelCenter™.....	153
6.1.2	Inlet Geometry Local Optimization and Cowl Geometry Iterations	154
6.1.3	ZONAIR Aerodynamics Model	156
6.1.4	Engine Exhaust-Washed Structures Model	158
6.1.5	FLOPS Modules	159
6.1.6	Constraints and Radar Cross Section Models.....	161
6.2	Optimization Approach.....	163
6.2.1	Large DOE Optimization Approach.....	165
6.2.2	Two Successive NSGA-II Optimizations in Full Design Space Method	170
6.3	Higher-Fidelity ESAV MDO Framework Optimization	170
6.3.1	DOE Method.....	171
6.3.2	Successive NSGA-II Method for Higher-Fidelity ESAV MDO Framework.....	173
6.4	Analysis of Final Optimal Higher-Fidelity ESAV MDO Configuration.....	181
6.4.1	Other Trapezoidal Wing Designs	181
6.4.2	Other Optimal ESAV Characteristics.....	182
6.4.3	Low Cost Range Extension of ESAV Design	183
6.5	Higher-Fidelity ESAV MDO Sensitivities	185
7.1	Conclusions.....	197
7.2	Future Work.....	202
	Bibliography	205

A.Appendix Optimization Background	223
B.Appendix Background Information on Four Propulsion Model Candidates	232

List of Figures

Figure 1.2-1. Notional ESAV configuration.....	7
Figure 1.2-2. ESAV mission profile.	8
Figure 1.2-3. Man performing maintenance on an embedded engine in a B-2 bomber.	9
Figure 1.4-1. Graphical representation of the constrained optimal problem.	13
Figure 1.5-1. First flight of the Wright Flyer.....	16
Figure 1.5-2. The Bell X-1 (NASA photo).	17
Figure 2.2-1. Sample two-dimensional airfoil geometries with varying $N1, N2$ values.	25
Figure 2.2-2. Geometry model drawing of a notional ESAV outer mold line including wing, fuselage, cowls, and inlets.	28
Figure 2.2-3. Increasing amplifier density with increasing Bernstein Polynomial order.	30
Figure 2.2-4. Various capabilities of the VT-CST model.....	31
Figure 2.2-5. Stereolithographic (rapid prototyping) format output by VT-CST.....	32
Figure 3.1-1. Initial low-fidelity ESAV MDO framework N^2 diagram.	39
Figure 3.1-2. Low-fidelity aerodynamics code, drag breakdown, and example interfaces within the ESAV MDO framework.	42
Figure 3.2-1. Twelve representative design variables in four 3D design space scatter plot representations. Plot a) has the cruise Mach number, bypass ratio, and semi-span. Plot b) has the cruise altitude, root chord, and the leading edge sweep angle. Plot c) has the cruise thrust per engine, the area ruling width parameter, and the fuselage length. Plot d) has the fuselage width, the wing dihedral, and the high pressure compressor pressure ratio.	57
Figure 3.2-2. Example Design Explorer TOGW convergence to a local stationary point.	60
Figure 3.2-3. Example CONMIN TOGW convergence to a local stationary point.	61

Figure 3.2-4. Planform shapes of identified six design points from the DOE, and their corresponding stationary points. They are in pairs (left is DOE design and right is stationary point design) in order of DOE designs increasing TOGW from top to bottom.....	62
Figure 3.2-5. DAKOTA Coliny DIRECT algorithm options for low-fidelity optimization.	66
Figure 3.2-6. NSGA-II algorithm options for low-fidelity optimization.....	68
Figure 3.2-7. NSGA-II TOGW convergence history over the run number.	69
Figure 3.2-8. Best ESAV configuration using NSGA-II algorithm for low-fidelity MDO.....	69
Figure 3.2-9. Optimum configuration using NSGA-II algorithm for low-fidelity ESAV using the fourth best DOE configuration result as the starting point.	70
Figure 3.2-10. Two final optimal designs found starting from the two configurations (top was best design and bottom was fourth best design) from NSGA-II optimizations, followed by a sequential quadratic programming gradient optimization.	73
Figure 3.2-11. Different views of the optimal low-fidelity configuration ESAV from performing the NSGA-II optimization followed by a gradient optimization in the reduced design space starting at the fourth best DOE ESAV design.	76
Figure 3.2-12. ESAV design for full design space NSGA-II starting point based on engineering judgment using some simple analysis and experience.....	77
Figure 3.2-13. Optimal ESAV planform configuration found using the successive NSGA-II optimization method using the full design space.....	81
Figure 3.2-14. Take-off gross weight sensitivity plot around the low-fidelity ESAV optimum..	88
Figure 3.2-15. Empty weight sensitivity plot around the low-fidelity ESAV optimum.....	91
Figure 3.2-16. Fuel weight sensitivity plot around the low-fidelity ESAV optimum.	92
Figure 3.2-17. Structural weight sensitivity plot around the low-fidelity ESAV optimum.....	93

Figure 3.2-18. Propulsion weight sensitivity plot around the low-fidelity ESAV optimum.	94
Figure 3.2-19. Aircraft noise sensitivity plot around the low-fidelity ESAV optimum.	96
Figure 3.2-20. Thrust-to-weight sensitivity plot around the low-fidelity ESAV optimum.	97
Figure 3.2-21. Wing loading sensitivity plot around the low-fidelity ESAV optimum.	98
Figure 4.1-1. Notional serpentine inlet in supersonic flight showing oblique shocks, the terminating normal shock, duct geometry, and the compressor face.	102
Figure 4.1-2. Notional low-observable ESAV nozzle structure with fixed throat.	103
Figure 4.3-1. Four shock inlet showing three ramps and the terminating normal shock.	112
Figure 4.3-2. FLOPS engine deck view file where blankPC is a whitespace string of fixed width required for rigid column format required by FLOPS.	115
Figure 4.3-3. Front end block diagram of the NPSS turbojet model.	116
Figure 4.3-4. Engine core block diagram of the NPSS turbojet model.	118
Figure 4.3-5. Back end block diagram of the NPSS turbojet model.	118
Figure 4.3-6. Example turbofan engine graphical output from WATE++.	119
Figure 4.3-7. Block diagram of the NPSS/WATE++ turbojet model.	120
Figure 4.3-8. Block diagram of the NPSS/WATE++ turbofan and variable cycle models.	121
Figure 5.1-1. The higher-fidelity N^2 diagram with added and changed disciplines from the low- fidelity version of the N^2 diagram.	126
Figure 5.1-2. Design challenges of an ESAV compared to the SR-71 supersonic aircraft.	127
Figure 5.2-1. The weight model of an example low-bypass engine with extended inlet.	129
Figure 5.3-1. ZONAIR CPU time over the nxn meshsize or paneling density for one run.	132
Figure 5.3-2. ZONAIR accuracy (based on assumption that 50x50 meshsize produces the correct results) over the nxn meshsize or paneling density for one run.	133

Figure 5.3-3. ZONAIR solution example, showing the coefficient of pressure results.....	134
Figure 5.4-1. Notional reference EEWS design with individual components and primary loading – flow goes from the engine on the right to ambient conditions on the left.	135
Figure 5.4-2. Process utilized to develop EEWS surrogate weight model from Ref. [104]......	136
Figure 5.5-1. Comparison of the original Matlab POFACETS code RCS results and C++ version without RAM.	139
Figure 5.5-2. Comparison of the original Matlab POFACETS code RCS results and C++ version with RAM.	140
Figure 5.6-1. Convair B-58 Hustler aircraft in flight.....	142
Figure 5.6-2. Representation of conical camber by chord slices over a wing with the conical camber effects becoming more evident closer to the wing tip.....	143
Figure 5.6-3. B-58 wing showing handles as “ * ” points, the root airfoil, and the tip airfoil....	144
Figure 5.6-4. The B-58 VT-CST model representation found from matching the root and tip airfoils and conical camber geometry of actual aircraft.....	145
Figure 5.6-5. F-5 wing geometry matching process and output using VT-CST.....	146
Figure 5.6-6. A representative aerodynamics run of many used for the B-58 tail drag surrogate model.....	148
Figure 6.1-1. Practical ModelCenter™ implementation of N ² diagram higher-fidelity ESAV MDO framework. The black curves with arrows show some of the data links and their direction between disciplines.	152
Figure 6.1-2. View of NPSS/WATE++ model with input and output links shown.	154

Figure 6.1-3. View of inlet CONMIN optimization model in ModelCenter™ with input and output links shown. The number 44 indicates the number of iterations it took to find the optimum.	155
Figure 6.1-4. View of cowl/inlet and propulsion design iteration in ModelCenter™ with input and output links shown. The number 6 in the boxes indicate the number of iterations it took to converge on the correct geometry for the case shown.	156
Figure 6.1-5. ZONAIR aerodynamic model flowchart as implemented in MDO framework. ..	158
Figure 6.1-6. View of EEWS discipline surrogate model with input and output links shown. ..	159
Figure 6.1-7. View of the FLOPS model. This includes the structures, weights, take-off & landing, mission performance, and noise modules in one box. There are many input links from previous models in the ModelCenter™ ESAV framework.	160
Figure 6.1-8. F-22 internal weapons storage volume.	161
Figure 6.1-9. Stereolithographic geometry output of a notional ESAV OML which is used by the RCS model as input.	163
Figure 6.2-1. Mixed fidelity optimization process represented by a flowchart.	166
Figure 6.2-2. Notional reduced design space process for a three-design-variable space.	169
Figure 6.2-3. Notional design space, already reduced, and its corresponding reduced-order space assuming the objective function is insensitive to one design variable.....	169
Figure 6.3-1. Starting point design for the first optimization run in the successive NSGA-II method using the higher-fidelity ESAV MDO framework. The left shows the planform drawn with the VT-CST geometry model using a field of points, the middle shows the ZONAIR C_p output for the $M=2.0$ at $C_L=0$ case, and the right shows the RCS format.	175

Figure 6.3-2. Best arrow wing design found in the second higher-fidelity ESAV MDO NSGA-II optimization run.	178
Figure 6.3-3. Final higher-fidelity ESAV MDO optimal configuration. From a) to e) it is the top view, *.stl ISO, ZONAIR M=2.2 C _p ISO, side view, and back-angled view.....	180
Figure 6.4-1. Northrop YF-23 side view and top view showing trapezoidal planform and area ruling features.	181
Figure 6.4-2. A Boeing concept for the F/A-XX aircraft reported by Aviation Week & Space Technology, April 1, 2013 on page 13.	182
Figure 6.5-1. Higher-fidelity ESAV MDO framework aspect ratio sensitivity chart.....	187
Figure 6.5-2. Higher-fidelity ESAV MDO framework TOGW sensitivity chart.	188
Figure 6.5-3. Higher-fidelity ESAV MDO framework fuel weight sensitivity chart.	189
Figure 6.5-4. Higher-fidelity ESAV MDO framework C _{D0} sensitivity chart.	190
Figure 6.5-5. Higher-fidelity ESAV MDO framework structural weight sensitivity chart.	191
Figure 6.5-6. Higher-fidelity ESAV MDO framework propulsion weight sensitivity chart.	192
Figure 6.5-7. Higher-fidelity ESAV MDO framework RCS sensitivity chart.	193
Figure 6.5-8. Higher-fidelity ESAV MDO framework noise sensitivity chart.....	194
Figure 6.5-9. Higher-fidelity ESAV MDO framework thrust-to-weight ratio sensitivity chart.	195
Figure 6.5-10. Higher-fidelity ESAV MDO framework wing loading sensitivity chart.	196

List of Tables

Table 2.2-1. Examples of Possible Global Design Variables in the Geometry Model.....	29
Table 3.1-1. Available FLOPS Modules.....	36
Table 3.1-2. Low-Fidelity Aerodynamic Codes and the Output Data.....	40
Table 3.1-3. Results from Phoenix Integration Showing the Best Algorithms for Solving a Simple but Similar Optimization Problem to the ESAV Problem.....	50
Table 3.2-1. Low-Fidelity Design Variables and their Lower and Upper Bound Limits.....	52
Table 3.2-2. Low-Fidelity Design Variables with Limited Lower and Upper Bounds Rounded to Nearest Hundredths Place	64
Table 3.2-3. NSGA-II Multidisciplinary Design Optimization Results from First Design Starting Point (NSGA-II 1 st), and from Fourth Design Starting Point (NSGA-II 4 th)	71
Table 3.2-4. Final Low-Fidelity Optimal Rounded Design Variables after SQP Optimization with the Final Optimal Design Highlighted by the Box	74
Table 3.2-5. ESAV Full Design Space NSGA-II Test Starting Point Design Variable Values ...	78
Table 3.2-6. Low-Fidelity Successive NSGA-II Method Optimal ESAV Design Variables.....	82
Table 4.2-1. NPSS, GasTurb, ACSYNT, and FLOPS Propulsion Modules vs. ESAV Propulsion Requirements.	108
Table 4.4-1. Validation Results of NPSS/WATE++ Turbojet Model	123
Table 5.1-1. Overview of Changes from the Low-Fidelity to the Higher-Fidelity N ² Diagram	125
Table 5.6-1. Comparison of Results from the Framework's B-58 Validation Analysis with the Actual B-58.....	149
Table 6.3-1. Results of Higher-Fidelity Checks on Low-Fidelity Stationary Points.....	171

Table 6.3-2. Higher-Fidelity ESAV MDO Framework Design Variables and the Corresponding Design Space Constraints	174
Table 6.3-3. Initial Design Variables Describing the Starting ESAV Design for the First Optimization Run of the Higher-Fidelity Successive NSGA-II Method.....	176
Table 6.3-4. Higher-Fidelity Optimal Design Found Using the Successive NSGA-II Method .	179
Table 6.4-1. Other Parameters of Interest for Final Higher-Fidelity ESAV Design	183
Table 6.4-2. New Optimum Resulting from Small Thirty Minute Mission Optimization	185

List of Abbreviations

3D	Three Dimensional
A2/AD	Anti-Access and Area-Denial
AFRL	Air Force Research Laboratory
AIC	Aerodynamic Influence Coefficient
ANOPP	Aircraft Noise Prediction Program
ANOVA	Analysis of Variance
BPR	Bypass Ratio
CAD	Computer Aided Drafting
C_D	Coefficient of Drag
CG	Center of Gravity
C_L	Coefficient of Lift
CST	Class Shape Transformation
db	Decibels
dBsm	Decibel per Square Meter (decibel measure of RCS relative to 1 m ²)
DOE	Design of Experiments
EEWS	Engine Exhaust-Washed Structures
EEWS	Engine Exhaust-Washed Structures
ESAV	Efficient Supersonic Air Vehicle
FAA	Federal Aviation Administration
FAR	Federal Aviation Regulations
FEA	Finite Element Analysis
FLOPS	Flight Optimization System
GA	Genetic Algorithm
GUI	Graphical User Interface
IEEE	Institute of Electrical and Electronics Engineers
JANAF	Joint Army-Navy-Air Force
LCC	Life Cycle Cost
LHS	Latin Hypercube Sampling
LO	Low Observable

M	Mach Number
MDAO	Multidisciplinary Design Analysis and Optimization
MDO	Multidisciplinary Design Optimization
MPI	Message Passing Interface
NASA	National Aeronautics and Space Administration
NIST	National Institute of Standards and Technology
NPSS	Numerical Propulsion System Simulation
NURBS	NonUniform Rational B-Splines
OML	Outer Mold Line
OpenVSP	Open Source Vehicle Sketch Pad
PID	Proportional Integral Derivative (controller)
RAM	Radar Absorbing Material
RCS	Radar Cross Section
RPM	Revolutions per Minute
SAE	Society of Automotive Engineers
SFC	Specific Fuel Consumption
SQP	Sequential Quadratic Programming
STL	Stereolithographic
T/W	Thrust-to-Weight
T_4	Turbine Inlet Temperature
TOMS	Transactions of Mathematical Software
TOGW	Take-Off Gross Weight
UAV	Unmanned Aerial Vehicle
USAF	United States Air Force
VT-CST	Virginia Tech Class Shape Transformation
W/S	Wing Loading
WATE	Weight Analysis of Turbine Engines

1.Introduction and Background

1.1 Putting Design and Optimization Together

The design of an aircraft is generally a highly nonlinear and highly coupled problem. It is highly nonlinear in that the design space is shaped by many nonlinear relationships between the aircraft design parameters. There are many nonlinear relationships that exist between various aspects of the aircraft that are both intuitive and nonintuitive. It is highly coupled in that all discipline models required to analyze an aircraft both supply input and receive output to and from other discipline models, respectively.

Changing one parameter of one subsystem will affect many different parameters in many other subsystems. If the collection of discipline analyses used to characterize an aircraft design is thought of as a system of equations, then nearly all of these equations are coupled to several, if not all, other equations in the system. A consequence to aircraft design that results from these complicated relationships is that regions of infeasibility in the design parameter space can sometimes be in non-intuitive locations. This can be described as a *disjointed* design space. This results in design space pockets of infeasibility surrounded by feasible regions.

Trade studies are a common part of engineering practice for any system. Generally, the improvement of one aspect of a system degrades another (or several others), hence the desire to find the best balance between competing design parameters. A trade study is one tool that can be used by an engineer to understand what the best balance should be. Trade studies are not confined to design parameters but are also routinely used to pick between entire systems. For example, should the United States Air Force (USAF) buy a few expensive stealthy fighter aircraft that require a large commitment to maintenance, or many less-expensive and less

survivable fighter aircraft to perform the same mission? Many of the important considerations for this example trade study are indeed technical, but some of the most important are operational details, human life, and also considerations based solely on the USAF preferences and leanings. It might favor the few stealthy aircraft, or may be inclined to lean towards lower procurement costs and more simple maintenance. These intangible, but highly relevant and central considerations can indeed make or break the success of a design. At the same time they are exceedingly difficult to mathematically model, and thus rely on a human choice based on experience (albeit the choice is often made as a mathematical or numerical estimate on a scale within a decision matrix).

Trade studies can only effectively consider a small number of competing designs. Design optimization is structured to handle many at the same time. There are three main headings which are encompassed by the scope of design optimization. These are analysis, design, and optimization. Indeed, the process of aircraft multidisciplinary design analysis and optimization (MDAO) requires all three of these concepts. Analysis determines the response of a system to inputs, or in other words, its environment. Design is the process one goes through in order to specify the system [1]. These two are naturally and routinely joined together in engineering, because the response of a system to its environment is critical information required to properly specify that system's design. Optimization is the third concept which gives the designer tools needed to rationally search for and identify the best design, rather than just a design that is feasible. Optimization is a mathematical exercise, but at the same time, optimization cannot work without a human engineer. In fact, without the human engineer, optimization would be meaningless.

A very important concept was just touched upon. The word “best design” was used, but the best design can mean totally different things to different users of the resulting system or even designers. This subtle but necessary part of MDAO requires more explanation. Although MDAO strives to minimize human intervention through automation of code execution and programming, it still requires an objective. Optimization is futile without it. The objective function is the aircraft aspect that will either be minimized or maximized by the optimization process, and is usually contrived through engineering judgment and/or past work. The current state-of-the-art requires a person to set up the design optimization problem by:

- choosing the optimization method/algorithm to be used for the problem at hand,
- choosing variables the optimizer can control and the optimization objective function,
- applying appropriate constraints to the problem,
- choosing the appropriate engineering/physics models and their fidelity level to capture all relevant/significant features of the system and the environment in which it will operate,
- setting up (wrapping) appropriate discipline models for analysis to determine the variable values for a given design, and
- balancing the usually competing desires of reasonable simulation accuracy and reasonable computational costs.

Therefore, this current state-of-the-art in design optimization is still completely reliant on a skilled human to have any success. However, at the same time, without the computer, the optimal design problem would remain hopelessly out of reach.

Let $X^0 \subset \mathfrak{R}^n$ and let $f: X^0 \rightarrow \mathfrak{R}$, $g: X^0 \rightarrow \mathfrak{R}^m$, and $h: X^0 \rightarrow \mathfrak{R}^k$. The optimization problem can be succinctly stated as [2]

$$\min_{\vec{x} \in X^0} f(\vec{x}),$$

$$\text{subject to: } h_i(\vec{x}) = 0, i = 1, \dots, k, \quad (1)$$

$$g_j(\vec{x}) \leq 0, j = 1, \dots, m.$$

$f(\vec{x})$ is called the objective function, dependent on the vector $\vec{x}$ of design variables. The problem is constrained by equality constraints, $h_i(\vec{x}) = 0$, and inequality constraints, $g_j(\vec{x}) \leq 0$. The set

$$X = \{\vec{x} \in X^0 \mid h(\vec{x}) = 0, g(\vec{x}) \leq 0\}, \quad (2)$$

is called the feasible set. The global optimal design is

$$\bar{x} \in X = \{\vec{x} \mid \vec{x} \in X^0, h(\vec{x}) = 0, g(\vec{x}) \leq 0\} \quad (3)$$

such that,

$$f(\bar{x}) = \min_{\vec{x} \in X} f(\vec{x}). \quad (4)$$

The measure of what makes one design better than another, or the objective function, is an important choice made by the designer(s) in the design process. The traditional design goal for aircraft is to minimize the take-off gross weight (TOGW). Jensen, Rettie, and Barber [3] selected seven different, but valid, figures of merit to optimize, and made comparisons. These figures of merit are

- TOGW,
- life cycle cost (LCC),
- acquisition cost,
- flyaway cost,
- LCC/productivity,
- direct operating cost, and
- fuel.

An accumulative penalty was calculated for each figure of merit by summing penalties imposed by the other six figures of merit on the design with a given minimum (optimal) figure of merit. It

was assumed that each of the seven figures of merit had the same weighting and was therefore equally important. The results showed that minimizing the TOGW had a relatively low accumulative penalty. That is, minimizing the TOGW caused only minor detrimental effects to the other important considerations.

Choosing the TOGW as the optimization objective function, $f(\vec{x})$, and minimizing it, is more than just a traditional design goal; it does well to balance the other important considerations represented by the other figures of merit. The TOGW not only indicates the aircraft cost to the first order [4], but also indicates the efficiency of the whole resulting aircraft system in performing the proposed mission. Therefore, the definition of the best ESAV design herein will be the feasible design that has the minimum TOGW.

Finding the balance between reasonable accuracy and reasonable computational cost has become a central problem in aircraft MDAO. Traditionally, semi-empirical rules that are very computationally cheap were used exclusively in the early phases of aircraft design [5]. These types of models are often referred to as low-fidelity models. However, their limited range of validity, accuracy and flexibility have been causing them to be replaced, where appropriate (depends on the problem), with more general and physics-based models [5]. The cost of introducing physics-based models, which are usually more accurate and flexible, is that they are generally more computationally expensive. These types of models are often referred to as medium-fidelity or high-fidelity models depending on the number of assumptions made to the governing physics equations. Fidelity can be thought of as a measure of how close the model comes to representing or predicting the actual physical situation. Solving the problem must remain tractable, but the solution must also be accurate to the tolerance determined by the designer(s). The solution can only be as good as the method(s) used to obtain it.

Multidisciplinary design optimization (MDO) can be applied to systems that do not yet exist or to systems that do. The only difference between MDO and MDAO is that MDO is used for optimization and MDAO is used to perform multidisciplinary analysis without optimization or to perform optimization.

The bulk of this current effort has been applied to a system that does not yet exist – a particular advanced military air vehicle concept. However the ideas here can be applied to “modify [an existing] design to improve the performance.” [6] Thus, an existing system can also be optimized. However, the difference lies in the cost of applying design modifications to a system that already exists as opposed to a system that does not yet exist. Often, the cost to modify a design far outweighs the performance improvement of the modification. For a system not yet in existence, the only cost for changes are setting up and running the MDO itself and the time it takes to run it. This fact highlights an important design aphorism that has often proven too evasive in actual practice for many reasons – design it right before building it. The introduction of an appropriate MDO process sooner into the design timeline has the possibility of getting to better designs earlier. Indeed, “more detailed design analyses are being required and applied in earlier phases of the aircraft design process where there are still very many degrees of freedom and few restrictions on the design space.” [5]

1.2 Multidisciplinary Design Optimization of an Efficient Supersonic Air Vehicle

This work concerns itself with a particular aircraft, and its multidisciplinary design optimization and design space characterization. This aircraft has been come to be known as an Efficient Supersonic Air Vehicle (ESAV). An ESAV and its mission have been described verbally and by documents by Air Force Research Laboratory (ARFL) researches to the author. It has a long

range, carries and delivers a military payload, is low-observable (LO), and flies at a Mach number in the range of $2.2 \leq M \leq 2.5$. Figure 1.2-1 shows a notional ESAV concept. For LO reasons, there is no tail, and the engines are embedded into the wing (or fuselage) under their respective cowls.

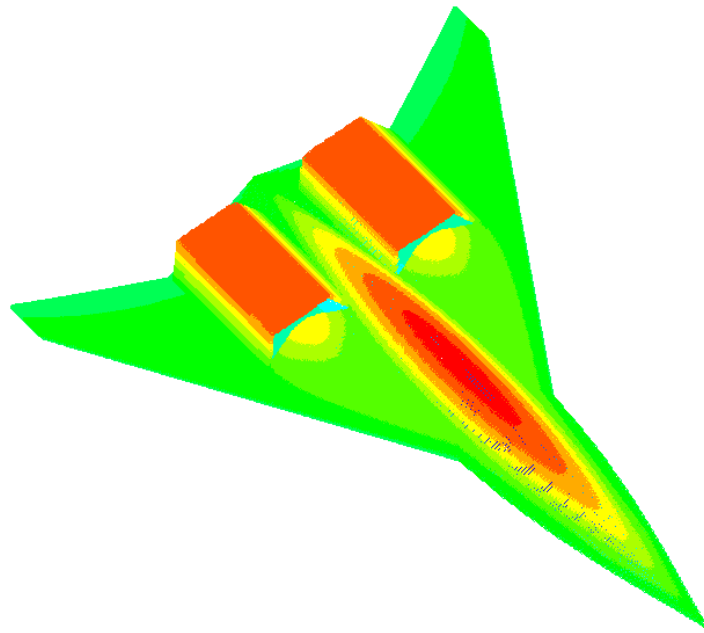


Figure 1.2-1. Notional ESAV configuration.

The basic mission of an ESAV is shown in Fig. 1.2-2. There is no stated constraint on the altitude at any point in the mission. After taxi and take-off, the ESAV climbs and accelerates to its 2000 n.mi. supersonic cruise-out leg. Once it has reached its combat radius distance, a 180° turn is initiated at a maximum load factor of 2.5. At this point, its military payload is either dropped or the decision is made to abort the mission and carry it back. To obtain conservative results, the assumption is made throughout this work that the payload is not released. The turn is followed by the 2000 n.mi. supersonic return cruise. Finally the ESAV descends, has a 20 minute hold constraint, and then lands. The amount of unusable fuel is fixed at 5% and the standard atmosphere is assumed throughout the mission profile.

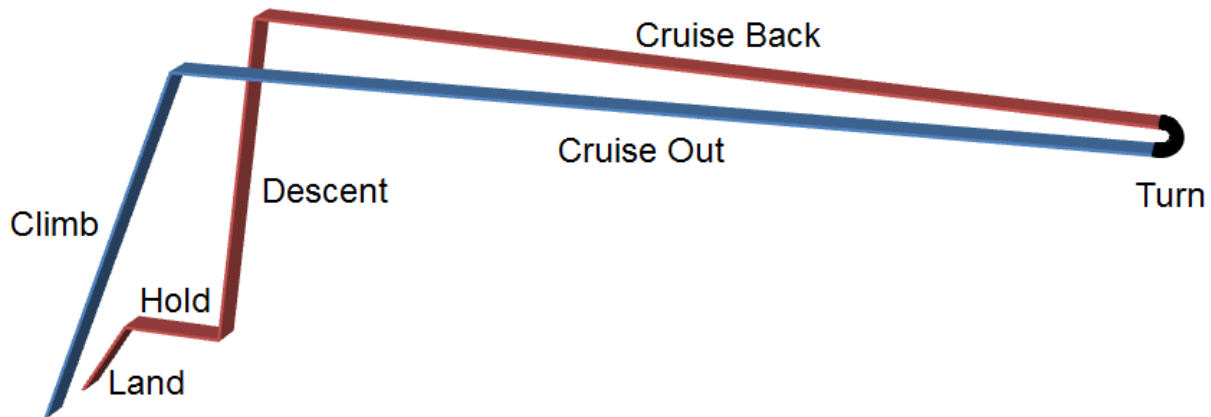


Figure 1.2-2. ESAV mission profile.

An ESAV may be manned, unmanned, or capable of either of these flight modes, the so-called "optionally-manned" configuration. With the advent and popularity of many various unmanned aerial vehicles (UAV) for military applications, each new military aircraft design, now and in the future, must weigh the costs between an unmanned or manned configuration. Since some design methodologies and constraints are different depending on this choice, it must be made earlier rather than later in the design process. The UAV design choice has the advantages of,

- not being limited by human frailties so usual design restrictions can be loosened,
- no risk of human crew loss of life,
- no search and rescue resources needed for downed crew members, and
- pilot training is done with simulators so flight time on vehicle is significantly reduced [4].

Except for the advantage of eliminating the crew loss of life risk, each of these advantages needs to be qualified. Although design constraints can be relaxed, the design is still subject to physics. For instance, the maximum g-load may be increased with no human pilot, but this does not mean that a corresponding increase in weight to the structure will not be required. In addition, the ride quality may not be as stringent without a pilot, but the added electronics

required for a remote pilot and software will have to be designed and qualified to withstand the more extreme environment, thus adding cost and complexity to the avionics design and the test program. Although no search and rescue resources may be needed for downed crew recovery, the operator may be very interested in not letting an advanced UAV design fall into the hands of a foreign nation [7]. Therefore, this would require resources for the recovery or confirmed destruction of any downed UAV nonetheless.

Another important consideration for the ESAV is that embedded engines will almost certainly complicate any repair of the engines (increase time and cost). The difficulty of working on embedded engines can be seen in Fig. 1.2-3. Since training of UAV pilots is typically done through simulations, the flight time of UAV's is cut back to only actual missions. This offers a chance to design for a limited flight-time life and could perhaps include the assumption that the engines will not need service that requires removal from the airplane during this shorter lifetime.



Figure 1.2-3. Man performing maintenance on an embedded engine in a B-2 bomber.

This is potentially an important advantage for the military's cost-aware acquisition process, since the cost to design and manufacture an aircraft that does not need a way to remove embedded

engines can be reduced. However, for the particular ESAV mission considered in this effort, the propulsion system complications stemming from flying in the supersonic flight regime, the long-range missions, the non-disposable nature of this aircraft, and the expectancy of a large dollar investment to procure a fleet of aircraft all make it highly unlikely that the ESAV would be designed without the capability to remove the engine for maintenance or replacement.

The main advantage of a manned military air vehicle is its capacity to handle the ambiguity and uncertainty of armed conflict situations, although work is in progress to improve UAVs in this area [8]. The dynamic ability of a pilot to deal with the unexpected as well as his or her ability to think (and often think creatively) about various changing situations that the pilot could possibly be faced with make this advantage of great consequence. Specifically, for anti-access and area-denial (A2/AD) missions [9] [10], the intelligence gathered upon which the mission is based may be incomplete and contain uncertain details. The bulk of this intelligence would probably have been obtained from orbit-based assets that can misrepresent the situational circumstances. Under these conditions, a military commander would most likely prefer a pilot in the cockpit rather than an UAV pilot at a remote station interacting with the aircraft through electronic sensors only.

From the previous arguments, it is easy to see where both the unmanned option and manned option might be the right answer for an ESAV. It is proposed in this work that the ESAV be designed to accommodate the operation of the vehicle by either a human pilot or remotely; an optionally-manned aircraft. This decision is supported by statements that ensure the place of manned aircraft in the military for many years to come [11]. Either of these options may be the right choice for an ESAV's various possible uses. However, this design decision is accompanied by a penalty in that both the life support equipment and remote pilot electronic equipment need

to be installed and maintained. In addition, the volume of a cockpit and UAV support electronics must be carved out and occupied at all times. This is volume that might otherwise have been used if one or the other options were chosen. Fortunately, the equipment is not always mutually exclusive. For instance, the air conditioning system required for a human could also be used to keep the UAV electronics cool. Designing the ESAV for a future mixed fleet (some of the airplanes would be manned only and some unmanned only) was not chosen because of the attractiveness of the versatility of having both options available at any time in all aircraft in the fleet.

An ESAV holds many challenging and complex interactions that must be accounted for in the design process. These interactions occur because the change in physical response involving one design parameter invariably affects many other parameters throughout the whole aircraft system. This problem, like other aircraft design problems, is nonlinear, highly coupled, and disjointed. Competing design parameter effects that live in this environment must be correctly captured, including their dependence on other design parameters, and balanced to find the optimum configuration. This is a non-trivial problem.

The process of building a suitable framework to capture the physics of an ESAV design and finding the optimum configuration is the subject of this effort. A background in mathematical optimization is required to first understand what the definition of optimal is, and second, how to determine optimal solutions to a given problem.

1.3 The Beginnings of Optimization

Nature itself has often been found to be an optimizer. Many natural systems try to get the best outcome or state for the least cost or energy. Often the “objective function” is the minimization energy for a given system in nature. The Clausius statement of the second law of

thermodynamics asserts that heat energy transfer cannot go from a cooler body to a hotter body [12], or in other words, heat energy transfer direction drives the system temperature to a minimum everywhere. Single soap bubbles and “double bubbles” [13] are another example of nature’s optimization tendencies. In this case, the bubble’s surface area is minimized around the enclosed bubble volume [13].

Optimization, as a mathematical problem tackled by humans, dates back to the 17th century [14]. In 1696, Johann Bernoulli published a famous problem known as the “brachistachrone problem” that asked what path should a bead follow between two points (only under the gravitational force) in order to minimize the time of travel. In 1697, Jacques Bernoulli gave the mathematical proof that the path was indeed a cycloid rather than a straight line [14]. Johann and Jakob Bernoulli, Newton, Leibniz, and L’Hôpital provided the solution path. In particular, Jakob Bernoulli’s solution provided the beginnings for what became known as the calculus of variations [14].

The practice and theory of the calculus of variations deals with real-valued functionals. Specifically, it characterizes the minima, maxima, and other critical values of these functionals. It also is concerned with the existence of these values [15]. The main idea is the development of necessary conditions and sufficient conditions which accompany local extrema.

1.4 Constrained Multi-Variable, Nonlinear Optimization

Almost all real engineering systems will have one or more constraints imposed upon it. It could be material strength limits, geometrical or size limits, weight limits, power limits, or some combination of these and other possible limits. Aircraft design problems are certainly posed with many constraints that must be intelligently dealt with. A more detailed background in

optimization principles is given in Appendix A, and may be beneficial for the reader to review this material before reading this section.

To understand the process of incorporating constraints, Fig. 1.4-1 [2] shows an example of a one-dimensional function, $\theta(x)$, with three inequality constraints, g_{1-3} . This figure could also include equality constraints but their behavior is just like active inequality constraints and would therefore be redundant for the purposes here. The gradient of the function at the extremum point, $\nabla\theta(\hat{x})$, is exactly canceled by the sum of the two gradients of the active constraints, ∇g_1 and ∇g_2 , multiplied by some constant. This can be seen in Fig. 1.4-1 in an intuitive graphical manner with the three green vectors.

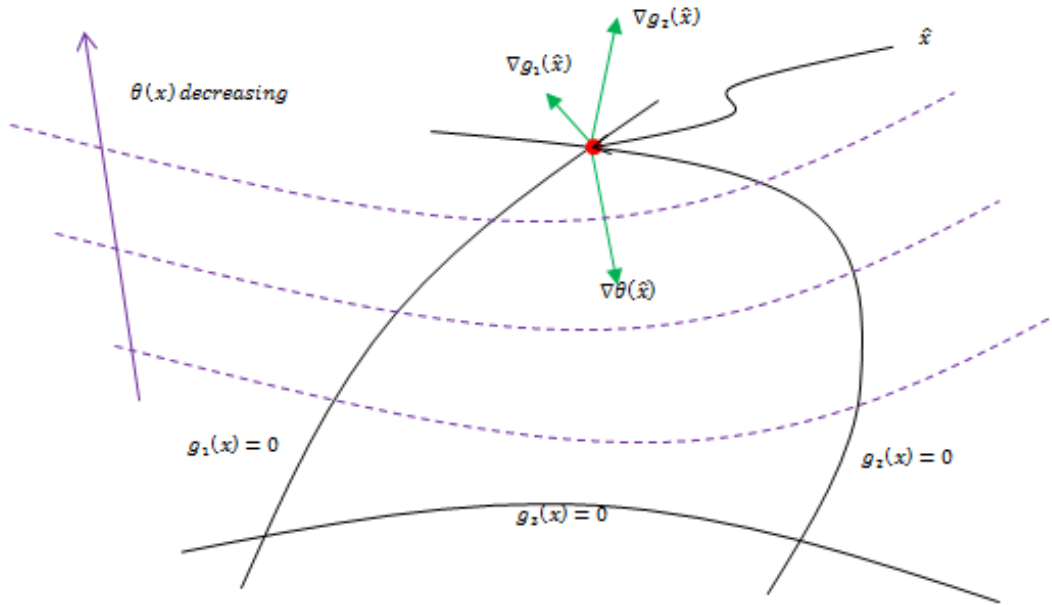


Figure 1.4-1. Graphical representation of the constrained optimal problem.

Mathematically, this can be expressed as,

$$\nabla\theta(\hat{x}) = \sum_{i=1}^3 \hat{\mu}_i \nabla g_i(\hat{x}), \quad (5)$$

where $\hat{\mu}_i = 0$ for the inactive constraint and some constant scalar value for the two active constraints. This condition requires that the sum of the gradients of the constraints at the extremum point be in the negative direction of the function gradient at the same point. These multiplicative scalar constants, $\hat{\mu}_i$, are called Lagrange multipliers.

The first step to solving the constrained problem is to form the Lagrange function, which for a problem with four inequality constraints and one equality constraint is written as,

$$L(x_1, x_2, \mu_1, \mu_2, \mu_3, \mu_4, \lambda) = f(x_1, x_2) - \sum_i \mu_i g_i(x_1, x_2) - \sum_j \lambda_j h_j(x_1, x_2). \quad (6)$$

The μ_i coefficients are the Lagrange multipliers for the inequality constraints, $g_i(x_1, x_2)$, and the λ_j coefficients are the Lagrange multipliers for the equality constraints, $h_j(x_1, x_2)$. The inequality constraints are written in a form so that $g_j(x_1, x_2) \geq 0$. Likewise, the equality constraints are written so that they are equal to zero.

The constrained optimization problem is larger than the unconstrained optimization problem because of the addition the Lagrange multipliers. These multipliers increase the number of unknowns that must be solved for. The stationary points formulated with the Lagrange function are found by taking partial derivatives equal to the number of unknowns, and setting them equal to zero (or less than or equal to zero in the case of inequality Lagrange multiplier derivatives).

The necessary conditions for a solution to the minimization problem are captured in the Karush–Kuhn–Tucker (KKT) necessary conditions [16]. These conditions are necessary to identify the stationary points of the Lagrange function. The optimality conditions (Eq. 7a), feasibility conditions (Eq. 7c), non-negativity conditions (Eq. 7d), and complementarity conditions (Eq. 7e), are enforced for the inequality constraints. The equality constraints must

satisfy the optimality conditions and feasibility conditions (Eq. 7b) only. The full set of conditions is stated mathematically in Eq. 7 [17].

$$\nabla L(\hat{\mathbf{x}}, \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\lambda}}) = \nabla f(\hat{\mathbf{x}}) - \sum_i \hat{\mu}_i \nabla g_i(\hat{\mathbf{x}}) - \sum_j \hat{\lambda}_j \nabla h_j(\hat{\mathbf{x}}) = \vec{0}, \quad (7a)$$

$$h_j(\hat{\mathbf{x}}) = 0, \quad (7b)$$

$$g_i(\hat{\mathbf{x}}) \geq 0, \quad (7c)$$

$$\hat{\mu}_i \geq 0, \quad (7d)$$

$$\hat{\mu}_i g_i(\hat{\mathbf{x}}) = 0. \quad (7e)$$

Given a solution to the optimality condition that satisfies all of the KKT conditions, $(\hat{\mathbf{x}}, \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\lambda}})$, then it can be said that this solution describes a stationary point.

Many different optimization schemes have been devised and implemented over the years, but still no true global optimization method exists. In other words, there is no guarantee, or necessary and sufficient condition, for a nonconvex problem that a global optimum has been found. This fact requires some ingenuity and thought in order to find the best design possible within the design space. This is especially true for the complex, large, and nonconvex ESAV design space that is simultaneously pervaded with local optimal points.

1.5 A Brief History of Aircraft Multidisciplinary Analysis, Design, and Optimization

Now that the basics of mathematical optimization have been reviewed, the modern problem of aircraft design using optimization principles can be more fully introduced. There have been several different approaches to aircraft design taken over the years. Certainly, aircraft were being designed long before the modern electronic computer was available, which is necessary for nonlinear optimization. For instance, the Wright Brothers of Dayton, Ohio used a combination of

mechanical expertise, new and lighter engine technology, imagination, and extensive experimental studies to become the first to accomplish controlled powered flight in 1903 on the beach in Kitty Hawk, North Carolina. They discovered what worked and what didn't work through their experiments. This can be thought of as the trial-and-error way of aircraft design. As can be well imagined, this is the most expensive and time-intensive way to approach the problem, but it was really the only available way for the Wright Brothers to tackle it in their particular circumstances.



Figure 1.5-1. First flight of the Wright Flyer.

As aerodynamic theory began to develop through research, mathematical equations began to replace some of the experimental trial-by-error methods of the Wright Brothers. Many early aerodynamicists contributed to the ever-clearer picture of fluid flows and how they interact with bodies (like wings). Of particular note is Prandtl who came up with the notion of a thin viscous layer (boundary layer) formed by fluid flow over a surface. The many other fields of study that

are also necessary to design an aircraft also progressed. These include structural dynamics, stability and control, combustion and propulsion, and material science to name just a few. Likewise, technology was ever-improving; therefore new mechanical equipment became available to the aircraft designer. An aircraft designer now had the ability to use these mathematical formulations to understand the underlying physics, and thus make improved aircraft designs at a much lower cost. Trial and error was still necessary for many projects. A good example is the Bell X-1 which required extensive flight tests and reworks based on the experimental data those engineers were collecting before finally breaking the mythical “sound barrier” on Oct. 14, 1947. Even then, this first supersonic aircraft design (see Figure 1.5-2) would seem impractical with what can be accomplished using modern design tools and physical theory.



Figure 1.5-2. The Bell X-1 (NASA photo).

Moving to the 1990's, computers began playing a larger role in aircraft design with the first aircraft assembled entirely in CAD without the need for a mockup was the Boeing 777 [18] [19].

Many aircraft MDO efforts, too numerous to mention here, have been completed since the 1990's. In particular, there have been some examples of supersonic aircraft designed with MDO. The early supersonic efforts focused on parts of the overall aircraft. For instance, Hicks et. al. [20] performed airfoil design optimization of a Mach 1.3 flow. This was due to the large computational cost that would be required if a whole aircraft system were designed that was associated with the computers of that time. However, modern supersonic optimization studies that also only focus on a particular aircraft component [21] are due to the use of very high-fidelity models and the very expensive computational effort involved even on the faster modern computers available to researchers.

One concept that has received a lot of attention is the High Speed Civil Transport (HSCT). Virginia Tech's Aerospace and Ocean Engineering department was one of these efforts [22] [23] during the 1990's. An even earlier HSCT work only incorporated aerodynamics and structures [24]. Other HSCT works were completed too. A large and long-lived effort has been undertaken at NASA Langley [25] [26] to use MDO for the design of a supersonic transport aircraft. This NASA work has been the beneficiary of collaborative efforts with other organizations like Georgia Tech [27].

Into the 2000's even more HSCT work was done, but a shift occurred in the field towards performing MDO for supersonic business jets. On the HSCT side, higher-fidelity tools were incorporated into the MDO framework like in Alston et. al. [28], and this was also true for the NASA Langley effort [29] [30] [31]. The supersonic business jet work began with aerostructural

multidisciplinary design optimization [32]. A new discipline, sonic boom analysis, was added to the MDO disciplines [33] [34] because the major limiting factor of introducing a supersonic business jet for commercial use was the regulatory restriction of supersonic flight over land. Conceptual and preliminary design with MDO techniques were also used for the supersonic business jet [35]. Propulsion was recognized as an important discipline in supersonic aircraft MDO beyond the traditional aerodynamics and structures disciplines [36]. Also, a large NASA-funded study was also undertaken to explore ways in which to reduce the design time and computational cost of supersonic aircraft MDO [37].

The ESAV, which is a long-range, supersonic, penetrating bomber aircraft, is an important concept that has not been explored with multidisciplinary tools in the open literature. This work seeks to fill in the gaps so that such a concept can be analyzed and characterized correctly.

Unique design challenges exist for this aircraft not seen in previous low observable bombers. The previous stealth bombers are all primarily subsonic aircraft. The supersonic requirement presents a very large challenge that did not have to be considered on similar subsonic aircraft. One area that this can be seen in is the embedded engine propulsion subsystem. Supersonic flight will make the design of the inlet much more difficult. In addition, the embedded nozzle will exist in an extreme environment due to the hot supersonic exhaust jet. It is very inefficient to use an augmenter on the ESAV mission that spends most of the time in supersonic cruise, and so the engines must be designed properly with this design goal. Similar supersonic issues could be stated for the aerodynamic, stability and control, structures, and LO configuration designs. The MDO tool which has been built and used in this work to explore the design space of an ESAV, and find an optimum configuration, is unique in the literature and a

valuable asset to further the understanding of the complicated relationships between aircraft design parameters in this nonlinear, disjoint, and highly coupled design space.

The rest of this work is organized according to the following brief outline. Chapter 2 details the basic tools needed to perform ESAV MDO. Chapter 3 describes the low-fidelity ESAV MDO framework and its results. Chapter 4 uses the propulsion discipline as an example to show the process to choose a higher-fidelity model than what is available in the low-fidelity ESAV MDO framework. Chapter 5 expands on the details of all discipline models included in the higher-fidelity ESAV MDO framework, and Chapter 6 discusses the practical implementation of the higher-fidelity ESAV MDO framework on a computer. Chapter 6 also reports the results of the higher-fidelity ESAV optimization result, and some further analysis on that aircraft. Finally, Chapter 7 draws conclusions about the work herein.

2. Basic Tools for an MDO Framework: Environment & Geometry

Before a practical implementation of a multidisciplinary framework can be constructed, two issues need to be decided upon. The first is what individual discipline computational programs will be chosen and connected together, and the second is how the vehicle geometry will be represented so that the disciplinary analyses will be working with the correct ESAV physical representation. Without these two questions resolved, individual discipline software development can only be done in a nonconcrete environment, although development can technically still go forward to a point.

2.1 MDO Environment

There are many ways to construct a practical implementation of a computational MDO framework. The options vary from using non-commercial tools [38] [39], commercial tools [40] [41], or constructing one's own tool. However, each must perform the basic tasks of executing the selected software models, linking the optimizer algorithm to its constraint values and objective function, and performing data input and output while taking into account the data relationships with the other various software models [42]. Of course, there are many other desirable features that could be included, and all open-source or commercial products have some combination of these.

The tool chosen for the ESAV low-fidelity MDO framework is Phoenix Integration's ModelCenter™ environment [41]. This tool offers an easy graphical interface to set-up the order of execution of analysis codes, the computer on which they will run, and any logic required between code interfaces. It also provides for the easy data transfer between analysis codes. Other

desirable features are tools that allow easy execution and set-up of parametric studies, design of experiments, and optimization. Data visualization of results is straightforward and intuitive. In addition to these features there are many others which make this a complete tool and capable of doing all that is required for this effort.

2.2 Parametric Geometry Model

During the conceptual design stage, the primary goals of an aircraft designer are to define the weight, size, performance, and configuration of the vehicle [43]. The configuration and size are explicitly related to the aircraft's outer mold line (OML), and weight and performance can be calculated from the various disciplinary analyses that are, in turn, largely based on the geometry. Thus, the centrality of the geometry model to the MDO process makes this discipline very important to the complete framework. It is with this view that the majority of the chosen global design variables for the aircraft configuration naturally reside within the geometry representation of the aircraft.

The *idealized* parametric geometry model would be intuitive to its users, detailed enough to support high fidelity analysis, and simple enough to be altered hundreds or thousands of times in a single optimization study without excessive computing time. There are several existing parametric geometry methods including Kulfan's class shape transformation (CST) method [44], Bezier curves [45], nonuniform rational B-splines (NURBS) [45], NASA's Vehicle Sketch Pad (OpenVSP) [46], and full CAD packages. Both Bezier Curves and NURBS offer good options for the purposes of the ESAV MDO framework. However, neither of these methods allows intuitive adjustments to the aircraft shape. Also, both are relatively difficult to implement. OpenVSP has had sizeable resources dedicated to its development, and it works with intuitive aircraft design variables. However, the geometrical interfaces with analysis programs are still

cumbersome and are limited to a few pre-defined formats. Full CAD packages have been used for geometry in past efforts for aircraft MDO [47] [48], but developing models for automation is very challenging. Despite this, the power of CAD tools remains quite promising. The CST method was chosen as the foundation for the geometry generation module due to its alignment with the idealized requirements [44]. The development of the practical implementation of the CST method was done in collaboration with Craig Morris, and resulted in the code now known as VT-CST [49].

CST produces continuous and smooth surfaces with relatively fewer design variables than other geometry methods. The geometrical bodies that it produces are watertight, and local control of their shape is done through amplifiers, but not at the cost of destroying the continuous and smooth surface inherent to this geometry representation. Kulfan's application of the class shape transformation method is for aircraft, and therefore two broad classes of bodies are defined. These are the airfoil shape and a body shape. The method was first developed for two-dimensional shapes and then extended to three-dimensional (3D) shapes [44].

The Kulfan CST method is based on an underlying assumption that the “general and necessary” [44] mathematical equation to describe an airfoil geometry must take the form

$$\frac{z}{c} = \sqrt{\frac{x}{c}} \left(1 - \frac{x}{c}\right) \sum_{i=0}^N \left[A_i \left(\frac{x}{c}\right)^i \right] + \left(\frac{x}{c}\right) \left(\frac{\Delta z_{TE}}{c}\right). \quad (8)$$

Here, x and z represent Cartesian coordinates, c is the airfoil chord length, A_i represents a vector of constants, N is the vector length, and Δz_{TE} is the trailing edge thickness. The pieces of this equation represent the fundamental equations of the CST method. The coefficient of the first term on the right-hand-side is the class function,

$$C \left(\frac{x}{c}\right) = \sqrt{\frac{x}{c}} \left(1 - \frac{x}{c}\right), \quad (9)$$

and it provides the rounded nose and tapered ends necessary to draw typical airfoil shapes. The shape function,

$$S\left(\frac{x}{c}\right) = \sum_{i=0}^N \left[A_i \left(\frac{x}{c}\right)^i \right], \quad (10)$$

describes the distribution of thickness along the chord length of the airfoil. The A_i vector represents the coefficients in front of each term in the N^{th} order polynomial and will be elaborated on later. Kulfan notes that while a polynomial shape function is not necessary, it is among the simplest to implement and is used in the work to be presented here. Another possible example is given by Straathof et al. [50], where in addition to the Bernstein polynomials, B-splines are also included.

The second term of Eq. 8 represents the trailing edge thickness of the airfoil as a fraction of the local chord,

$$TE_{thickness} = \left(\frac{x}{c}\right) \left(\frac{\Delta z_{TE}}{c}\right). \quad (11)$$

This term allows for a controlled finite thickness at the trailing edge, a technique that can be very useful in the design of transonic airfoils. Therefore, Eq. 8 is really the class function times the shape function with the trailing edge thickness added on. Again, the class function defines either an airfoil or a body class of object and the shape function defines the local shape control parameters called handles. It is easy to see that the handles are the A_i coefficients and there are “ N ” of them since a N^{th} order polynomial was chosen to represent the shape.

Further control of the shape can be had by changing the exponents of the two terms in Eq. 9. These are represented by two parameters called N_1 and N_2 . The class function can then be rewritten with these exponents as

$$C\left(\frac{x}{c}\right) = \left(\frac{x}{c}\right)^{N_1} \left(1 - \frac{x}{c}\right)^{N_2}. \quad (12)$$

For the standard round-nosed-airfoil, the class function exponents are $N_1 = 0.5$ and $N_2 = 1.0$.

By modifying these exponents, however, many different classes of shapes can be defined. Figure 2.2-1 shows several examples of shape classes that can be created using the (N_1, N_2) sets of

- $(0.5, 1.0)$
- $(0.75, 0.75)$
- $(0.75, 0.25)$, and
- $(1.0, 0.001)$.

For each of these plots, $\Delta z_{TE} = 0$, $c = 5$, and $S\left(\frac{x}{c}\right) = 1$. The lower surfaces are also plotted by taking the mirror image of the upper surface.

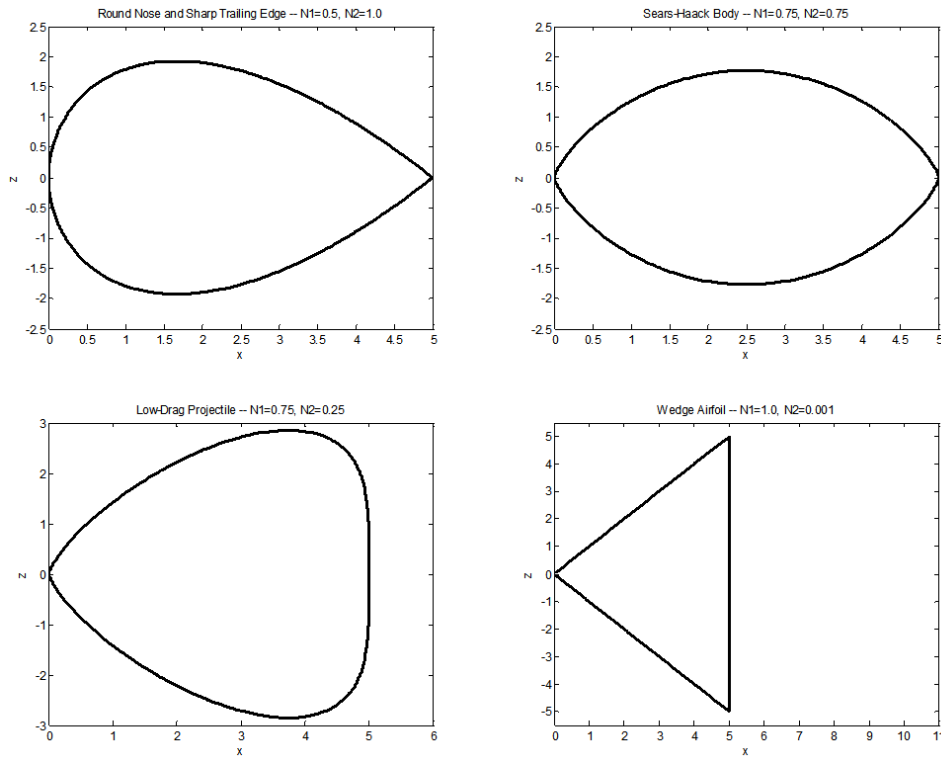


Figure 2.2-1. Sample two-dimensional airfoil geometries with varying (N_1, N_2) values.

Bernstein polynomials are chosen to define the shape function because of their utility as a partition of unity [44]. They are special functions so that when evaluated on the domain, $\frac{x}{c} \in$

$[0,1]$, they will always return unity. Thus, when used as the shape function, the Bernstein polynomial provides a means of parameterizing the geometry without modifying the general class shape (only multiplying by unity). The r^{th} term of a n^{th} degree Bernstein polynomial ($n + 1$ total terms) can be written as

$$S_r\left(\frac{x}{c}\right) = K_{r,n} \left(\frac{x}{c}\right)^r \left(1 - \frac{x}{c}\right)^{n-r}. \quad (13)$$

Here, r is an integer with values from zero to n , and $K_{r,n}$ are coefficients defined as

$$K_{r,n} \equiv \binom{n}{r} \equiv \frac{n!}{r!(n-r)!}. \quad (14)$$

Once parameterized, the unit shape function can then be modified from unity by including amplifying coefficients, A_r , such that the composite non-unity shape function becomes

$$S_{r,n}\left(\frac{x}{c}\right) = \sum_{r=0}^n \left[A_r S_r\left(\frac{x}{c}\right) \right]. \quad (15)$$

Note the similarity with Eq. 10. The level of shape manipulation achievable is controlled through the polynomial degree, n , in Eq. 15. Increasing n will increase the geometry model fidelity by increasing the number of available amplifiers (A_r) and allow for finer control of the class function geometry. Decreasing n has the opposite effect. Therefore, a careful trade must be made to determine the best value of n for a given application such that the number of design variables is minimized while the required geometric flexibility is still obtained.

The amplifiers behave as "handles" placed chordwise along the surface to increase or decrease the local thickness while maintaining a smooth shape. Therefore, these handles can be used to arbitrarily change the shape of the basic class function category. In addition, the upper and lower surfaces can be defined with different values for their respective handles and trailing edge thicknesses, enabling both camber and variable thickness distributions. An airfoil's

geometry is described by two related equations for the upper and lower surfaces. Together these make up one complete shape.

$$\left(\frac{z}{c}\right)_{upper} = C \left(\frac{x}{c}\right) S_r^{upper} \left(\frac{x}{c}\right) + \left(\frac{x}{c}\right) \left(\frac{\Delta z_{TE}^{upper}}{c}\right) \quad (16)$$

$$\left(\frac{z}{c}\right)_{lower} = -C \left(\frac{x}{c}\right) S_r^{lower} \left(\frac{x}{c}\right) + \left(\frac{x}{c}\right) \left(\frac{\Delta z_{TE}^{lower}}{c}\right) \quad (17)$$

It can be seen that since the upper and lower shapes can be different, almost any type of airfoil can be represented. This method can be extended to a three-dimensional geometry with many additional design parameter options like twist, taper, sweep, leading edge breaks, trailing edge breaks, conical camber, etc. The details follow from the two-dimensional theoretical derivation above [44].

The CST method offers intuitive design variables, water tight models, and flexible outer mold lines capable of describing many different aircraft configurations. The developed practical implementation of the CST geometry model for this work is called VT-CST [49]. This module is constructed using the object-oriented language, C++, and executes very quickly. It is simple to use as it only requires one small ASCII input file to describe the desired aircraft geometry.

Appropriate assumptions were made that simplify the framework's application of CST to represent the OML of an ESAV geometry, like not including tail geometries. These assumptions permitted the timely coding of the geometry model while maintaining its applicability to the ESAV MDO objective. An example of a notional ESAV drawn with the VT-CST geometry program is shown in Figure 2.2-2.

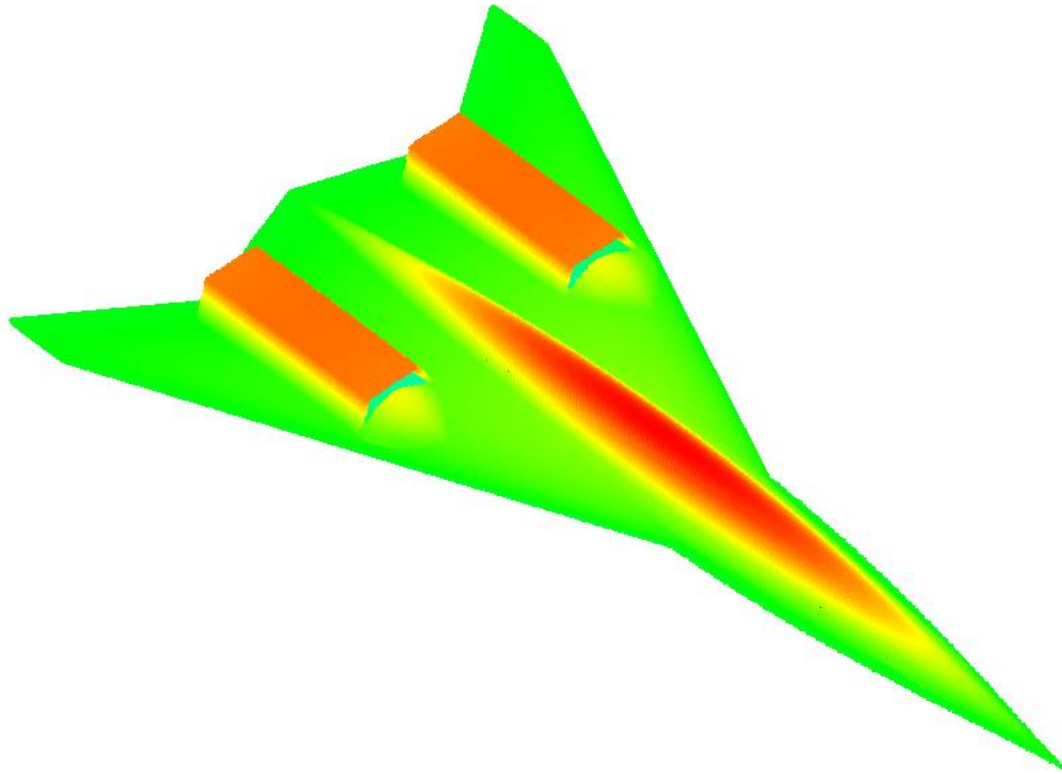


Figure 2.2-2. Geometry model drawing of a notional ESAV outer mold line including wing, fuselage, cowls, and inlets.

The program's simple single text file input method is easily incorporated into the ESAV MDO framework through file wrapping. File wrapping is the process of writing a small code to either read input values from a file or parse output values from a file. The geometry representation is parameterized so each parameter can potentially become a global design variable. Table 2.2-1 lists a subset of these possible parameterized geometry global design variables.

Many enhancements to the VT-CST geometry model have been made, either specifically suited to an ESAV, or just generally useful. Examples include area-ruling, continuity at the airplane centerline, control surfaces, and preservation of traditional and intuitive aircraft design variables [49].

Table 2.2-1. Examples of Possible Global Design Variables in the Geometry Model

Global Design Variable	Description	Global Design Variable	Description
L	Fuselage total length	W	Max. fuselage width
N _C	Fuselage cross section class function exponent	N _D	Fuselage distribution class function exponent
H _{top}	Fuselage height above X-Z plane	H _{bot}	Fuselage height below X-Z plane
X ₀	Starting X location of fuselage	K _{loc}	Area ruling location gain
b _{semi}	Wing semi-span	K _{mag}	Area ruling magnitude gain
C _{tip}	Tip chord length	K _{width}	Area ruling width gain
C _{root}	Root chord length	LE _{break}	Nondimensional location of a leading edge break
C _{flap}	Flap chord length	TE _{break}	Nondimensional location of a trailing edge break
Shear ₁	1 st order wing shear coefficient	λ _{LE}	Leading edge sweep angle
Shear ₂	2 nd order wing shear coefficient	λ _{TE}	Trailing edge sweep angle
H _{cowl}	Height of cowl	H _{ramp}	Height of ramp
L _{cowl}	Length of cowl	L _{ramp}	Length of ramp
Twist ₁	1 st order twist coefficient	Twist ₂	2 nd order twist coefficient
Twist ₃	3 rd order twist coefficient		...and so forth

The choice of the analyses models takes into consideration several levels of fidelity to balance the cost of computation time with computational accuracy. The VT-CST module is, in itself, a multifidelity geometry module. This is because the CST method has at its core the

polynomial partition of unity [44] as just discussed. The order of these polynomials effectively controls the number of design variables distributed about the surface of the OML that are available to locally vary the surface shape, and thereby the airplane's thickness. The coefficients of the polynomial terms are therefore aptly called thickness amplifiers.

By changing just one integer in the input file, the order of the polynomial can be increased or decreased, and the designer can work with a more or less refined distribution of thickness amplifiers. The cost for a more refined distribution is an increase in the number of design variables. A great benefit to easily changing polynomial order is that early efforts in the design process to narrow the design space can be performed with fewer design variables. This means that geometric refinement can take place after the scope of the design space is reduced for later design efforts. This feature offers a possible solution to balancing fidelity and computational cost, as fewer design variables are used for the larger design space and more design variables are used for the smaller design space. Figure 2.2-3 shows an example of the increase in amplifier density. When extended to three dimensions, the order of the Bernstein polynomial, n , is required in each axis. Therefore, n in the x -direction (chordwise coordinate) is denoted N_x , and n in the y -direction is denoted N_y (spanwise coordinate).

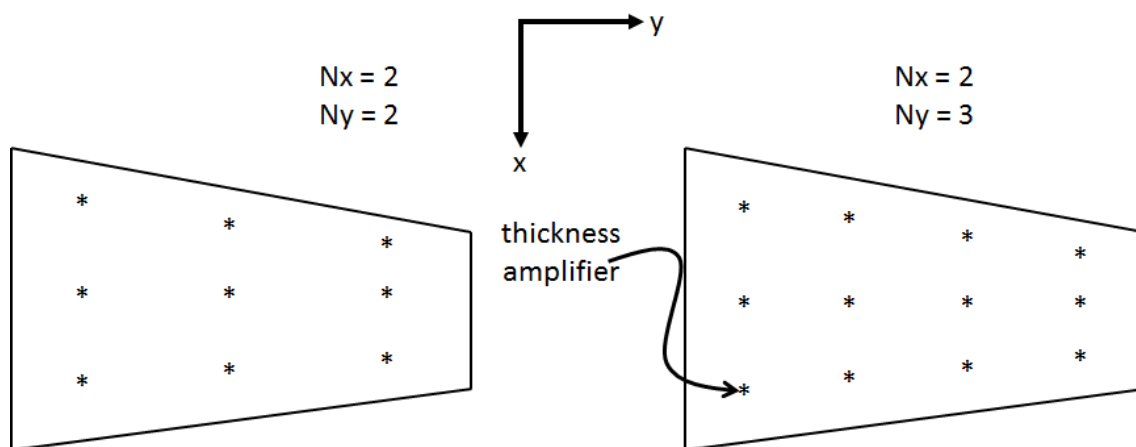


Figure 2.2-3. Increasing amplifier density with increasing Bernstein Polynomial order.

VT-CST includes many additional three-dimensional design parameter options that are theoretically possible with the CST method. These include twist (up to 2nd-order), shear (up to 2nd-order), sweep angles, leading-edge and trailing-edge breaks, conical camber, and several others. Some of these possibilities are shown in Fig. 2.2-4 below.

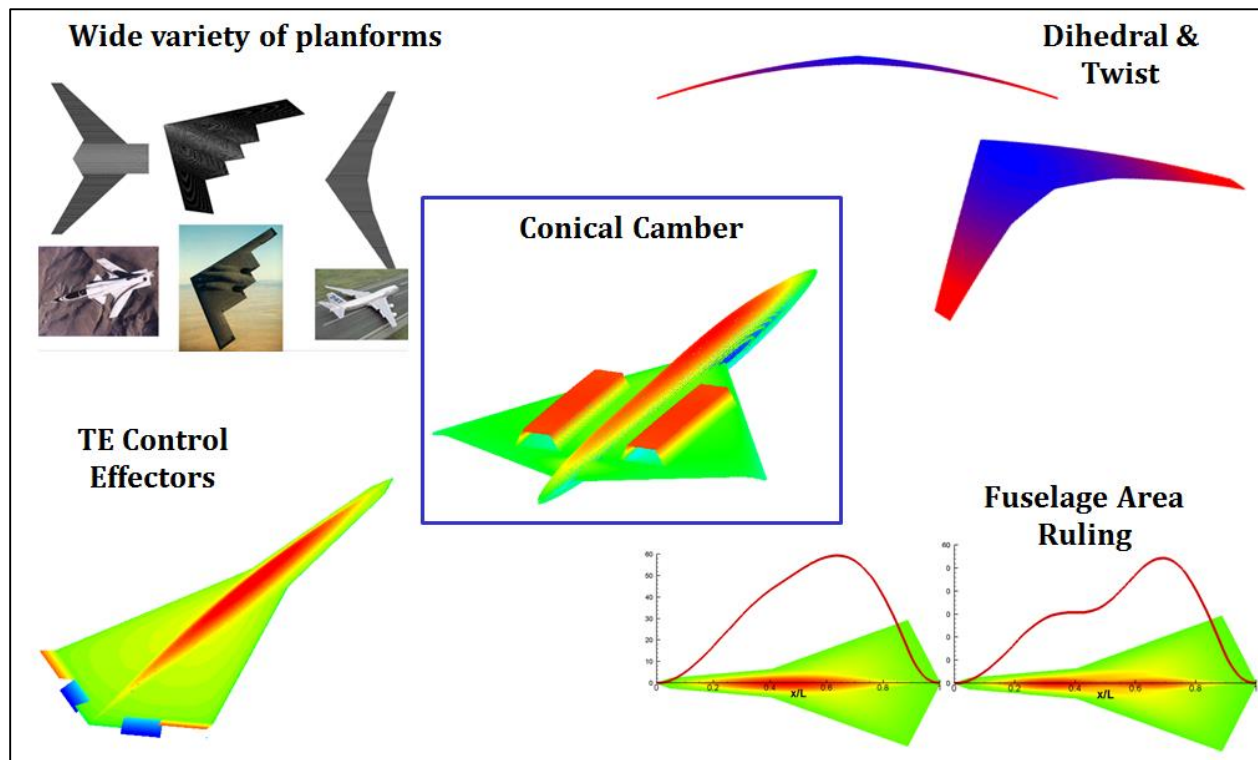


Figure 2.2-4. Various capabilities of the VT-CST model.

Each discipline analysis in the MDO framework must rely on the geometry generation tool's ability to interface with it in order to perform any analysis. While there are many currently available geometry formats, each different discipline's analysis model has its own interface requirements. Even within a single discipline, different models for different fidelity levels have their own unique geometry interface requirements. The power of the VT-CST module is realized in this critical task. The CST geometry model is an analytic function, therefore the model is able to represent almost any discretized geometry format. A stereolithographic (STL) file is an ordered triangular paneling of the surface that is generated using the analytic geometry

representation and nested for loops within VT-CST – along with logic. An example STL rendering of a notional ESAV geometry can be seen in Figure 2.2-5.

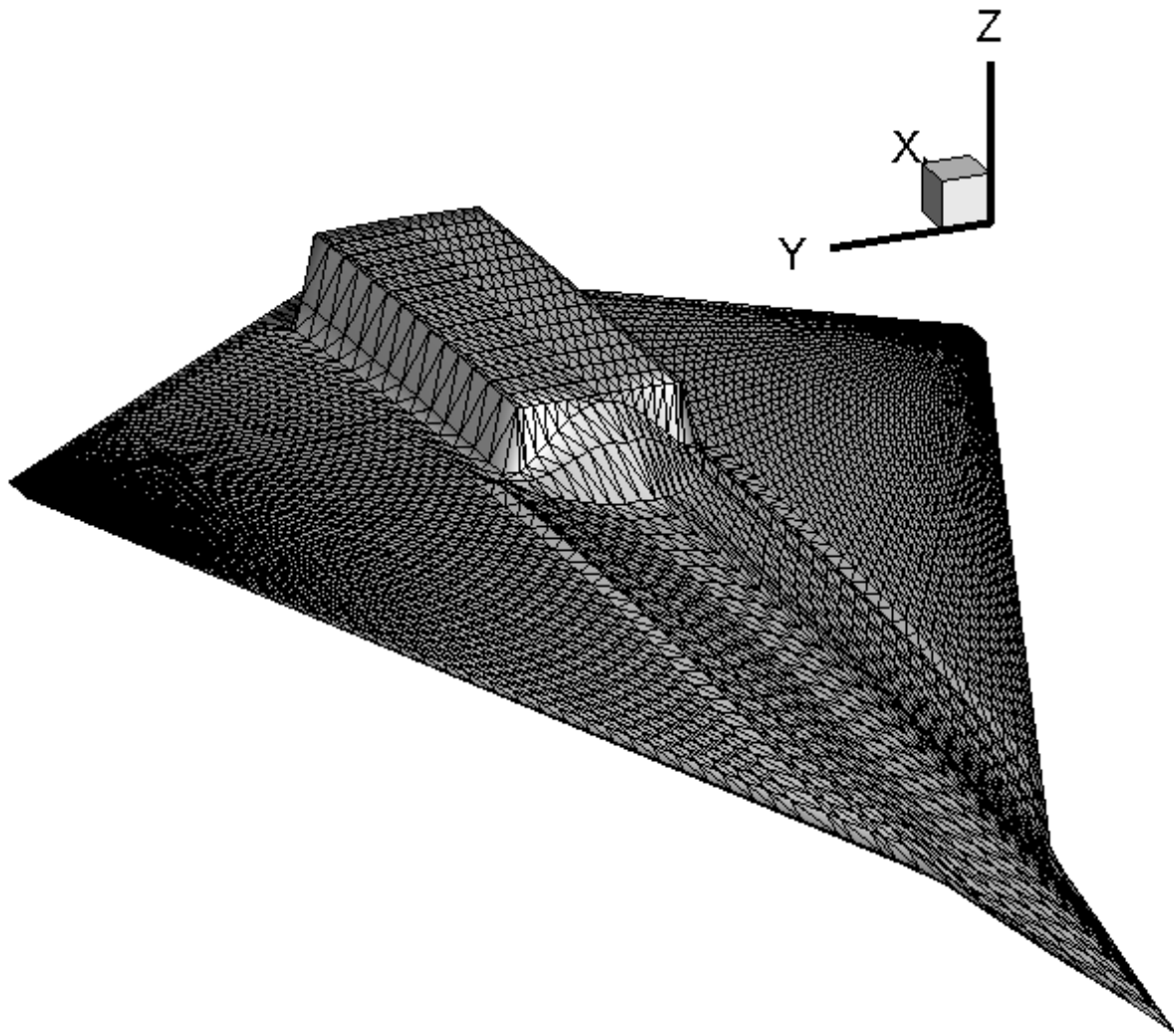


Figure 2.2-5. Stereolithographic (rapid prototyping) format output by VT-CST.

Many aerodynamic solvers have specialized means by which the OML information must be input. For instance, the lowest fidelity aerodynamic program that is part of the low-fidelity ESAV MDO framework makes use of a simplified version of the aircraft and is based on a format known as the Craidon geometry format [51]. The power of VT-CST is its ability to not only interface with this particular geometry format, but also with all other disciplinary modules

in the framework to facilitate the design optimization process. Point cloud formats are also regularly used for configuration visualization, and any other discretized format is possible. The key is that one parameterized geometry model is used to provide the design variable interface with each disciplinary analysis within the ESAV MDO framework.

3. Low-Fidelity ESAV MDO Framework

3.1 Low-Fidelity Framework

Any multidisciplinary design optimization (MDO) effort must start from somewhere – some baseline design or guess. For an aircraft, this could be a detailed design or a crude guess at certain well-known engineering parameters, like aspect ratio, wing area, fuselage length, span, and engine thrust. Generally, the closer the initial design is to the optimum design, the easier it is to reach an optimal design with optimization algorithms. This might not always be true, but it is certainly a good rule-of-thumb. Thoroughly searching a design space with an optimizer for a reasonably-sized aerospace vehicle design problem is computationally expensive. In order to make the process tractable, the designer must balance analysis run times and the fidelity at which a physical process can be adequately represented by the simulation tool.

Some discipline analysis models do not require a high level of fidelity in order to produce very good results, but others must be raised to a higher-fidelity level. These are usually the disciplines that have the greatest effect on the design. The approach taken in this work is to start at the lowest reasonable fidelity level (this usually translates to the fastest computational time), and then decide which disciplines need to have an increased level of fidelity or which disciplines need to be added to the framework to capture physical responses not modeled previously.

This tiered approach has at least two distinct advantages. The first is that the final framework is most likely to also be the least expensive computationally, but still capture all pertinent physics. The second is the advantage of establishing data relationships between disciplines early in the framework construction process. Defining these relationships early helps center all discipline leads on what is most important and how they fit into the overall process, or in other words, the big picture comes into focus for all stakeholders.

A low-fidelity MDO framework was constructed for the ESAV supersonic bomber concept. The starting point was to use existing models for the various disciplines. A robust and well-established NASA code called Flight Optimization System (FLOPS) offers a promising suite of aircraft conceptual design modules ready for low-fidelity calculations. The supported disciplines are: 1) weights, 2) aerodynamics, 3) engine cycle analysis, 4) propulsion data scaling and interpolation, 5) mission performance, 6) takeoff and landing, 7) noise footprint, 8) cost analysis, and 9) program control [52]. These modules are based on a wealth of information from previous aircraft designs (25 military aircraft and 17 commercial aircraft) [53].

3.1.1 Flight Optimization System

The Flight Optimization System is a well-developed aircraft conceptual and preliminary design tool [52]. Although the nine primary modules that make up the suite of multidisciplinary design tools all are useful in certain situations, only five are used in the present low-fidelity ESAV MDO framework as shown in Table 3.1-1.

Table 3.1-1. Available FLOPS Modules

FLOPS Module	Used in Low-Fidelity ESAV MDO Framework
Weights	Yes
Aerodynamics	No
Engine Cycle Analysis	Yes
Propulsion Data Scaling and Interpolation	No
Mission Performance	Yes
Take-off and Landing	Yes
Noise Footprint	Yes
Cost Analysis	No
Program Control	No

The FLOPS aerodynamics module provides drag polars for performance calculations and is a modified version of the empirical drag estimation technique [54]. It is not used in this low-fidelity framework, because the ESAV concept is a supersonic aircraft. Although FLOPS is capable of supersonic flight aerodynamics calculations they were not recommended [55] by the FLOPS developer. Instead, more suitable supersonic aerodynamics tools, based on physics, are employed for the aerodynamics analysis. The drag polars produced by the FLOPS aerodynamics module are still required for other FLOPS modules, therefore any other aerodynamics model used in the ESAV MDO framework must conform to the right format in order to use the FLOPS modules that rely on this aerodynamic information.

The interface between other FLOPS modules used in the low-fidelity framework that require aerodynamic results is handled through code written to “wrap” the low-fidelity aerodynamics model. This wrapping code automatically runs the model correctly to solve for all required aerodynamic characteristics at predefined points throughout the flight envelope. It then places the data into the correct format stipulated by the FLOPS program. The chosen aerodynamics model for the low-fidelity ESAV MDO framework is discussed in the next section.

In the low-fidelity ESAV MDO framework the FLOPS engine cycle analysis module was initially employed because the engine analysis is critical to any aircraft design (especially a supersonic aircraft) and also thrust is desired as a design variable so that the optimal engine design for the ESAV can be found. The propulsion data scaling and interpolation module in FLOPS uses an engine deck that has been input or one that has been generated by the engine cycle analysis module, fills in any missing data, and uses linear or nonlinear scaling laws to scale

the engine data to the desired thrust. This thrust-scaling function is turned off via a variable available in the input deck in order to preserve the ability to use thrust as a design variable.

In addition to not using the aerodynamics module from FLOPS, the geometry and optimization modules are likewise taken from other sources. The geometry model has already been discussed at length in this work. It is the in-house Virginia Tech code [49] called VT-CST that was developed specifically for the ESAV concept, but could be extended to represent geometries from a host of different aircraft configurations.

The FLOPS cost analysis module is not used in the ESAV MDO framework for two reasons. First, this module's primary purpose is for subsonic transport aircraft. Second, the focus of the current ESAV MDO framework is to explore the technical design space only, rather than including cost constraints at this stage.

Through the program control module, FLOPS may be used to analyze a point design, parametrically vary certain design variables, or optimize a configuration with respect to these design variables. The optimization objective (or cost) function can be defined to find the minimum gross weight, minimum fuel burned, maximum range, minimum cost, minimum NO_x emissions, or some weighted combination of these metrics. The optimal solutions are found using nonlinear programming techniques. Although these features exist in FLOPS, the program control module is not used in the low-fidelity ESAV MDO framework because these functions are handled by the framework's chosen optimizer package. In addition, the ModelCenter™ environment offers many advantages to the designer that are not available if the FLOPS program control module was used, and external design variables in the geometry model could not be controlled if using the FLOPS program control module – severely limiting the available ESAV design space.

The optimization model is chosen from over 30 choices available in Phoenix Integration's ModelCenter™ environment [41]. The optimization model controls each design variable by determining its value before each objective function evaluation.

The N^2 diagram [56] of the initial low-fidelity framework is shown below in Figure 3.1-1. The ordering of analysis models from left to right down the diagonal is deliberate. The guiding principle for the ordering is to minimize data pass-back relationships (data passing relationships underneath the diagonal) to speed execution. The resulting N^2 diagram is shown to have no data pass-back relationships in its final form. The aerodynamics model, FLOPS weights, mission performance module, take-off and landing module, and noise footprint are part of the ESAV MDO framework and are explained in the following sections.

Low Fidelity ESAV N^2 Diagram

Geometry	Planform, Mach #, Cruise Alt.	Thrust, Mach #, Cruise Alt.	Planform	Planform	Planform	Wing area	Planform
	Aerodynamics Low Fidelity					Aero. data in mission envelope	
		FLOPS Propulsion Module	Engine weight	Engine weight	Characteristics for engine noise	Thrust data in mission envelope	
			FLOPS Structures Module	Structural Weight			
				FLOPS Weights Module		Aircraft weight in mission envelope	TOGW
					FLOPS Noise Module	Noise Levels	
						FLOPS Mission and Constraints	Feasibility
Planform	Mach #, Alt.	Engine parameters					Optimization

Figure 3.1-1. Initial low-fidelity ESAV MDO framework N^2 diagram.

3.1.2 Aerodynamics Model

Initial choices for low-fidelity aerodynamic software focused on simplicity of implementation and minimizing run times. A suite of legacy codes were integrated together to provide the necessary lift and drag coefficients at both subsonic and supersonic Mach numbers. These codes are WINGDES [57], AWAVE [58], and FRICTION [59]. Combining their individual output

data into the required FLOPS drag polar table format is performed, so that the data can be used in the mission performance FLOPS module. Each code was required to provide different pieces of the total drag coefficient of a given ESAV design. Each of these codes works for both the subsonic and supersonic flow regimes. The codes' output is summarized in Table 3.1-2.

Table 3.1-2. Low-Fidelity Aerodynamic Codes and the Output Data

Code	Drag Output	Lift Output	Pressure Output
WINGDES	Induced drag coefficient – C_{Di}	Aircraft lift coefficient – C_L	Coefficient of pressure change over wing – ΔC_p
AWAVE	Volumetric wave drag coefficient – C_{Dw}	N/A	N/A
FRICTION	Friction drag coefficient – C_{Df} Form drag coefficient – C_{Dp}	N/A	N/A

The WINGDES code calculates the induced drag portion of the total drag coefficient. It determines the aerodynamic properties of a given wing using a panel method [60]. WINGDES accepts a wing design that can include camber and twist operating at specified flight conditions in either the subsonic or supersonic regimes [57]. It includes estimations for attainable leading-edge thrust. The coefficient of lift (C_L), and the change in the coefficient of pressure (ΔC_p) between the upper and lower surfaces over the whole wing are also output. This code is able to handle flow situations where the lifting surfaces are in a condition where they have partially separated and partially attached flow [60].

The AWAVE code is a modified version of the Harris wave drag code [61] from the 1960's that executes an order of magnitude faster than the original code [62]. In addition, while keeping all the same capability of the original code, it includes approximate effects of non-zero angles of attack and reduced numerical integration errors [62]. This program calculates the

volumetric wave drag coefficient portion of the total drag coefficient and makes the valid assumption that the induced wave drag contribution is negligible for highly twisted and cambered wings [61].

The FRICTION code estimates the laminar and turbulent skin friction and form drag portions of the total drag coefficient [63]. Flight conditions are used to find the skin friction along with equivalent flat plate skin friction methods. The laminar flow compressibility effects are found using the Eckert reference temperature method. The turbulent flow compressibility effects are found with the van Driest II formula. The Eckert reference temperature (T^*) formula is [64]

$$T^* = T_s + 0.5(T_w - T_e) + 0.22(T_r - T_s), \quad (18)$$

where T_e is the free-stream temperature, T_w is the wall temperature, and T_r is the recovery temperature. The 0.5 and 0.22 coefficients have been refined in more recent times since they were 0.58 and 0.19 respectively in Eckert's original paper [64]. For an adiabatic wall,

$$T_r = T_e + r \frac{U_e^2}{2c_p}, \quad (19)$$

where r is the recovery factor (approximately equal to the square root of the Prandtl number), U_e is the free-stream flow velocity, and c_p is the specific heat at constant pressure [64] [65].

Each of these legacy FORTRAN codes uses its own modified version of the Craidon geometry format [51]. The VT-CST geometry module shown in the low-fidelity N² diagram (see Fig. 3.1-1) creates the required geometry format for each of these codes and writes the appropriate input files. This is a critical step because any mistakes in the geometry will cause incorrect results to propagate throughout the entire design space negatively affecting such things as engine sizing, fuel burn, range, and mission performance, to name but a few.

These three aerodynamic codes together provide all necessary aerodynamic data to characterize a supersonic aircraft throughout the whole mission envelope – lift, drag, and the wing pressure distribution. Figure 3.1-2 portrays this graphically. It gives an overview of the low-fidelity aerodynamics model, shows which codes the components of the total drag come from, and gives example interfaces with other disciplines.

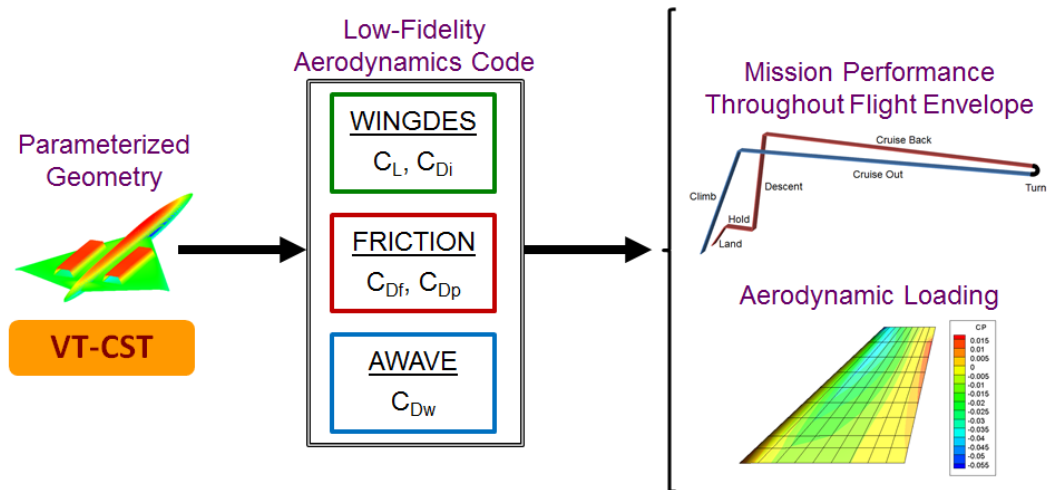


Figure 3.1-2. Low-fidelity aerodynamics code, drag breakdown, and example interfaces within the ESAV MDO framework.

3.1.3 Engine Cycle Analysis FLOPS Module

The FLOPS engine cycle analysis module is user-friendly and executes quickly. It is based on Quick NEP (QNEP) [66], which is, in turn, based on the Navy Engine Performance Computer Program (NEPCOMP) [67] and its successors like the Navy NASA Engine Program (NNEP) [68]. It is a small propulsion code designed specifically for conceptual and preliminary design studies. It provides the capability to internally generate an engine deck consisting of thrust, fuel flow, and other engine data at a variety of Mach-altitude conditions. Several built-in cycle files exist, but a user can optionally generate a new one.

FLOPS allows the user some limited design variables for the engine cycle chosen. For a turbojet, these design variables are maximum sea-level thrust, maximum turbine inlet temperature (T_4), and the overall pressure ratio. For a cycle with a bypass flow, additional design variables are the fan pressure ratio and the bypass ratio.

The engine data is passed internally to other FLOPS modules, but it can also be output as a stand-alone engine deck file. Part power off-design points are allowed. An afterburner (augmenter) is also an option that can be turned on. There are many controls in FLOPS for individual component tailoring. For example, the user can set the turbine cooling flow fraction from the high pressure compressor or change the fuel by adjusting the fuel heating value. Also, aircraft-propulsion integration issues can also be included through empirical relationships like boattail drag and spillage drag. Of course, the drag must be carefully documented so that the aerodynamics analysis in the MDO framework does not double count drag if it is being supplied by propulsion and *vice versa*.

In a later section, a higher-fidelity propulsion model is constructed for the MDO framework. When tests were conducted to see how well the FLOPS propulsion model compared to the higher-fidelity model, discrepancies were discovered in the estimation of the engine weight. A more thorough investigation discovered three reasons for the two models not matching. The reasons are that the FLOPS low-fidelity engine weight is limited due to:

1. its baseline engine weight being calculated by dividing the sea-level static thrust by one of only two possible constants – a subsonic transport constant or a fighter constant (neither of which are the same class of aircraft as an ESAV),
2. the serpentine inlet weight is not properly accounted for due to the built-in assumption of a traditional podded engine inlet, and

3. the assumption that changes to the overall pressure ratio do not affect the engine weight.

The third effect is small, and is, therefore, an appropriate assumption for conceptual design.

However, the other two assumptions by the low-fidelity model produce non-negligible discrepancies for an ESAV.

These two assumptions were overcome by the construction of a surrogate model that approximates the higher-fidelity propulsion model. The higher-fidelity model will be introduced in a later section. The surrogate model is constructed with a modified version of the Shepard algorithm [69]. Specifically, the SHEPPACK [70] code, which is part of the Transactions of Mathematical Software (TOMS) collection as algorithm 905 [71], was used to build the surrogate from the data generated by the higher-fidelity code.

The modified Shepard algorithm uses the inverse distance weighted interpolation method, but limits the weighting to those data points that are local to the point of interest, x . The weighting functions, $w_k(x)$, are defined as

$$w_k(x) = \left[\frac{(R_w^{(k)} - \|x - x^{(k)}\|)}{R_w^{(k)} \|x - x^{(k)}\|} \right]^2, \quad (20)$$

and the interpolated value, $\tilde{f}(x)$, is computed by

$$\tilde{f}(x) = \frac{\sum_{k=1}^n w_k(x) P_k(x)}{\sum_{k=1}^n w_k(x)}. \quad (21)$$

Here, $R_w^{(k)}$ is a strictly positive constant defining a radius of influence about the $x^{(k)}$ point and $P_k(x)$ is a local approximation to the function $f(x)$ centered at $x^{(k)}$. There are several different functions that can be used to create P_k that are mentioned in the SHEPPACK paper [70]. A linear function was chosen for this surrogate model due to the assumption that the propulsion data is locally linear, and for the linear function's smaller computational expense.

3.1.4 Structures and Weights FLOPS Module

The FLOPS structures and weights model is empirical, which has advantages and disadvantages. The disadvantage, especially to a new concept aircraft like the ESAV, is that it potentially breaks the curve fit trends of past aircraft. However, the main advantage to empirical estimation techniques for weights is the ability to account for all the unknowable weights during the initial design levels that appear on a real aircraft. A good example of a weight that can only be estimated empirically at the conceptual/preliminary design level is the fastener weight.

The FLOPS aircraft database, from which the empirical models are derived, includes the following supersonic or supersonic-capable bomber aircraft:

- Convair B-58A Hustler (Mach 2),
- North American B-70 Valkyrie (Mach 3),
- North American A-5 Vigilante (attack bomber),
- General Dynamics F-111A/B Aardvark (tactical fighter-bomber),
- Republic F-105D Thunderchief (tactical fighter-bomber), and
- McDonnell Douglas F-4B/E Phantom (multi-mission including ground-support bomber).

The B-58 and B-70 airplanes are important inclusions in the database as far as the ESAV configuration is concerned. This is because they are supersonic bomber aircraft like the ESAV, albeit from the mid-20th century. In addition, a study supersonic transport called the Boeing 969-336C was included in the FLOPS database. This is significant in that a supersonic transport has a similar mission to a supersonic bomber (deliver a payload using supersonic cruise). Several other fighter aircraft capable of supersonic flight are also included (e.g., F-104G, F-16A, F-18A).

The database captures the weights of these various aircraft by their major system components. These include the weights of the fuselage, wing, tails, landing gear, nacelles, and

engines. The secondary weights are also captured and include the thrust reverser, propulsion system weight, fuel system, surface controls, APU, instruments, hydraulic system, avionics, electrical system, air conditioner, anti-icing, engine oil, etc. Center of gravity coordinates and moments of inertia can also be calculated for multiple fuel conditions. The ultimate load factor is input as 3.75 (factor of safety of 1.5 with a maximum maneuver load of 2.5 g), which is a major factor in determining the structural weight.

3.1.5 Take-off and Landing FLOPS Module

The FLOPS take-off and landing module used in the low-fidelity ESAV MDO framework calculates the [52]

- field length of the all-engine take-off,
- balanced field length,
 - one-engine-out take-off,
 - aborted take-off for one-engine-out and all-engines-operating,
- landing field length, and
- approach speed.

Almost all FAR Part 25 and MIL-STD-1793 requirements are guaranteed to be met by this module for any feasible ESAV design. For take-off, the differences from MIL-STD-1793 are that the gear begins retraction at lift-off rather than when the climb gradient has reached 2.5%, and there is no ramp-up time from the start of braking to get the full benefit of the spoiler and thrust reverser effects for an aborted take-off. However, these effects are assumed to be minor for the ESAV, as its designs have been observed to consistently not be constrained by the field length during the exercise of the framework over the course of years. However, if the field length does

become a constraining factor in the optimization, then these departures from MIL-STD-1793 should be revisited.

Correctly calculating the rotation speed is critical to determine if the design meets the take-off requirements, and requires several iterations within the module. The requirements include the

- second segment climb gradient,
- missed approach climb gradient,
- constraints on lift-off speeds for different conditions,
 - one engine out,
 - all engines operating,
- constraint on speed at the obstacle, and
- no degradation in one-engine-out take-off performance for aircraft rotated 5 knots early.

The approach speed is the main parameter necessary to correctly analyze the landing performance. The critical condition to get the approach speed is the flare altitude that results in a soft landing. To this end, an additional constraint beyond the standard requirements is applied in the FLOPS module. This constraint is that the velocity cannot fall below the maximum of either

1. the minimum power-off unstick speed (i.e., the minimum velocity where the ESAV can actually take off at the maximum possible rotation angle) or
2. 1.1 times the stall speed (V_{stall}).

There is a subroutine that captures the ground effect based on a NASA contractor report [72] within this module. The take-off and climb profiles calculated in this FLOPS module are used in the FLOPS noise footprint module, which is also included in the low-fidelity ESAV MDO framework.

3.1.6 Mission Performance and Noise Footprint FLOPS Modules

The FLOPS mission performance module is based on an energy analysis and takes inputs from several different disciplines in an attempt to fly the ESAV through the prescribed mission profile (see Figure 1.2-2). The FLOPS mission module has all the functionality required (plus more if changes to the profile are made in the future) for the ESAV mission and runs very fast (on the order of a second). A standard atmosphere is assumed throughout the mission, although options exist to include deviations from the standard day.

The data this module requires before it can be executed come from the aerodynamics, propulsion, and weights analyses performed previously in the MDO framework (see Figure 3.1-1). In addition to the implicit limits set by a particular configuration, which is embedded within the data that comes from the other disciplines, the mission design variables of maximum cruise altitude and cruise Mach number are used to define the mission.

The mission profile is segmented into its various main parts -- start (taxi, etc.), climb, accelerate, cruise, turn, cruise, decelerate, hold/loiter, and descend (among others available in FLOPS but not required in the ESAV mission profile). Optimal climb, descent, and cruise flight paths are calculated based on the energy method. The cruise segments are not constrained to fly at a fixed altitude but can vary (usually increases up to the defined maximum cruise altitude) based on the changing weight during the flight. Alternate missions to the primary mission can also be specified and analyzed in the same call to this module for the same ESAV design as used in the primary mission profile.

The mission performance analysis holds most of the feasibility constraints. That is, a particular aircraft design is “feasible” only if it can fly through the entire specified mission. For

supersonic aircraft concepts, it is often not the supersonic cruise part of the mission profile but rather the low-speed flight segments that can be the most challenging.

The noise footprint module uses the take-off and climb-out profile generated in the take-off and landing module to compute noise footprint contour data or noise levels at user specified or FAA locations. It is based on the FOOTPR program [73] and is in line with the Aircraft Noise Prediction Program (ANOPP) results [74]. The FLOPS noise module calculates the noise levels at the FAR 36 observer locations [75].

Again, like the other FLOPS modules used in the ESAV MDO framework, this module runs very fast and outputs all required information about the ESAV. The noise sources are assumed to be the fan, exhaust jet, burner, turbine, flaps, and the airframe. By far, the jet exhaust is the greatest contributor to ESAV noise and could potentially be the only noise source included for simplicity. Ground reflection and attenuation effects as well as atmospheric attenuation are included. Noise shielding effects are also allowed in the noise calculation. The ESAV concept can take advantage of engine noise shielding because it has embedded engines mounted on the top of the wings – a benefit to the required low-observability engine design.

3.1.7 Optimization Models

An optimization algorithm must be chosen to perform the ESAV optimizations. The commercial ModelCenter™ software has many built-in optimization methods to choose from, each with convenient, ready-to-use, ModelCenter™ interfaces. A test optimization problem called the “Bird Problem” characterized with exclusively continuous variables and multiple local optima, similar to the ESAV MDO problem, was solved with each of the available ModelCenter™ optimization algorithms. The search space is constrained and the global optimal point is known [76]. The results showed a virtual tie for finding the optimal point among four of the algorithms,

but the number of function evaluations to get the solution varied greatly. Table 3.1-3 shows the results garnered from the Phoenix Integration website [76].

Table 3.1-3. Results from Phoenix Integration Showing the Best Algorithms for Solving a Simple but Similar Optimization Problem to the ESAV Problem

Algorithm	Avg. Error in Objective	Avg. Number of Function Evaluations
DAKOTA COLINY DIRECT	0.0013	194
NSGA-II	0.0087	815
Darwin	0	1649
EVOLVE	0.0072	3828

Darwin and EVOLVE are deemed too computationally expensive based on the number of function evaluations required to find the optimum. Therefore, the DAKOTA Coliny DIRECT and NSGA-II algorithms will be used to solve for the ESAV optimal design using the low-fidelity ESAV framework, and the best algorithm will be chosen for ESAV work based on which one finds the best objective function value. The number of function evaluations will also be taken into account, especially if it is excessive for either algorithm, regardless of whether the algorithm happens to find a better feasible ESAV design.

3.2 Low-Fidelity Design of Experiments and Optimization Problem

The collection of models that make up the low-fidelity ESAV MDO framework have been chosen and deliberately ordered via the N^2 diagram. Each model has been described in detail in the previous sections. In addition, the environment in which this framework was built has been chosen to be ModelCenter™. This MDO framework is exercised to produce a low-fidelity design of experiments study towards an optimal ESAV design.

In an effort to reduce the number of design variables, a fixed wing shape was used for all ESAV wings. This effectively reduces the number of design variables by about half because all of the shape handles in VT-CST discussed previously are no longer variable. This leaves the more critical geometric design variables, like span length, fuselage width, and area ruling, free to be used during optimization. Dr. William Mason was consulted to decide upon a good choice for the fixed airfoil sections. From this discussion, and confirmation that previous aircraft using these types of airfoils flew supersonically (e.g. F-16), the NACA 65A(1.5)04 and NACA 65A(1.5)03 airfoils were chosen for the root and tip respectively. Dr. Mason also confirmed that the airfoil choice was a minor consideration for the ESAV multidisciplinary design purposes herein.

3.2.1 Multidisciplinary Design Optimization Problem Set-Up

Based on earlier discussion, the definition of the best ESAV design herein will be the feasible design that has the minimum TOGW. Having made this choice, it should be pointed out that the MDO framework is designed to be flexible and is, therefore, capable of easily working with a different objective function (including a multiobjective function). However, TOGW is deemed the appropriate choice for these initial ESAV design space studies.

For the low-fidelity framework, the chosen design variables are geometry parameters, engine parameters, and mission parameters. In total, there were twenty-six design variables chosen. They are listed in Table 3.2-1 along with the upper and lower bounds that presume to encompass the reasonable design space, that is, the global optimum solution vector, $\bar{x}$, is expected to exist within the design space defined with these side constraints. These upper and lower bounds constitute the side constraints of the optimization problem [77].

Table 3.2-1. Low-Fidelity Design Variables and their Lower and Upper Bound Limits

Design Variable	Units	Lower Bound	Upper Bound
1) Mission Maximum Cruise Mach Number	N/A	2.2	2.5
2) Mission Maximum Cruise Altitude	ft	45000	70000
3) Engine Maximum Cruise Thrust	lbf	7500	35000
4) Bypass Ratio	N/A	0.001	0.5
5) Fan Pressure Ratio	N/A	2.9	3.5
6) Low Pressure Compressor Pressure Ratio	N/A	2.9	3.5
7) High Pressure Compressor Pressure Ratio	N/A	2.9	3.5
8) High Mach Number BPR Cutoff	N/A	1.8	2.5
9) Low Mach Number BPR Cutoff	N/A	0.8	1.6
10) Wing Semi-Span	ft	25	60
11) Wing Root Chord Length	ft	90	130
12) Wing Tip Chord Length	ft	0.05	40
13) Wing Leading Edge Sweep Angle	deg	55	78
14) Wing First-Order Shear (Dihedral)	ft	-1.5	1.5
15) Wing Second-Order Shear	ft	-1.5	1.5
16) Wing Zero-Order Twist	deg	-1.5	1.5
17) Wing First-Order Twist	deg	-1.5	1.5
18) Wing Second-Order Twist	deg	-1.5	1.5
19) Fuselage Length	ft	130	165
20) Fuselage Maximum Width	ft	6	12
21) Fuselage Upper Height	ft	4	9
22) Fuselage Bottom Height	ft	4	9
23) Fuselage Starting X Coordinate	ft	-30	0
24) Fuselage Area Ruling Location	N/A	0	0.7
25) Fuselage Area Ruling Magnitude	N/A	0	0.5
26) Fuselage Area Ruling Width	N/A	0	0.9

Careful attention must be paid to the design variable constraints, since they are somewhat arbitrarily chosen by the designer(s). Although based on physical facts and engineering experience, if the optimal design becomes artificially constrained by any of these variables it should be revisited by the designer(s). However, some of the design variable constraints are not arbitrarily chosen by the designer. For instance, the ESAV Mach number range is a defined range and cannot be changed. The maximum cruise thrust design variable is per engine, and four engines are always assumed for the ESAV to avoid combinatorial optimization complications. If three engines were desired, another optimization can be run with this new baseline design and compared to the four engine version done herein.

3.2.2 Design of Experiments for Low-Fidelity ESAV MDO Framework

The design of experiments (DOE) method offers a powerful tool to perform design space exploration for both sampling (design exploration) and to determine which design variables have the most effect on the objective function [78]. The design variables' importance can only be analyzed in the design space neighborhood around the optimum ESAV configuration. Such knowledge leads to insights about this type of aerospace vehicle and can provide designers and strategists/managers with decision-making data and trends. For instance, the question, “which technologies deserve the most investment for development because they affect the ESAV the most” or “which design variables have the most effect on the aircraft” may have light shed on them with this type of analysis. The default way to explore a design space is to look at every possible point within the space. This, of course, means an effectively infinite number of points, which is unrealistic. Even a discretized space where each of the 26 variables are allowed 5 different values turns into

$$5^{26} = 1,490,116,119,384,765,625 \quad (22)$$

permutations, or this extraordinarily large amount of objective function evaluations using the ESAV MDO analysis framework. This reduced number of runs, from infinite number required to sample every point in the design space, is still not feasible.

The design of experiments field, and more specifically its subset the design of computer experiments, has developed tools to intelligently populate the design space in order to perform meaningful analysis to determine the actual design spaces' behavior, without requiring an infeasible number of runs [78] [79].

Originally, DOE was developed to help plan physical experiments and tests. Distinctive characteristics of computer experiments require a different approach in the design of these experiments [78]. Specifically, a computer experiment typically produces deterministic results (no stochastic analyses in this effort), that is, the same input will always yield the same output. This deterministic output is certainly not true of physical experiments as there are always uncontrolled/unaccounted variables present that fluctuate. Also, the number of design variables is generally greater in computer experiments than in physical experiments. A large number of design variables imposes the challenge of an exploding number of required analysis framework runs by using traditional DOE methods as seen in the equation above. Another issue is that often computational models are very expensive to run. This challenge is the reason for starting at a lower-fidelity level for the discipline models, which lowers the computational expense. In later chapters, higher-fidelity models will be implemented, but always with the goal to control the computational expense so that it is at the minimum required.

Based on these computer experiment characteristics, the desirable properties of the chosen DOE method is that it will be a space-filling design that does not have any repeated points, and keeps the number of required points to a minimum. Space-filling designs are

appropriate for computer experiments because, the resulting points are spread out in some uniform way throughout the entire design space [80]. This allows the study of possible interesting regions throughout the design space, which is crucial when considering an advanced aerospace vehicle concept like the ESAV that does not physically exist yet.

The method known as the “full-factorial method” runs into the problem highlighted by Eq. 22 – exponential growth with design variables. However, intelligent methods can be employed to find a subset of the points that would be part of the full-factorial design by a careful selection of points. The subset is still considered a space-filling design. Since a computer experiment is deterministic, it is highly desirable to have non-replicated points and, in fact, the DOE design method chosen should not have replicated points, even if they were to be projected into lower dimensions [80].

The method known as “the orthogonal array” has been widely used in practice and is endorsed in the literature [79]. The orthogonal array method is a space-filling design and does indeed find a subset of the full-factorial design with no replicated points. This is the primary DOE method chosen for this ESAV study with its ability to fill the design space without replicated points, and at the same time use a small number of sampling points to do this (relative to the full factorial number of required sampling points). The DOE method chosen for this effort also includes Latin Hypercube Sampling (LHS) to add points (on a need-only basis) to reach the number of points specified by the user as the maximum-allowable number of runs.

Finding a global optimum design is so challenging that there has been no general method to do this for nonconvex problems to date. However, a logical methodology can be constructed to give a reasonable expectation that if the optimum found is not a global optimum, it is probably very close to it. One major challenge to finding the global optimum is being able to identify a

smaller region of the design space in which to look for it. A smaller region to search, especially with many local optima, increases the probability that the finite number of sampling points used by an optimizer will find the region (if using a non-gradient based optimizer) where the optimal design resides. DOE can be utilized to help narrow down the search domain by performing a large space-filling design analysis using the low-fidelity MDO framework. The number of runs included in this space-filling design exploration is determined by the computational resource availability. For this effort, this was chosen to be 10,000 low-fidelity runs.

The orthogonal array with LHS DOE space-filling design methodology was utilized to choose which 10,000 design points would be analyzed by the low-fidelity framework. This hybrid approach considers the boundary conditions while also minimizing “clumping” of design points, or in other words, it spaces the sampling locations fairly evenly. For the 10,000 runs, this method uses 9409 runs generated by the orthogonal array supplemented with 591 runs generated by the LHS algorithm. A few sample design space results are shown in Fig. 3.2-1 on a 3D axes plot where each axis is for one unique design variable. Each point represents a complete aircraft design with its own value of TOGW. Only the feasible points were plotted. These particular variables were chosen if they demonstrated any interesting behavior in the feasible design space and otherwise to represent common behaviors. The units were given in Table 3.2-1.

Figure 3.2-1a shows a typical scatter distribution throughout the design space for three variables; the maximum cruise Mach number, the bypass ratio, and the semi-span. The typical scatter fills the entire space with feasible runs. Figure 3.2-1b shows that at lower altitudes, the feasible designs found were sparse. A rough estimate of the lowest maximum cruise altitude where a good scatter of feasible ESAV designs starts is about 55,000 ft. It also shows the relationship between leading edge sweep and the maximum cruise altitude, that is, the lower the

altitude, the higher the sweep angle needed to be and the higher the altitude, the lower the tolerable sweep angle could be. Figure 3.2-1c shows that if the cruise thrust value gets too low, the number of feasible designs is significantly decreased. Figure 3.2-1d shows another typical view across three other design variables, again demonstrating the good space-filling designs obtained by this DOE method.

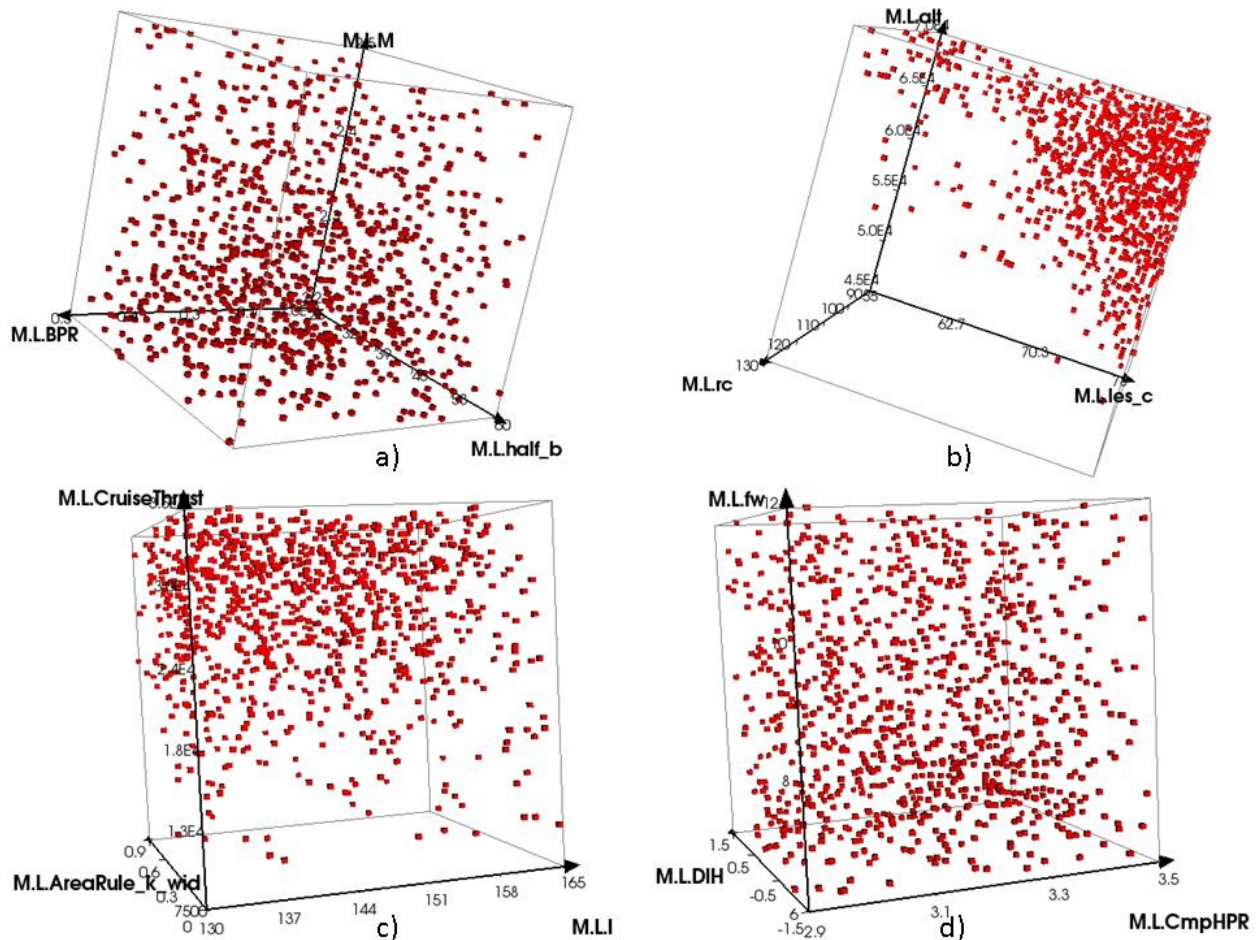


Figure 3.2-1. Twelve representative design variables in four 3D design space scatter plot representations. Plot a) has the cruise Mach number, bypass ratio, and semi-span. Plot b) has the cruise altitude, root chord, and the leading edge sweep angle. Plot c) has the cruise thrust per engine, the area ruling width parameter, and the fuselage length. Plot d) has the fuselage width, the wing dihedral, and the high pressure compressor pressure ratio.

Clearly, the orthogonal array with LHS does a good job of sampling the entire design space without replicating points. All of this data can now be used to identify a smaller set of feasible points where the TOGW is smallest, thus providing a path forward to the goal of reducing the size of the design space. Again, a smaller design space increases the likelihood that an optimizer can converge on the global optimum.

The six best designs resulting from the large DOE were chosen in an effort to find a reduced design space with a high likelihood that the global optimal design is within it. Since a DOE was used to find these six points, there is no reason to think that any of them are stationary points. That is, there is probably a better design (local optimum) in some local neighborhood close to the DOE point. Therefore, a gradient optimization was performed from each of these chosen six points to find their corresponding local optimum. The stationary points are of more interest in the overall objective to bound the reduced design space, since by the necessary conditions, it is known that there exists no point in the neighborhood of these stationary points with a better objective function value. Therefore the bounds can be applied with greater confidence to these points rather than just to the DOE-produced design points.

3.2.3 Optimization for Low-Fidelity ESAV MDO Framework

The strategy used so far for the low-fidelity ESAV MDO framework has been to capitalize on its low computational cost, and initially perform an extensive DOE search of the entire design space. The next step completed was to identify a small set of points from the complete DOE results that have the most desirable TOGW (objective function) values. Since these designs were obtained from a DOE, there is no guarantee that they are optimum, or stationary, points. Therefore, a gradient-based optimization is performed from each of the identified designs to find the corresponding local optimum design.

There are several gradient optimization schemes available. Two gradient methods are readily and conveniently available in ModelCenter™. The first is called Design Explorer, which uses the Sequential Quadratic Programming (SQP) method [17]. The second is the DAKOTA CONMIN algorithm [81].

The CONMIN algorithm finds the minimum of a constrained optimization problem, and the code in which this algorithm has been implemented has been developed to a high level of robustness over the years.

The CONMIN algorithm utilizes the method of feasible directions to perform gradient-based optimization. Inequality constraints are the only type of constraints on the ESAV problem and are handled by this optimizer, but it does struggle with equality constraints [38]. This method excels at finding local minima close to the starting point [38]. It moves from one feasible point to another better feasible point. That is, for a feasible solution, $\vec{x}_i$, a better solution, $\vec{x}_{i+1}$, is found (if not already converged to a stationary point) by moving in the direction $\vec{d}_i$ with a step magnitude determined from the strictly positive scalar λ , or written in one statement,

$$\vec{x}_{i+1} = \vec{x}_i + \lambda \vec{d}_i. \quad (23)$$

The details of the method are documented elsewhere [81]. This method is chosen before Design Explorer because it generally converges much faster, or in fewer objective function evaluations. However, it cannot handle infeasible points (or failed runs within the disciplinary models that reside in the N^2 diagram), and therefore is not used if this situation is encountered.

The Design Explorer method generates a quadratic model of the response surface using data from design points within the defined design space. It then uses the SQP algorithm to optimize the quadratic response surface model. It next chooses new points to explore, and the data gathered from these points is used to refine the quadratic approximation. This continues

until convergence to a stationary point has been obtained. Due to the need to build and refine a response surface approximation, this method generally takes many more objective function evaluations than a more direct gradient optimization technique, like CONMIN. Conversely, this robust method can handle failed runs and still continue. Therefore, it is used if the faster CONMIN method encounters a failed run, and therefore cannot continue to completion.

An example of the Design Explorer convergence from a starting point to a local TOGW minimum point can be seen in Fig. 3.2-2. This can be compared to the example of the CONMIN convergence from a starting point to a local TOGW minimum point in Fig. 3.2-3. Note the fewer runs required for convergence and different convergence paths for the CONMIN algorithm compared to Design Explorer's SQP algorithm.

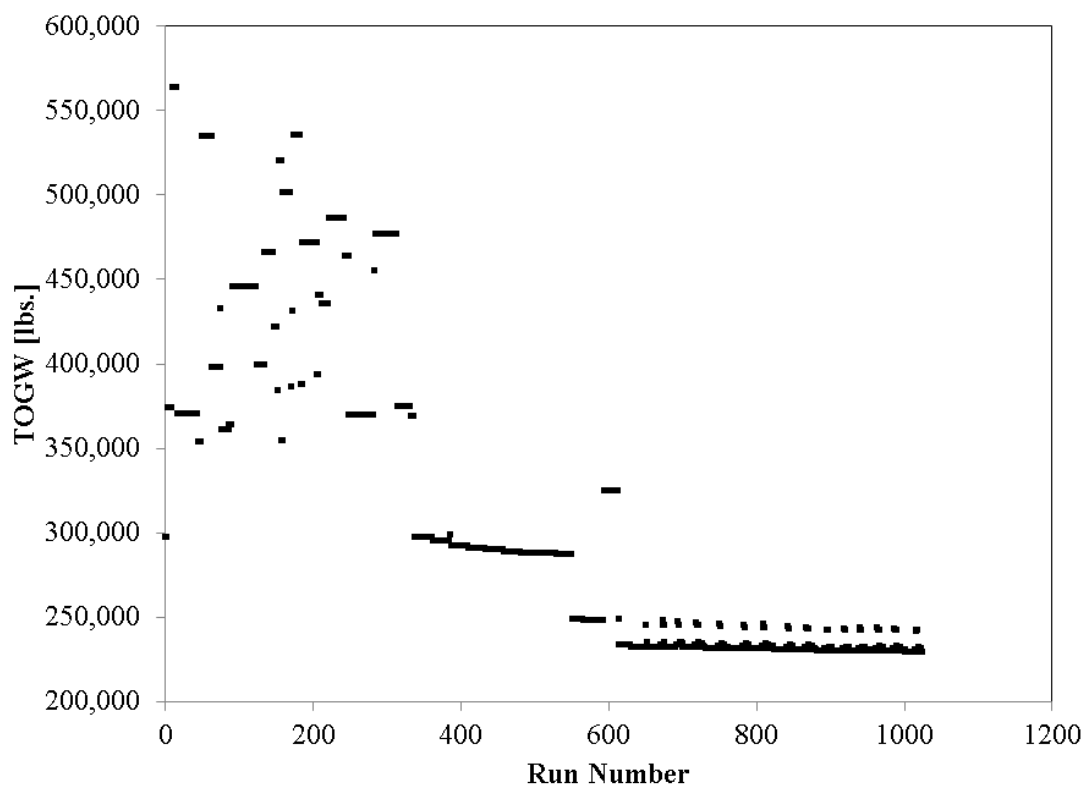


Figure 3.2-2. Example Design Explorer TOGW convergence to a local stationary point.

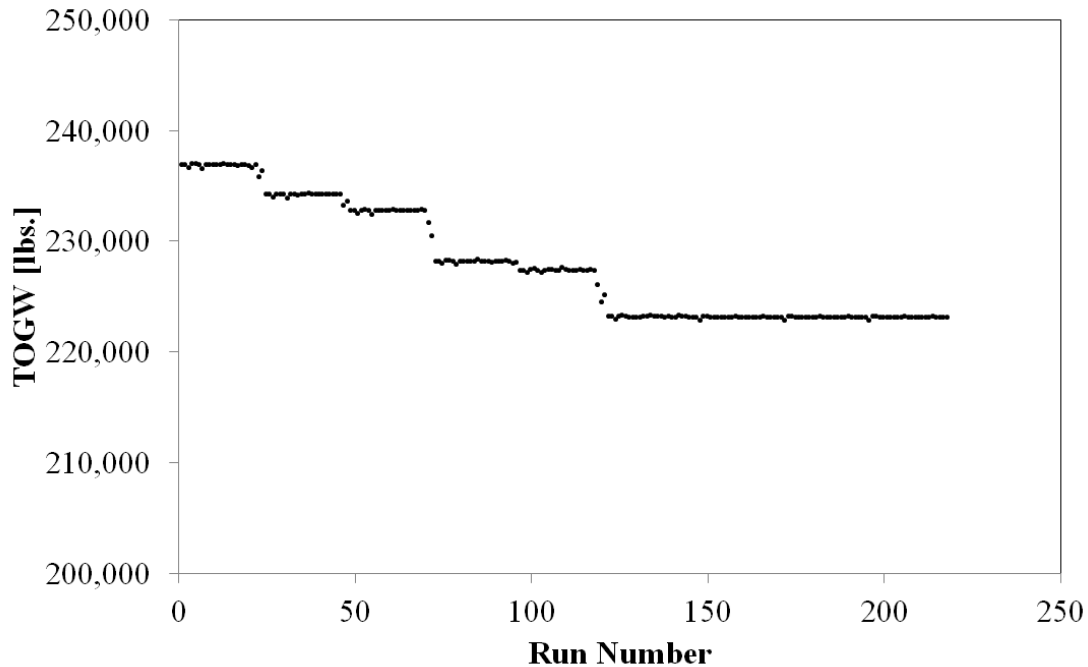


Figure 3.2-3. Example CONMIN TOGW convergence to a local stationary point.

The identified six design points from the 10,000 run DOE, and their corresponding stationary points from the gradient optimizations are shown in Fig. 3.2-4. They are shown in pairs (left is DOE design and right is stationary point after gradient optimization) in order of the DOE designs increasing TOGW from top to bottom. The only stationary point that could be found using CONMIN was configuration five (Fig. 3.2-4 e). The other configurations required Design Explorer with SQP to find their corresponding stationary points because of failed runs. The CONMIN algorithm did not move very far from the starting point to the local minimum, so the stationary point planform configuration is very similar to the starting configuration. On the other hand, the other configurations stationary points generally had significant changes from the DOE design – both in planform and objective function value.

If these six starting points happened to all converge to the same design after the gradient optimization, then either more points could have been chosen from the DOE results to obtain a better spread of local optimum points with which to bound the reduced design space, or the

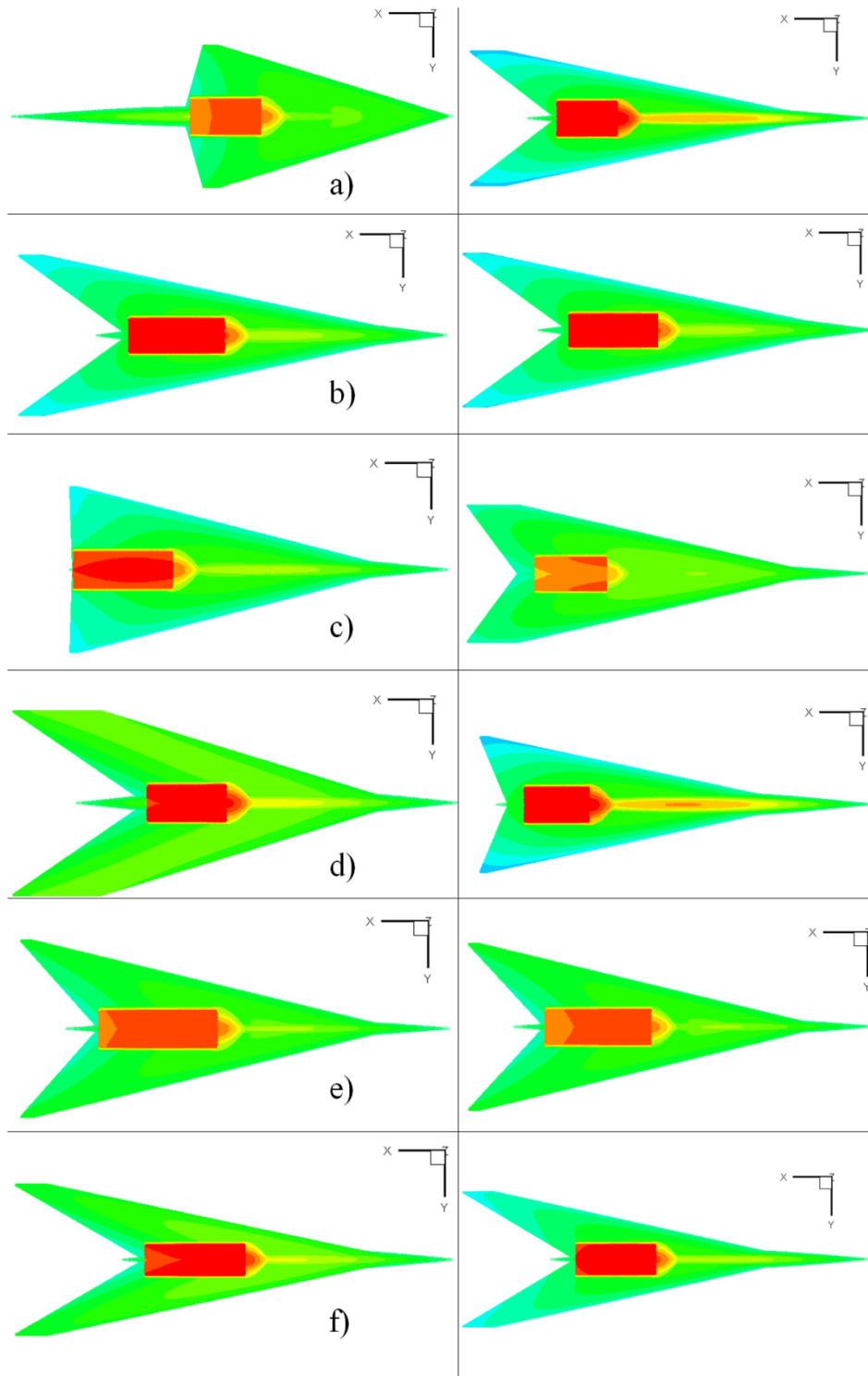


Figure 3.2-4. Planform shapes of identified six design points from the DOE, and their corresponding stationary points. They are in pairs (left is DOE design and right is stationary point design) in order of DOE designs increasing TOGW from top to bottom.

single converged point could have been chosen as the best guess for the global optimum with no further steps. The final designs to be used to reduce the design space are these local optimum stationary points associated with these six configurations.

Finding the local optimum points strives to prevent the situation that the bounds on the reduced design space will exclude the global optimum if a chosen DOE point happens to block the downward slope to the global optimum. Due to the fact that these are the best designs that resulted from the large DOE sampling, it is a good indication (although without a guarantee) that the global optimum will lie within the bounds set by these stationary points. The minimum and maximum values of each design variable across these best designs, which defines the boundaries, was then found and used to restrict the design space as shown in Table 3.2-2. Although rounded to the nearest hundredths place for ease of reading in the table, they are implemented to a much higher significant figure in the actual problem. The upper and lower bounds from the best designs define the new side constraints on the optimization problem.

After the design space has been reduced using the best designs found in the DOE, and subsequent gradient optimization starting at each of these points, it is conjectured that the likelihood that the global optimum can be found within this reduced design space is increased. Some of the design space limits are the same, or appear to be the same to the nearest hundredths place, as the original limits. If the eventual optimum is found to also be on a boundary that is not externally imposed (e.g. the Mach number limits), then they will need to be expanded. If, however, the eventual optimum point does not have an active side constraint, then these bounds can be considered acceptable. Therefore, the next step is to run an optimization within the reduced design space bounds as indicated in Table 3.2-2.

Table 3.2-2. Low-Fidelity Design Variables with Limited Lower and Upper Bounds**Rounded to Nearest Hundredths Place**

Design Variable	Units	Lower Bound	Upper Bound
1) Mission Maximum Cruise Mach Number	N/A	2.31	2.5
2) Mission Maximum Cruise Altitude	ft	59892.58	67753.91
3) Engine Maximum Cruise Thrust	lbf	9702.148	28697.92
4) Bypass Ratio	N/A	0.14	0.5
5) Fan Pressure Ratio	N/A	3.00	3.42
6) Low Pressure Compressor Pressure Ratio	N/A	3.04	3.49
7) High Pressure Compressor Pressure Ratio	N/A	2.9	3.29
8) High Mach Number BPR Cutoff	N/A	1.88	2.27
9) Low Mach Number BPR Cutoff	N/A	1.17	1.6
10) Wing Semi-Span	ft	25.21	30.46
11) Wing Root Chord Length	ft	94.30	113.91
12) Wing Tip Chord Length	ft	2.00	19.089
13) Wing Leading Edge Sweep Angle	deg	76.43	78
14) Wing First-Order Shear (Dihedral)	ft	-1.07	0.08
15) Wing Second-Order Shear	ft	-1.09	0.97
16) Wing Zero-Order Twist	deg	-1.22	1.01
17) Wing First-Order Twist	deg	-1.43	1.02
18) Wing Second-Order Twist	deg	-1.24	0.83
19) Fuselage Length	ft	130.0	139.78
20) Fuselage Maximum Width	ft	6.36	11.31
21) Fuselage Upper Height	ft	4.02	7.08
22) Fuselage Bottom Height	ft	4.0	7.30
23) Fuselage Starting X Coordinate	ft	-29.00	-10.32
24) Fuselage Area Ruling Location	N/A	0.29	0.69
25) Fuselage Area Ruling Magnitude	N/A	0.01	0.40
26) Fuselage Area Ruling Width	N/A	0.42	0.82

The DAKOTA Coliny DIRECT method was one of the two optimization algorithms chosen here based on its performance in the benchmark Bird Problem. However, a major drawback with this method is that it cannot handle objective function evaluation failures. That is, if an infeasible run occurs and/or an analysis model/program does not complete as expected (a failed run), the optimization algorithm cannot continue and stops the whole ModelCenter™ optimization process at that point. CONMIN has similar behavior.

The DAKOTA Coliny DIRECT method is derivative-free and works by the Diving RECTangles (DIRECT) algorithm wherein local searches are performed in more promising regions while simultaneously searching in under-explored regions to find more possible promising regions. Constraint violations are handled with a fixed penalty factor. The design space is split into rectangular regions and, a point is chosen within each rectangle for objective function analysis, which also determines if that area is one of interest or not. Any area found to be of interest (by the measure of the objective function) is further subdivided into smaller regions over and over, thus allowing the method to drill down to a possible global optimum point. Several convergence tests are performed including maximum number of iterations, smallest maximum box (rectangle) size limit, solution target, and smallest minimum box size limit. The settings for the DAKOTA Coliny DIRECT algorithm are shown in Fig. 3.2-5.

The design space of the ESAV is disjoint. Therefore, even in the reduced design space bounds, there are still possible infeasible combinations of the design variables that can be encountered. Even with the evaluation failure-means-crash limitation of the Coliny DIRECT method, this algorithm was still attempted. Unfortunately, for this case, an infeasible result was encountered on the second run, prematurely stopping the optimization algorithm. This method

may be tried for other cases. Actually, DAKOTA Colony DIRECT was found for another similar MDO framework to obtain good optimization results.

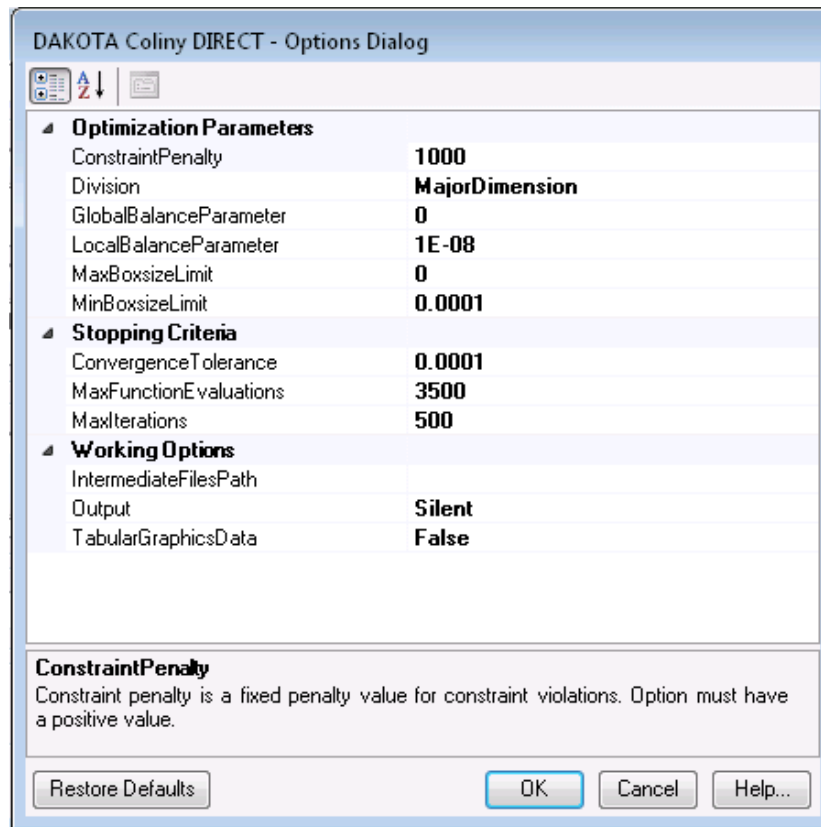


Figure 3.2-5. DAKOTA Colony DIRECT algorithm options for low-fidelity optimization.

The NSGA-II optimization algorithm was also identified by its performance on the Bird Problem as a very promising ESAV optimization algorithm. This algorithm does not suffer from the evaluation failure problem like the Colony DIRECT algorithm, therefore, it was chosen to run within the limited design space boundaries (side constraints) using the stationary point with the lowest TOGW as the starting point. The starting point is important because it becomes the first member of the first randomly-selected initial generation and is compared to ensuing generations' individuals. The reduced design space limits were shown in Table 3.2-2.

The NSGA-II algorithm is a genetic algorithm (GA) but works differently from other GA algorithms. The NSGA-II algorithm's identifying features are its fast nondominated sorting

technique, elitist preserving strategy, and a parameterless niching operator [82] (method used to preserve design diversity/spacing). It handles constraints with the constrained dominance approach [83]. For the ESAV, there is only a single objective function rather than a multiobjective function, although NSGA-II can be used for either case.

When a multiobjective objective function is used as opposed to a single objective function, an optimal value is not a simple minimum or maximum, but is a Pareto optimal point. A succinct way to describe a Pareto optimal point is that improving on one of the objectives in the multiobjective function cannot be achieved without being detrimental to another objective. A Pareto front is the group of Pareto optimal points found by the optimizer.

For a single objective function, which is the case for the ESAV problem, the non-dominated sorting technique means that the individuals in the generation are sorted from best to worst based on the objective function and the next Pareto front is just the next best design. The diversity preservation technique permits a good spread of designs in the design space to be sampled while the nondominated sorting technique lets the next generations focus on the more optimal areas of the design space.

The algorithm initially generates a random population for the first generation (except for the one user-specified starting point individual), and then sorts the individuals by the non-domination technique. Binary tournament selection is used for selecting the parents that will produce the next generation. This particular method to approach the selection of the parents for crossover works by first randomly selecting two parents from the same generation. Then, the dominated parent is discarded, while the other is used for crossover. Once both parents are chosen the crossover (recombination) and mutation operators are applied to produce the second generation. The default ModelCenter™ settings for this algorithm are shown in Fig. 3.2-6.

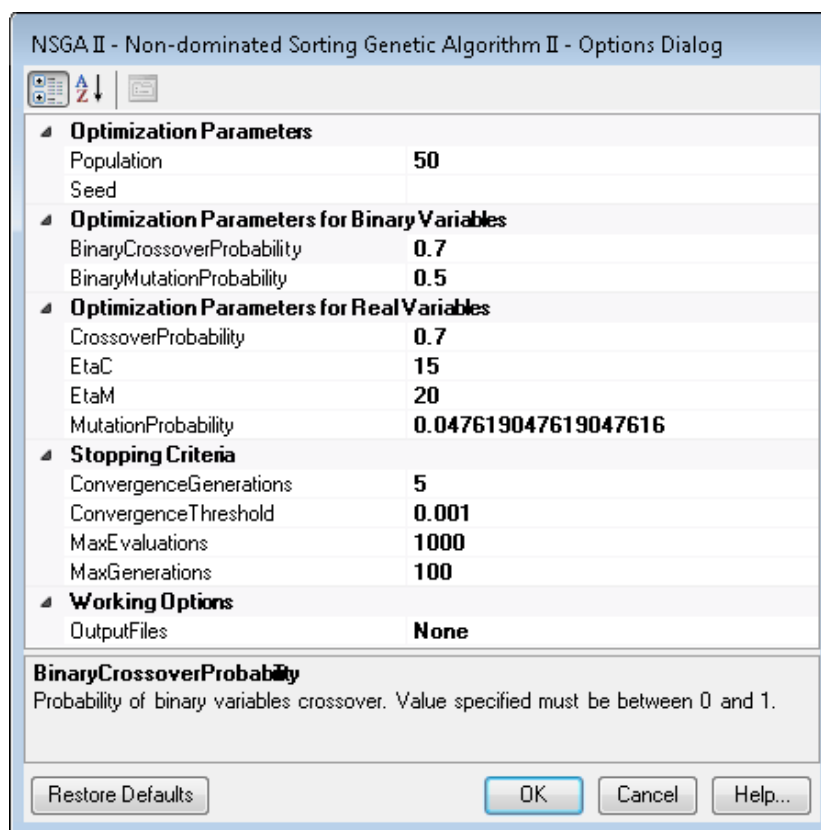


Figure 3.2-6. NSGA-II algorithm options for low-fidelity optimization.

The succeeding generations, after these initial two generations, follow a similar pattern, but with the important elitism strategy included. Elitism in NSGA-II ensures that only the best designs are kept by comparing any new designs in the latest generation to all the best designs found before that generation. NSGA-II includes a crowding distance check to make sure points retain some variety.

Using the smallest TOGW configuration from the large DOE results as a starting point, the NSGA-II algorithm was able to converge (with some infeasible runs) after 861 individuals. The convergence history is shown in Fig. 3.2-7, and the configuration with the lowest TOGW is shown in Fig. 3.2-8. Note the reduced amount of scatter in the last half of the convergence history plot. This is due to the NSGA-II algorithm producing only very similar individuals from elite parents shortly before convergence.

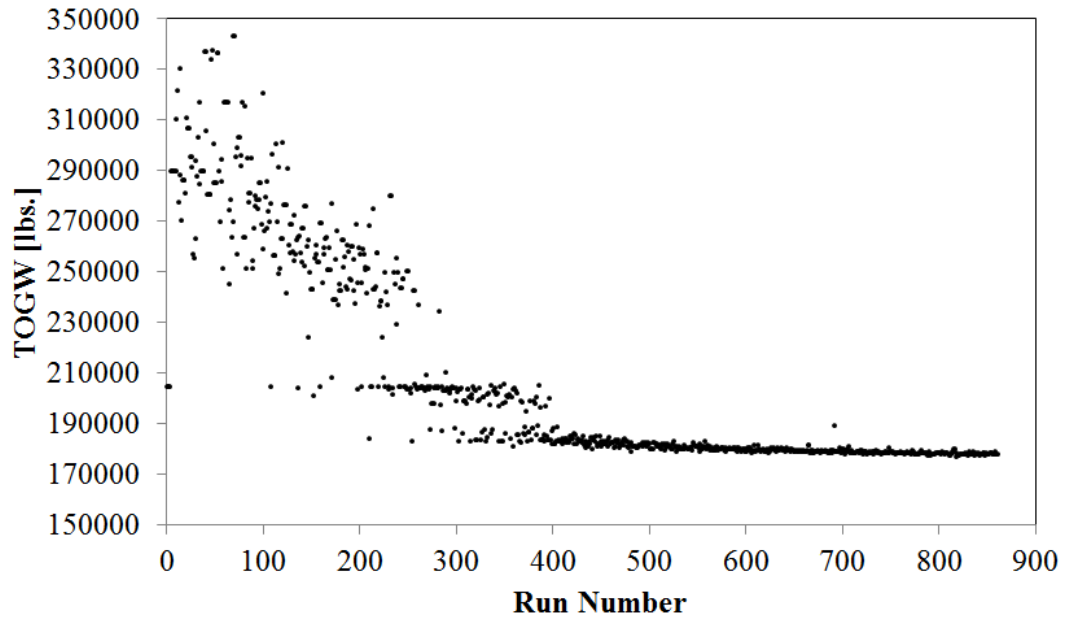


Figure 3.2-7. NSGA-II TOGW convergence history over the run number.

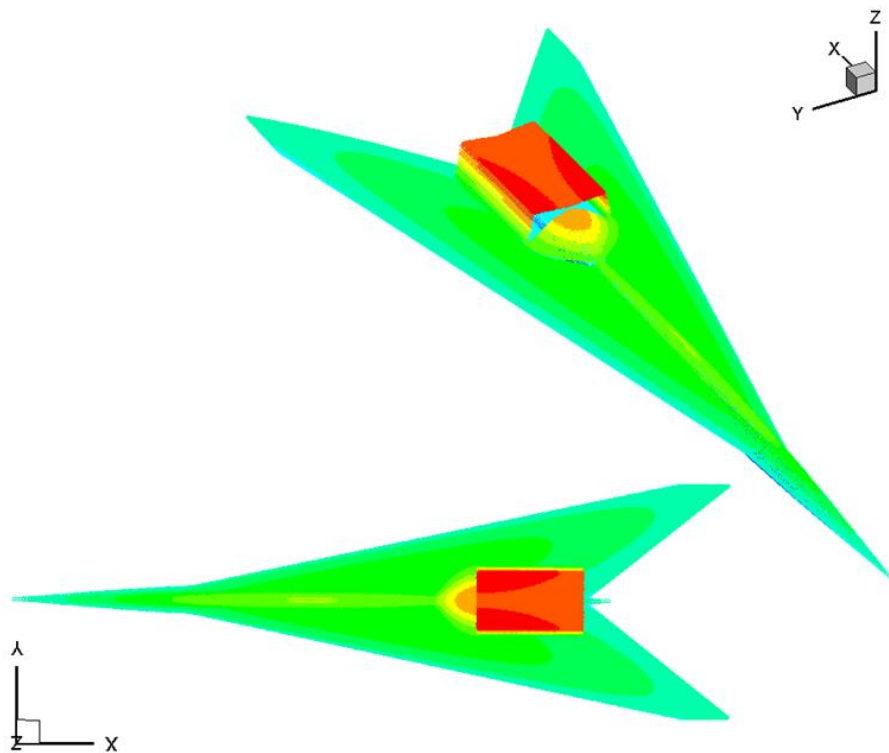


Figure 3.2-8. Best ESAV configuration using NSGA-II algorithm for low-fidelity MDO.

This optimum result was found using the best configuration from the DOE (after the gradient optimizations) as a starting point, and it looks very similar to that configuration (see Fig.

3.2-4a on the right side). To test the dependence on the final result on the starting point using the NSGA-II optimization technique, the fourth best design from the DOE (see Fig. 3.2-4d on the right side) was used as the starting point in another optimization run. This configuration was chosen because it looks different from the best DOE design (see Fig. 3.2-4a) used previously as the starting point.

The resulting lowest TOGW configuration with the fourth best starting point found after 900 individuals is shown in Fig. 3.2-9. This configuration is very similar to the optimal solution found using the best design as the starting point (see Fig. 3.2-8). Table 3.2-3 has the actual values for the final design variables, rounded to (at most) the nearest thousandths place, of these two NSGA-II optimizations. No side constraints were active in either of the designs although it may look like it with the rounded values in Table 3.2-3.

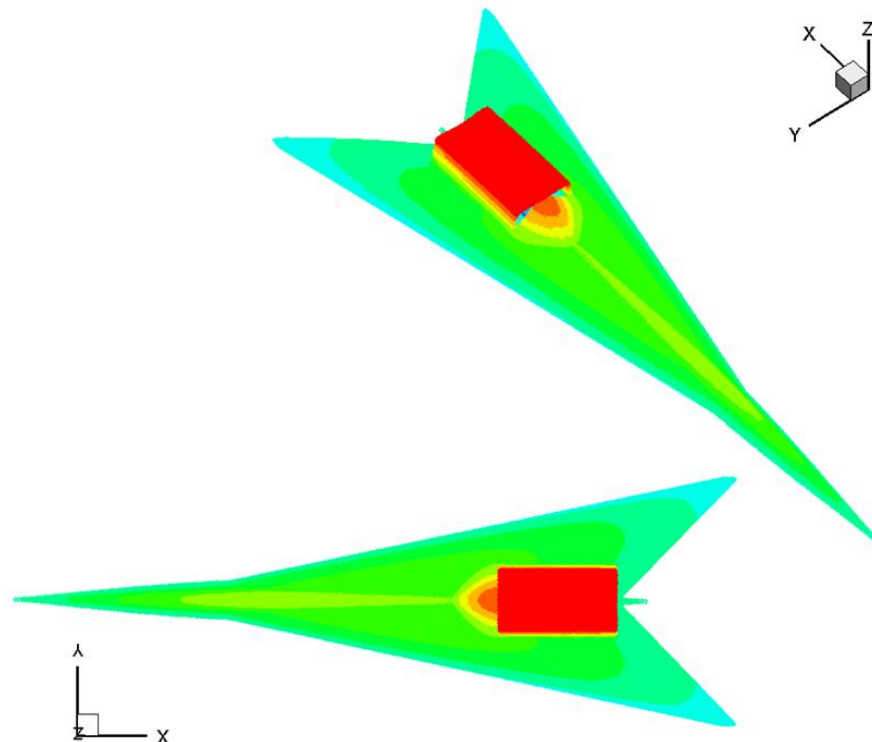


Figure 3.2-9. Optimum configuration using NSGA-II algorithm for low-fidelity ESAV using the fourth best DOE configuration result as the starting point.

Table 3.2-3. NSGA-II Multidisciplinary Design Optimization Results from First Design Starting Point (NSGA-II 1st), and from Fourth Design Starting Point (NSGA-II 4th)

Design Variable	Units	NSGA-II (1st)	NSGA-II (4th)
1) Mission Maximum Cruise Mach Number	N/A	2.498	2.470
2) Mission Maximum Cruise Altitude	ft	67,070	67,743
3) Engine Maximum Cruise Thrust	lbf	9,702	10,312
4) Bypass Ratio	N/A	0.500	0.490
5) Fan Pressure Ratio	N/A	3.234	3.414
6) Low Pressure Compressor Pressure Ratio	N/A	3.421	3.371
7) High Pressure Compressor Pressure Ratio	N/A	2.969	3.013
8) High Mach Number BPR Cutoff	N/A	1.927	1.968
9) Low Mach Number BPR Cutoff	N/A	1.473	1.556
10) Wing Semi-Span	ft	25.355	25.390
11) Wing Root Chord Length	ft	97.993	96.953
12) Wing Tip Chord Length	ft	10.038	2.167
13) Wing Leading Edge Sweep Angle	deg	78.000	77.934
14) Wing First-Order Shear (Dihedral)	ft	0.002	-0.220
15) Wing Second-Order Shear	ft	-0.482	-0.861
16) Wing Zero-Order Twist	deg	0.303	0.069
17) Wing First-Order Twist	deg	-0.862	-0.964
18) Wing Second-Order Twist	deg	-1.085	-0.459
19) Fuselage Length	ft	130.000	130.251
20) Fuselage Maximum Width	ft	6.709	8.438
21) Fuselage Upper Height	ft	4.043	4.100
22) Fuselage Bottom Height	ft	4.000	4.115
23) Fuselage Starting X Coordinate	ft	-26.107	-27.055
24) Fuselage Area Ruling Location	N/A	0.342	0.339
25) Fuselage Area Ruling Magnitude	N/A	0.010	0.322
26) Fuselage Area Ruling Width	N/A	0.793	0.505
TOGW	lbs.	176,869	177,010

Most of the design variables between the two optimal points were very similar. The variables that had the greatest disparity were the thrust, tip chord length, and wing shape. There were also some differences in the area ruling gains. The percent difference between the two objective functions, TOGW, is a mere 0.08%, even with dissimilar starting ESAV designs. The NSGA-II algorithm, in this reduced design space, is therefore capable of finding a location in the design space where the TOGW is small, regardless of the starting point. This cannot be offered as a general conclusion, but as an observation of the identified low-fidelity reduced design space.

The two NSGA-II results are significantly better than their respective starting points found from the DOE and subsequent gradient optimizations. The best DOE stationary point used to define the reduced design space boundaries had a TOGW of 204,135 lbs. and the fourth best had a TOGW of 228,760 lbs. Thus, using the NSGA-II optimization gives a 13.4% and 22.6% improvement over the original DOE stationary points used as starting points.

Both resulting designs from the two NSGA-II optimizations have almost an identical TOGW. This result gives confidence in the ability of the NSGA-II algorithm to find a global optimum, or at least a design very close to it (with respect to the objective function). However, there is no guarantee that the NSGA-II has found a stationary point, since individual designs are selected based on initial randomness, then recombination by previously randomly chosen parents. That is, no KKT conditions are checked at the NSGA-II identified points. A Design Explorer SQP optimization was therefore performed starting from each of the designs identified by NSGA-II. This was done within the identified reduced design space as shown in Table 3.2-2.

Both SQP optimizations from both starting point designs do indeed find better points in the local vicinity of the NSGA-II identified optimum value, with a relatively large improvement in the TOGW. That is, the TOGW was reduced from 176,869 lbs. to 170,198 lbs. (3.8%

reduction), and from 177,010 lbs. to 169,763 lbs. (4.1% reduction) for the first and fourth best starting designs respectively. Clearly, there is value in performing an additional gradient optimization after a NSGA-II (genetic algorithm) optimization, if the computational expense can be afforded.

The design variable values for these two configurations are given in Table 3.2-4, and their corresponding ESAV configurations are both shown in Fig. 2.3-10. These two designs are almost identical, with the largest difference being in the area ruling gains with other small discrepancies in the variable cycle bypass configuration and fuselage size. The percent difference between these two configurations in the TOGW objective function is only 0.26%.

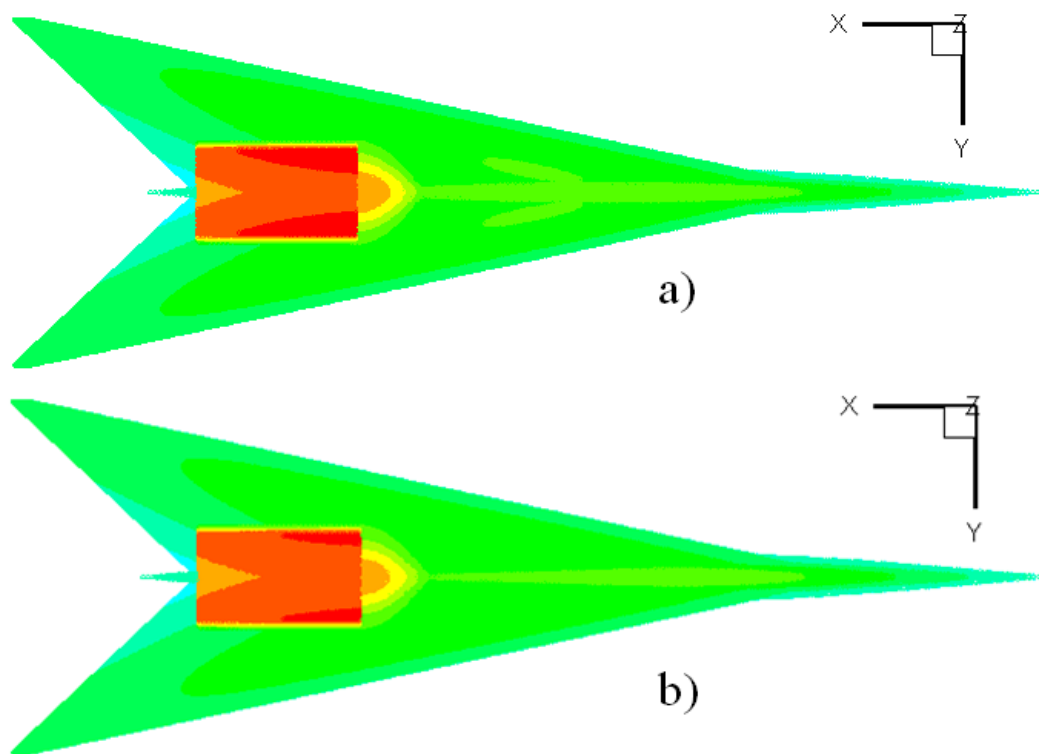


Figure 3.2-10. Two final optimal designs found starting from the two configurations (top was best design and bottom was fourth best design) from NSGA-II optimizations, followed by a sequential quadratic programming gradient optimization.

Table 3.2-4. Final Low-Fidelity Optimal Rounded Design Variables after SQP**Optimization with the Final Optimal Design Highlighted by the Box**

Design Variable	Units	NSGA-II (1st)	NSGA-II (4th)
1) Mission Maximum Cruise Mach Number	N/A	2.50	2.50
2) Mission Maximum Cruise Altitude	ft	67,754	67,754
3) Engine Maximum Cruise Thrust	lbf	9,702	9,702
4) Bypass Ratio	N/A	0.45	0.5
5) Fan Pressure Ratio	N/A	3.42	3.41
6) Low Pressure Compressor Pressure Ratio	N/A	3.49	3.49
7) High Pressure Compressor Pressure Ratio	N/A	3.12	3.29
8) High Mach Number BPR Cutoff	N/A	1.88	2.14
9) Low Mach Number BPR Cutoff	N/A	1.60	1.60
10) Wing Semi-Span	ft	25.21	25.21
11) Wing Root Chord Length	ft	94.30	94.37
12) Wing Tip Chord Length	ft	2.00	2.00
13) Wing Leading Edge Sweep Angle	deg	78.00	78.00
14) Wing First-Order Shear (Dihedral)	ft	0.08	0.08
15) Wing Second-Order Shear	ft	0.16	0.24
16) Wing Zero-Order Twist	deg	1.01	0.80
17) Wing First-Order Twist	deg	-1.43	-1.43
18) Wing Second-Order Twist	deg	-1.24	-1.04
19) Fuselage Length	ft	130.00	130.00
20) Fuselage Maximum Width	ft	6.90	7.16
21) Fuselage Upper Height	ft	4.02	4.02
22) Fuselage Bottom Height	ft	4.00	4.41
23) Fuselage Starting X Coordinate	ft	-28.64	-27.54
24) Fuselage Area Ruling Location	N/A	0.29	0.33
25) Fuselage Area Ruling Magnitude	N/A	0.06	0.34
26) Fuselage Area Ruling Width	N/A	0.82	0.51
TOGW	lbs.	170,198	169,763

The final low-fidelity optimal ESAV design (NSGA-II 4th with box around it in Table 3.2-4) has few surprises in the final design variable values. However, two seemingly peculiar results were that the mission was flown at the highest cruise Mach number allowed, and the engine had a high bypass ratio. From engineering experience, the lowest supersonic Mach number might be expected to be the most efficient choice, but a higher Mach number has two virtues. The first is that the mission takes less time to complete (engine is being used less), and the second is that the faster the ESAV flies, the better its survivability becomes. The supersonic regime at $M = 2.5$ is usually dominated by turbojets, so the high bypass ratio on the optimal low-fidelity ESAV configuration may seem atypical. However, this is a variable cycle engine, so it is actually operating like a turbojet at $M = 2.5$, but is allowed the higher bypass ratio at lower Mach numbers. The high bypass ratio is therefore reasonable when this variable cycle viewpoint is remembered.

The geometrical wing shape has a slight positive dihedral and a negative higher-order twist. This can be seen in Fig. 3.2-11b and 3.2-11e as the wing is curved – becoming more pronounced the closer it is to the wing tip. This wing configuration, produced with twist and shear, is similar to conical camber. The fuselage is just over seven feet wide and is over eight feet tall, but this is difficult to distinguish in the plane drawings because the fuselage is 130 ft. long and the wing span is 50.42 ft. wide. The fuselage does have area-ruling applied through its three gain design variables, which can be somewhat noticed, especially in Fig. 3.2-11e.

The clipped arrow wing is found to be the optimum planform shape using the low-fidelity ESAV MDO framework as shown in Fig. 3.2-11a. This is a known aerodynamically efficient supersonic wing design, but a hidden problem is also known to occur with this type of wing.

Chacksfield et. al. report that at low and high incidences the center of pressure shifts causing instabilities and pitch nonlinearities [84]. One suggestion to alleviate this problem is the addition of a lower sweep angle section at the tip of the wing – the so-called cranked wing.

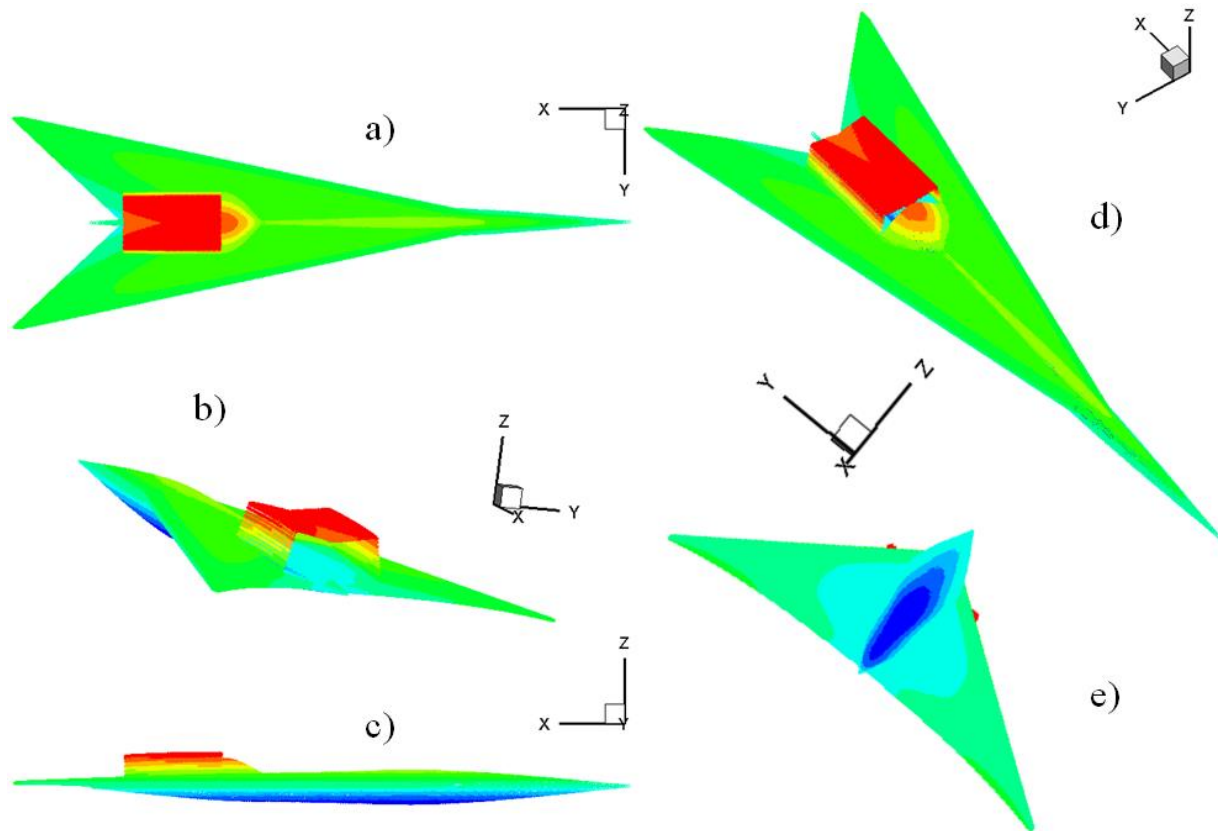


Figure 3.2-11. Different views of the optimal low-fidelity configuration ESAV from performing the NSGA-II optimization followed by a gradient optimization in the reduced design space starting at the fourth best DOE ESAV design.

A cranked wing reduces the aerodynamic efficiency, but also can ease the low-speed stability and control issue. The low-fidelity framework does not have a stability and control module and is not suitable for performing this design trade-off study. Hence, the unmodified clipped arrow wing is found by the optimizer (uninhibited by a stability and control analysis) as the optimal design.

The noise module in the low-fidelity framework calculates that the maximum combined noise is within the FAR 36 noise limits. The flyover noise is significantly lower (by almost 29 db); however, the sideline noise exceeds the limit by about 7 db. The shielding offered by the over-wing engine location and arrow wing shape, the variable cycle engine which has a bypass flow at low speeds, and the optimal (small) engine size contribute to these noise results.

The optimal low-fidelity ESAV framework design point was identified with a large amount of effort, starting with the 10,000 run DOE. During this process, it was seen that the NSGA-II optimization algorithm was able to consistently find a good design, even given different starting configurations. A test was next performed to see if the expensive DOE step was necessary, or if a straight optimization using NSGA-II in the original (non-reduced) boundaries is all that is required. In order to perform this test, engineering judgment based on some simple analysis and experience was used to choose a starting configuration. The chosen design variable values are shown in Table 3.2-5. The corresponding configuration was drawn with the VT-CST drawing functions and is shown in Figure 3.2-12.

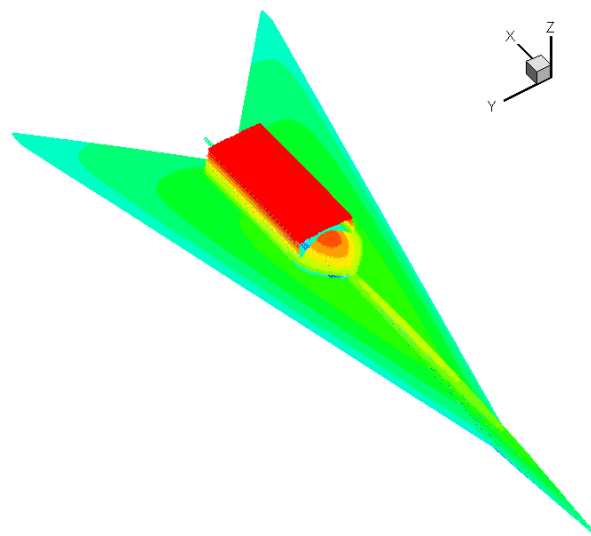


Figure 3.2-12. ESAV design for full design space NSGA-II starting point based on engineering judgment using some simple analysis and experience.

Table 3.2-5. ESAV Full Design Space NSGA-II Test Starting Point Design Variable Values

Design Variable	Units	Starting Point Design Variables
1) Mission Maximum Cruise Mach Number	N/A	2.4
2) Mission Maximum Cruise Altitude	ft	63,000
3) Engine Maximum Cruise Thrust	lbf	14,000
4) Bypass Ratio	N/A	0.3
5) Fan Pressure Ratio	N/A	3.0
6) Low Pressure Compressor Pressure Ratio	N/A	3.4
7) High Pressure Compressor Pressure Ratio	N/A	3.2
8) High Mach Number BPR Cutoff	N/A	2.4
9) Low Mach Number BPR Cutoff	N/A	1.2
10) Wing Semi-Span	ft	30
11) Wing Root Chord Length	ft	100
12) Wing Tip Chord Length	ft	3.0
13) Wing Leading Edge Sweep Angle	deg	77.0
14) Wing First-Order Shear (Dihedral)	ft	0.0
15) Wing Second-Order Shear	ft	0.0
16) Wing Zero-Order Twist	deg	-1.0
17) Wing First-Order Twist	deg	-0.5
18) Wing Second-Order Twist	deg	0.0
19) Fuselage Length	ft	135
20) Fuselage Maximum Width	ft	6.0
21) Fuselage Upper Height	ft	4.5
22) Fuselage Bottom Height	ft	5.0
23) Fuselage Starting X Coordinate	ft	-25.0
24) Fuselage Area Ruling Location	N/A	0.4
25) Fuselage Area Ruling Magnitude	N/A	0.2
26) Fuselage Area Ruling Width	N/A	0.75

Although this design based on engineering judgment seems quite reasonable, and turns out to be feasible, its TOGW is 240,214 lbs.; significantly heavier than the optimum 169,763 lbs. found previously by the DOE design space search, design space reduction, and subsequent optimization process. The NSGA-II algorithm for this optimization process check case was run within the full design space boundaries defined previously in Table 3.2-1 and using the same NSGA-II parameters as shown in Fig. 3.2-6.

The number of individual NSGA-II objective function evaluations required to find this optimum was 898. The first couple of generations had many infeasible points, but after these generations almost all points were feasible. The optimal design had a TOGW of 206,136 lbs., or 21.4% heavier than the optimum found with the much more intensive reduced design space DOE process. However, there is a substantial improvement over the initial engineering judgment ESAV configuration guess shown in Fig. 3.2-12. The TOGW was reduced from this design by 34,078 lbs. or 14.2%.

There is still much room for improvement to reach closer to the optimal DOE method design. Continuing the optimization process with another NSGA-II optimization will still be about an order-of-magnitude less runs than that required by the DOE method. Therefore, taking this direct approach one step further, the optimal design from this first NSGA-II optimization was used as the starting point of another NSGA-II run in the same original design space.

The original design space was used, because the best designs from the first NSGA-II optimization are nearly identical due to the NSGA-II algorithm details, therefore these design points make borders of a design space that is too small to justify looking for a global optimum. This is to be expected, because this algorithm tends to converge on the most interesting location within the design space in the later generations. By the same token, using the larger original

design space is therefore hoped to not be a very large impediment to finding the minimum TOGW ESAV design.

The optimal ESAV design after running the second NSGA-II optimization has a TOGW of 186,838 lbs. This is an improvement over the previous NSGA-II optimization design's TOGW of 9.4%. This second NSGA-II optimization, using the first optimum as the starting point, converged after 899 individual runs.

Comparing this result to the DOE method shows that the successive NSGA-II optimization method does very well considering that it requires about an order-of-magnitude fewer objective function evaluations. For instance, after the NSGA-II optimization in the reduced design space in the DOE method, the TOGW was found to be 177,010 lbs. The result from the successive NSGA-II optimization was only 5.6% away from this result.

The successive NSGA-II optimization method is fairly sensitive to the initial guess, because it can often focus the algorithm near that area of the design space if the first few random initial generations of other designs do not produce very good individuals based on the objective function. If a different initial design had been chosen, the successive optimization method could have been in a similar area of the design space as was identified by the DOE method. In fact, in other tests of a slightly different framework, the successive NSGA-II optimization method found an optimal aircraft configuration that was better than that found using the DOE method.

These results clearly point to the effectiveness of the NSGA-II optimization algorithm for this ESAV design problem. In general, if the computational resources of an optimization problem allow the more methodical DOE method, and its much higher computational cost, it should be undertaken. However, if computational resources are very limited, then the successive NSGA-II method can find a very good design at a cost an order-of-magnitude less than the DOE

method. This conclusion is valid for the ESAV minimum TOGW optimization problem and may not be applicable to any optimization problem. The design variables of the final ESAV design found using the successive NSGA-II optimization method are in Table 3.2-6. The corresponding planform is shown in Fig. 3.2-13. This planform has both a back view to show the wing curvature and a full-planform view. It is similar to both the starting guess (as expected) shown before in Fig. 3.2-12, and the DOE method optimum found in Fig. 3.2-11.

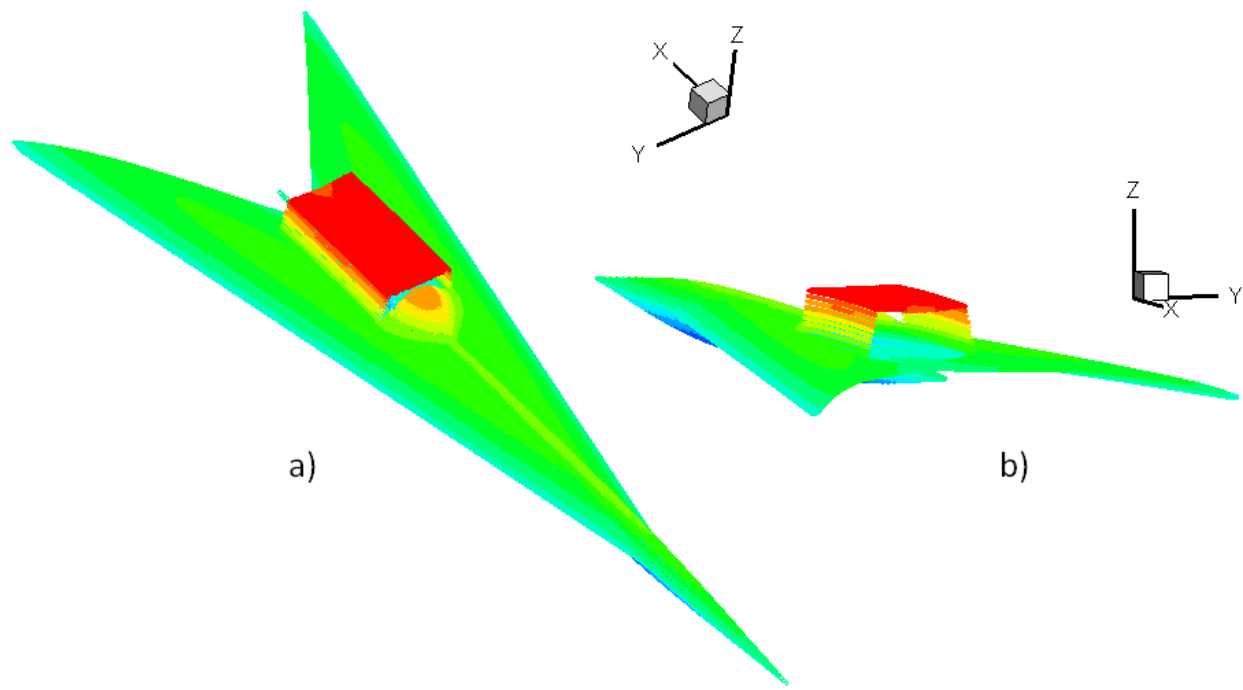


Figure 3.2-13. Optimal ESAV planform configuration found using the successive NSGA-II optimization method using the full design space.

The design variables with the greatest difference between the DOE method optimal and the successive NSGA-II optimal were mostly in the engine parameters, the wing shape, and the area ruling shaping values. The Mach number was higher for the DOE method, but the cruise altitudes were only about 900 feet different.

Table 3.2-6. Low-Fidelity Successive NSGA-II Method Optimal ESAV Design Variables

Design Variable	Units	Successive ESAV Design
1) Mission Maximum Cruise Mach Number	N/A	2.4
2) Mission Maximum Cruise Altitude	ft	66,838
3) Engine Maximum Cruise Thrust	lbf	11,352
4) Bypass Ratio	N/A	0.42
5) Fan Pressure Ratio	N/A	3.37
6) Low Pressure Compressor Pressure Ratio	N/A	3.45
7) High Pressure Compressor Pressure Ratio	N/A	3.20
8) High Mach Number BPR Cutoff	N/A	2.21
9) Low Mach Number BPR Cutoff	N/A	1.22
10) Wing Semi-Span	ft	28.67
11) Wing Root Chord Length	ft	96.87
12) Wing Tip Chord Length	ft	1.79
13) Wing Leading Edge Sweep Angle	deg	77.99
14) Wing First-Order Shear (Dihedral)	ft	0.11
15) Wing Second-Order Shear	ft	-0.09
16) Wing Zero-Order Twist	deg	-0.73
17) Wing First-Order Twist	deg	-0.50
18) Wing Second-Order Twist	deg	-0.00
19) Fuselage Length	ft	131.28
20) Fuselage Maximum Width	ft	6.00
21) Fuselage Upper Height	ft	4.15
22) Fuselage Bottom Height	ft	5.00
23) Fuselage Starting X Coordinate	ft	-25.22
24) Fuselage Area Ruling Location	N/A	0.15
25) Fuselage Area Ruling Magnitude	N/A	0.30
26) Fuselage Area Ruling Width	N/A	0.75
TOGW	lbs.	186,838

The design point thrust value was 17% larger for the successive NSGA-II method than the DOE method, and this is probably the biggest direct contributor to the weight difference since the propulsion weight is 4,700 lbs. greater and the fuel weight is 5,766 lbs. greater. This combined 10,467 lbs. then causes a ripple effect into the ESAV empty weight which is 11,244 lbs. greater. These effects together create the 17,075 lbs. difference in TOGW. However, this is the direct effect. Parameters such as the area ruling and wing shaping are the source of the need for the larger engines in the successive NSGA-II design. This comparison highlights the highly coupled nature of aircraft design and the worth of considering the whole aircraft system together at once in the MDO framework rather than the compartmentalized discipline design approach.

3.2.4 Reduced-Order Design Space

Often some subset of the design variables has more influence on the objective function than the other design variables. It is likely that in different areas of the design space, the most influential design variables (and combinations of design variables) will be different than in another area. The optimum ESAV configuration found using the large DOE optimization method (as opposed to the direct NSGA-II successive optimizations method) was identified in the last subsection. The current subsection studies the design variables' local effect on the ESAV TOGW in the neighborhood of the optimum design.

Another design of experiments method was used to choose points in an n -dimensional ball surrounding the optimum, where n is the number of design variables. These points are then used to build a Kriging model to characterize the relative effect of every design variable (independent variables) on the ESAV characteristics (dependent variables). The most obvious ESAV characteristic to analyze is the TOGW objective function, but other characteristics are also of interest, like noise levels. The method chosen to analyze the data relationships is based on

an analysis of variance (ANOVA) type decomposition. The complicated Kriging basis functions that are used as the response surface approximation to the n -dimensional ball around the optimal design point makes turning to ANOVA decomposition the right choice to compute the design variable sensitivities [79].

The ANOVA method is very common in the field of computer DOE sensitivity analysis. It is also desirable specifically for use on the ESAV design problem, because the expected form of the design space response surface is nonlinear [79]. This form is expected based on the data obtained while performing the optimization steps detailed above.

The decomposition (or partitioning) of the sum-of-squares is the main mathematical analysis behind the ANOVA method. The variance of a design variable is partitioned into different categories, or variance components, that are associated with different variation sources. The following follows the mathematical development in Fang et. al. [79] and Santner et. al. [78].

The Kriging model can be represented by $y(\mathbf{x})$ and it is assumed to be an integrable function. For simplicity of the mathematical development, the ball around the optimal design value is the d -dimensional unit cube, $C^d = [0,1]^d$. Let

$$y_0 = \int_{[0,1]^d} y(x_1, \dots, x_d) dx_1 \dots dx_d \quad (24)$$

be the overall mean of $y(\mathbf{x})$. The unique functional ANOVA decomposition can be written as

$$y(\mathbf{x}) = y_0 + \sum_{i=1}^d y_i(x_i) + \sum_{1 \leq i < j \leq d} y_{ij}(x_i, x_j) + \dots + y_{1,\dots,d}(x_1, \dots, x_d) \quad (25)$$

that satisfies

$$\int_0^1 y_{i_1, \dots, i_s}(x_{i_1}, \dots, x_{i_s}) dx_k = 0, \quad k = i_1, \dots, i_s. \quad (26)$$

Integrating Eq. 25 over C^d and using the relationship in Eq. 26 gives,

$$\int y(\mathbf{x}) d\mathbf{x} = y_0,$$

specifying that $k \neq i$ gives,

$$\int y(\mathbf{x}) \prod_{k \neq i} dx_k = y_0 + y_i(x_i),$$

and specifying that $k \neq i, j$ gives,

$$\int y(\mathbf{x}) \prod_{k \neq i, j} dx_k = y_0 + y_i(x_i) + y_j(x_j) + y_{ij}(x_i, x_j),$$

and so on. All summands in Eq. 25 are orthogonal in that

$$\int y_{i_1, \dots, i_u} y_{j_1, \dots, j_v} d\mathbf{x} = 0, \quad (27)$$

whenever $(i_1, \dots, i_u) \neq (j_1, \dots, j_v)$.

Equation 25 gives the first-order main effects, $y_i(x_i)$, and second-order interaction effects, $y_{ij}(x_i, x_j)$, of the design variables on the model output. If $y(\mathbf{x})$ is assumed to be square integrable, and drawing upon the orthogonality relation in Eq. 27, then all $y_{i_1, \dots, i_l}$ in Eq. 25 are also square integrable. Thus, we can write

$$\begin{aligned} \int y^2 d\mathbf{x} &= y_0^2 + \sum_i \int y_i^2(x_i) dx_i + \sum_{i < j} \int y_{ij}^2(x_i, x_j) dx_i dx_j + \dots \\ &+ \int y_{12\dots d}^2(x_1, \dots, x_d) dx_1 \dots dx_d. \end{aligned} \quad (28)$$

Define

$$V = \int_{[0,1]^d} y^2(x_1, \dots, x_d) dx_1 \dots dx_d - y_0^2 \quad (29)$$

as the total variance of $y(x_1, x_2, \dots, x_d)$. Recall from Eq. 24 that y_0 is the overall mean of $y(\mathbf{x})$.

The partial variances are computed from each of the terms in the decomposition and are written as

$$V_{i_1, \dots, i_s} = \int_0^1 \cdots \int_0^1 y_{i_1, \dots, i_s}^2(x_{i_1}, \dots, x_{i_s}) dx_{i_1} \cdots dx_{i_s} \quad (30)$$

for $s = 1, \dots, d$ and $1 \leq i_1 < \dots < i_s \leq d$. Squaring both sides of the decomposition in Eq. 25, using Eq. 29, and using the orthogonality properties of the decomposition terms the integration of both sides over $[0,1]^d$ gives

$$V = \sum_{i=1}^d V_i + \sum_{1 \leq i < j \leq d} V_{ij} + \cdots + V_{1,2,\dots,d}. \quad (31)$$

The sensitivity indices (measures) that we want can now be defined using Eq. 31 as,

$$S_{i_1, \dots, i_s} = \frac{V_{i_1, \dots, i_s}}{V}. \quad (32)$$

The proportion of variation on the output due to input x_i is measured by the first-order sensitivity index, S_i . The proportion of variation on the output due to inputs x_i and x_j that cannot be explained by x_i and x_j alone are the second-order sensitivity indices, S_{ij} , for $i < j$. The sum of the sensitivity indices from all design variables for any one output must satisfy

$$\sum_{i=1}^d S_i + \sum_{1 \leq i < j \leq d} S_{ij} + \cdots + S_{1,2,\dots,d} = 1. \quad (33)$$

The sensitivity indices are sometimes referred to as ‘‘Sobol’ indices’’ in the literature.

The local neighborhood in which the DOE will sample points to create the Kriging model was chosen to be $\pm 2\%$ on each design variable, but no design variable bound was allowed to exceed any of the global design space boundaries. The DOE orthogonal array in this local

neighborhood samples 125 points with an additional 25 sampled points generated with the LHS method. This gives a total of 150 points that will be sampled.

The statistically significant sensitivity indices for each output of interest were plotted. All others are put into a general category called “Other.” The most sensitive design variables have the higher sensitivity measure. These measures are plotted in percent values. The ESAV optimization objective function is the take-off gross weight, and it is therefore the first plot shown. Figure 3.2-14 shows that the leading edge sweep (les_c) angle and the fuselage length (l) are the most sensitive design variables to TOGW near the low-fidelity optimum. The design variables, in order, that are most sensitive to the TOGW are

- the leading edge sweep angle (les_c),
- the fuselage length (l),
- the area ruling width gain ($AreaRule_k_wid$),
- the fan pressure ratio ($FanPR$),
- the maximum cruise altitude (alt),
- the maximum cruise Mach number (M),
- the low pressure compressor pressure ratio ($CmpLPR$),
- the root chord (rc), and
- the height of the fuselage top (fh_top).

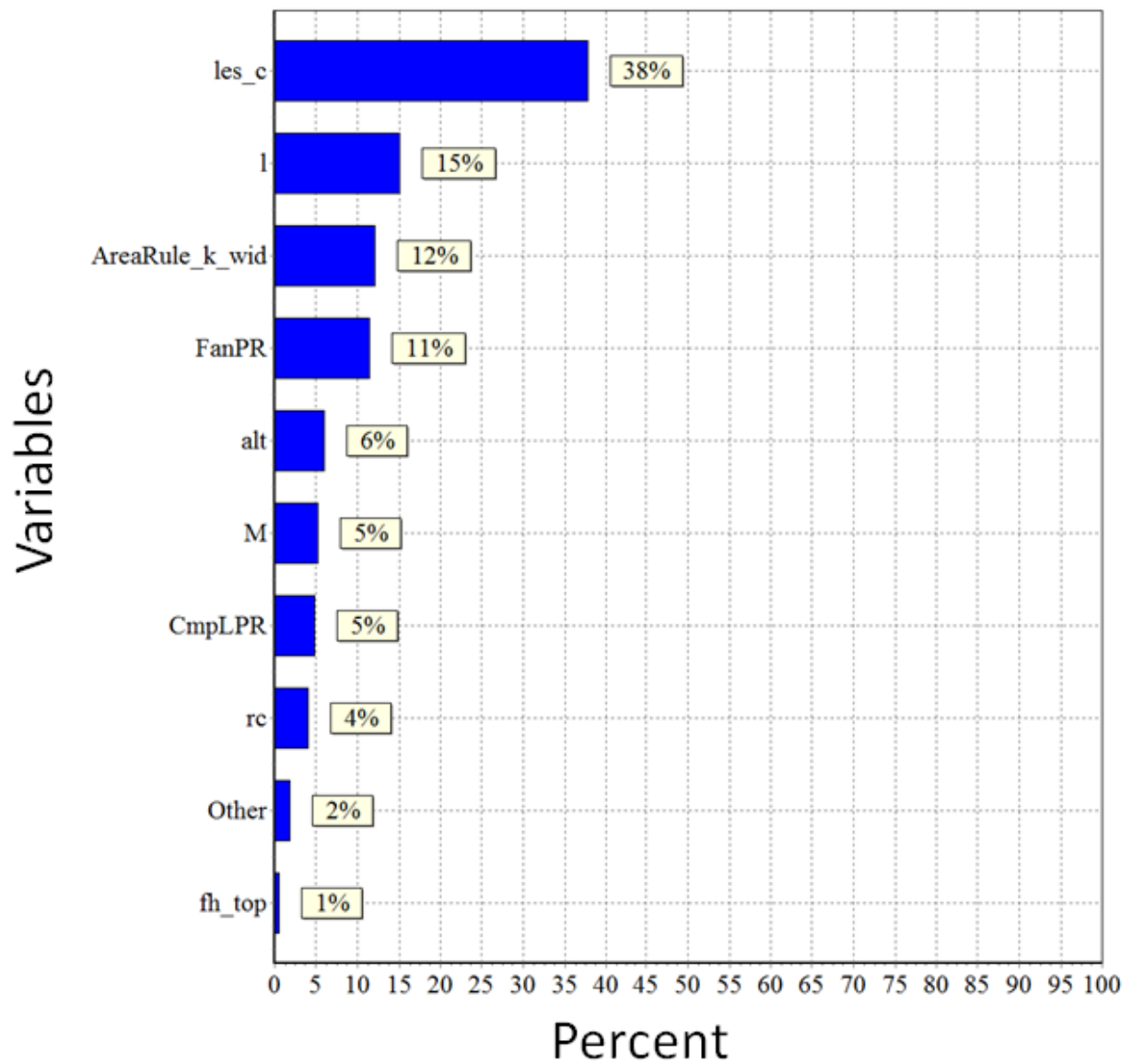


Figure 3.2-14. Take-off gross weight sensitivity plot around the low-fidelity ESAV optimum.

The design variables in “Other” do not have a statistically significant influence on the TOGW.

They are (in no particular order)

- the area ruling magnitude gain (AreaRule_k_mag),

- the area ruling location gain (AreaRule_k_loc),
- the maximum bypass ratio (BPR),
- the high pressure compressor pressure ratio (CmpHPR),
- the high Mach number bypass ratio cutoff (MhighBPRcutoff),
- the low Mach number bypass ratio cutoff (MlowBPRcutoff),
- the semi-span (half_b),
- the tip chord (tc),
- the dihedral (DIH),
- the second order shear (shear2),
- the zero order twist coefficient (twist0),
- the first order twist coefficient (twist1),
- the second order twist coefficient (twist2),
- the maximum fuselage width (fw),
- the height of the fuselage bottom (fh_bot),
- the distance from the wing front tip and the fuselage front tip (fuse_from_wing_x0), and
- the cruise thrust (CruiseThrust).

Looking at the empty weight, the fuel weight, the structural weight, and the propulsion weight sensitivity plots (Figs. 3.2-15 to 3.2-18, respectively) helps to clarify the relationship between the design variables and the TOGW.

The dominant design variable on the fuel weight plot is the leading edge sweep angle – the same as the TOGW plot. The most plausible explanation for this behavior is that the subsonic leading edge (sweep angle is 78° for $M = 2.5$) experiences some transonic effects that are approximated by the low-fidelity aerodynamic codes with changing sweep angle. A higher-

fidelity aerodynamics code that can handle transonic flow directly could help determine if this is, indeed, the case.

The second most sensitive variable on the TOGW plot, the fuselage length, is due to it being the dominant sensitive design variable on the structural weight plot. The fuselage length also has a large effect on the empty weight – due to this structural effect. This finding is not surprising, since the fuselage is a very large contributor to the total weight of the aircraft.

The third most sensitive variable on TOGW, the area ruling gain on the width of the “coke bottling” effect on the fuselage, is customarily tied to wave drag. However, the ANOVA analysis shows the empty weight, not propulsion weight or fuel weight, to be most sensitive to this design variable. That being said, the propulsion weight plot shows an 8% sensitivity measure for the area ruling width gain, but the fuel weight is only at 1%. This means that the engine weight/size is more affected than the fuel burn to changing the gain on the width of the area ruling. The propulsion weight is affected by the engine size required to handle the change in wave drag. The structural weight plot shows a sensitivity of 4% to the area ruling width gain. These values start to tell a story of how this variable is affecting the structural and propulsion weights in a significant way. For the leading edge sweep angle and the fuselage length sensitivity on the TOGW, they could be traced to one dominant effect on a major portion of the TOGW (the fuel weight and structural weight respectively). For this third most sensitive variable, a combined effect on different components of the TOGW is what causes the total higher effect. This combined effect works to produce the 12% value seen on the TOGW plot.

The other statistically significant effects have to do with the engine design, mission, and fuselage size. The mission variables consistently show up on the weight plots that are components to the TOGW, but not as the dominate design variable. The engine parameters

likewise are consistently part of the weight component sensitivity plots. The fuselage is a major weight component, and therefore, it is not surprising that the TOGW is sensitive to these variables.

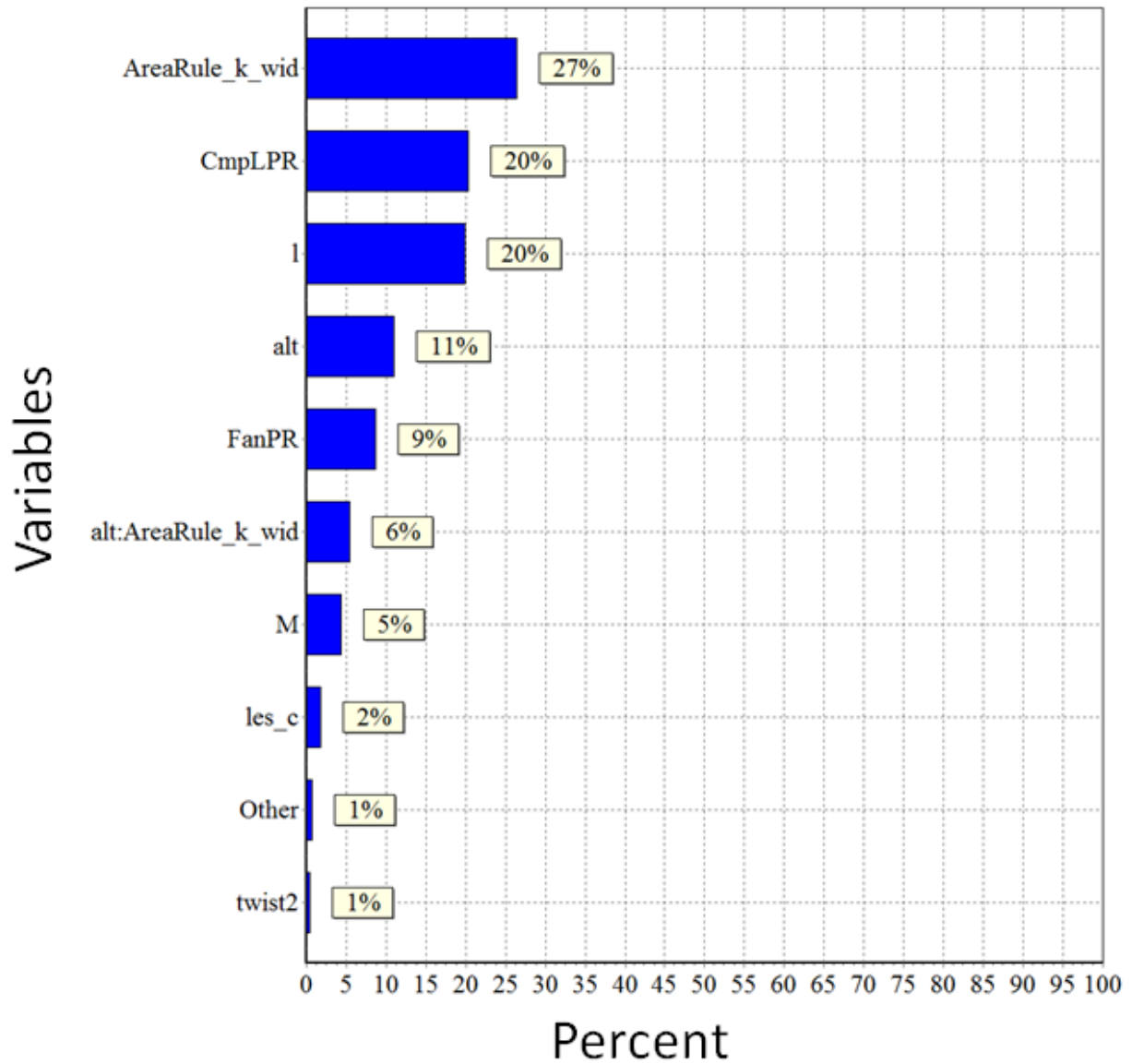


Figure 3.2-15. Empty weight sensitivity plot around the low-fidelity ESAV optimum.

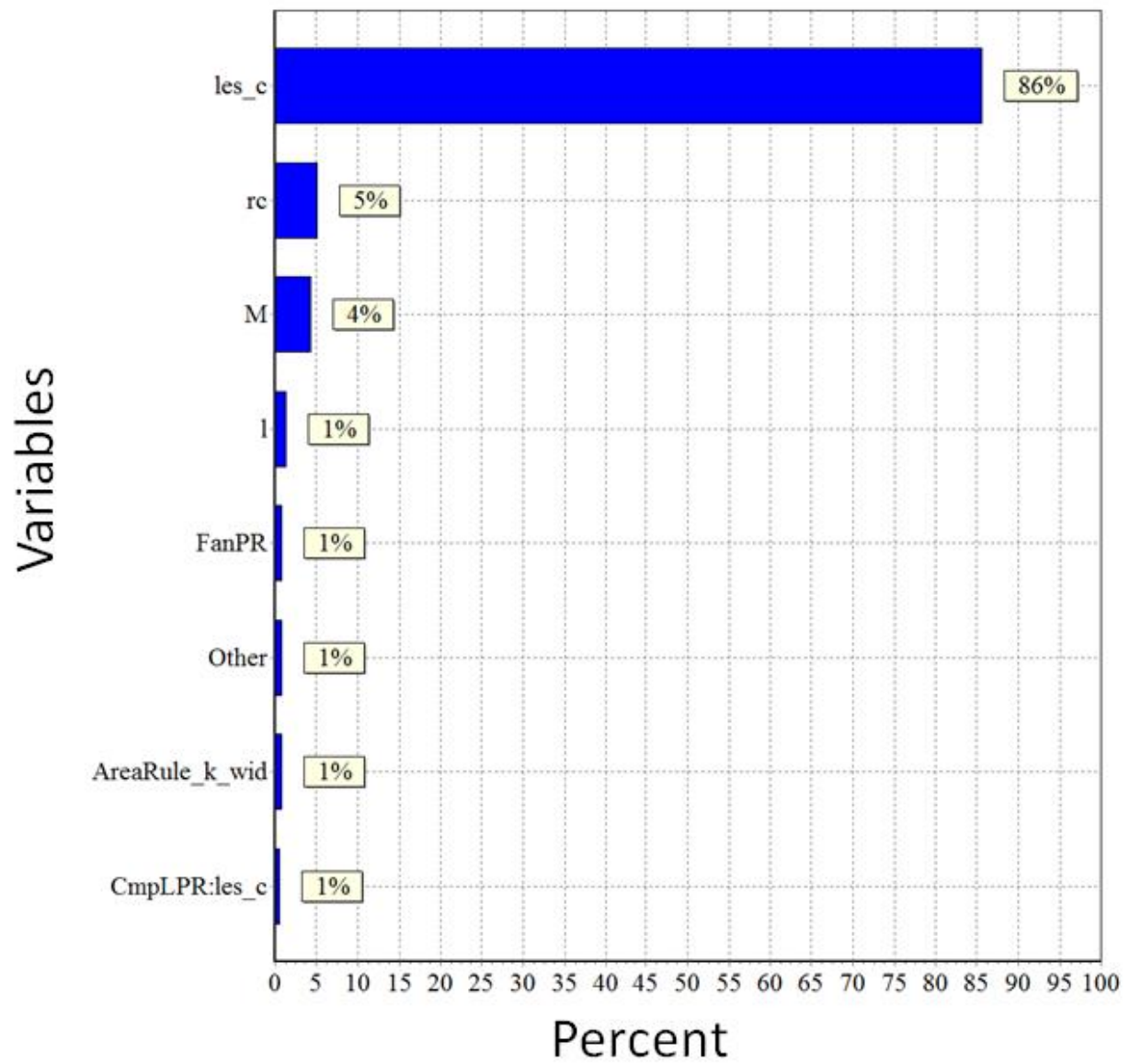


Figure 3.2-16. Fuel weight sensitivity plot around the low-fidelity ESAV optimum.

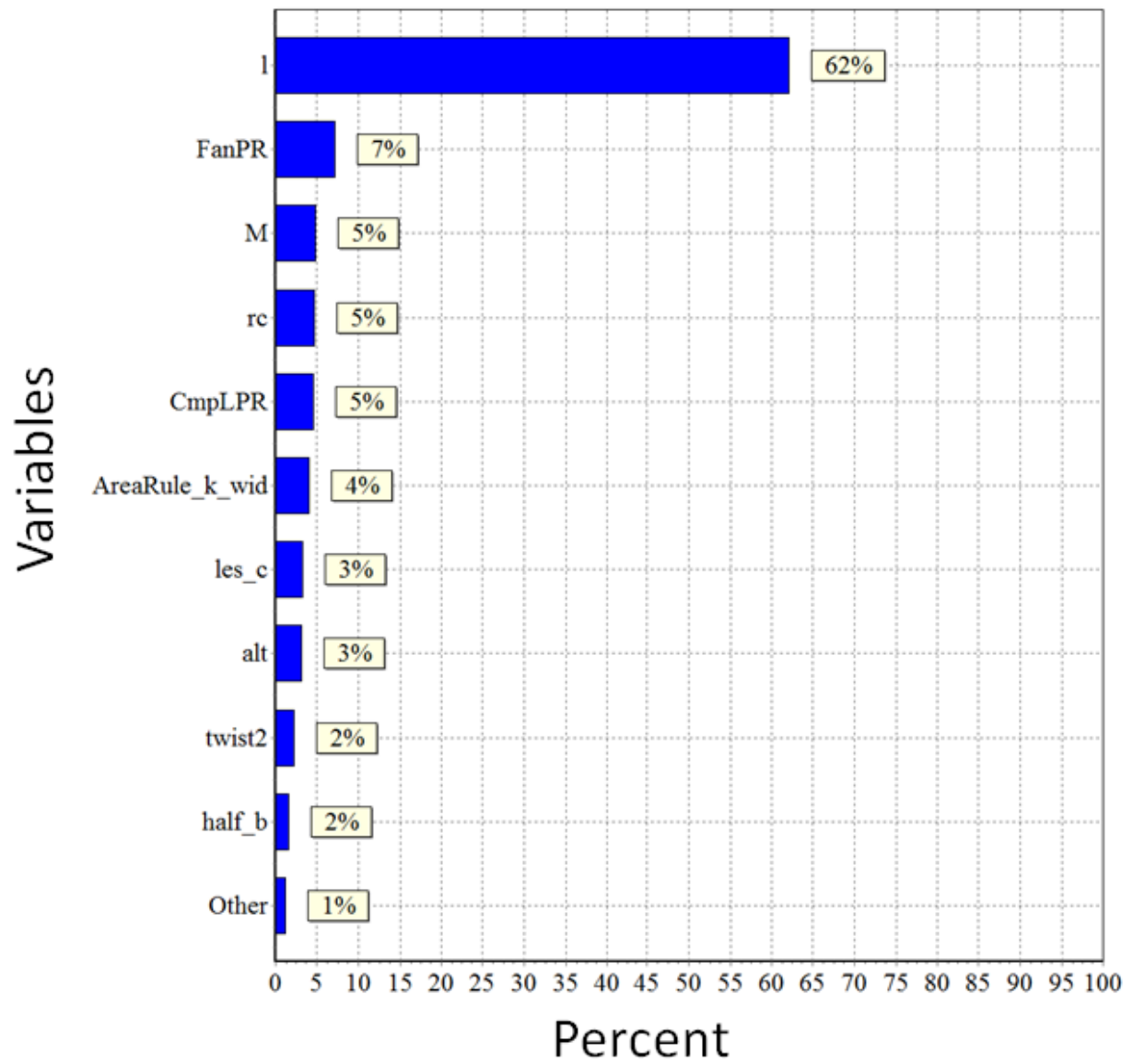


Figure 3.2-17. Structural weight sensitivity plot around the low-fidelity ESAV optimum.

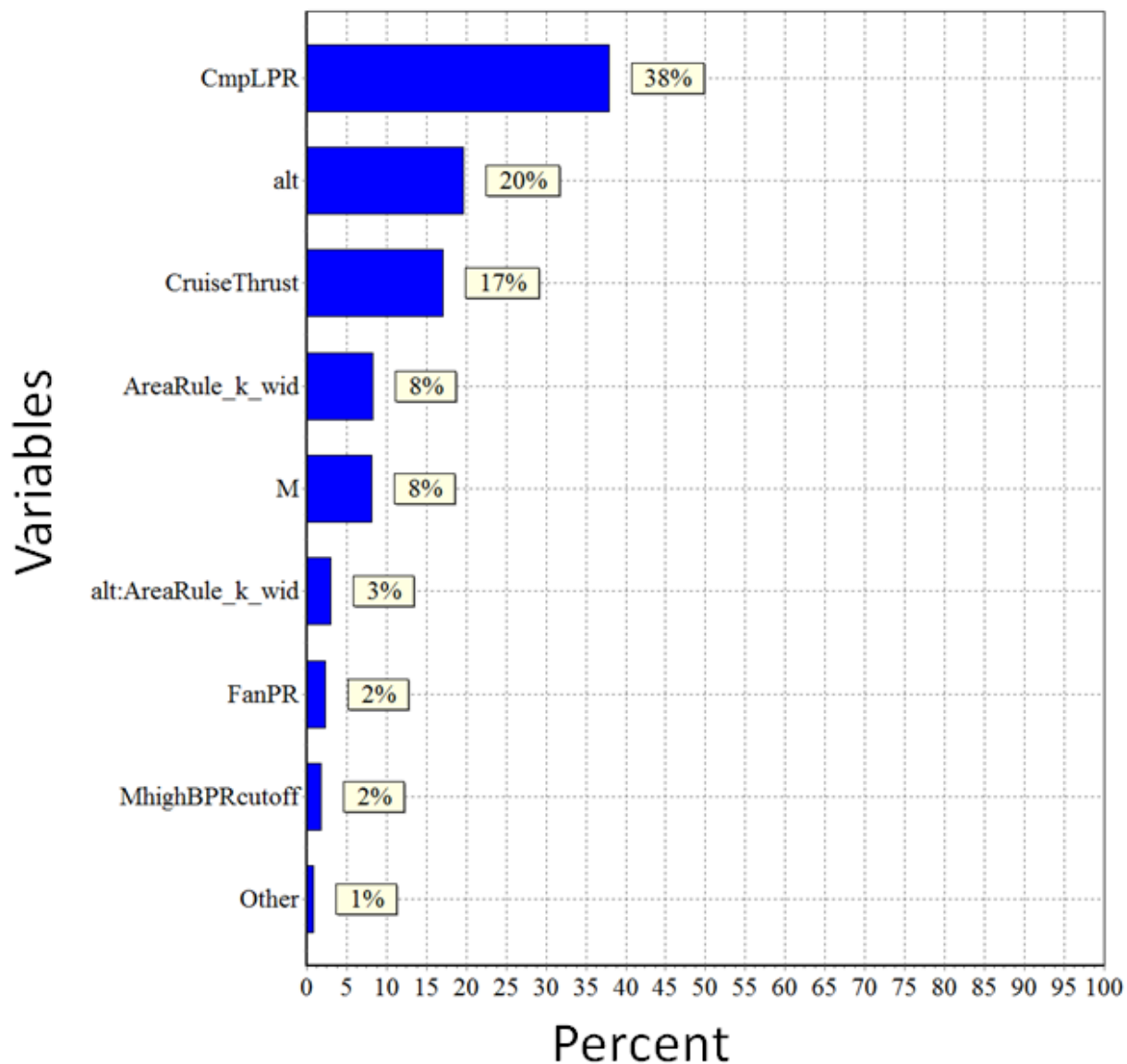


Figure 3.2-18. Propulsion weight sensitivity plot around the low-fidelity ESAV optimum.

The design variables that influence the TOGW the least are also candidates to be eliminated from future optimization studies. A common threshold for elimination is a sensitivity indice of less than 1%. This means that only 9 of the 26 design variables are statistically significant in determining the TOGW. However, it must be remembered that this is based on a

local Kriging surrogate model around the optimum, therefore, without the additional design variables, the optimal location in the design space may not be found by the DOE or optimizer.

Other interesting sensitivity plots show the effect of the design variables on other important ESAV characteristics. The FAR 36 noise levels are a good example. Fig. 3.2-19 shows the sensitivity of the combined noise level (sideline and flyover noise) to the design variables. As expected, the most influential design variables have to do with the engine size and the wing shape. The engine is important, because the exhaust velocity is the largest factor for how much noise an aircraft with a gas turbine engine produces. The wing parameters are important for both airframe noise and engine noise shielding, and even engine size.

Two other important design features that are very influential to aircraft conceptual designs are the thrust-to-weight ratio (T/W) and the wing loading (W/S). Fig. 3.2-20 shows that indeed, the cruise thrust per engine is the most influential design variable for the aircraft T/W . In fact, it appears that the rest of the most influential design variables for the T/W have a greater effect on the thrust than the weight – that is a focus on the numerator rather than the denominator in this ratio. The leading edge sweep, which was identified earlier as the most influential design variable for the take-off gross weight, also has a high sensitivity indice. The zero-order twist angle affects the engine size required to fly, because it affects the lift capability of the wing. This is why it has a statistically significant sensitivity measure on the T/W .

The wing loading (W/S) sensitivity plot is shown in Fig. 3.2-21. Like the T/W plot, the most influential design variable is related to the numerator, or the weight. In the case of the leading edge sweep angle though, it has an effect on both the numerator and the denominator. The second most sensitive design variable on the plot, the root chord, has the largest influence on the wing area. An interesting trend appears on this sensitivity plot. That is, many of the most

sensitive indices are in fact second-order sensitivity indices, S_{ij} . Therefore, the combinations of the design variables together have a statistically significant influence on the W/S. The gain for the magnitude of the area ruling, when in combination with the leading edge sweep angle, has a greater sensitivity indice than the area ruling magnitude gain alone.

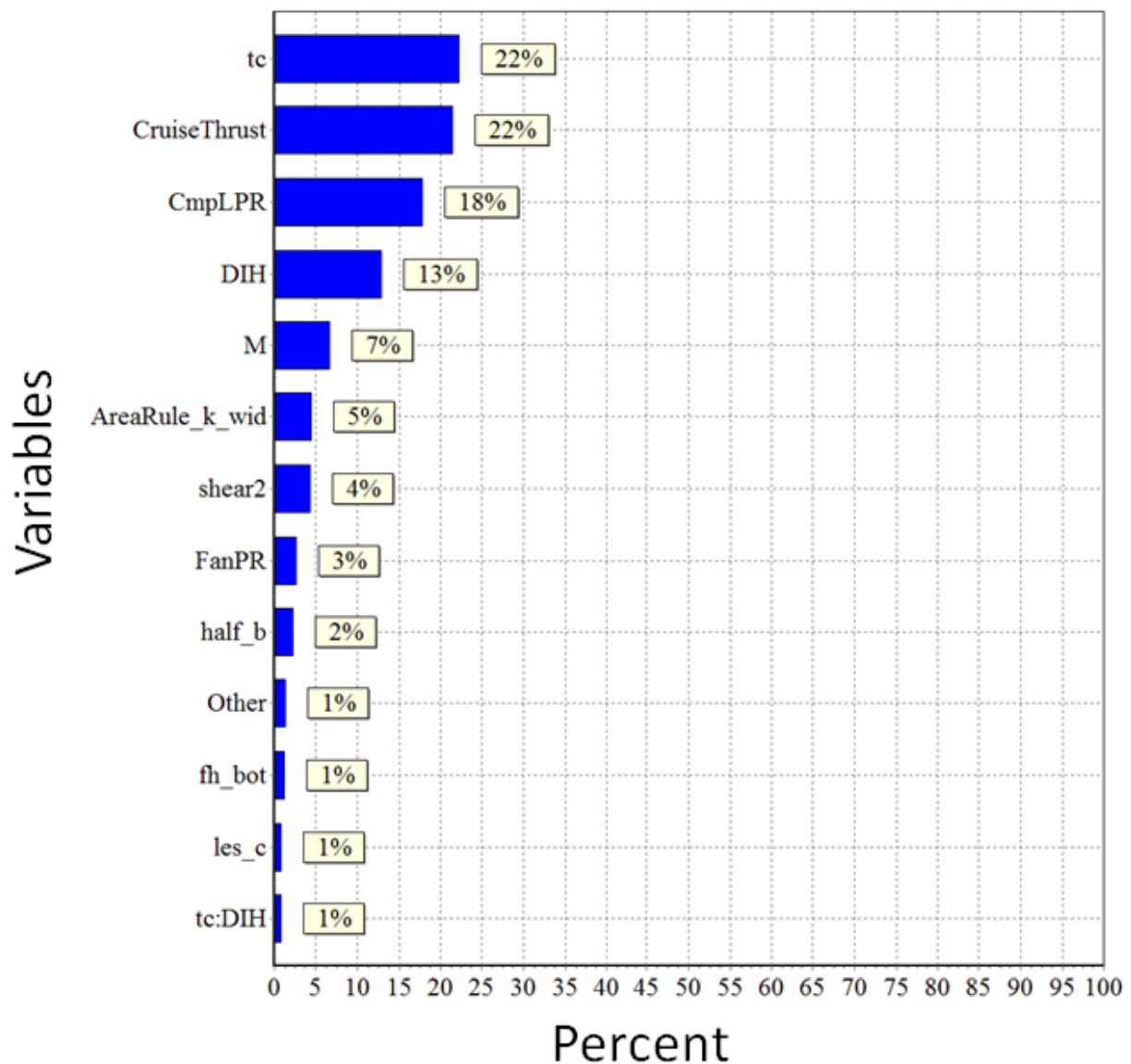


Figure 3.2-19. Aircraft noise sensitivity plot around the low-fidelity ESAV optimum.

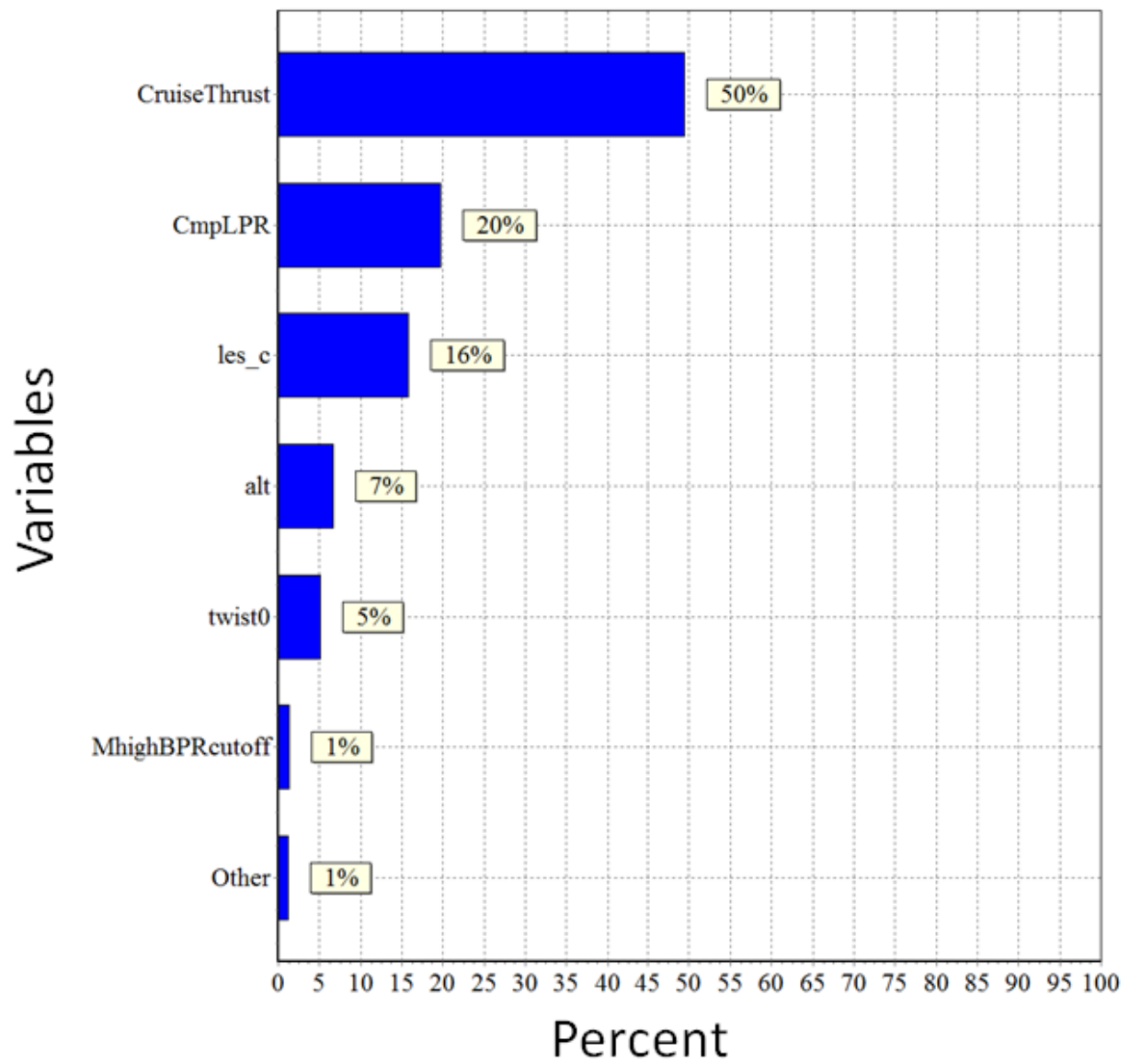


Figure 3.2-20. Thrust-to-weight sensitivity plot around the low-fidelity ESAV optimum.

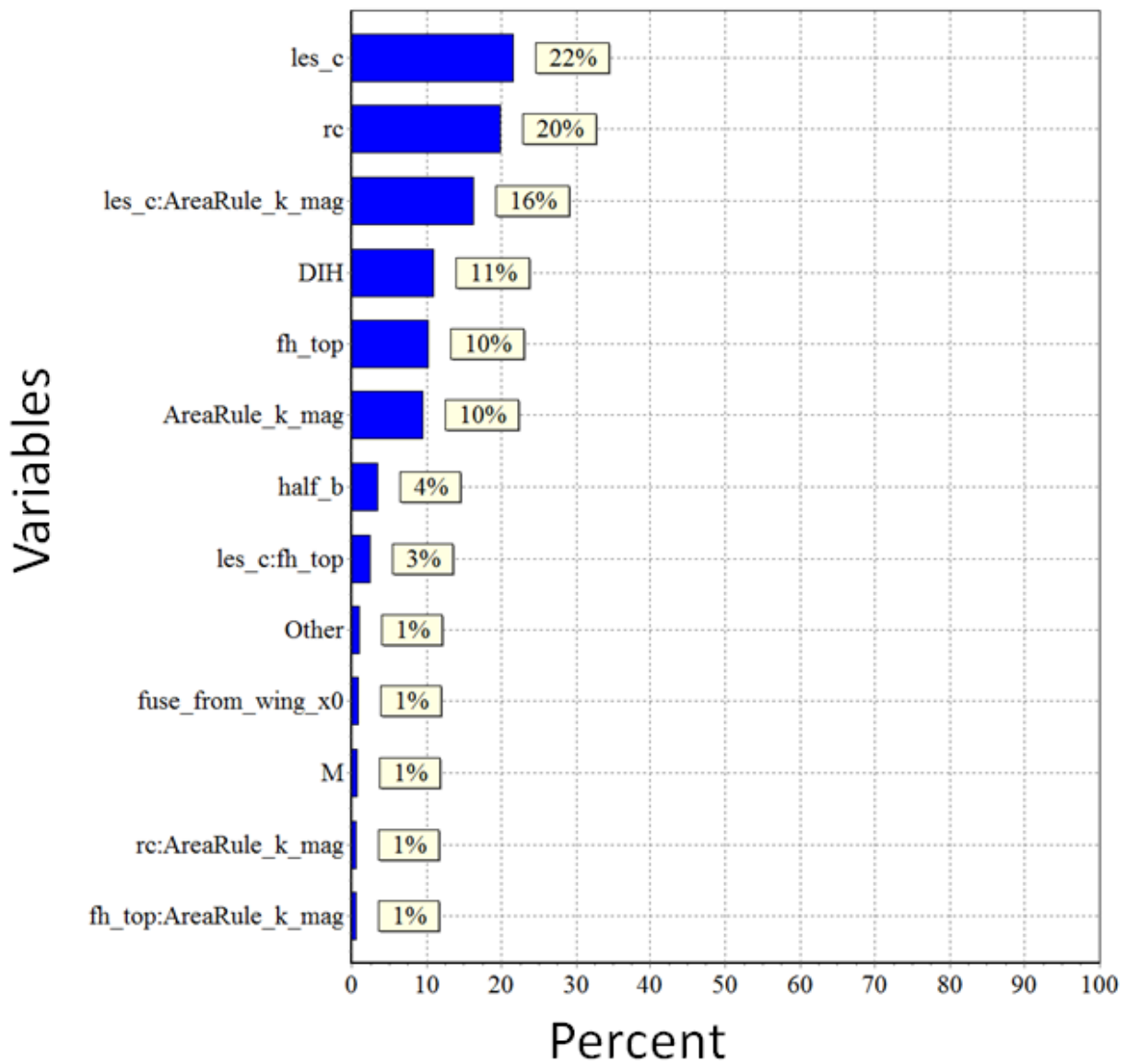


Figure 3.2-21. Wing loading sensitivity plot around the low-fidelity ESAV optimum.

3.2.5 Low-Fidelity Optimization Conclusions

The ESAV low-fidelity framework was exercised, along with an involved design space exploration and optimization process, to identify the best estimation at the global ESAV optimum. This configuration is shown in Fig. 3.2-11 and its final design variables are given in

the boxed column in Table 3.2-4. The planform was an arrow wing with negative higher order twist and a high cruise altitude. The maximum cruise Mach number for this configuration was slightly smaller than 2.5, which is nearly at the top of the allowable range.

The DOE process was computationally expensive. Another method that employed two successive NSGA-II optimizations using the low-fidelity MDO framework using a sensible starting configuration was completed. The second of the successive NSGA-II optimizations used the optimum identified from the first NSGA-II optimization as the new starting point. This method proved to find a good ESAV design in relation to the objective function, and it was likewise an arrow wing configuration. Its TOGW was about 10% higher than the DOE optimum. Figure 3.2-13 shows this ESAV configuration and Table 3.2-6 lists its final design variables. Although this configuration proved to be heavier than the DOE optimum, the number of objective function evaluations required to find it was more than an order-of-magnitude less than the DOE method. This method may, therefore, be useful with a computationally expensive MDO framework. In fact, when the author exercised a slightly different aircraft design framework using the successive NSGA-II method, the final configuration had a better objective function value than that found using the DOE method.

A sensitivity study was carried out in the design space local neighborhood around the DOE optimal ESAV. This study employed the well-known ANOVA decomposition. It was found that only 9 of the 26 total design variables were statistically significant around the optimum ESAV, suggesting that the order of the design space may be able to be reduced from 26 to 9. This makes the large assumption that the DOE method would still be able to identify the optimum ESAV without allowing the optimizer and experimental design to vary and search the other 15 design variables. This may, therefore, be invalid because the sensitivity study is done

only in the local neighborhood of the optimum configuration and therefore does not apply to the larger design space.

4. Discipline Model Enhancement Example - Propulsion

This section uses an individual discipline, propulsion, to show the detailed and involved process that is followed to choose an appropriate disciplinary model for the MDO framework. Although propulsion is chosen as the example, other disciplines could have been used for this chapter, like aerodynamics and structures. It is hoped that the process seen here can be applied by the practitioner to whatever disciplinary model decisions that are relevant to them.

4.1 Selection Criteria for the Propulsion Model to be used in the ESAV MDO Framework

The propulsion analysis is critical to the design of an ESAV. By looking at the technical specifications of past supersonic aircraft, like the B-58, the XB-70, and the Concorde, a general trend emerges that about half of their TOGW is fuel weight. This highlights the importance of having a propulsion model that is able to accurately predict engine operation (and thereby fuel burn). In addition to fuel weight, the dry engine weight is not a negligible part of the TOGW of the aircraft as seen in previous sensitivity studies. Its effects are from the weight of the actual engine, its placement on the aircraft (e.g., wing engines tend to decrease the weight of the wing), and its secondary effects like structural weight, drag, and inertial properties that affect the control system design. The engine dimensions are also an important factor in determining the aircraft size. This includes the inlet, cowl, and engine itself. The fuel burn, measured with the specific fuel consumption (SFC) and time throughout the specified mission, is the most important factor in determining the interior volume required for fuel tanks.

The low-observability constraints placed on the ESAV have led to the requirement of embedded engines. In some ways, embedded engines must be analyzed differently from the traditional nacelle-pylon assembly that usually hangs off the wings or the fuselage, and in other ways the analysis is the same. In addition, the supersonic flight regime that the ESAV will mostly operate in also greatly affects the inlet design. The propulsion model for the ESAV MDO framework, therefore, needs to be able to handle both the general engine model requirements common to all aircraft and the unique and demanding needs of the ESAV concept.

For an embedded engine, the inlet will have a serpentine duct to eliminate radar scattering from the engine compressor face. A serpentine inlet is more likely to exhibit flow aberrations like separation and/or pressure distortion that would not be present in a conventional inlet simply because of the necessity for the flow to turn in the duct from the inlet to the compressor face, whereas a traditional inlet does not turn the flow. Figure 4.1-1 shows a cartoon of a notional serpentine inlet geometry.

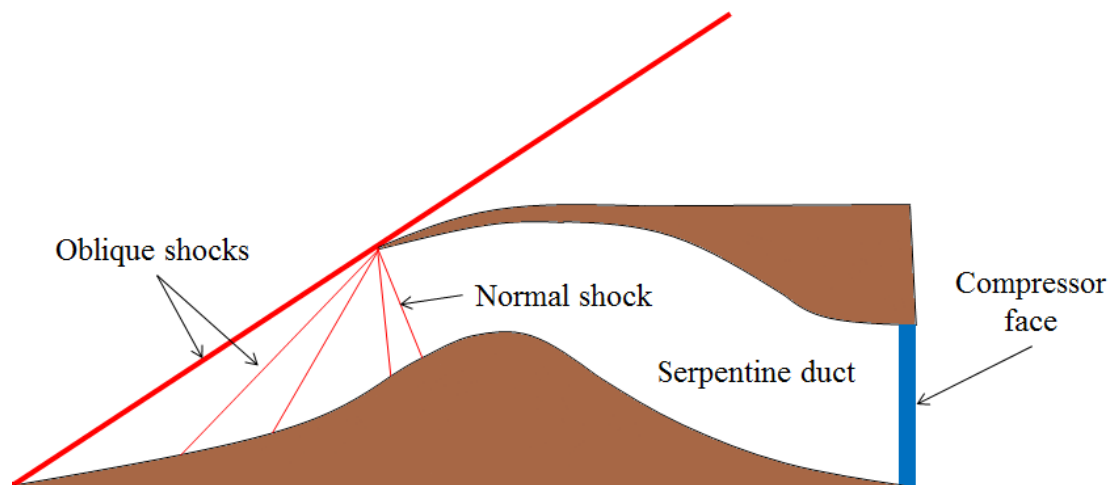


Figure 4.1-1. Notional serpentine inlet in supersonic flight showing oblique shocks, the terminating normal shock, duct geometry, and the compressor face.

The design methodology for the ESAV with its low-observable (LO) nozzle is to fix the exhaust path from the engine exit to the atmosphere so that radar beams will not be able to exit

the channel. The ideal LO nozzle would expand the exhaust flow until the exhaust temperature is the same as the ambient temperature, which eliminates plume heat observability. This cannot be achieved in practice for several reasons including the most obvious of nozzle weight and size issues.

A variable-geometry throat is not required for the ESAV because the engine design will not include an afterburner (augmenter). An afterburner might be expected to be part of the engine design based on the ESAV's high mission cruise Mach number. For instance, the Concorde had four engines with a cruise Mach number of 2.0 (less than the expected Mach range of the ESAV), and still required afterburners [85]. Modern fifth-generation fighter jets, like the F-22, have the capability of supercruise where no inefficient re-combustion of relatively low-pressure air in the nozzle (afterburning) is required to maintain supersonic flight. The majority of the ESAV mission is at the supersonic cruise Mach number. The inclusion of an afterburner would, therefore, significantly degrade the fuel burn efficiency of the engines. In addition, the re-combustion of the already hot exhaust gas in the nozzle increases the temperature of the exhaust further. This is unacceptable for a low-observable vehicle, which must be designed for infrared detection avoidance in addition to radar. For these reasons (and others) no augmenter is included with the ESAV engine core. A notional nozzle structure is shown below in Figure 4.1-2.

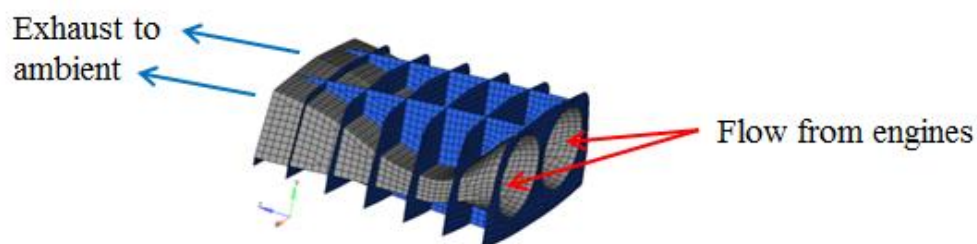


Figure 4.1-2. Notional low-observable ESAV nozzle structure with fixed throat.

As already discussed, the propulsion analysis is critical to the determination of the ESAV design objective function, TOGW. Therefore, the low-fidelity FLOPS propulsion module

underwent an exhaustive evaluation. Several different simulation models were analyzed and weighed against the propulsion needs of the ESAV MDO framework. These criteria are the questions:

1. Can the model be easily automated to incorporate it into the ESAV MDO framework?
2. Can the code results be trusted (validation)?
3. Can the weight and size of the engines be computed?
4. Are both design and off-design points treated?
5. Can the SFC be accurately computed?
6. Is the engine effect of the flow distortion caused by the LO inlet analyzed?

A “yes” answer to each of these questions is favorable to the model. A “no” answer is unfavorable. The only requirement that is not considered critical is the last one that deals with flow distortion. Other methods can be implemented that can model these effects. However, it is still desired that the propulsion model be able to account for this inlet effect.

4.2 Candidate Propulsion Models for the ESAV MDO Framework

There are several propulsion models that have been developed over the years and are available to use. However, these propulsion models will have to meet the requirements laid out above in order to be found adequate to be chosen as the propulsion model for the framework. Four well-known and well-used models were chosen as candidates and were reviewed. The first is the propulsion model that is part of the NASA Langley Flight Optimization System software and was chosen for the initial low-fidelity framework. The second is the NASA/Navy Engine Program which is part of the ACSYNT aircraft design software [86]. The third is the GasTurb 11 or 12 software which has been developed by J. Kurzke as an outgrowth from his years of experience in the aircraft engine business. The last program that has been considered is the

Numerical Propulsion System Simulation (NPSS) software which has been developed at the NASA Glenn Research Center in partnership with major U.S. propulsion companies and academia [87] over many years. These codes represent a good cross section of available propulsion modeling approaches and will be examined in depth one-by-one.

There is too much detailed information about each separate propulsion model to include in the body of the text. Therefore, much of this information is available in Appendix B for the readers interested in more details about the capabilities of each individual propulsion model.

The FLOPS propulsion analysis model capabilities and features are the first to be compared to the propulsion model requirements for the ESAV. The FLOPS propulsion module can be easily automated and incorporated into the ESAV MDO framework. It is based on an older, well-used and well-validated code. The weight of the engine is calculated, but the dimensions are not. As mentioned previously, the weight calculation was not adequate and was replaced by a surrogate model. Both off-design and design points are treated throughout the flight envelope. The engine fuel flow is calculated for the mission. The FLOPS propulsion module cannot calculate the flow distortion caused by the low-observable inlet, or the engine characteristics with such inlet flow distortion. This code was not written for embedded engines, therefore this technology is not treated.

The ACSYNT propulsion analysis module is next compared to the propulsion model requirements for the ESAV. In addition, the ACSYNT model is also compared to the FLOPS propulsion module. The ACSYNT propulsion model can be automated and incorporated into the ESAV MDO framework. This includes the ability to extract the information generated by the model needed by other disciplines, and input the information from other disciplines required by the model. It has been in existence for many years and is therefore well-used and well-validated.

ACSYNT has continually upgraded the propulsion code to the newest version when available up until the NNEP89 code [88]. The weight of the engine is calculated, but the dimensions are not. Both engine design and off-design points are treated. The engine fuel flow is calculated for the mission giving the fuel weight portion of the TOGW. This propulsion program cannot calculate the flow distortion caused by the low-observable inlet, or the engine characteristics with such inlet flow distortion. This is because the ACSYNT propulsion model was not written for embedded engines and the required inlet and nozzle geometry are not provided.

Comparing the FLOPS and ACSYNT propulsion modules points to the fact that they are very similar in capability and application. However, one more subtle difference is in regards to the input deck in the area of ease of automation. While both can be automated, the input deck for ACSYNT is more accurate but at the expense of being more time consuming to construct [89]. Also, it has been pointed out that the ACSYNT approach is less scalable [89].

GasTurb is the third propulsion analysis software compared to the propulsion model requirements for the ESAV. The biggest drawback is that GasTurb cannot be automated and incorporated into the ESAV MDO easily because it is based on a graphical user interface (GUI). However, it is well-validated and in common use among most major aero-propulsion companies (although there is a disclaimer that the author is not responsible for any incorrect results that the code might produce). The weight and dimensions of the engine can be calculated, although the engine weight estimate is admittedly lower than the actual weight. This would require some compensating calculation or assumption to overcome this deficiency. The more computationally-intensive “more” mode is required for dimensions. Both off-design and design points are treated and can be conveniently pre-defined in a mission envelope design table using a spreadsheet or ASCII file. The engine fuel flow can be calculated for a given profile mission, which is then able

to supply the fuel weight portion of the TOGW. Unlike the previous two propulsion models, GasTurb can calculate the flow distortion caused by the low-observable inlet. This includes both the pressure distortion and the temperature distortion, although the temperature distortion model is not to the same level as the pressure distortion model. Therefore, GasTurb meets all demands made by an ESAV except for its ability to be easily integrated into the ESAV MDO framework given its GUI dependence. In comparison to the FLOPS propulsion module and the ACSYNT propulsion model, the GasTurb program is superior in every way except for its ability for automation in the MDO framework.

A comparison of the NPSS/WATE++ software and the propulsion model requirements for the ESAV is the final one to be carried out. Although NPSS, by virtue of its object-oriented design, has the ability to incorporate specialized codes to overcome any deficiency, the following analysis is done with just the core executable code itself because large external codes are not desirable in conceptual design.

NPSS can be easily automated and incorporated into the ESAV MDO framework, because all of its input and output is done with files. It is well-validated and in common use among all major United States aero-propulsion organizations (including industry, government, and academia). The weight and dimensions of the engine can be calculated with the NASA-integrated NPSS/WATE++ code. WATE++ is based on WATE, which is a validated and longstanding code. Both off-design and design points are easily treated. The engine fuel flow can be calculated for a given profile mission which is then able to supply the fuel weight portion of the TOGW. NPSS can either calculate the pressure distortion and the temperature distortion caused by the low-observable serpentine inlet through the integration of an external flow analysis or with assumptions incorporated into the construction of the engine model itself. Thus it does

not need to rely on the external code to incorporate these effects – at least approximately.

Therefore, a NPSS/WATE++ model meets all demands made by the ESAV design environment.

Table 4.2-1 below summarizes the results from the comparisons above.

Table 4.2-1. NPSS, GasTurb, ACSYNT, and FLOPS Propulsion Modules vs. ESAV

Propulsion Requirements.

Propulsion Model Requirements	FLOPS	ACSYNT	GasTurb	NPSS/WATE++
Automated	Yes	Yes	No	Yes
Validated	Yes	Yes	Yes	Yes
Weight & Dimensions	Not Adequate, No	Yes & No	Yes & Yes	Yes & Yes
Design and Off-Design	Yes	Yes	Yes	Yes
Accurate Fuel Flow	Yes	Yes	Yes	Yes
LO Inlet Distortion	No	No	Yes	Yes

The results of an exhaustive study of available software to use in the ESAV MDO framework for just one discipline concluded that the best choice for ESAV MDO framework propulsion simulation is a NPSS/WATE++ model. This conclusion is clear based on the data shown in Table 4.2-1. Therefore, NPSS/WATE++ was chosen as the propulsion model for the MDO framework. GasTurb came in at a close second, only lacking in its ease to be incorporated into the automated framework due to its GUI interface.

A similar exercise was done for other disciplines in the ESAV MDO framework, but this type of detailed analysis is not given for each discipline in this work due to space considerations. The next section provides details of the ESAV NPSS/WATE++ propulsion model used in the ESAV MDO framework. This will again be a more thorough treatment than given for other

disciplines. However, similar processes were employed for each discipline appropriate to the model and its fidelity level.

4.3 Detailed Propulsion Model Development

Once the NPSS/WATE++ propulsion simulation software was chosen, a model had to be developed and integrated into the ESAV MDO framework. This model must be able to interface in both input and output with all other coupled disciplines. The initial low-fidelity N² diagram, introduced in Fig. 3.1-1, is a good starting place to determine what the required input and output data are. However, as will be seen in the next chapter, this low-fidelity view of the framework does not capture all of the coupling for the multidisciplinary analysis required for an ESAV. This speaks to the main obstacle for the creation of a modular MDO framework. Fortunately, the object-oriented nature of NPSS makes the incorporation of new data input or output easy. Therefore, the core of the propulsion analysis and its parameterization is of the greatest importance. This section strives to show the development and details of the NPSS/WATE++ propulsion analysis model, and describe how the input and output of data is nominally handled in the practical ESAV MDO framework.

4.3.1 Choice of Propulsion Cycle Models

Three different cycles were chosen to be developed for the ESAV MDO:

1. turbojet cycle,
2. low-bypass turbofan cycle, and
3. variable cycle.

The reason for these three choices can be understood by considering the overall gas turbine engine efficiency, η_0 , which is made up of both the propulsion efficiency, η_p , and the thermal efficiency, η_{th} [90]. Assuming a single stream, these equations are,

$$\eta_p = \frac{Tu}{\dot{m}_a \left[\frac{u_e^2}{2} (1 + f) - \frac{u^2}{2} \right]} = \frac{\text{thrust power}}{\text{production of kinetic energy rate}}, \quad (34)$$

$$\eta_{th} = \frac{\dot{m}_a \left[\frac{u_e^2}{2} (1 + f) - \frac{u^2}{2} \right]}{\dot{m}_f Q_R} \equiv \frac{\text{production of kinetic energy rate}}{\text{energy consumption rate}}, \text{ and} \quad (35)$$

$$\eta_0 = \eta_p \eta_{th} \equiv \frac{\text{thrust power}}{\text{energy consumption rate}}. \quad (36)$$

The variable T is the thrust, u is the aircraft velocity, u_e is the exhaust velocity from the engine, $\dot{m}_a$ is the mass flow rate of air into the engine, $\dot{m}_f$ is the mass flow rate of fuel into the engine for combustion, f is the fuel-to-air ratio, and Q_R is the heat of reaction (a measure of how much energy the fuel can release during combustion per mole of fuel).

The exhaust velocity must be greater than the aircraft velocity if any thrust is to be produced. As $u_e \rightarrow u$, the propulsion efficiency approaches unity or 100%. However, the other part of the overall efficiency is the thermal efficiency. As $u_e \rightarrow u$, the thermal efficiency approaches zero (as does the thrust produced). Thus, these two components of η_0 compete with each other as the flight Mach number varies for a fixed engine.

The equations become more complicated with the bypass flow in a turbofan, but the net result is a slower exhaust velocity due to the bypass air. The bypass air is handled with a separate flow velocity from the core flow and designated as, u_{ef} . At higher flight Mach numbers (higher u), the turbojet, which has a higher u_e , will also have a higher overall efficiency. At lower Mach numbers, however, the turbofan will be the more efficient choice. An important threshold is the Mach number where the turbojet η_0 begins to exceed the turbofan η_0 . It turns out that, in the general case, there is an overlap region in the Mach number space so that both cycles are potentially valid for an ESAV. Raymer [43] includes a chart on engine selection over Mach

number that shows the ESAV cruise Mach number is indeed in the overlap Mach number range for the low-bypass turbofan and turbojet.

The turbojet is a traditional choice for a high-speed aircraft due to its higher exhaust velocity. However, the low-bypass turbofan cycle model was also developed in this ESAV MDO effort, because the Mach number at which the turbojet will surpass it in efficiency is not set in stone and depends on the engine design (e.g. the bypass ratio). The variable cycle was chosen, because it blends the best of both the turbojet and turbofan at the differing flight regimes encountered during a supersonic cruise mission profile. Certainly, at low speeds, a bypass flow is desirable. However, at the higher Mach numbers the bypass flow reduces the overall engine efficiency. The variable cycle essentially changes the bypass ratio as a function of aircraft flight Mach number. The way that the variable cycle does this has been parameterized within the NPSS/WATE++ model, and it is therefore available as design variables in the ESAV MDO framework.

4.3.2 Turbojet Model

The NPSS/WATE++ model of the turbojet can be broken up into different parts, each contained within its own file to ease bookkeeping. The file that is actually passed to the NPSS executable contains a list of all files needed for the turbojet to be completely defined and solved. These files are defined in this section. They are added to the main file with the familiar C++ syntax of “#include.” The engine is designed by NPSS in such a way so that it matches the design point maximum thrust per engine, which is a global design variable.

The thermodynamic package used for the turbojet model is the Joint Army-Navy-Air Force (JANAF) package. An online repository of the JANAF database can be found on the National Institute of Standards and Technology (NIST) website [91]. This model is well known

and validated. A thermodynamics package must be specified for NPSS to run. It is initialized with a call to the NPSS setThermoPackage function.

A non-standard NPSS file that contains all possible parameterized variables was created in an attempt to ease the integration of NPSS into the ESAV framework. For the turbojet model, this file is called “GlobalVariables.in.” Parameters such as the maximum cruise Mach number and altitude, the thrust desired at the design point, pressure ratios, nozzle area ratio, extracted power to run aircraft systems, bleed flow parameters, and maximum engine temperatures are all defined and set in this file. In this way, the design variable input interface with the ESAV MDO framework only requires one file.

NPSS requires an external code to be integrated for supersonic inlets. The purpose of this code is to calculate the shock compression process of an inlet through the terminating normal shock, and then pass the subsonic flow conditions behind this shock into the engine model. For conceptual design, a four-shock inlet was chosen due to its efficiency based on the ESAV’s high cruise Mach number and that the vast majority of the mission is at this Mach number. A cartoon of the inlet is shown in Figure 4.3-1.

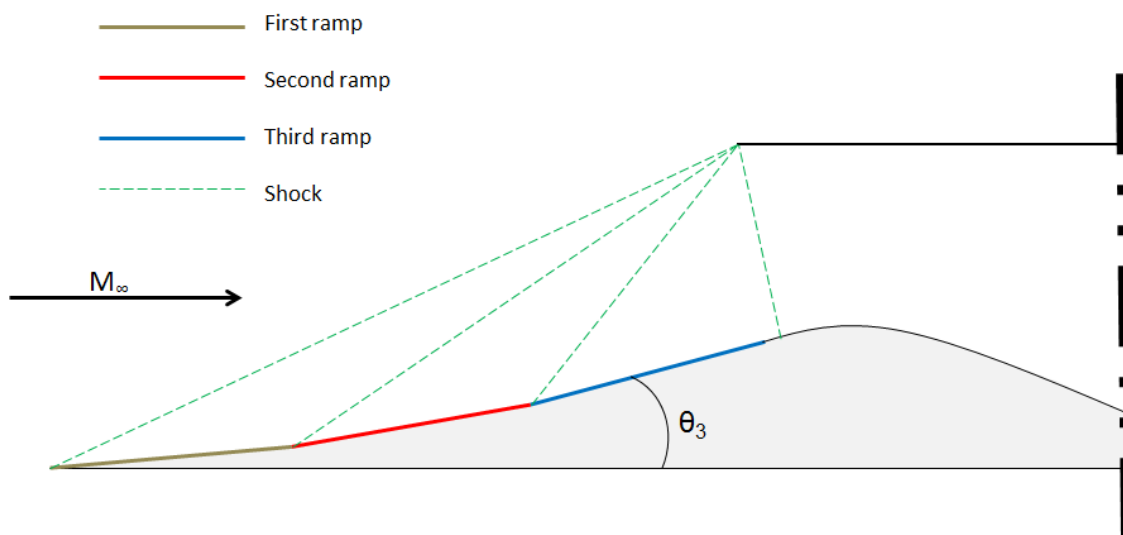


Figure 4.3-1. Four shock inlet showing three ramps and the terminating normal shock.

An additional oblique shock could have been added for a five-shock design, but the increase in total pressure recovery compared to the increase going from three to four shocks is small [92], and an additional ramp would increase complexity and weight of the aircraft. The resulting two-dimensional flow problem's simplicity aided in reducing the computational expense required of a more complex three-dimensional inlet.

The design parameters are the three angles, $\theta_1 - \theta_3$. A local optimization is performed in order to choose the three angles that maximize the value of the total pressure ratio from the freestream to the flow behind the normal shock at the cruise Mach number and altitude. The height of the cowl and length of the ramp are determined based on the idealized constraint that all shocks meet at the cowl lip. These values then determine the dimensions of the inlet geometry. The cowl width is determined based on the required mass flow for an engine to obtain the required thrust as determined by NPSS and the computed cowl height. The necessary data comes from the solved compression problem's flow density behind the normal shock, which is used to calculate the required inlet area and thus the cowl width. This is an iteration that is performed between the VT-CST geometry model and the NPSS code.

The method chosen to integrate this code into the NPSS/WATE++ turbojet model for off-design computations was in the form of a NPSS table element, or in other words, a surrogate model. This model was built by running optimization cases over differing flight Mach number and altitude pairs, (M, h) . The optimal angles, $\theta_1 - \theta_3$, which maximize the total pressure recovery, were found by the optimizer, and the flow conditions immediately behind the terminating normal shock were recorded. Linear interpolation is used by NPSS for the practical use of this surrogate. This surrogate table look-up is used for the off-design cases.

The design point case uses the same code as was used to build the surrogate, but it employs an external inlet optimization in an iteration loop with the geometry model. This reduces the error for the design point engine thermodynamic and dimensional design because the code is run directly and no surrogate is used. It also reduces the error for the cowl and inlet geometrical designs.

The “bleed_macros” function file is included in the cycle models. This is a standard file that is distributed along with the copy of the NPSS executable. It allows a user to conveniently use bleed ports. Rules are applied to the creation and linking of these ports that the user must follow. This limits the number of bleed situations that can be modeled to just four, but no additional situations were needed for the ESAV NPSS/WATE++ model. Thus, these four situations were completely adequate for the ESAV engine models.

1. Bleed from compressor to turbine (e.g. cool turbine blades)
2. Bleed from compressor to a bleed element (e.g. surge bleed)
3. Bleed from a bleed element to turbine
4. Bleed between two bleed elements

View files are used to collect the data that is desired to be output by NPSS and WATE++. The view files for the turbojet model pull out many various flow conditions from the thermodynamic stations. These include, but are not limited to, Mach numbers, total temperatures and pressures, static temperatures and pressures, fuel-to-air ratio, fuel flow rate, nozzle dimensions, the specific fuel consumption, and thrust values. All of these output values are then available to any other discipline as required through the ESAV MDO framework.

A specific view file was written for the engine deck required by the FLOPS mission performance module. This FLOPS module requires the engine input deck in a specified rigid

format. In fact, all engine off-design point runs in the NPSS model are for the specific reason to fill up the engine deck specified by the FLOPS mission performance model. This NPSS view file is shown in Fig. 4.3-2. Syntax and other details are in Reference [93].

```

OutFileStream FLOPSheetStream {
    filename = "FLOPS_engine.out";
}

DataViewer CaseRowViewer FLOPSheet {
    string BlankString = "    ";
    caseHeaderBody = "";
    caseHeaderVars = {}

    variableList = { "Minf:?.??",
                     "altitude:?????",
                     //"blankPC:?????",
                     "Perf.PowCode:?????",
                     "Perf.Fg:??????.?",
                     "Perf.Fram:??????.?",
                     "Perf.Wfuel:??????.?",
                     "BlankString:?????????",
                     "Perf.SFC:???.?????",
                     "NozPri.Aexit:??????.??"
    }

    showMarks = FALSE;
    showHeaders = FALSE;
    pageWidth = 124;
    pageHeight = 800;

    outStreamHandle = "FLOPSheetStream";
}

```

Figure 4.3-2. FLOPS engine deck view file where blankPC is a whitespace string of fixed width required for rigid column format required by FLOPS.

The next file included in the run file is the model file, and is the core of the NPSS propulsion model. It defines all turbojet engine components, and also acts as the governing file for how the components are linked. Beginning with the external compression inlet discussed previously, the results of this external code are passed into an “Ambient” element which is defined behind the terminating normal shock. Next, the flow goes through an “Inlet Start”

element and then an “Inlet” element. Finally, the flow goes through a “Splitter” element that takes a boundary layer bleed flow from the main stream and dumps it overboard. This process is shown in Fig. 4.3-3. Explanation notes and engine design variables are included along the flow chart. The model file also links all elements that make up the turbojet engine. These are represented as the arrows between boxes. The subsonic Mach number, pressure, and temperature fed into the “Ambient” element are calculated with this inlet surrogate model, and are not design variables. The cruise Mach number and altitude that feed into the external compression inlet block are global design variables.

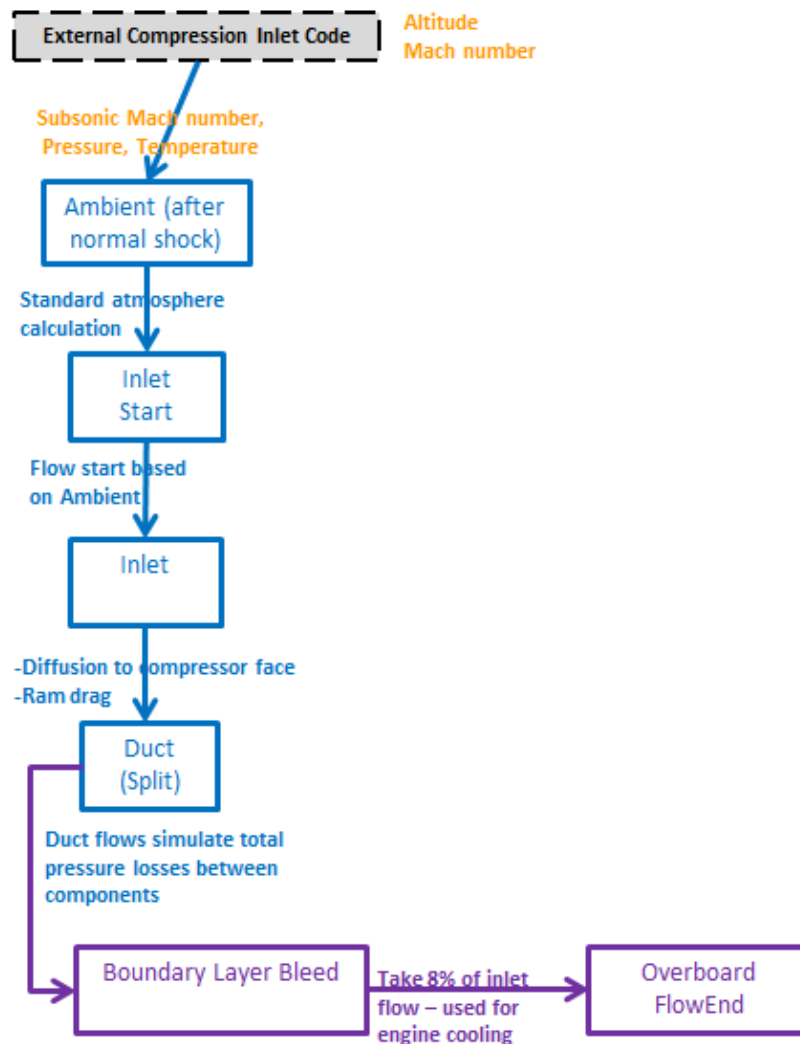


Figure 4.3-3. Front end block diagram of the NPSS turbojet model.

Boundary layer bleed has been identified as a good candidate to control the flow through the LO inlet duct [94] [95]. The internal flow must be controlled to limit flow distortion to acceptable levels. Therefore, the boundary layer bleed dumped overboard represents the presence of actual flow losses incurred by using this method to control the flow through the serpentine inlet. In addition to this penalty for the LO inlet, a total pressure loss is also factored in based on high-fidelity computations and engineering judgment [96].

The next part of the turbojet model file defines the engine core. First, the bulk of the split flow (that portion not taken as a boundary layer bleed) goes through the compressor. Part of this flow is bled off to cool the turbine blades. This is necessary given the high turbine inlet temperature (T_4) assumed for this advanced, high-performance engine.

The remaining flow is referred to as the core flow and constitutes most of the air that was compressed into the engine from the ambient atmosphere. Since this is a single-spool turbojet cycle, there is only one high-pressure compressor. A surge bleed element is added to protect against surge conditions. The core flow then goes through a duct and a burner. It is here where combustion is modeled with fuel fed into the flow with the “FuelStart” element. The NPSS-default Jet-A fuel is assumed, but could easily be changed by changing the FuelStart parameters appropriately. The fuel composition is not limited to only existing fuels with this functionality, so alternative fuels (e.g. biofuels) can be simulated with an ESAV.

Finally, the core flow expands through the turbine. Several duct elements are inserted between major components to simulate total pressure losses throughout the engine, as would be experienced in a real engine. The top level solver uses the necessary power balance of the shaft between the turbine and the compressor to find a converged engine solution. A flowchart of the turbojet core is shown in Fig. 4.3-4 with explanation notes and engine design variables included.

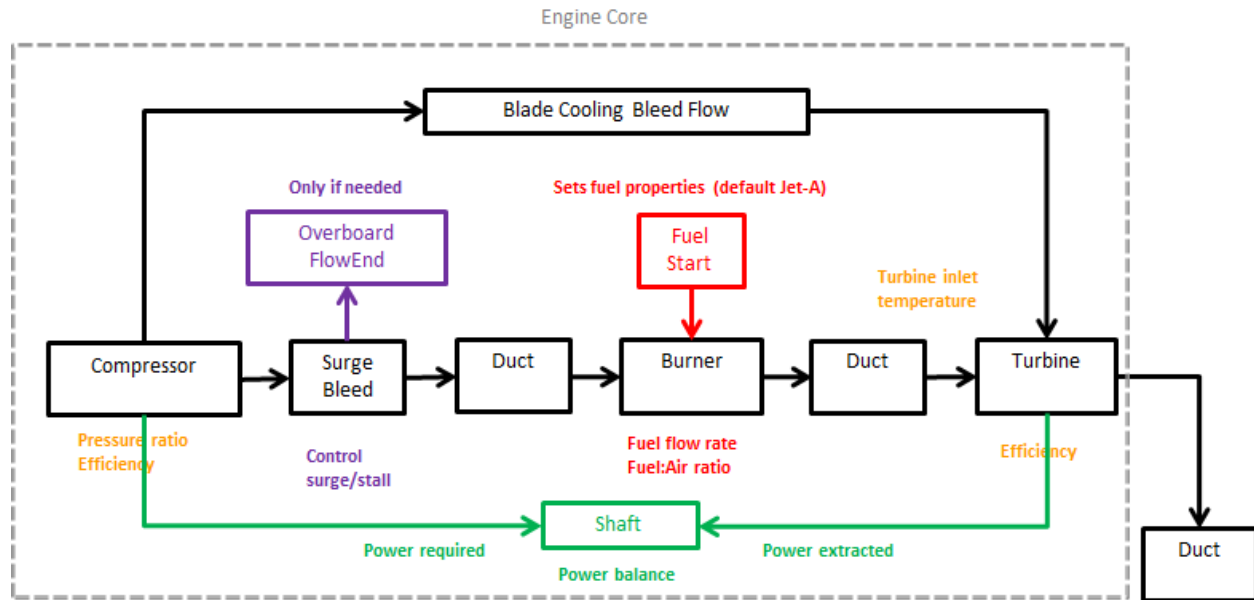


Figure 4.3-4. Engine core block diagram of the NPSS turbojet model.

The flow must exit through the nozzle and terminate for the propulsion simulation. Although the ESAV will nominally not include an afterburner, one is optionally included in the engine model, if the user desires this engine feature. This afterburner can be turned on and off with the setting of a single variable defined in the GlobalVariables.in file. This allows the possibility that an augments could be included in a future ESAV MDO study. The afterburner flow exits to the converging/diverging nozzle element. NPSS requires a flow ending element as it did a flow starting element. The block diagram of the engine back end is shown in Fig. 4.3-5.

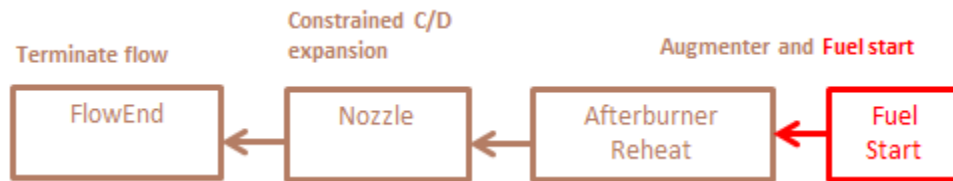


Figure 4.3-5. Back end block diagram of the NPSS turbojet model.

So far, each of these files has been for NPSS by itself. The next file included defines the WATE++ model. This model is built up in an almost one-to-one fashion with the NPSS model file. Each of the components in the NPSS model file that will add either weight or dimension

must be included in the WATE++ model. Two exceptions are the WATE++ engine mount element and the accessories element, both of which are self-explanatory as to their purpose in the weight model. Like the NPSS model, an optional augments is included if desired. WATE++ has its own standard view file for data output. In the NASA version of NPSS, WATE++ is seamlessly integrated so that all required thermodynamic data from the NPSS solution is immediately available for the engine weight and size computations. WATE++ has a graphical output available for the user to visually view the engine and an example turbofan can be seen in Fig. 4.3-6.

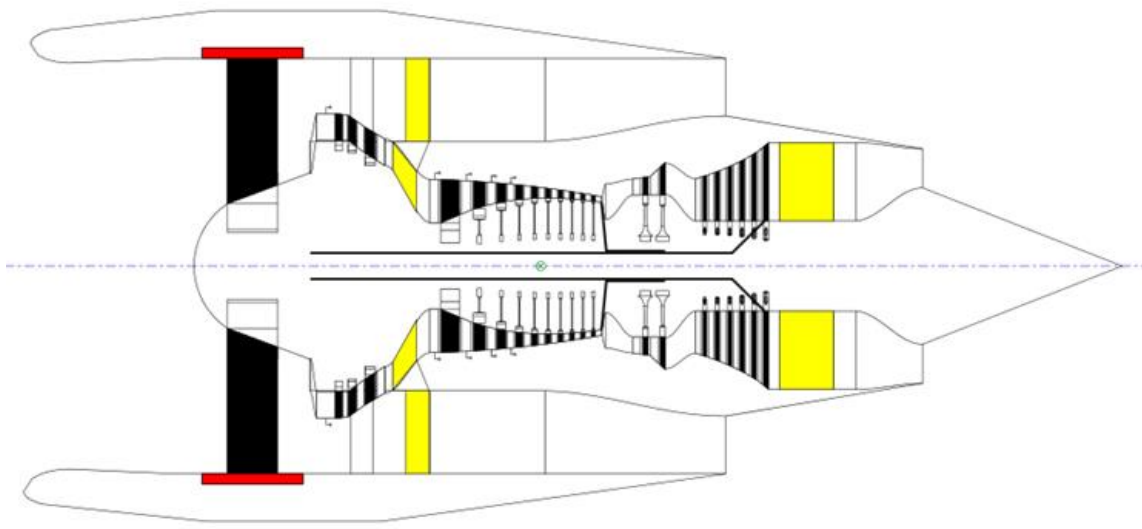


Figure 4.3-6. Example turbofan engine graphical output from WATE++.

The last file included in the run file is called the case file. The case file sets up the constrained engine design problem for the top level solver. First, the design point engine is solved. This is subject to constraints and the values of design variables chosen by the optimizer. In addition, technology assumptions are part of the engine model file (e.g. maximum T_4). Once the engine is designed, the engine performance for all off-design cases is calculated. In practice, this is an unstable process if all off-design cases are done in one case file. Therefore, a Python

file wrapper was written which changes the case file so that NPSS solves for the same design point engine first then solves for one unique off-design case (and any power code cases at the off-design point). The off-design cases are defined in the Python code. As it turns out, limiting the off-design cases to just one for each design point solution makes the solve procedure very stable. Of course, the design point engine does not change from one case to the next so the WATE++ model is only run once to save computational effort.

The engine input deck file (required by the FLOPS mission performance module) at different Mach number and altitude points in the flight envelope is output by the Python wrapper code. As discussed previously, this file is in the rigid format required by the mission performance module. The complete turbojet cycle block diagram shown in Figure 4.3-7 includes the solver constraints, the top level solver, data viewers, and the turbojet model itself (from ambient flow to nozzle exhaust) as one complete NPSS/WATE++ model.

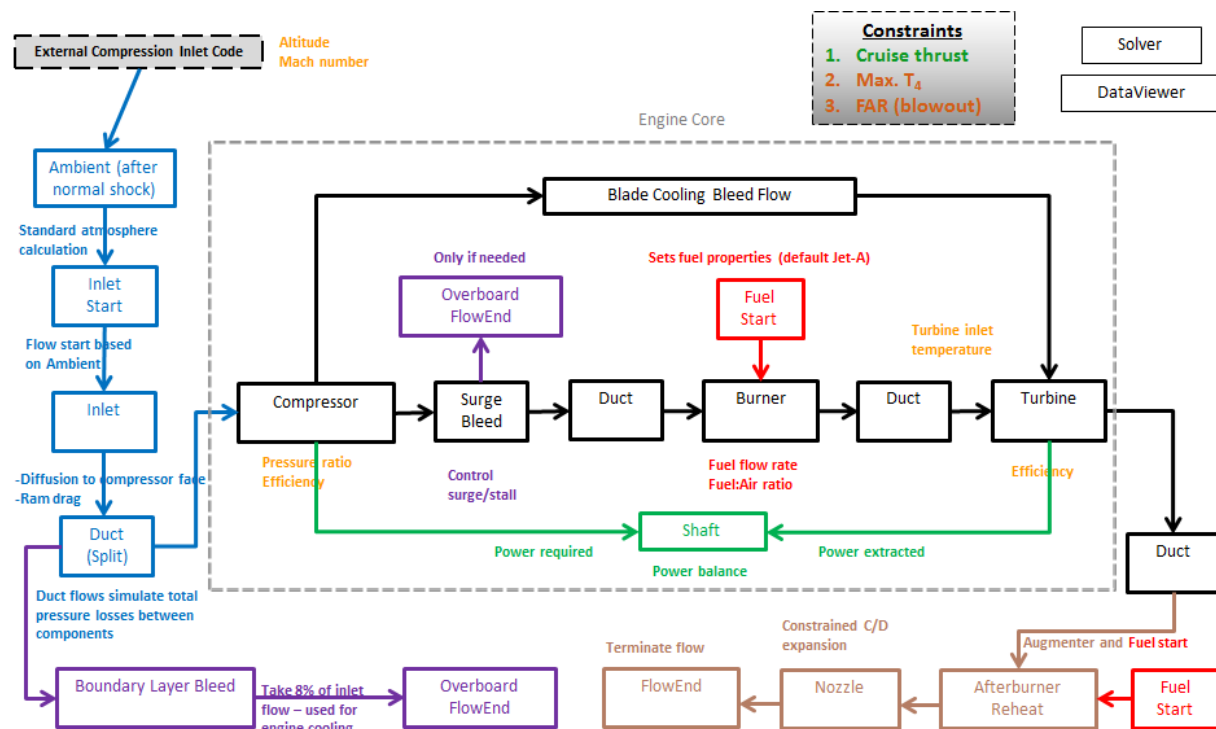


Figure 4.3-7. Block diagram of the NPSS/WATE++ turbojet model.

4.3.3 Low Bypass Turbofan and Variable-Cycle Models

The low bypass turbofan and variable cycle NPSS/WATE++ engine models are both very similar to the turbojet model described in the last section with the same file structure. Additional design variables are added for the bypass flows and for the description of the behavior of the bypass ratio (BPR) over different Mach numbers in the variable cycle model. Of course, the model file is more complex than the turbojet, and the solution must perform a power balance with more than one shaft. A new element called a “FuelSplitter” is added that is only used for the optional afterburner element. A splitter element is used to split the core flow and the bypass flow. A fan, high pressure, and low pressure compressor along with a turbine are new requirements. The block diagram of the low bypass gas turbine engine is shown in Fig. 4.3-8. It is broken up into the front end, core, bypass, and back end sections with the same color coordination as Fig. 4.3-7.

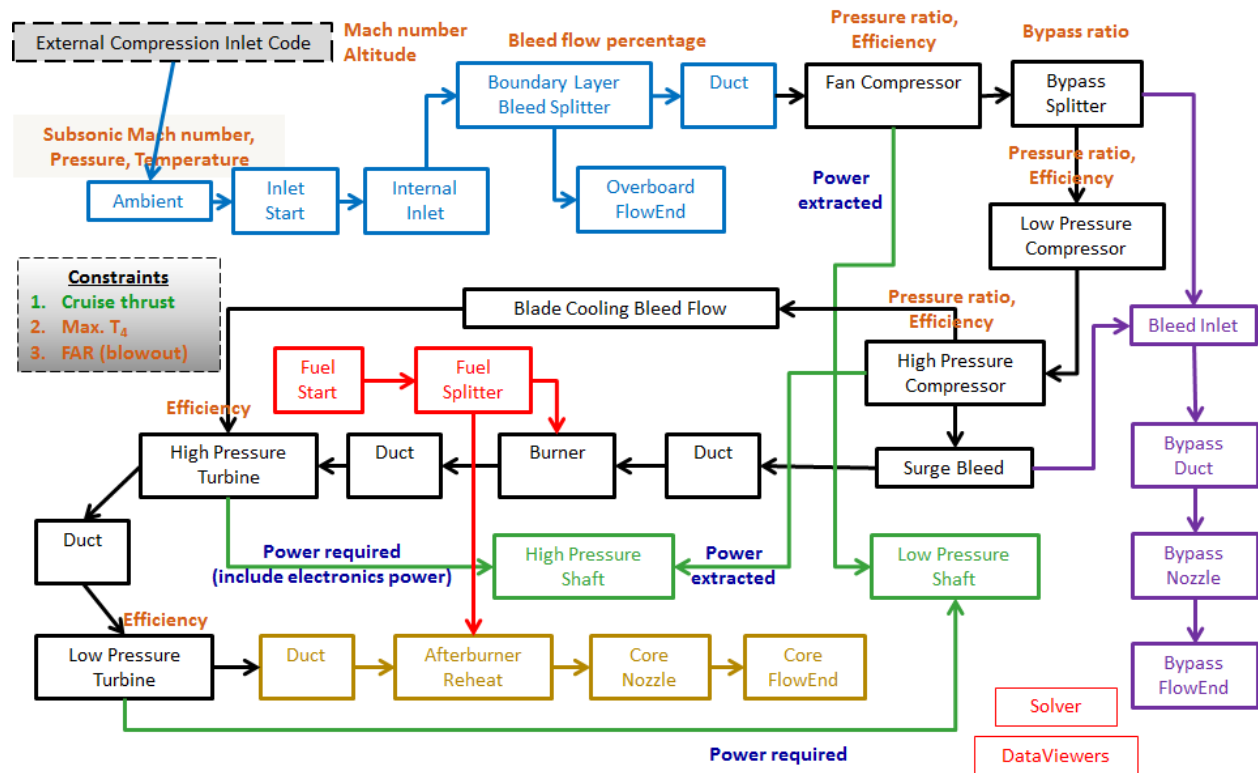


Figure 4.3-8. Block diagram of the NPSS/WATE++ turbofan and variable cycle models.

The low bypass turbofan model and the variable cycle model are essentially the same. However, the Python wrapping code originally used to create the engine input deck for FLOPS input has added design variables and functionality to change the off-design bypass ratio (not the design point geometry like the inlet size) for the variable cycle model. Currently, a linear decrease in the bypass ratio as Mach number increases from a minimum to a maximum value is assumed. The minimum and maximum Mach number values that mark full bypass and no bypass flow paths respectively are design variables, as well as the maximum BPR. For the linear assumption, these three parameters are all that is required for the simple parameterized variable cycle model. It is conceivable that another function may be defined to describe the bypass behavior that is not linear and that also includes altitude effects. This is left as future work.

4.4 Final Propulsion Model

The NPSS/WATE++ model designs a new engine for each run through the ESAV MDO framework shown in the N^2 diagram. Each pass through the framework corresponds to a unique ESAV design. Other methods have been used to integrate these propulsion tools into existing MDO frameworks that involve building a surrogate model [97]. A surrogate model offers execution time advantages during the actual MDO process. For the ESAV conceptual design, the high-fidelity codes that could be linked to NPSS are not required. It is these external high-fidelity codes that cause NPSS to have a large computational cost – not NPSS itself. Since NPSS is not computationally expensive (although it is considered medium-fidelity), and no external high-fidelity codes artificially slow it down in this ESAV MDO framework, NPSS can be run directly within the ESAV MDO framework iterations to design the engine and supply engine data to other disciplines. This mitigates the error that otherwise would creep in by employing a surrogate

model. The generation of the entire engine deck takes on the order of a minute, and is thus acceptable.

4.4.1 Validation of NPSS/WATE++ Model

The NPSS/WATE++ turbojet model was validated by comparing its results to an existing engine. The engine chosen was used on a past supersonic bomber, the Convair B-58 Hustler. This aircraft used the General Electric J79-GE-5 engine. The J79-GE-5 engine is a turbojet with an afterburner. It has a maximum augmented thrust of 15,600 lbf. The compressor has seventeen stages and the turbine has three stages. Both are axial flow components. The afterburner is a short type with a maximum temperature of 4060°R. The length is 202.17 inches and the maximum width is 31.63 inches. It has a dry weight of 3635 lbs. [98].

The optional afterburner feature of the model turned out to be very convenient for the purposes of this validation, since the J79-GE-5 engine happens to have an afterburner. This meant that no new development was needed to model the engine with NPSS/WATE++. The model was constructed with the parameters of the actual engine as described above, including the maximum thrust. This model was run with NPSS/WATE++ and the output was compared to the actual engine characteristics. The results reported in Table 4.4-1 show good agreement, and give confidence in the validity of the NPSS/WATE++ code beyond other validations that occurred during its development, and in the specific engine cycle models constructed for this effort.

Table 4.4-1. Validation Results of NPSS/WATE++ Turbojet Model

Specification	Percent Error
Overall length	< 2%
Maximum diameter	< 1%
Dry weight	< 2%

4.4.2 Summary of Final ESAV MDO Framework Propulsion Model Process

Previously, four propulsion models were analyzed in depth for the purpose of choosing the right model to be used in the ESAV MDO framework (see Appendix B for details). Based on the requirements of such a challenging embedded engine design and computational environment, the state-of-the-art Numerical Propulsion System Simulation (NPSS) model was chosen. NPSS, coupled with WATE++, provide all required functionality. This model is considered a medium-fidelity model and has been validated.

Three parameterized NPSS/WATE++ engine cycle models were constructed. The parameterization was straightforward in the NPSS environment using the flexible object-oriented NPSS programming language and file structure. Parameterization is necessary for optimization studies. In addition, all multidisciplinary framework-required (and more) output data is available and easily obtained from the NPSS/WATE++ models using the NPSS view file feature. The default NPSS/WATE++ propulsion cycle used for ESAV multidisciplinary design is the variable cycle engine. The variable cycle engine combines the best of the turbojet and low bypass turbofan engine throughout the flight envelope.

5.Higher-Fidelity ESAV Multidisciplinary Design Optimization Framework

Within the preceding chapter detailing the selection and development of the ESAV MDO framework's propulsion model, there were hints at an expanded framework than that given as the low-fidelity framework in Fig. 3.1-1. By taking a closer look at the characteristics of the ESAV, certain disciplines are certainly missing from that framework if it is to be properly analyzed. This chapter expands the low-fidelity framework into the final, or higher-fidelity ESAV MDO framework, and gives more details into the discipline models which make it up.

5.1 Expanded N² Diagram

The N² diagram is again invoked in order to represent the expanded, higher-fidelity ESAV MDO framework. The disciplines that remain the same are the VT-CST geometry model, weights, mission performance, take-off & landing, noise, constraints, and optimization. Justification for not changing them, plus justification for changing or adding the other disciplines will be given in the following sections. Table 5.1-1 below shows the unchanged and changed disciplines from the low-fidelity ESAV MDO framework to the higher-fidelity ESAV MDO framework. The N² diagram of the framework is shown in Fig. 5.1-1.

Table 5.1-1. Overview of Changes from the Low-Fidelity to the Higher-Fidelity N² Diagram

Discipline	Changes	Discipline	Changes
Propulsion	New Model	Mission Performance	No Changes
Geometry	No Changes	Noise	No Changes
Aerodynamics	New Model	Radar Cross Section	Added
Engine Exhaust	Added	Constraints	No Changes
Washed Structures		Optimization	No Changes
Weights/Structures	No Changes		

Expanded ESAV N² Diagram

Propulsion	Flow behind inlet shocks	Flow-through panels' data	Temp., Press., Alt., M, Dimen.	Engine weight	Engine weight	Engine data in flight envelope	Exhaust speed & temp., Noz. Dim.			Thrust, Altitude, Mach #, BPR, etc.
Cowl and Inlet	Geometry	Configuration, Mach #, Alt.	Cowl, Aft deck	Configuration	Tank and engine	Wing area	Noise shielding factor	Configuration	Configuration, Avail. fuel	Configuration
		Aerodynamics	Skin temp., Loading			Aero. data in flight envelope				
			EEWS	EEWS weight	Structural weight - EEWS					
				Structures	Structural weight - other					
					Weights	Aircraft weight in flight envel.				TOGW
						Mission Performance	Detailed take-off parameters		Req. fuel volume	Feasibility
							Noise		Noise output	
								Radar Cross Section	RCS metric	
									Constraints	Feasibility
Thrust, Altitude, Mach #, BPR, etc.	Configuration									Optimization

Figure 5.1-1. The higher-fidelity N² diagram with added and changed disciplines from the low-fidelity version of the N² diagram.

The geometry model is still the VT-CST model. This shows the versatility and power of this geometry representation as it can provide input to multi-fidelity discipline models, all with the same fast-running code. There is a stability and control model in development [99], at the time of writing, which is intended to handle the difficult lateral-directional control issues at low-speed flight given the tailless design, as well as all other normal static and dynamic stability and control issues. A higher-fidelity structures model is also in development, at the time of writing, to improve the structural weight estimate. However, the similar aircraft in the well-vetted FLOPS structural weight database are expected to provide suitable structural weight estimates for the ESAV. The sensitivity of the wing structural weight is examined with the framework in an ensuing section.

Figure 5.1-2 shows the SR-71 military supersonic aircraft, and the two differences that require higher-fidelity models for at least propulsion and stability and control. These are the ESAV embedded engines and the tailless design – both for LO reasons.



Figure 5.1-2. Design challenges of an ESAV compared to the SR-71 supersonic aircraft.

5.2 Propulsion

The ESAV MDO framework has two available propulsion models. The first is the propulsion module that is part of NASA's Flight Optimization System (FLOPS) software [52] with its weight surrogate as discussed previously.

Due to the criticality of the propulsion analysis for a supersonic aircraft, the principal propulsion analysis in the higher-fidelity framework is the state-of-the-art NPSS/WATE++ software. NPSS, described in great detail previously in Chapter 4, was developed by a collaborative partnership between NASA Glenn, the U.S. aeropropulsion industry, and the U.S. Department of Defense [93]. WATE++ was integrated into NPSS by NASA researchers.

The higher-fidelity ESAV MDO framework has three different parameterized NPSS engine cycle models available for use in optimization studies. The first is a simple, single-spool, turbojet engine, the second is a low-bypass, two-spool engine, and the third is a variable-cycle engine. The variable-cycle engine model is the default in the framework because of its increased efficiency over the three flight regimes encountered in an ESAV mission over the other two fixed-geometry engines.

The combined NPSS/WATE++ propulsion model is considered a medium-fidelity model and runs relatively quickly for this level of fidelity, on the order of a minute. This run time is acceptable for the ESAV conceptual design level studies employing the model.

Embedded engines employ a serpentine inlet, or S-bend duct. These inlets are known to cause flow distortion and swirl. These undesirable flow effects can lead to degraded compressor stability, compressor surge, and overall engine performance losses [100]. Within the ESAV MDO framework, a penalty is imposed that acknowledges the existence of the serpentine inlet before the compressor, but can be applied without the involved and expensive CFD analyses

usually employed to determine the complicated duct flow [96] [95]. One promising method for controlling the flow in the serpentine inlet is boundary layer suction [94] [101]. Therefore, in order to assess a penalty for the serpentine inlet, a significant boundary layer bleed flow was taken from the inlet's core stream in the NPSS model of the engine as well as imposing a total pressure loss through the inlet.

The weight of the serpentine inlet was modeled by adding an extra-long standard inlet onto the WATE++ analysis, due to the fact that a longer serpentine inlet is less prone to flow disturbances [95]. This WATE++ inlet model does not affect the calculation of the flow into the engine, but merely acts on the weight model. This allows the core of the engine to be calculated correctly by NPSS, while extra weight expected by the serpentine inlet is added at the correct location on the engine, which has the beneficial effect of moving the total propulsion CG in the right direction. A picture of the WATE++ model of a low-bypass engine cycle with the extended inlet is shown in Fig. 5.2-1.

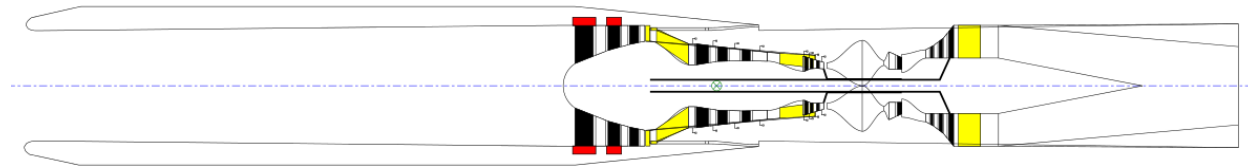


Figure 5.2-1. The weight model of an example low-bypass engine with extended inlet.

The propulsion model is integrated into the ESAV MDO framework via the developed NPSS/WATE++ model, a local optimization, and a convergence loop. The model analysis designs the engine and writes an engine input deck over the flight envelope that is suitable for input to the FLOPS mission performance module. Next, the local optimization determines the three best angles of the four-shock inlet to minimize the total pressure loss for the design-point engine. Finally, the inlet and cowl dimensions are determined for the design-point engine through iteration between the geometry model and the cowl entrance area. This iteration

converges to the correct engine air flow solution demanded by the designed engine. Once converged, the propulsion subsystem is fully defined and able to capture the effects of the propulsion system on the larger ESAV aircraft system. Many detailed parameters are output from the NPSS/WATE++ propulsion model, but only those required by other MDO framework models are placed into the design space.

5.3 Aerodynamics

Initial choices for aerodynamic software used in the low-fidelity ESAV MDO framework focused on simplicity of implementation and minimizing run times. A suite of legacy codes was pieced together to provide the necessary lift and drag coefficients at both subsonic and supersonic Mach numbers. The speed and simplicity of the initial analysis package helped establish the data input/output relationships within the MDO framework, but its limitations quickly began to inhibit other disciplines. None of these low-fidelity codes was capable of producing aerodynamic results for any configuration other than a clean wing. That is, there could be no flaps, no control surfaces nor any stability derivatives output. This was first noticed in the mission analysis module where ESAV designs regularly failed to complete the climb segment of the mission due to the unnecessarily large aircraft angles of attack that needed to be achieved to generate sufficient lift. Then, as the stability and control module began development [99], the absence of aerodynamic derivatives made stability analyses impossible. For these reasons, another aerodynamic tool was deemed necessary.

The ZONAIR [102] aerodynamics solver was chosen as a higher-fidelity solution to the deficiencies encountered from the lower-fidelity legacy aerodynamics codes. ZONAIR provides desired capability not available in the lower-fidelity legacy aerodynamic model, like high-lift devices and stability derivatives. In addition to ZONAIR software, Zona Technologies produces

many different analysis packages, each with a particular focus and varying levels of fidelity. ZONAIR is a panel method and was chosen as a medium-fidelity analysis tool for its relative speed of computation and capability. With both subsonic and supersonic panel methods, ZONAIR produces all necessary stability and control derivatives, computes a three-dimensional load distribution, and offers an interface with finite element method software to include static aeroelastic effects. Unlike other panel codes that make use of thickness approximations, ZONAIR actually panels the true aircraft outer mold line (OML) generated by the geometry model, at the cost of longer run times. This is appealing from an aerodynamics computational accuracy standpoint, which will be shown to be important later.

Using a purchased special educational license from Zona, and the Message Passing Interface (MPI) standard, this higher-order panel method does indeed become tractable in conceptual aircraft multidisciplinary design optimization. The special license allows as many concurrent ZONAIR analyses as desired on one machine. However, in order to construct the aerodynamic drag tables required by the mission performance model, these concurrent analyses must be organized and controlled. MPI was used to provide the parallel structure needed to construct the formatted drag tables with ZONAIR. An eight-core desktop machine was purchased in order to be able to run eight concurrent Mach number cases and a parallel MPI code was written to execute Python scripts in parallel. The Python scripts contain the logic to produce the required drag tables for the mission performance module.

A performance gain can be had by organizing the ZONAIR analyses by Mach number since the Aerodynamic Influence Coefficient (AIC) matrix for a given configuration and Mach number need only be calculated once. After the initial calculation of the AIC, it can be reused for any subsequent altitude and angle-of-attack combinations, which is all that is conveniently

required to be changed to compute the values needed for the drag tables. The computation of the AIC is the most expensive part of the ZONAIR panel method. Thus, ZONAIR has been made more tractable by taking advantage of the AIC re-use strategy employed in the ZONAIR parallel framework model. This model can fill up the required drag tables on the order of ten to twenty minutes. Of course, this is highly dependent on the chosen paneling density.

A computational study was conducted once the ZONAIR model was completed. This study varied the panel density of the same simple wing-like aircraft configuration to find the best paneling density. The figures of merit used to choose were both the accuracy of the results and the time it took to obtain the results. The baseline for the correct solution was the 50x50 mesh results. Figure 5.3-1 shows the relationship between the CPU time and the panel density. Figure 5.3-2 shows the accuracy of the computed C_D and C_L results over the various meshsizes.

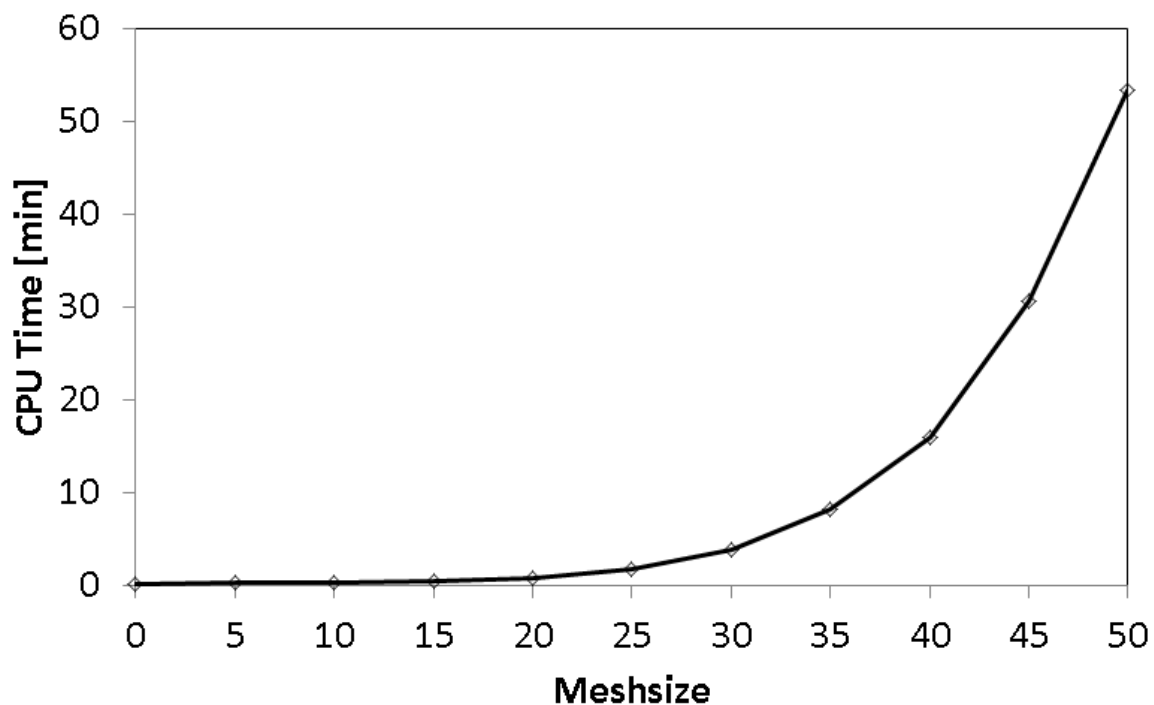


Figure 5.3-1. ZONAIR CPU time over the nxn meshsize or paneling density for one run.

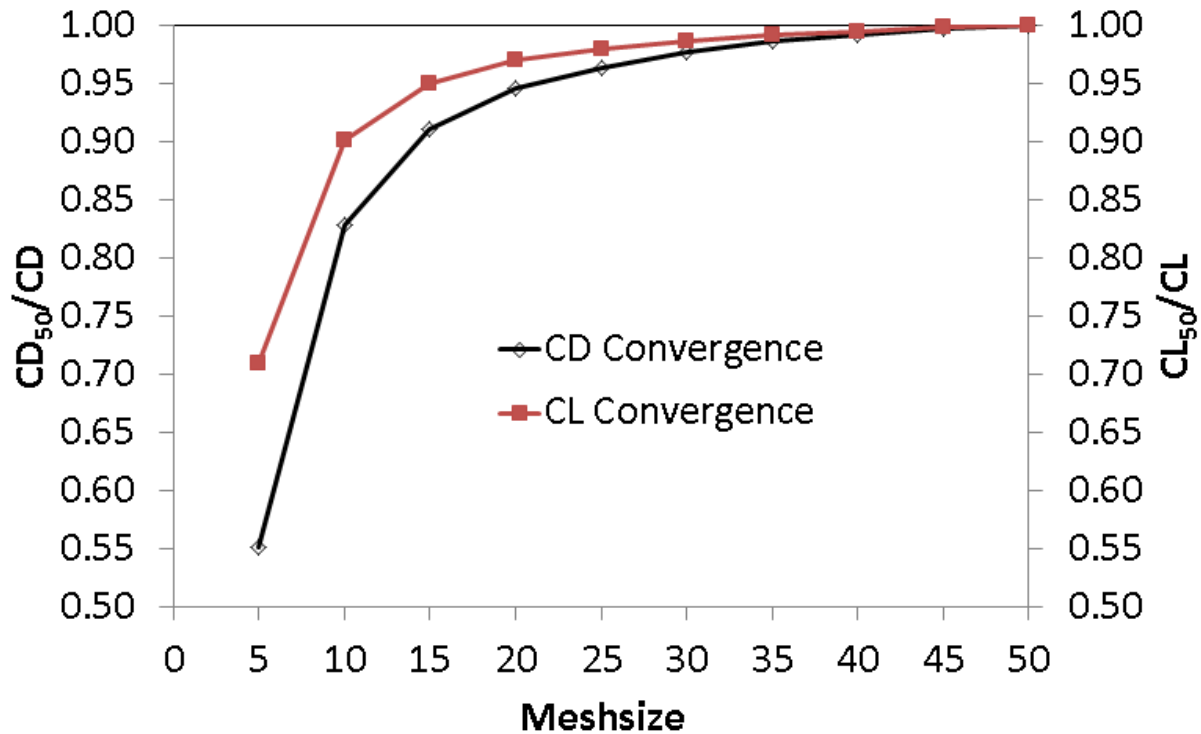


Figure 5.3-2. ZONAIR accuracy (based on assumption that 50x50 meshsize produces the correct results) over the nxn meshsize or paneling density for one run.

The desired time for one simulation of one Mach number, altitude, and angle of attack case was less than one minute. For conceptual design, the desired accuracy was to have an error less than 5% in both C_D and C_L . Examining the results of the parametric study in these two figures shows that to meet both the time and accuracy requirements, the meshsize should be about 20x20. In practice, the extra communication overhead imposed from the parallelization using MPI, and the extra iterations required in order to fit the ZONAIR output to the drag table format requires a slightly smaller meshsize of 18x18. This smaller meshsize does not reduce the accuracy beyond a reasonable level and allows the ZONAIR computation to proceed in a time requisite for the MDO framework.

ZONAIR has been implemented into the ESAV MDO framework. It is fully automated, with the geometry model supplying the automatic panel generation, to obtain the desired aerodynamic results. A sample ZONAIR configuration is shown in Fig. 5.3-3. This example ZONAIR aerodynamic solution is a Mach 1.6 case that was part of a larger B-58 validation case for the whole ESAV framework discussed later.

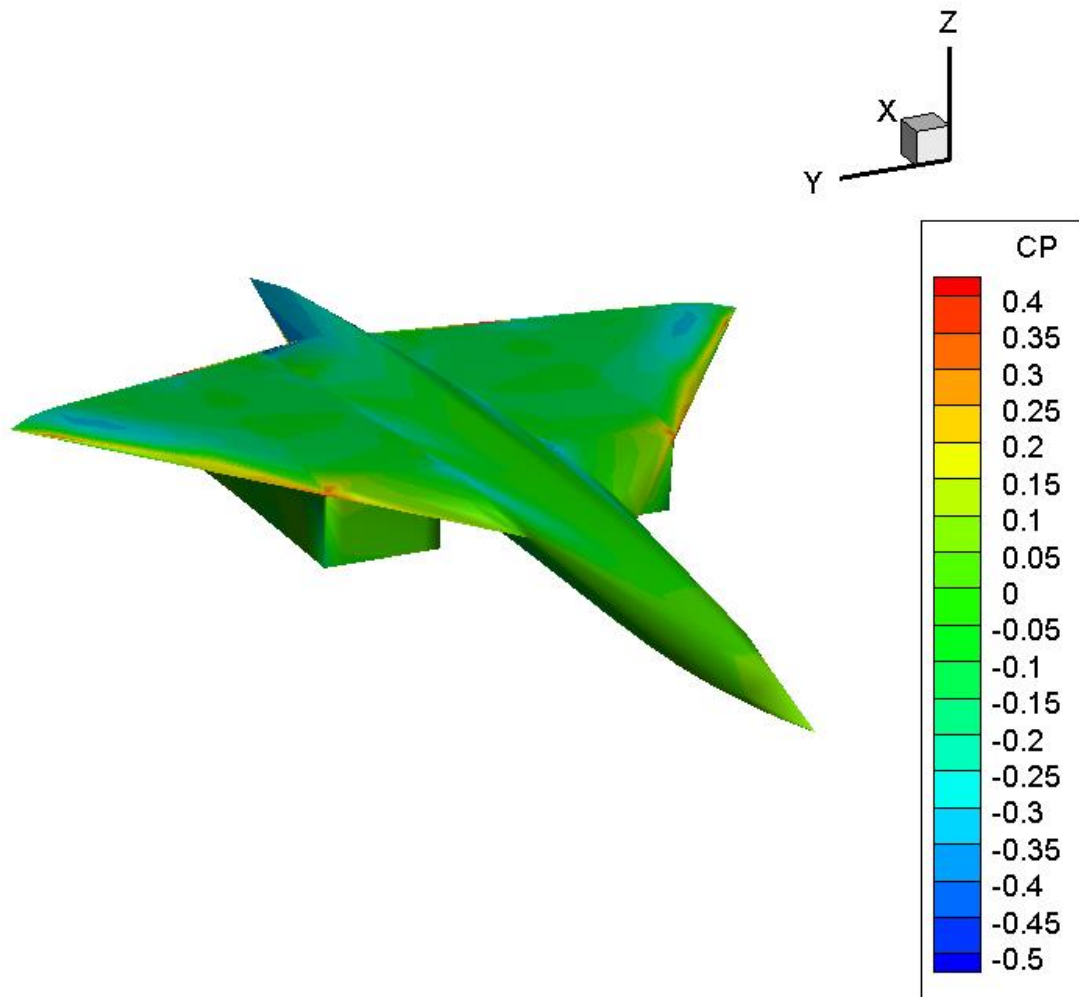


Figure 5.3-3. ZONAIR solution example, showing the coefficient of pressure results.

5.4 Engine Exhaust-Washed Structures

Engine exhaust-washed structures (EEWS) are structural components located aft of the ESAV embedded engines (and other embedded engine aircraft), and are therefore situated in the hot

exhaust flow path from the engines. The structural components include a nozzle and all supporting substructure. EEWS are subjected to a severe loading environment from the high temperatures and other mechanical loading. ESAV design requirements related to low-observability place constraints on the EEWS that supersede traditional design practices for structural design in a thermal environment [103]. A notional reference EEWS configuration is shown in Fig. 5.4-1.

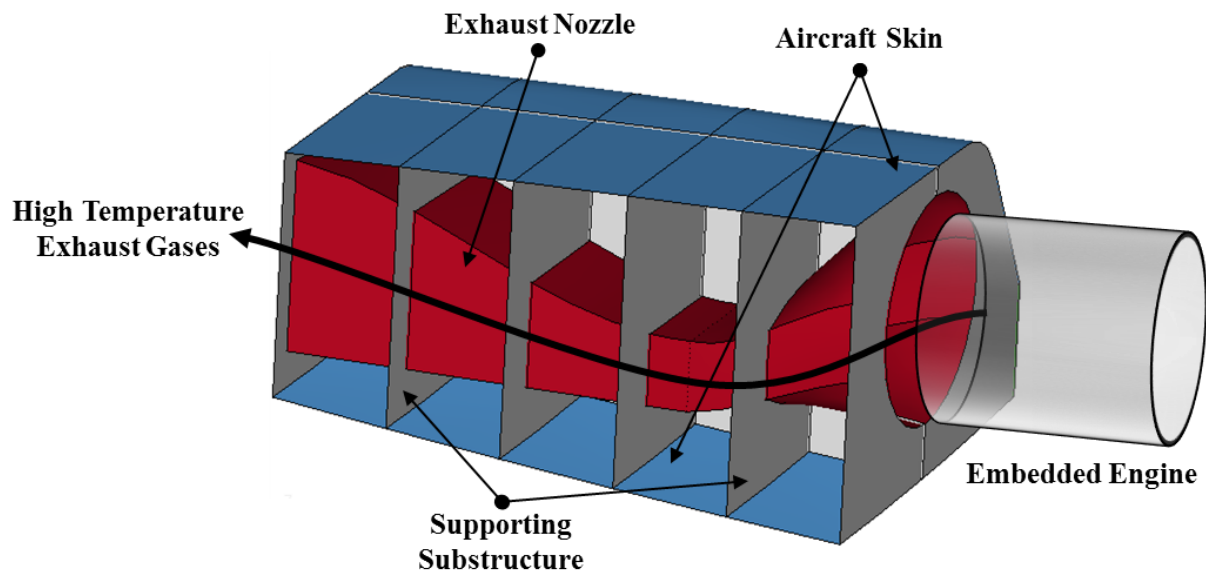


Figure 5.4-1. Notional reference EEWS design with individual components and primary loading – flow goes from the engine on the right to ambient conditions on the left.

As may be expected, the analysis required to model this intricate problem is very computationally expensive, putting this type of analysis beyond the scope of conceptual design and inclusion of the full analysis in the MDO framework. However, this effect has been deemed too important to ignore from the very expensive lesson-learned on the B-2 bomber program [103]. Therefore, the framework includes EEWS by using a response surface model approximation of the more exact direct simulation. This EEWS model was developed by Deaton [104].

EEWS analysis must include both heat transfer and structural analysis because the loading is dependent on both. Therefore, the approach to the EEWS optimization problem is to use a thermal/structural optimization framework previously developed to account for the design dependency of thermal loads in structural optimization [105]. Boundary conditions are related to inputs from the propulsion model and mission design variables. To maintain flexibility in the overall aircraft design space by not subjectively limiting its size based on preconceived notions, a large design space for the EEWS must be correctly expressed in a surrogate model. The actual EEWS analysis is a finite element analyses (FEA) and a structural optimization to find the minimum EEWS weight for the given boundary conditions. The surrogate EEWS model was created in a standard fashion by fitting a second-degree polynomial to the optimal EEWS weight responses over the defined design space.

The baseline EEWS model is similar to the one shown in Fig. 5.4-1. This baseline model is parameterized with the input variables, which include the temperature and pressure of the engine exhaust, the nozzle area, the Mach number, and the cruise altitude. Upper and lower bounds are imposed on these parameters based on engineering experience and insight. A DOE selected points throughout the design space for the high-fidelity EEWS optimizations. The constraints that are included as parts of the optimization problem are the stresses and bifurcation buckling. The minimum EEWS weight at each of these points is the output of interest. This process is summarized in Figure 5.4-2.

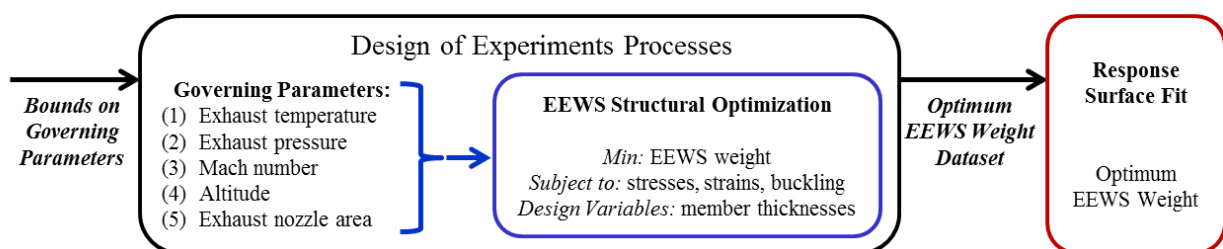


Figure 5.4-2. Process utilized to develop EEWS surrogate weight model from Ref. [104].

By simply running the surrogate model, which was coded up in C++, the EEWS can rapidly (less than a second) provide the required EEWS weight. This weight is then passed to the FLOPS weight module. This surrogate model allows for the basic, first-order effects of the important EEWS analysis at the conceptual design level of the ESAV aircraft. By including the first-order EEWS effects, the ESAV design should not have any major weight surprises as the conceptual surrogate model lends confidence to a reliable EEWS weight estimation early on. This surrogate also helps to more realistically shape the ESAV design space by not excluding the EEWS effects.

5.5 Radar Cross Section Estimation

An important consideration in the design of any LO air vehicle is its radar cross section (RCS). An ESAV is designed to be an advanced military bomber concept vehicle, therefore radar stealth is a critical metric for its final optimal configuration. In fact, the assumptions which make the design of an ESAV so difficult, being tailless and having embedded engines are mainly for low-observable reasons.

The measure of whether one configuration has a lower radar cross section as compared to another is an important consideration. However, the availability of RCS codes is very sparse, given the expected export and proprietary restrictions on such a code. Providently, freely-available Matlab software [106], developed at the Naval Postgraduate School [107] offers a way to analyze the RCS of a particular aircraft configuration, and it has, therefore, been integrated into the higher-fidelity ESAV MDO framework.

The basic radar range equation [108] is

$$R_{max.}^4 = \frac{P_t G^2 \lambda^2 \sigma}{(4\pi)^3 P_{min.}} \quad (37)$$

where, $R_{max.}$ is the maximum detection range, P_t is the power transmitted, G is the gain of the radar antenna, λ is the radar wavelength, $P_{min.}$ is the smallest power level that can be detected by the radar antenna given a signal-to-noise ratio, and σ is the aircraft RCS that is the ratio of reflected power per unit solid angle to the incident power per 4π . Assuming the radar has fixed characteristics, then the relationship between the maximum detection range and the aircraft RCS becomes simply

$$R_{max.} = R_0 \sigma^{\frac{1}{4}} \quad (38)$$

where R_0 is the maximum detection range for a 1 m^2 perfectly conducting object. Equation 38 can be used to understand a RCS rule of thumb, that is, for every 12 dBsm of RCS increase ($\text{dBsm} \equiv 10 \log_{10} \left[\frac{\sigma}{1 \text{ m}^2} \right]$), $R_{max.}$ will double. Conversely, every 12 dBsm reduction in RCS means $R_{max.}$ will be halved [108]. The smaller the detection range the better for the aircraft survivability.

Detailed design is very important to the RCS signature of the final aircraft [109] because small details can affect the RCS significantly, but this is beyond the scope of the current early-design-stage effort. As far as mission profiles go, the higher and faster an aircraft flies the higher the survivability [108]. Fortunately, both of these work in the ESAV's favor because of its optimal cruise altitude being much higher than subsonic aircraft, and its fast supersonic cruise.

For conceptual and preliminary design, the two main tools available to reduce the RCS of an aircraft are the OML shaping design and the use of radar absorbing material (RAM). An earlier work lists several design guidelines for shaping the aircraft: minimizing the size of the aircraft, placing edge returns away from threat sectors, and causing unavoidable spikes from body lines to be aligned in the same direction, to name but a few [108]. The POFACETS code,

along with the VT-CST geometry code, allows the techniques of shaping and RAM analysis to be explored in the MDO framework.

The original POFACETS code is based on a Graphical User Interface (GUI) and is, therefore, not suitable for easy automation and integration into the ESAV MDO framework. In addition, Matlab is a script language and does not execute as quickly as a compiled code. To address these challenges, a compiled language code, C++, was used to rewrite the monostatic RCS part of the original POFACETS Matlab code so that the GUI was eliminated and the execution speed was increased. Using the default perfect electric conductor as the aircraft skin material, the C++ RCS output of a full notional ESAV configuration was compared to the original Matlab POFACETS code with excellent agreement, as shown in Fig. 5.5-1.

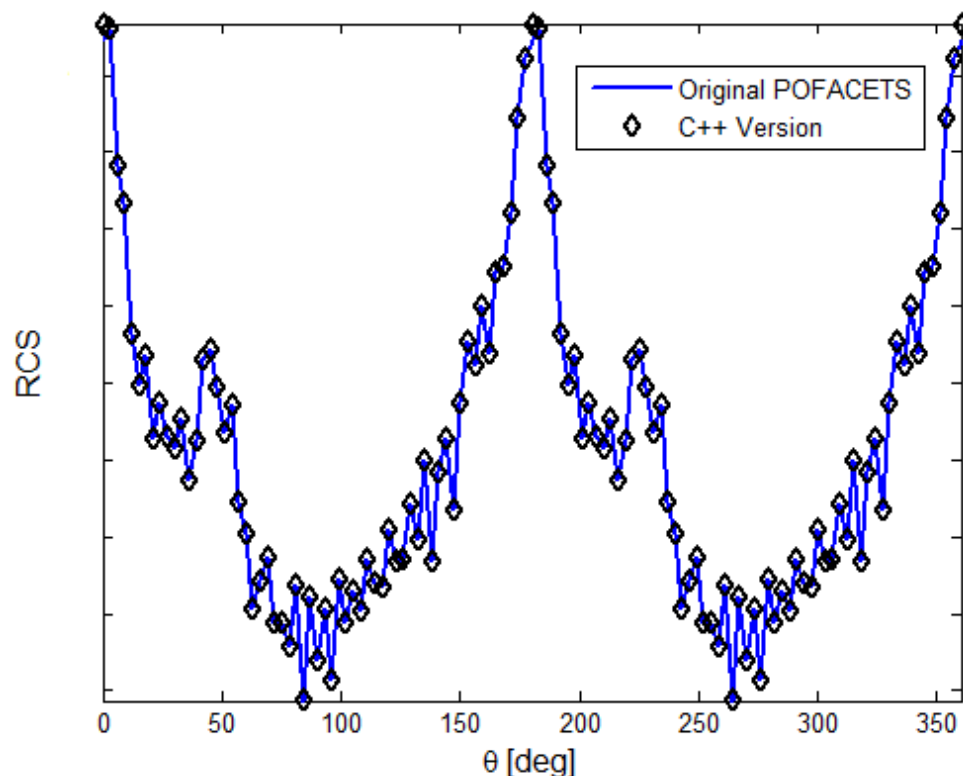


Figure 5.5-1. Comparison of the original Matlab POFACETS code RCS results and C++ version without RAM.

To include a more realistic material than a perfect electrical conductor to address the RAM method of RCS reduction, the material polyphenylene oxide was chosen from the POFACETS database as a representative generic radar absorbing material. This particular material has a specific gravity close to 1.0. This was then likewise transferred to the C++ RCS code, and another comparison between the original Matlab code and the C++ code with the new material was done. A thin film of this material is assumed to cover the entire outer skin of the ESAV for the RCS analysis. The results again show good agreement as seen in Fig. 5.5-2.

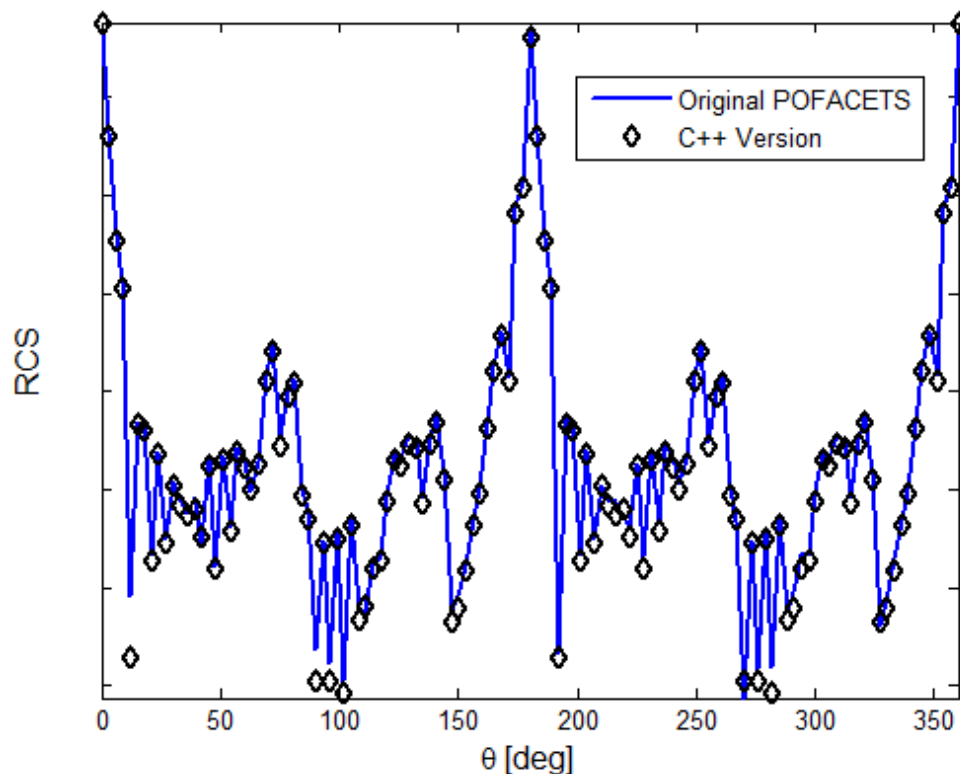


Figure 5.5-2. Comparison of the original Matlab POFACETS code RCS results and C++ version with RAM.

The only discrepancies are at extremely small RCS values, and are most likely due to differences in real number representations between Matlab and C++. The IEEE standard for

floating-point arithmetic (IEEE 754) tolerates error in the last bit for transcendental functions [110], which are used heavily in the RCS computations.

No weight penalty was explicitly assigned to the RAM material because of future technology development considerations. An assumption of future RAM material that can double as structural composite material is made. This assumption implies that such material will be available for aircraft construction when the ESAV is actually built. Polyphenylene oxide itself, as a specific RAM, is only a placeholder used as a vehicle to compare one ESAV RCS to another, but no assumption that an ESAV will actually use this particular RAM is made here. Therefore, since the RAM is assumed to be the same as the structural skin material, no weight penalty is necessary.

5.6 Validation of Analysis Codes in Higher-Fidelity Framework by Comparison with the Convair B-58 Hustler Supersonic Bomber

In order to validate all of the analysis codes together in the higher-fidelity ESAV MDO framework, an existing legacy aircraft was chosen that had a similar mission as that proposed for the ESAV. The numbers of aircraft that have been manufactured as long-range supersonic bombers are few. The aircraft chosen for validation was the Convair B-58 Hustler Mach 2 supersonic bomber. The B-58 had four, under-wing J79-GE-5B engines and a large vertical tail. A picture of this aircraft is given in Fig. 5.6-1.

A NPSS/WATE++ afterburning turbojet J79-GE-5B model was constructed to model the propulsion system using documented engine characteristics [98]. A separate validation of the NPSS/WATE++ propulsion model was done at the same time. There was very good agreement between the results of the model and the real engine as was shown in Table 4.4-1.



Figure 5.6-1. Convair B-58 Hustler aircraft in flight.

In order to allow for the simulation of the B-58 in the framework analysis models, the geometry model needed to be able to represent the B-58 geometry. To this end, the VT-CST geometry model was enhanced to allow for engine cowls under the wing. In addition, since the B-58 employs conical camber in the wings as well as having specific NACA airfoil shapes in the wing cross section, the VT-CST geometry model needed to be enhanced to model these characteristics. In effect, the geometry model was needed to work backwards from its intended direction in that an existing geometry needed to be represented. A notional conical camber design in a wing is shown in Fig. 5.6-2 from Reference [111] by progressing airfoil cross sections from the wing root to wing tip.

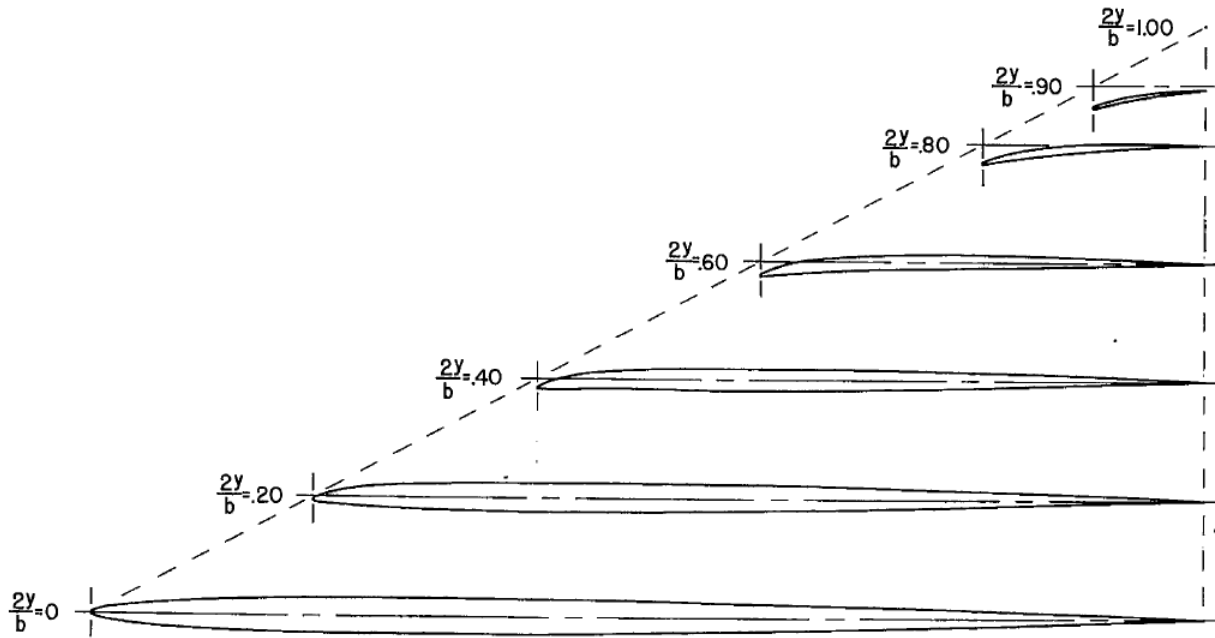


Figure 5.6-2. Representation of conical camber by chord slices over a wing with the conical camber effects becoming more evident closer to the wing tip.

The power of the VT-CST geometry model was tapped to match the existing three-dimensional wing shape of the original B-58 bomber. The planform shape was a matter of changing the span, sweep angle, root chord, and tip chord to match the B-58 wing. The three-dimensional (3D) shape matching, however, is done with handles as explained previously during the VT-CST geometry model discussion. The 3D matching process began with the given B-58 NACA root and tip airfoils and the conical camber profile at several different span stations. The conical camber dimensions were found in an early wind tunnel test paper for a scale model of the B-58 [112]. These coordinates were nondimensionalized for use in the full-scale geometry model. The B-58 wing root and tip airfoil sections were also found [112] [113] and are

- wing root → NACA 0003.46-64.069 and
- wing tip → NACA 0004.08-63.

These airfoils are illustrated along with the 3D shape handles in Fig. 5.6-3.

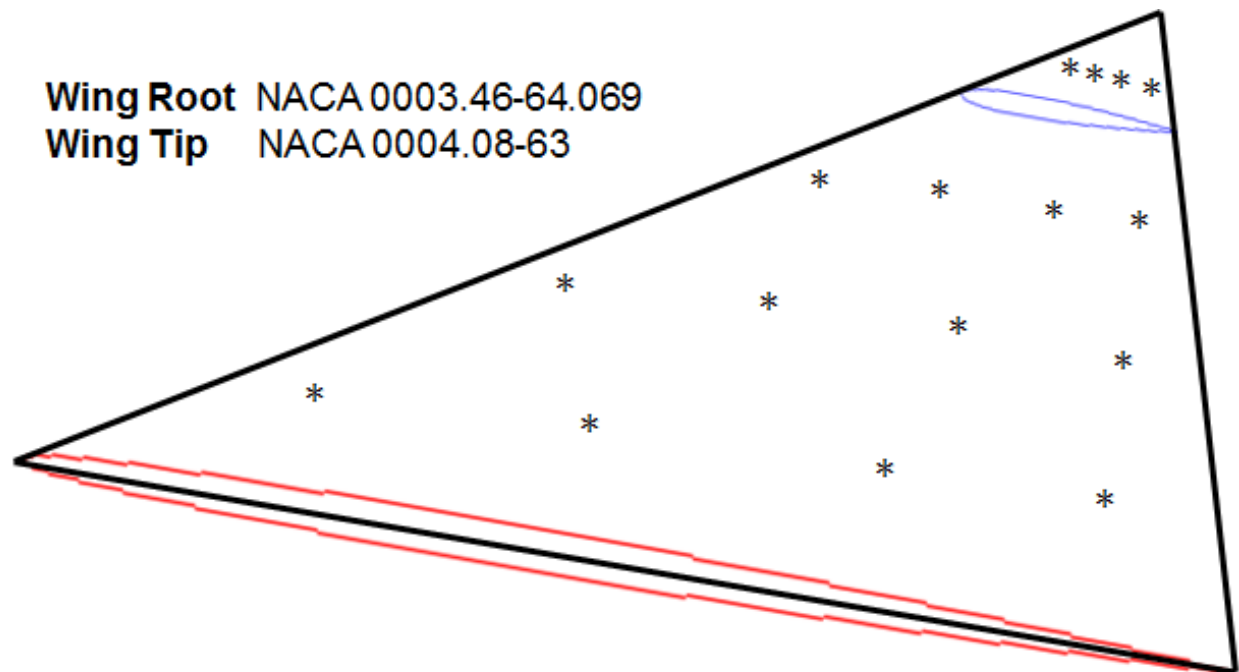


Figure 5.6-3. B-58 wing showing handles as “ * ” points, the root airfoil, and the tip airfoil.

New functionality to define the conical camber was added to the VT-CST geometry model by the addition of two angle parameters. The known airfoil sections and conical camber geometry were then used as constraints in an optimization process in order to match the VT-CST geometry model definition of the wing to the actual wing. The design variables in this matching optimization problem were the wing shaping handles, and the objective function was the difference (error) between the geometry model shape and the actual shape. The VT-CST geometry representation of the B-58 geometry after these model enhancements matched very closely with the actual B-58 geometry. The optimization process was able to find the wing 3D shape handles so that the final optimum wing OML varied smoothly between the defined root and tip chord airfoil shapes. The final VT-CST geometry model representation of the B-58 is shown in Fig. 5.6-4 (note that the conical camber can be observed in the wing shape).

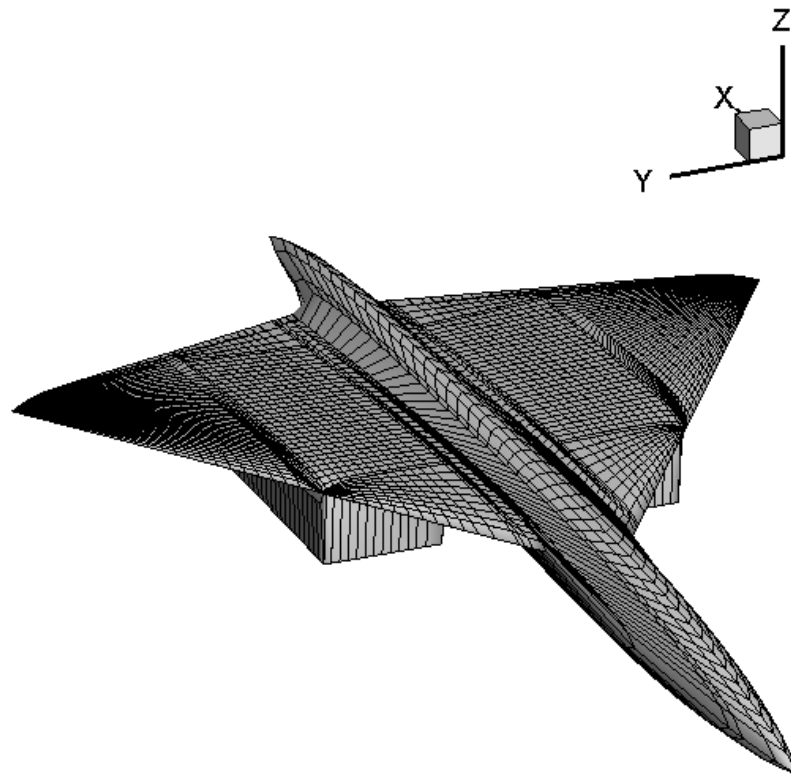


Figure 5.6-4. The B-58 VT-CST model representation found from matching the root and tip airfoils and conical camber geometry of actual aircraft.

This process has also been applied to other shapes like the F-5 aircraft wing as shown in Fig. 5.6-5. Again, something must be known about the planform dimensions, and at least two airfoil cross sections. Then, the geometry handles for the 3D shaping are manipulated by an optimizer to match the VT-CST geometry representation to the known shape. Finally, the smoothness of the surface is checked over the whole wing through the VT-CST drawing functions. Once defined, the VT-CST geometry model can output the final matched shape into a variety of geometrical representations for the discipline analyses within the MDO framework.

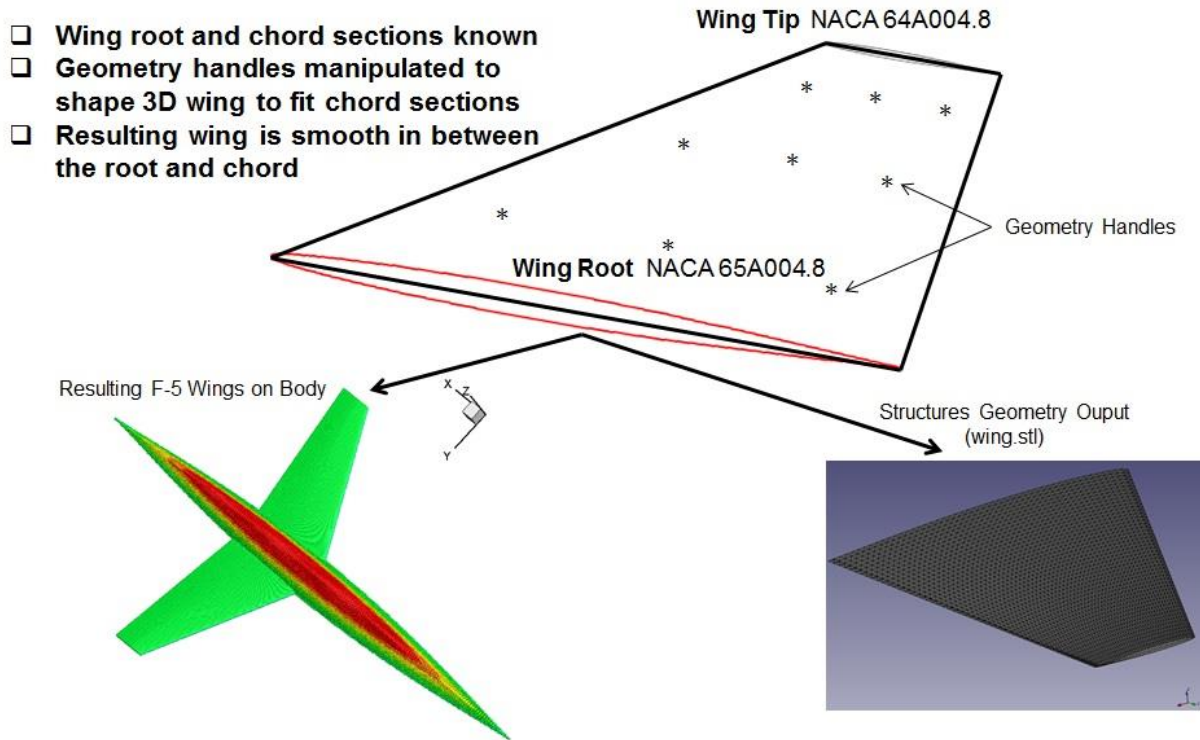


Figure 5.6-5. F-5 wing geometry matching process and output using VT-CST.

With the correct B-58 geometry the validation case study could then be performed. There was no need for the EEWS analysis (i.e. there are no embedded engines on the B-58), and no RCS comparison can be performed with the B-58, so these were omitted from the validation analysis. The FLOPS weight and mission performance modules were used.

The mission profile chosen for the mission performance FLOPS module was a demanding 1750 n.mi. combat radius broken up into three cruise segments. These segments, in order from start to finish, are

1. taxi and take-off,
2. climb,
3. subsonic cruise at $M = 0.78$ for 1167 n.mi.,
4. accelerate to $M = 2.0$
5. supersonic cruise at $M = 2.0$ for 1166 n.mi.,

6. decelerate to $M = 0.78$
7. subsonic cruise at $M = 0.78$ for 1167 n.mi.,
8. hold for 20 min.,
9. land, and
10. taxi.

Lower altitude subsonic cruise segments act as bookends to the high altitude supersonic cruise because of the B-58 requirements for low-altitude bombing approaches due to surface-to-air missile threats. There is no B-58 published mission profile known to the author beyond a general combat radius, and therefore this profile was chosen with the intention to find the peripheral weight design. There was no release of the fixed-weight stores assumed during the mission, which makes this analysis the worst-case mission in regards to weight. The weights output from the analysis are, therefore, expected to be near the maximum published design weights for the B-58.

The aerodynamics module used for the validation was the medium-fidelity ZONAIR code. It may be noticed that there is no vertical tail on the B-58 geometry representation (see Fig. 5.6-4), while the actual B-58 has a large tail (see Fig. 5.6-1). This is due to the assumption during the development of VT-CST that an ESAV would be tailless, therefore a vertical tail function was not required to be included in VT-CST. However, the CST method is versatile enough that the ability to model a vertical tail could be included if needed in the future.

In order to improve the validation comparison, a stand-alone wing was constructed in VT-CST that matched the dimensions of the B-58 vertical tail. This tail representation then employed the VT-CST functionality to automatically panel it for a ZONAIR analysis. The ZONAIR panel output was used to build a surrogate model of the tail drag coefficient. The

altitude and Mach number were parametrically changed, and the drag coefficient at each case was recorded. This data was then used to construct a table-lookup surrogate model. This surrogate is able to add the tail drag to the drag of the rest of the aircraft computed by ZONAIR. The FLOPS weight module was set up so that the weight of the vertical tail was included even though it was not present in the geometry model. A single case of a tail aerodynamics run that was just one of the many required to build the surrogate model is shown in Fig. 5.6-6.

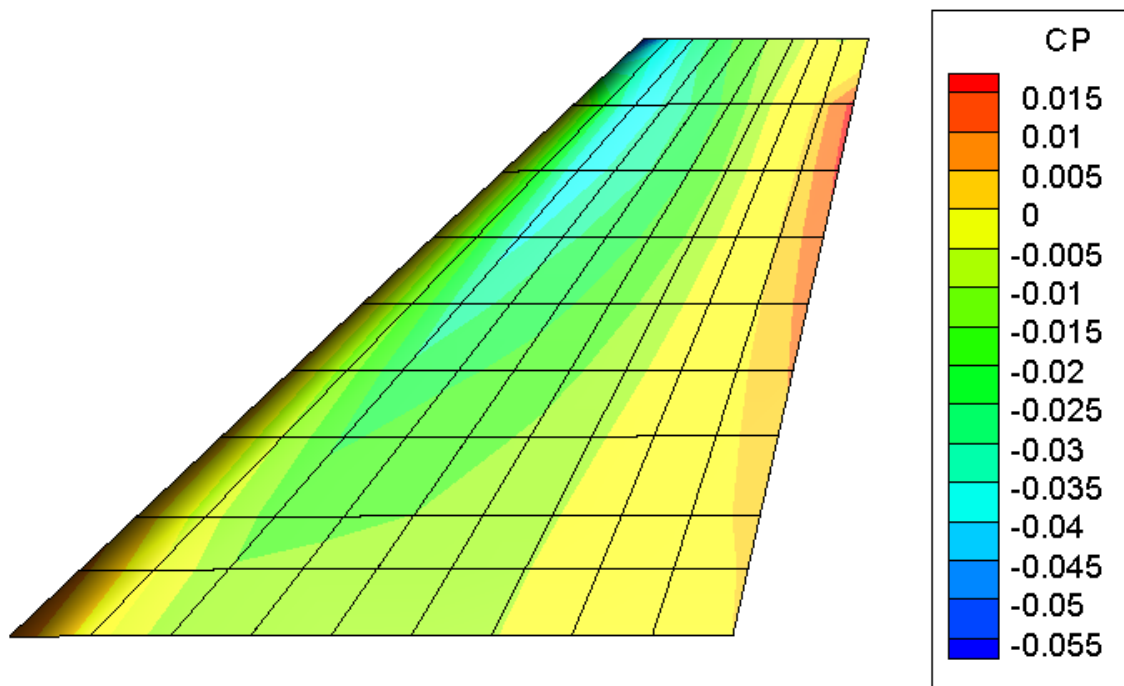


Figure 5.6-6. A representative aerodynamics run of many used for the B-58 tail drag surrogate model.

The B-58 analysis validation test case showed close agreement between the actual B-58 and the ESAV framework's analysis of the B-58. The chosen objective function for the ESAV MDO is the TOGW, therefore the validation case weights are compared in Table 5.6-1. In all cases, there is close agreement between the developed framework's results and the actual B-58 supersonic bomber specifications for the chosen mission profile.

Table 5.6-1. Comparison of Results from the Framework’s B-58 Validation Analysis with the Actual B-58

Weight Category	Actual B-58	Framework B-58
TOGW	163,000 lbs. (max.)	160,136 lbs.
Empty Weight	55,560 lbs.	58,364 lbs.
Fuel Weight	100,000 lbs. (max.)	90,073 lbs. (less than max.)

5.7 Higher-Fidelity ESAV MDO Framework Overview

A multidisciplinary design analysis and optimization framework tool has been built with the goal of optimally designing an efficient supersonic air vehicle. Specific physical challenges in designing an ESAV have been addressed in this higher-fidelity ESAV MDO framework that were not addressed in the low-fidelity ESAV MDO framework. Besides normal disciplinary analyses like aerodynamics, mission performance, and weights, new models and disciplines have been included in this framework specifically required by an advanced military aircraft that is tailless and has embedded engines. These are an NPSS/WATE++ propulsion model that takes into account performance and weight considerations inherent in an embedded engine (e.g. the serpentine inlet), an engine exhaust-washed structures analysis that designs the aft-deck affected by the supersonic hot gas exhaust environment from the engines, and a radar cross section analysis model that allows shaping and radar-absorbent material effects to be quantified so that different ESAV configurations can be compared based on their calculated RCS signatures. In addition, a stability and control module that is being designed to handle the issues inherent with a tailless supersonic aircraft and a structures model to provide a physics-based structural weight estimate are under development.

A validation of the ESAV framework with an existing aircraft that performs a similar supersonic bomber mission was undertaken. The Convair B-58 Hustler was modeled by the tools in the ESAV framework. The corresponding B-58 results from the ESAV framework analysis tools matched very closely with the actual B-58 data.

At this point, the higher-fidelity ESAV MDO framework was ready to be exercised by undertaking the exploration of the ESAV design space and optimizing the ESAV design for minimum TOGW. This will result in identifying an optimal ESAV design given the mission profile and other constraints.

6.ESAV Higher-Fidelity MDO Studies

6.1 Practical ESAV MDO Framework in ModelCenter™

The N^2 diagram is a good organizational tool to establish the data relationships and execution order of the MDO framework. Challenges arise, however, when the N^2 diagram must leave the theoretical N^2 format and be put into a practical working format, because an N^2 diagram is not a complete code. ModelCenter™ [41], a tool developed by Phoenix Integration Inc. was chosen to implement the current ESAV MDO frameworks. This software integration tool was introduced in the low-fidelity framework chapter.

ModelCenter™ offers two different high-level approaches to order and link the different disciplines, the process model and the data model. In an exceptionally large data space, it is convenient to pass data from one process to another by directly writing the data from one model into one or more output files that also serve as an input file to another model. This, of course, is only facilitated with code written in-house where custom output files can be written. The VT-CST geometry model is developed in-house and can, therefore, output custom file formats directly used by individual discipline models. Since the ModelCenter™ data model format is setup for output files transferring information to new input files created by ModelCenter™ filewrappers, the process model was chosen in order to additionally allow the direct output-to-input file structure for the ESAV MDO framework.

In an effort to tie the ModelCenter™ practical implementation of the framework to the N^2 diagram, the ModelCenter™ representation is shown in Fig. 6.1-1. The order in which the ModelCenter™ components are given follows the N^2 diagram reasonably well. It may seem mismatched with the FLOPS modules, because all of these are run at once with the single

FLOPS executable as seen in the ModelCenter™ implementation. The N^2 diagram lists the FLOPS modules in a certain order which may or may not be the actual order that FLOPS runs them in, but certainly the N^2 diagram lists them sequentially from the structures to noise.

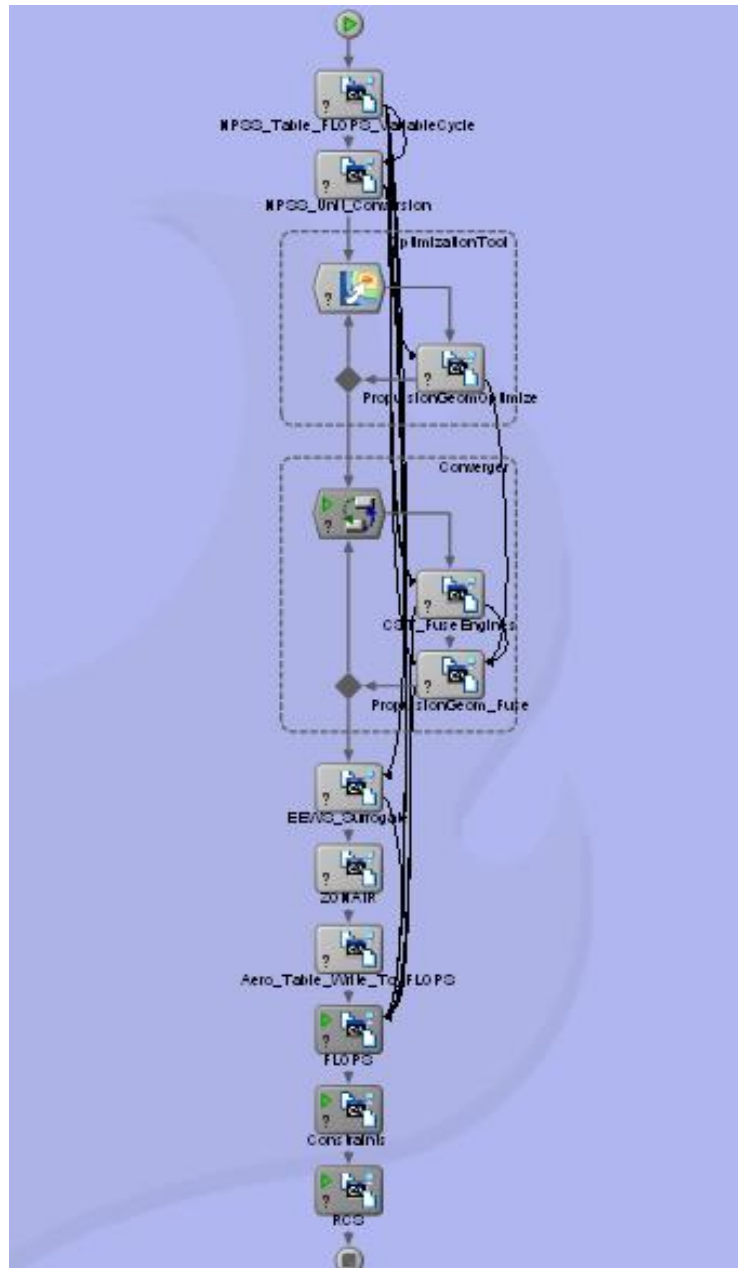


Figure 6.1-1. Practical ModelCenter™ implementation of N^2 diagram higher-fidelity ESAV MDO framework. The black curves with arrows show some of the data links and their direction between disciplines.

6.1.1 NPSS/WATE++ Propulsion Model in ModelCenter™

The NPSS/WATE++ model runs are to provide an engine input deck table throughout the mission profile, the weight, a center of gravity (CG) location, and engine dimensions as output data to be used by other discipline models. A Python script file is used to control the complicated series of NPSS runs needed to generate the output data. There are several design variables that can be chosen that are input into this Python script file, namely the cruise Mach number, the cruise altitude, the cruise thrust per engine, the maximum bypass ratio (BPR) for the variable cycle engine, the fan pressure ratio, the low/high pressure compressor pressure ratio, and the Mach numbers at which the BPR begins to decrease and where it finally goes to zero for the variable cycle. The WATE++ model is only run once at the design point to save time. All other off-design data points for the engine input deck are obtained with just the NPSS part of the model. The three separate output files that are created by NPSS and WATE++ are parsed for the required output data.

WATE++ outputs the engine length and diameter in inches, but it is required by the user discipline models in units of feet. Therefore, a small “linker” code is written to convert from inches to feet, called NPSS_Output_Unit_Conversion.exe. This simple program takes these two data from NPSS output, converts them to units of feet, and then outputs the new values in an output file. It also outputs noise values needed for the FLOPS noise module. These two programs are shown in Fig. 6.1-2 below as incorporated in ModelCenter™. This figure, and the several following figures, are nothing more than zoomed in views on particular parts of the larger view of the ModelCenter™ implementation of the ESAV MDO framework that was shown in Fig. 6.1-1.

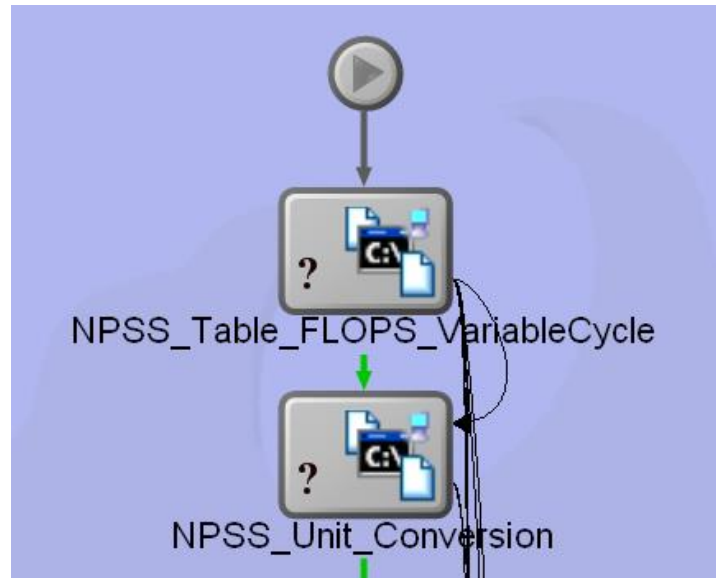


Figure 6.1-2. View of NPSS/WATE++ model with input and output links shown.

6.1.2 Inlet Geometry Local Optimization and Cowl Geometry Iterations

Once the design point of the engine and all off-design points have been computed by the NPSS/WATE++ model, the next step in the ModelCenter™ setup is to find the inlet geometry and cowl geometry. The inlet angles are found through a local CONMIN optimization described below. The cowl geometry is found through iteration between the geometry model and the VT-CST geometry model. This iteration requires data from the previously-run NPSS/WATE++ model to work.

The four-shock inlet model employs the DAKOTA project CONMIN [38] [81] optimization algorithm. This robust and well-established method of feasible directions [1] optimization code is used to find the three oblique ramp angles that maximize the total pressure ratio from the freestream to the flow behind the terminating normal shock (see Fig. 4.3-1). The ModelCenter™ implementation of this local optimization is shown in Fig. 6.1-3. This shows the value of 44 on the model's box, which indicates how many objective function calls were required

to find the optimum three angles. Although forty-four runs may appear to be computationally expensive, in actuality, it usually takes less than twenty seconds to converge.

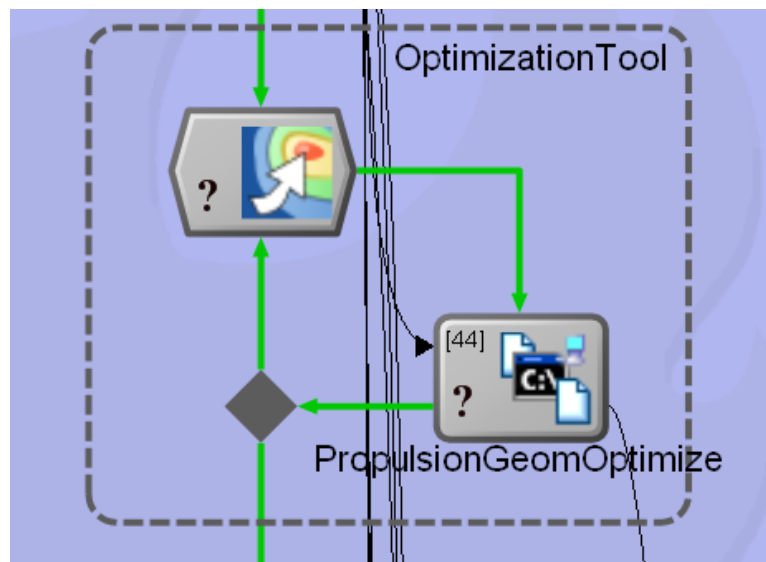


Figure 6.1-3. View of inlet CONMIN optimization model in ModelCenter™ with input and output links shown. The number 44 indicates the number of iterations it took to find the optimum.

With the inlet angles found, the inlet geometry can then be determined using the idealistic assumption that the oblique shock waves will all intersect at the cowl lip. The location of the cowl lip is not known a priori, and therefore must be calculated as part of the iteration. The cowl opening must accommodate the required mass flow demanded by the engine designed by NPSS based on design variables controlled by the global optimizer and the flow density computed by the four-shock inlet code. The baseline cowl design in the geometry input file is unlikely to match this equality constraint. Also, the inlet height and length geometry will change based on the cowl lip location and the unchanging optimal ramp angles. The iteration is performed between the geometry model and the propulsion geometry in order to find the correct cowl and ramp dimensions that match the engine air flow demand and the fixed optimal ramp angles.

ModelCenter™ has a built-in tool that makes adding an iteration setup convenient to the framework. It is called “Converger” and is found under the component plug-in menu item. Typically, the iteration takes about three to seven passes before the correct cowl and inlet geometry for the design engine is found. The geometry model does not disseminate any geometry to any other discipline until this iteration is complete. The ModelCenter™ implementation of this local iteration is shown in Fig. 6.1-4.

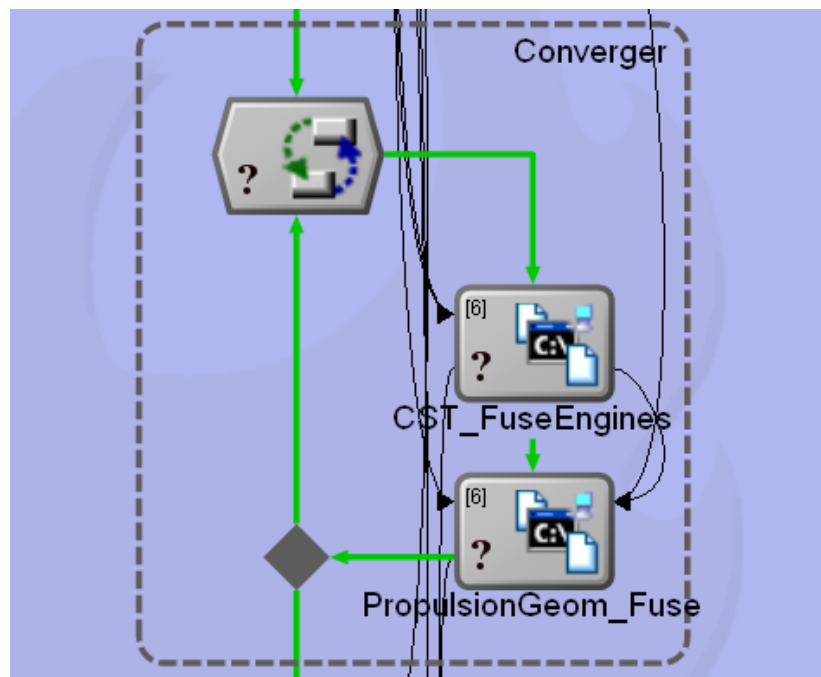


Figure 6.1-4. View of cowl/inlet and propulsion design iteration in ModelCenter™ with input and output links shown. The number 6 in the boxes indicate the number of iterations it took to converge on the correct geometry for the case shown.

6.1.3 ZONAIR Aerodynamics Model

The next discipline in the ModelCenter™ implementation of the N^2 diagram is the ZONAIR aerodynamic panel method. The ZONAIR model runs without any kind of variable input or output facilitated by ModelCenter™ filewrappers. The input is in the form of two files generated

by the geometry model, VT-CST. One file is the header file that contains the instructions on the case to be run. The other file holds the ESAV geometry data for ZONAIR. VT-CST automatically generates the panel discretization of the ESAV outer mold line for arbitrary ESAV designs. These two files are put together with a Python script to form the single ZONAIR input file. Other files allow ZONAIR to run in parallel using the Message Passing Interface (MPI) standard as explained below.

The mission performance model requires an aerodynamic table that spans multiple Mach numbers, coefficients of lift (angle-of-attack is iterated to match the required C_L), and altitudes. ZONAIR is capable of creating an aerodynamic influence coefficient (AIC) matrix file for a given Mach number and ESAV configuration. This task is relatively expensive, but must only be done once for a given configuration and Mach number. After the AIC file has been created, the altitude and angle of attack may be arbitrarily changed for new aerodynamic solutions, but instead of recalculating the AIC matrices, they can be read in from the existing file which speeds up these subsequent calculations on the order of two times faster. Thus, the first step is to find the AIC matrices for each given Mach number.

Finding the AIC matrices is done in parallel using a wrapping MPI program. Each separate Mach number becomes a parallel process and runs independently, but at the same time, as the other Mach number processes. A Python script directs each of these processes and gathers all required aerodynamic table data at the given Mach number. The call to each of these Mach number processes blocks for synchronization by using the `MPI_Barrier` routine. Therefore, not until all Mach number processes have completed and reached the barrier are the final aerodynamic tables assembled from the data computed from each Mach number process. This is done by a single Python script. Finally, a wrapping code takes this data and inputs it into the

FLOPS mission performance module input file. The entire ZONAIR aerodynamic process is shown as a flowchart in Fig. 6.1-5.

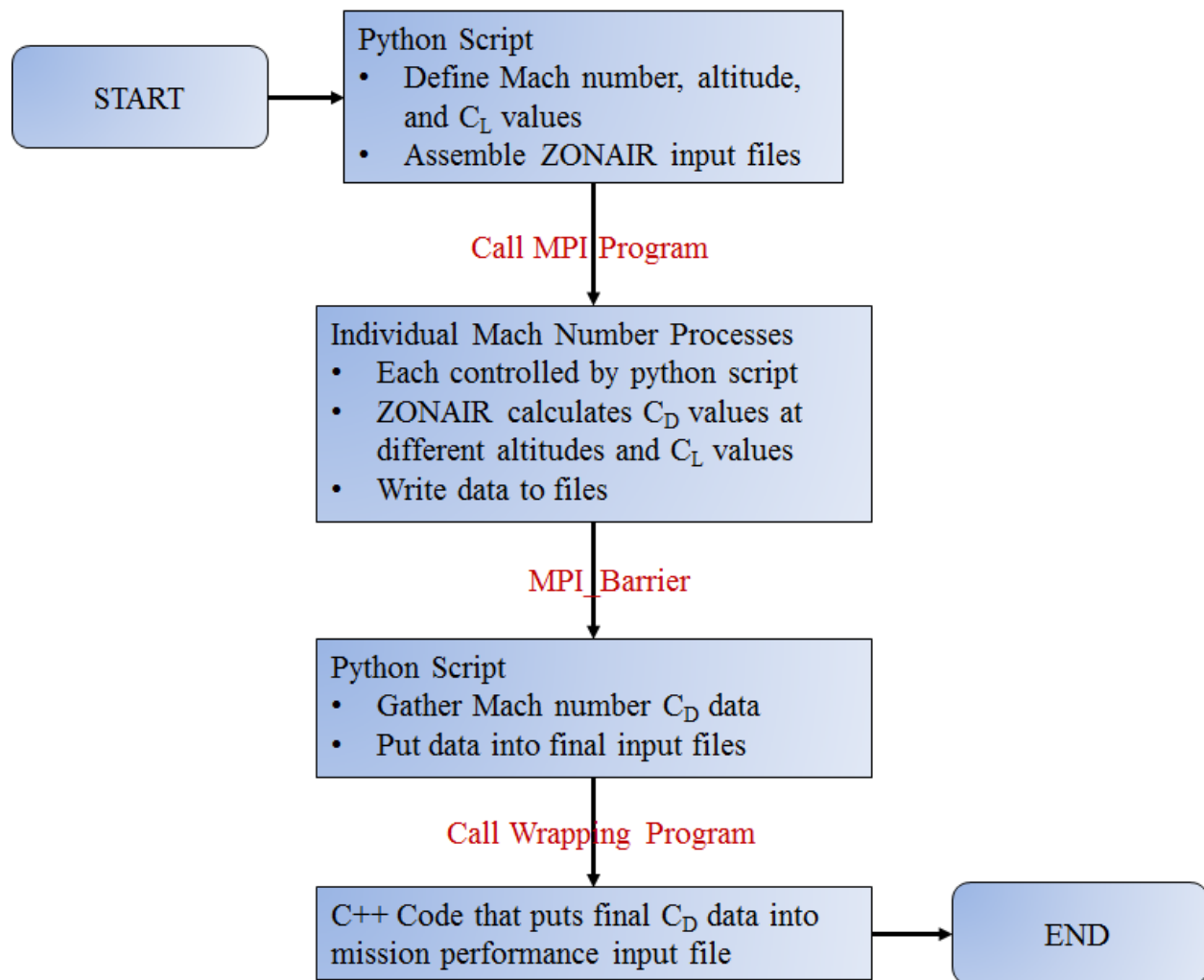


Figure 6.1-5. ZONAIR aerodynamic model flowchart as implemented in MDO framework.

6.1.4 Engine Exhaust-Washed Structures Model

The EEWS model takes the cruise altitude and Mach number as well as the nozzle throat pressure, temperature and area as inputs and then outputs a weight for the EEWS structure. This output weight is then input into the weights module downstream. References [114] and [104] detail the creation and integration of this discipline's surrogate model. Figure 6.1-6 shows the ModelCenter™ implementation with its single input and single output links.

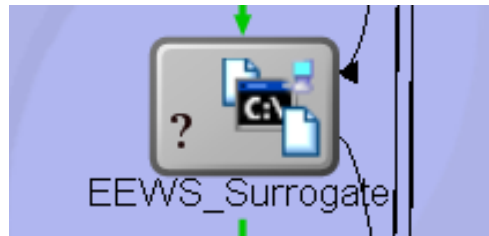


Figure 6.1-6. View of EEWS discipline surrogate model with input and output links shown.

6.1.5 FLOPS Modules

The Flight Optimization System program works with one controlling input file and one executable. Therefore, although several individual FLOPS modules are used in the framework, it is only run once in the practical ModelCenter™ implementation of the N^2 diagram. This one run calculates all of the output from all of these modules at once. These modules are structures, weights, noise, take-off & landing, and mission performance. The single input file is supplemented by the aerodynamic coefficient of drag tables and propulsion engine input deck required by the FLOPS mission performance module throughout the flight envelope. The aerodynamic tables are embedded directly into the single controlling input file, while the propulsion engine deck's file name is input into the FLOPS file, although the engine deck itself remains a separate file.

The FLOPS noise module takes inputs from the propulsion model as the engine noise is by far the most important factor in total aircraft system noise generation. The jet noise calculation method is chosen as the method developed by the SAE A-21 jet noise subcommittee [52] [115]. This method is based on an ANOPP [116] module. The inputs required by this method are the engine exhaust velocity and total temperature as well as the nozzle exit area. The output is the noise level in decibels as compared to the FAR 36 [75] standards. If the noise value is greater than zero it has violated the FAR 36 standards and *vice versa* for less than zero values.

The FLOPS executable is pivotal to the entire ESAV optimization process. One reason is because the objective function, the TOGW, is calculated in the FLOPS weights module. The aft deck and engine weights are passed into the weights routine from the EEWS and NPSS/WATE++ models respectively. In the future, the structural wing weight will also be input from a structures analysis model under development at the time of writing. The fuel weight comes from the fuel required to perform the mission with the appropriate time and SFC of the engines listed in the engine deck. All secondary subsystem weights and unobtainable weights (e.g. fasteners, welds, epoxies, etc.) are all packaged together with the primary weights to make up the TOGW.

Most ESAV design constraints are examined in the FLOPS mission performance and take-off & landing modules. That is, the feasibility of a design rests largely on the resulting aircraft's ability to fly through its intended mission. This includes take-off, climb, transonic drag rise, cruise, maneuvers, and landing. All disciplines positioned previous to the FLOPS box in the ModelCenter™ framework pass some data to the FLOPS box. This can be seen by the many data feed links going to the FLOPS box in Fig. 6.1-7, with the exception of the aerodynamics module, which passes the data by files rather than using the variable transfer done with file wrapping in ModelCenter™.



Figure 6.1-7. View of the FLOPS model. This includes the structures, weights, take-off & landing, mission performance, and noise modules in one box. There are many input links from previous models in the ModelCenter™ ESAV framework.

6.1.6 Constraints and Radar Cross Section Models

The constraints model is simply an executable file that can take inputs from any discipline outputs (whether it resides in a file or is part of the ModelCenter™ file wrapping variables), and then apply constraint conditions to those values, or any mathematical combination of those values. The output can then be added to the ModelCenter™ optimization tool, and limits imposed to ensure that the constraint is met.

The executable constraints code is a compiled in-house C++ code. Therefore, any additional constraints can be easily added at any time. Currently, the constraints model calculates the difference between the required fuel volume and the available volume for fuel, taking into account the significant internal volume required for the cockpit, landing gear, and internal weapons storage like the F-22 shown in Fig. 6.1-8. The ESAV internal weapons volume and storage method is likely to be similar to the F-22 because of LO considerations.



Figure 6.1-8. F-22 internal weapons storage volume.

The available volume is just the total volume available multiplied by a factor of 0.8 [43] to account for fuel tankage volume which is unusable for fuel. This value is then added to the optimization problem as a constraint (*available fuel volume* $\geq$ *required fuel volume*). Since there is no RCS constraint (RCS is just a metric at present), this comes before the RCS analysis. If a RCS constraint is added in the future, the order can be easily swapped.

The RCS model takes a geometry file input in stereolithographic (.stl) format from VT-CST. The stereolithographic representation of the OML is an ASCII file that contains a list of triangular panels with 3D coordinates of each vertex, as well as the triangle area vector. The vertices of coincident triangle edges are the same between triangles, thus the unmodified STL aircraft OML is a closed shape. VT-CST draws the ESAV OML (both starboard and port sides – symmetric configurations assumed), but does not include the panels on the face of the cowl inlet for the RCS analysis. These panels are missing because it is assumed that a properly designed serpentine inlet will not allow radar return once it enters the serpentine inlet.

A departure from reality is the fact that the geometry model currently does not include the ability to have saw-tooth edges on the faces that are normal to the non-turning flight direction, both on the fore and aft of the aircraft. This means that the radar return near 0° and 180° (where 0° is relative to the coincident vector with the ESAV x-axis in the positive direction through the aircraft nose) is higher than it would actually be on the final aircraft. For this reason, the θ angles between

$$355^\circ \leq \theta \leq 5^\circ \text{ and} \quad (39a)$$

$$175^\circ \leq \theta \leq 185^\circ \quad (39b)$$

are ignored for the RCS output. The infinity norm for the balance of the angles' RCS returns is the model output used in the design space, but all RCS values at each angle analyzed (every 3° is

the current step size) is available in the output file. A representative RCS STL file of a notional ESAV configuration is shown in Fig. 6.1-9. This figure shows the triangular paneling, not including the deleted panels on the cowl face. The default mesh density is 50x50 for the RCS analysis. The analysis takes on the order of 20 seconds for all angles between 0° and 360° (at 3° intervals) and with the 50x50 mesh density using the C++ code based on POFACETS.

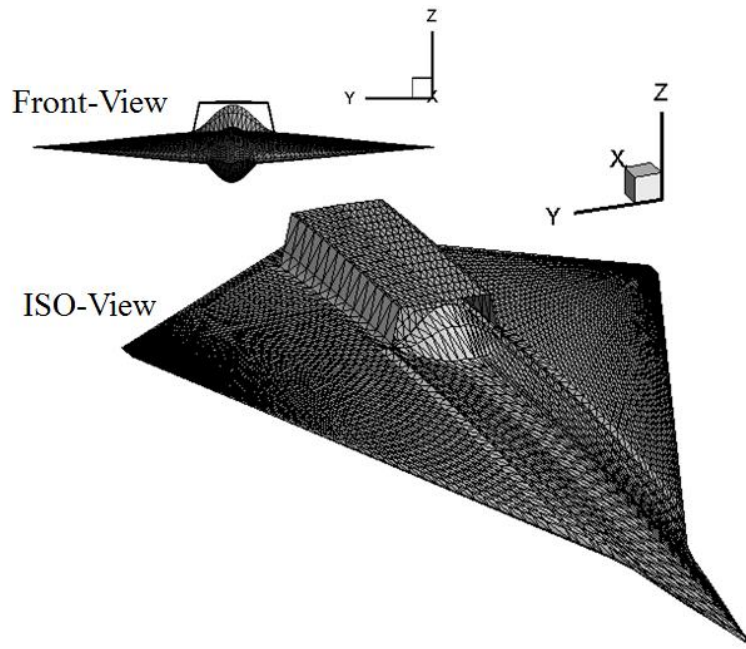


Figure 6.1-9. Stereolithographic geometry output of a notional ESAV OML which is used by the RCS model as input.

6.2 Optimization Approach

In a previous chapter, two different optimal ESAV designs were found using two different methods with the low-fidelity framework. The first, called the “DOE method”

- starts by exploring the whole design space with a large DOE,
- identifies the group of the best results (using objective function) from the DOE,
- performs a gradient optimization using the identified points as starting design points,

- reduces the design space boundaries based on the design variables of the stationary points found with the gradient optimizer,
- performs a NSGA-II optimization within the reduced design space using the best gradient optimal design point found as the starting point, and
- performs a final gradient optimization from the point identified by the NSGA-II optimization as optimal.

This long and computationally expensive procedure (many objective function evaluations) produced an optimal ESAV design with a low TOGW.

The second method, called the “sequential NSGA-II method”

- performs one NSGA-II optimization with a starting ESAV design point obtained through engineering judgment and some analysis with the low-fidelity framework, and
- runs a second optimization using the first NSGA-II optimal point as the new starting point.

This method also produced an optimal ESAV design, but it was shown to not be the global optimal design, because it had a higher TOGW than the ESAV found using the DOE method.

Both methods are useful. The second method takes more than an order of magnitude fewer objective function evaluations than the first method. However, one method could be more useful than the other depending on the optimization problem being solved.

With this in mind, both approaches will be expanded on in this section. The new ability to perform a higher-fidelity objective function evaluation, gained through disciplinary model enhancements, brings in a different dimension not seen in the low-fidelity framework alone. The different dimension introduced with the higher-fidelity framework is the computational expense of the higher-fidelity methods. This also offers a higher level of fidelity to analyze ESAV

configurations in order to determine the objective function value within the constrained optimization problem than what was obtainable with the low-fidelity ESAV MDO framework. Since the low-fidelity MDO framework is not as expensive as the higher-fidelity ESAV MDO framework, it is desired to use it as often as possible in ESAV design studies. The higher-fidelity framework can be used to check low-fidelity results to gauge their appropriateness to the ESAV problem.

6.2.1 Large DOE Optimization Approach

The first method used is called the DOE method. This method was used with the low-fidelity ESAV MDO framework to first perform a large space-filling DOE analysis that explored the design space. A small set of design points with the best (lowest) values of the TOGW were identified. Then, a gradient optimization was performed using each of the identified designs as the starting points to ensure that each of the points were local optimum, or stationary points. A reduced design space was identified using these stationary points to demarcate its boundaries. Within this reduced design space, a NSGA-II optimization algorithm was executed to find the optimal low-fidelity design. Realizing that there may be a local optimal point by the NSGA-II point, a final gradient optimization was performed to find the final optimal low-fidelity ESAV design.

In order to determine if the low-fidelity effort can be leveraged in conjunction with the higher-fidelity ESAV MDO framework, the low-fidelity results were assessed using the higher-fidelity framework. The process developed in this effort to do this is shown in Fig. 6.2-1. The first step is to check the low-fidelity stationary point results, used to reduce the design space, with the higher-fidelity framework. This step works to certify the feasibility of the points used to define the reduced design space boundaries. If several of these points are found to be infeasible

ESAV Mixed-Fidelity Optimization Process

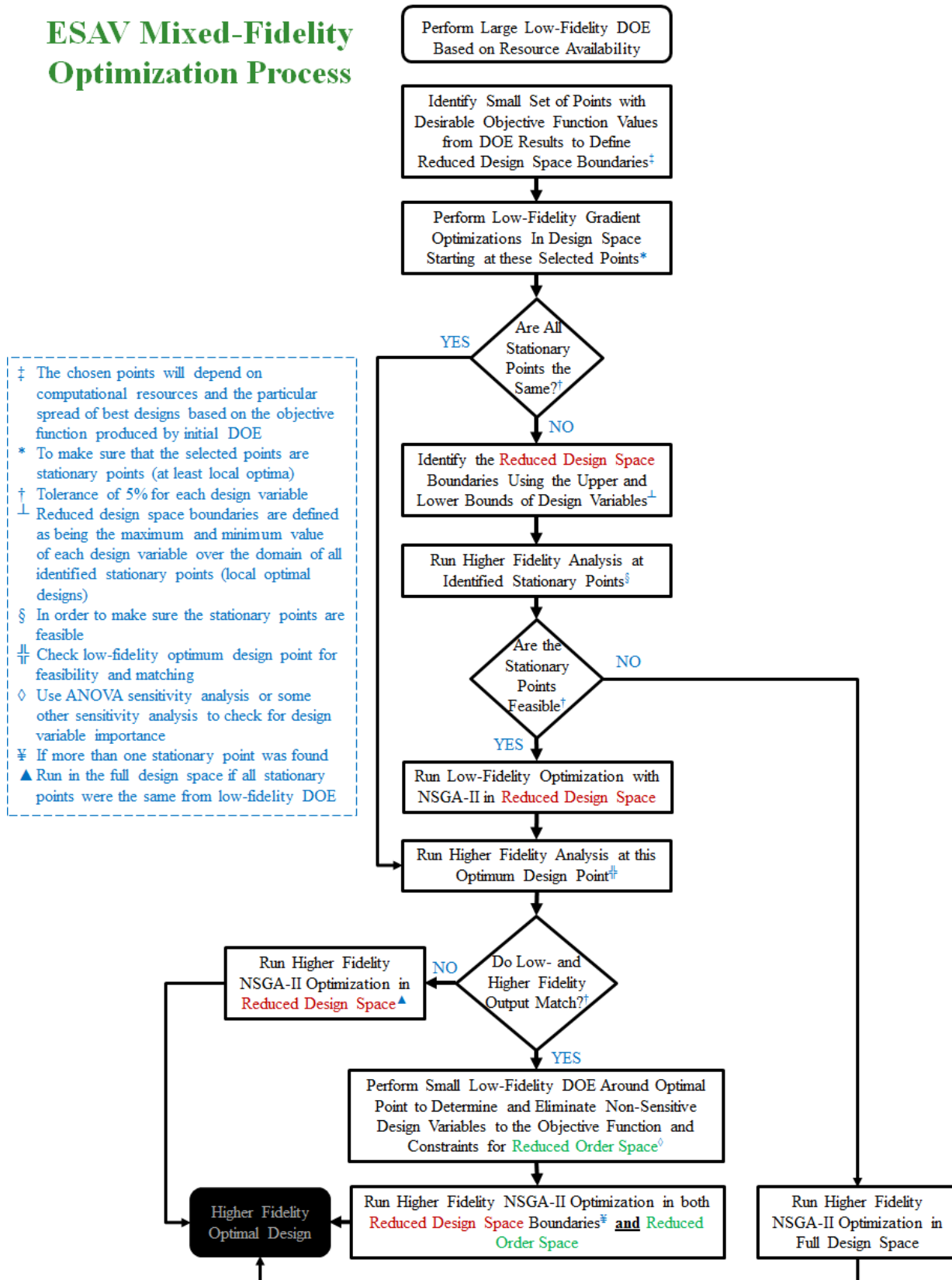


Figure 6.2-1. Mixed fidelity optimization process represented by a flowchart.

with the higher-fidelity analyses, or totally inconsistent with the higher-fidelity TOGW results, then they cannot be used along with the higher-fidelity ESAV MDO framework. In this case, the higher-fidelity ESAV MDO framework must be used alone to find the optimum ESAV design (see the successive NSGA-II method below). However, if they are found to be feasible, then the low-fidelity reduced design space can be used to reduce the higher-fidelity optimization design space.

If the reduced design space low-fidelity stationary point boundaries are discovered to be feasible, then the higher-fidelity ESAV MDO framework can be used to check if the optimal design found with the NSGA-II algorithm, followed by the local gradient optimization in the reduced design space, is feasible or not. If the optimal point proves to be infeasible, then this low-fidelity optimum result is not used.

If the low-fidelity optimal ESAV is feasible in the higher-fidelity ESAV MDO framework, the next level of checking the low-fidelity results requires seeing if the resulting optimal low-fidelity design *matches* the higher-fidelity results. This is the next step of confirmation beyond the less demanding feasibility check. This can be done by setting a tolerance percentage value that the two objective function results from the two frameworks must be within.

If the results do match, then a small low-fidelity DOE can be performed within a small ball centered at the optimal point to determine which (if any) of the design variables can be neglected using the ANOVA method. If this analysis shows that any of the design variables have little effect on the objective function or the constraints, then they may be discarded. The resulting fewer design variables will then define a reduced order design space for the higher-fidelity optimization.

If the low-fidelity and higher-fidelity designs do not match within the tolerance, then no attempt can be made to reduce the design space order. This is because the low-fidelity sensitivity DOE results around the low-fidelity optimum cannot be trusted to guide the reduction in design space order for the higher-fidelity ESAV MDO framework. In this case, all design variables must be kept when the higher-fidelity framework is used for optimization.

The higher-fidelity ESAV MDO framework is used to find the final optimal ESAV design. Using this framework to perform optimization is much more computationally expensive than the low-fidelity ESAV MDO framework. Up to this point in the DOE method, the higher-fidelity framework has only been used one analysis at a time to check low-fidelity results. The utility of using the low-fidelity framework has been to possibly define both a reduced design space and a reduced-order space depending on the results of the various higher-fidelity checks on the low-fidelity framework results. The reduced design space and reduced-order space narrow the optimization problem, making it less expensive to run the higher-fidelity optimization.

This process ensures that all paths lead to at least the higher-fidelity optimal design. If the low-fidelity ESAV MDO framework is accurate enough, the path to the higher-fidelity optimum will be both computationally cheaper and more likely to be the global optimum.

Reducing both the design space and order of the design space can be viewed graphically, assuming an initial 3D design space, that is, the number of design variables is three. Figure 6.2-2 shows a notional design space being reduced into a more manageable size, which is representative of the large DOE and stationary point boundary steps done with the low-fidelity framework. Figure 6.2-3 shows the now-reduced design space after having determined that the objective function is not sensitive to one of the original three design variables.

Clearly, the resulting 2D space in Fig. 6.2-3 is a significantly more practicable space in which to perform a higher-fidelity, and more computationally expensive, multidisciplinary design optimization. The principal guiding strategy for the DOE method has been to use the low-fidelity framework to find a reduced design space *and* a reduced-order space using relatively low-cost, higher-fidelity checks during the process. Once the final space has been defined, the higher-fidelity framework can then be used to find the sought-after optimal ESAV design.

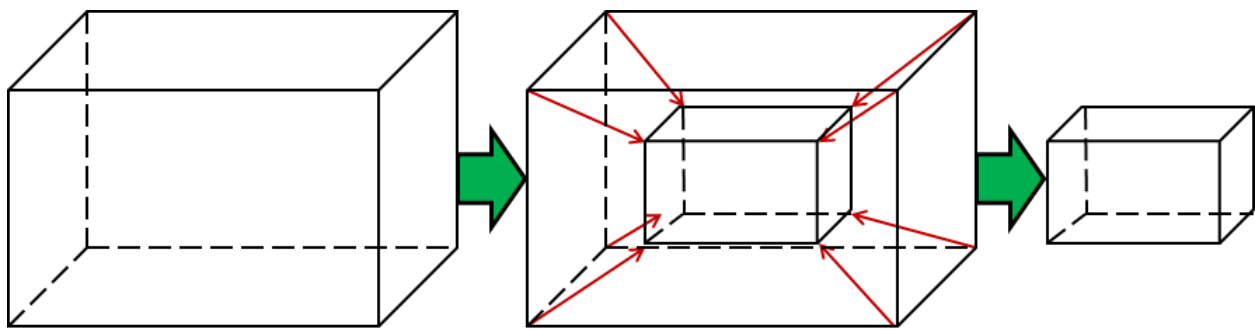


Figure 6.2-2. Notional reduced design space process for a three-design-variable space.

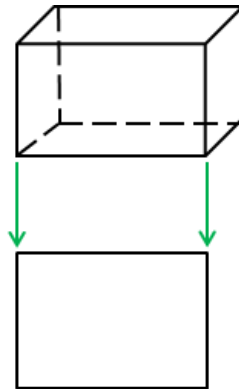


Figure 6.2-3. Notional design space, already reduced, and its corresponding reduced-order space assuming the objective function is insensitive to one design variable.

6.2.2 Two Successive NSGA-II Optimizations in Full Design Space Method

The previous ESAV low-fidelity MDO results showed that the two successive NSGA-II optimizations did a fairly good job at finding a low TOGW ESAV design. The result was about 10% from the much more intensive DOE method result. In previous studies with a similar framework, the successive NSGA-II method has sometimes even found better designs than the DOE method. In this effort, the first optimization used an engineering guess as the starting point, and the second used the optimal design found in the first NSGA-II run as the starting point. The successive NSGA-II optimization method used more than an order-of-magnitude less objective function evaluations to find its low TOGW design.

The successive NSGA-II optimizations method used the full design space in both optimization runs for the low-fidelity ESAV MDO framework, therefore it does not reduce the design space boundaries nor reduce the order of the design space for the second optimization. The boundaries are not reduced, because the best designs are all very similar and close together within the design space due to the nature of the NSGA-II algorithm, which tends to converge to the best location within the design space. The number of design variables was not reduced, because no extra runs were spared to build a surrogate model for sensitivity studies. Also, the utility of local sensitivity data on the larger design space is small.

6.3 Higher-Fidelity ESAV MDO Framework Optimization

The DOE method outlined in Fig. 6.2-1, and the successive NSGA-II method are both explored for identifying the higher-fidelity ESAV MOD framework optimal efficient supersonic air vehicle concept.

6.3.1 DOE Method

There were six design points identified from the 10,000 run DOE using the low-fidelity ESAV MDO framework. Each of these points had either a CONMIN or SQP gradient optimization applied to them using the identified designs as starting points. The resulting six stationary points were then used to define the reduced design space boundaries, and each is now checked for feasibility using the higher-fidelity ESAV MDO framework. The results of this checking process are shown in Table 6.3-1.

Table 6.3-1. Results of Higher-Fidelity Checks on Low-Fidelity Stationary Points

Stationary Point ESAV Design	Feasible in Higher-Fidelity	Feasibility Issue	TOGW Deviation
1	Yes	N/A	+30.8%
2	No	Climb – Transonic	N/A
3	No	Climb – Take-off	N/A
4	No	Climb – Take-off	N/A
5	Yes	N/A	+24.7%
6	No	Cruise*	N/A

****Cruise segment is calculated before the climb segment – no weight convergence obtained***

The main difference causing the TOGW deviation from the low-fidelity ESAV MDO framework to the higher-fidelity ESAV MDO framework for the ESAV stationary point design number 1 appears to be the propulsion system weight and the structural weight. The ZONAIR aerodynamics model generally predicts higher drag at the lower C_L values than the low-fidelity aerodynamics model, which easily translates to a larger engine seen in the higher-fidelity results. The fuel weight was also somewhat different and is probably due to the near cancellation of the higher-fidelity engines lower SFC, but higher throttle setting. A good rule-of-thumb for the aircraft growth factor for weight is about 4 to 5 for supersonic aircraft [117] [118]. This very

high weight growth factor range shows the great necessity for accurately predicting all weight values during supersonic aircraft conceptual design.

The NPSS model calculated a lower SFC (more efficient engine), as expected, because of the more modern empirical engine maps used for NPSS. The main causes for the deviation of the ESAV stationary point design number 5 are the difference in the fuel weight and the corresponding difference in the structural weight. In this instance, the propulsion weights were almost identical. The drag coefficients calculated by ZONAIR were again higher than the linear, low-fidelity framework aerodynamics model. This caused a higher throttle setting (higher fuel burn rate) for the higher-fidelity ESAV MDO framework during the mission, and thus a higher fuel weight was obtained for this higher TOGW ESAV design.

The low-fidelity optimum ESAV was also checked with the higher-fidelity ESAV MDO framework. This design was also not feasible in this framework due to its inability to both take-off and climb through the transonic regime.

Looking at the poor feasibility results from the higher-fidelity checks and the large discrepancies in the objective function, TOGW, between the two cases that were feasible, it is clear that the low-fidelity ESAV MDO framework cannot be used to guide the higher-fidelity ESAV MDO framework optimization process.

According to the logical process in Fig. 6.2-1, the higher-fidelity optimization should be carried out in the full design space. The DOE method cannot be accomplished due to its high computational expense using the higher-fidelity ESAV MDO framework. Therefore, the successive NSGA-II process was chosen for this optimization. Even though the DOE process is not used here because of the mismatch between the low- and higher-fidelity ESAV optimization,

it may have merit with other multifidelity optimization efforts within the aircraft design field, or elsewhere.

6.3.2 Successive NSGA-II Method for Higher-Fidelity ESAV MDO

Framework

The first two tasks that need to be performed before the successive NSGA-II method is used to find the higher-fidelity ESAV MDO framework optimum result are:

1. define the design space, and
2. choose an initial guess ESAV design.

The higher-fidelity ESAV MDO framework uses the same design variable choices that were used in the low-fidelity framework. The same reasonable, but generous, bounds are used for each of the higher-fidelity design variables that were used to bound the design variables used in the low-fidelity case. These were shown previously in Table 3.2-1 and are repeated in Table 6.3-2 for convenience.

If the upper bounds of the three pressure ratios are multiplied it gives an overall pressure ratio for the engine of 42.875. This value for the overall pressure ratio is reasonable for today's engines and certainly for engine technology level expected to be the norm for the future timeframe that an ESAV would be expected to operate in. It is also low enough to not cause excessive compressor heating issues.

The initial ESAV design for this higher-fidelity successive NSGA-II optimization is chosen the same way it was chosen for the low-fidelity ESAV MDO framework's successive NSGA-II optimization – by engineering experience and some analysis. That is, a sound baseline design is chosen and then analyzed. The model output is scrutinized and minor modifications are then done to improve the baseline design until it is at least feasible. This is done to replicate a

Table 6.3-2. Higher-Fidelity ESAV MDO Framework Design Variables and the Corresponding Design Space Constraints

Design Variable	Units	Lower Bound	Upper Bound
1) Mission Maximum Cruise Mach Number	N/A	2.2	2.5
2) Mission Maximum Cruise Altitude	ft	45000	70000
3) Cruise Thrust	lbf	7500	35000
4) Maximum Bypass Ratio	N/A	0.001	0.5
5) Fan Pressure Ratio	N/A	2.9	3.5
6) Low Pressure Compressor Pressure Ratio	N/A	2.9	3.5
7) High Pressure Compressor Pressure Ratio	N/A	2.9	3.5
8) High Mach Number Bypass Ratio Cutoff	N/A	1.8	2.5
9) Low Mach Number Bypass Ratio Cutoff	N/A	0.8	1.6
10) Wing Semi-Span	ft	25	60
11) Wing Root Chord Length	ft	90	130
12) Wing Tip Chord Length	ft	0.05	40
13) Wing Leading Edge Sweep Angle	deg	55	78
14) Wing First-Order Shear (Dihedral)	ft	-1.5	1.5
15) Wing Second-Order Shear	ft	-1.5	1.5
16) Wing Zero-Order Twist	deg	-1.5	1.5
17) Wing First-Order Twist	deg	-1.5	1.5
18) Wing Second-Order Twist	deg	-1.5	1.5
19) Fuselage Length	ft	130	165
20) Fuselage Maximum Width	ft	6	12
21) Fuselage Upper Height	ft	4	9
22) Fuselage Bottom Height	ft	4	9
23) Fuselage Starting X Coordinate	ft	-30	0
24) Fuselage Area Ruling Location	N/A	0	0.7
25) Fuselage Area Ruling Magnitude	N/A	0	0.5
26) Fuselage Area Ruling Width	N/A	0	0.9

practicing engineer's available tools in choosing where to start. The initial higher-fidelity ESAV design for the successive NSGA-II method was the same as the low-fidelity ESAV design for the successive NSGA-II method. However, this was found to be infeasible using the higher-fidelity analyses codes, and so a different design was then converged upon using the method briefly outlined above.

All of the chosen design variables are whole numbers or have a small number of significant figures, because a human designer chose the initial configuration. Another good method might have been to choose the lowest TOGW feasible configuration from the low-fidelity ESAV MDO framework stationary point checks, since this data was readily available. The initial planform is shown in Fig. 6.3-1 (using different analysis models views of the geometry), and Table 6.3-3 gives the values of the initial design variables.

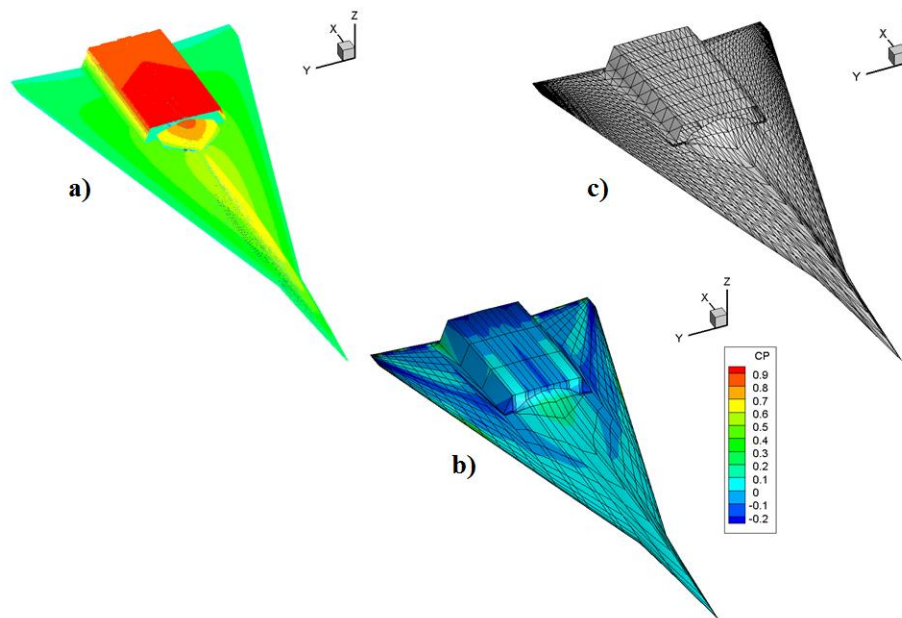


Figure 6.3-1. Starting point design for the first optimization run in the successive NSGA-II method using the higher-fidelity ESAV MDO framework. The left shows the planform drawn with the VT-CST geometry model using a field of points, the middle shows the ZONAIR C_p output for the $M=2.0$ at $C_L=0$ case, and the right shows the RCS format.

Table 6.3-3. Initial Design Variables Describing the Starting ESAV Design for the First Optimization Run of the Higher-Fidelity Successive NSGA-II Method

Design Variable	Units	Value
1) Mission Maximum Cruise Altitude	ft	58000
2) Mission Maximum Cruise Mach Number	N/A	2.2
3) Cruise Thrust	lbf	25000
4) Maximum Bypass Ratio	N/A	0.23
5) Fan Pressure Ratio	N/A	3.0
6) Low Pressure Compressor Pressure Ratio	N/A	3.0
7) High Pressure Compressor Pressure Ratio	N/A	3.333
8) High Mach Number Bypass Ratio Cutoff	N/A	2.1
9) Low Mach Number Bypass Ratio Cutoff	N/A	1.6
10) Wing Semi-Span	ft	30
11) Wing Root Chord Length	ft	115
12) Wing Tip Chord Length	ft	5
13) Wing Leading Edge Sweep Angle	deg	75
14) Wing First-Order Shear (Dihedral)	ft	-0.7
15) Wing Second-Order Shear	ft	0.5
16) Wing Zero-Order Twist	deg	0.1
17) Wing First-Order Twist	deg	0.5
18) Wing Second-Order Twist	deg	-0.5
19) Fuselage Length	ft	132
20) Fuselage Maximum Width	ft	8
21) Fuselage Upper Height	ft	5
22) Fuselage Bottom Height	ft	5
23) Fuselage Starting X Coordinate	ft	-25
24) Fuselage Area Ruling Location	N/A	0.5
25) Fuselage Area Ruling Magnitude	N/A	0.125
26) Fuselage Area Ruling Width	N/A	0.4
TOGW	lbs.	304,081

The TOGW of the initial ESAV design for the higher-fidelity ESAV MDO framework is far away from the optimal design found using the DOE method with the low-fidelity framework. In fact it is 79% heavier at 304,081 lbs. The first NSGA-II run took 609 objective function evaluations and about 14 days to complete. By far, the reason for this long run time was the ZONAIR aerodynamics model, even with it running in parallel on an eight-core machine. It would be desirable to run ZONAIR on a massively parallel computer to reduce its required run time even further than that achievable with the eight-core machine. The best TOGW from this first attempt was 246,784 lbs., a 18.8% reduction in TOGW from the initial design.

The second NSGA-II run used the best design from the first run as the new starting point. It ran 785 objective function evaluations in about 3 weeks. The final optimal ESAV design from the second NSGA-II optimization had a TOGW of 181,559 lbs. This amounts to a 40.3% improvement on the initial design, and a 26.4% improvement on the best design resulting from the first NSGA-II optimization. The utility of the successive NSGA-II method is shown with the low TOGW final ESAV design found using this method.

The NSGA-II optimizer spent most of the beginning generations going back and forth between an arrow wing (like the low-fidelity ESAV MDO optimum) and a trapezoidal wing design during the second optimization run. In the end, the trapezoidal wing turned out to be the better configuration based on the higher-fidelity ESAV MDO framework. The best arrow wing design had a TOGW of 200,735 lbs. and is shown in Fig. 6.3-2. The final optimal design variables are shown in Table 6.3-4 and its configuration in Fig. 6.3-3.

Previously, the low-fidelity optimal ESAV was shown to be infeasible when analyzed by the higher-fidelity framework. Likewise, when the higher-fidelity optimal ESAV is analyzed by the low-fidelity framework it is found to be infeasible as well. The reason for this is that cruise

thrust was not sufficient in the low-fidelity analysis to overcome the cruise drag. This is probably the reason that the low-fidelity optimal ESAV had such a high cruise altitude (67,754 ft.).

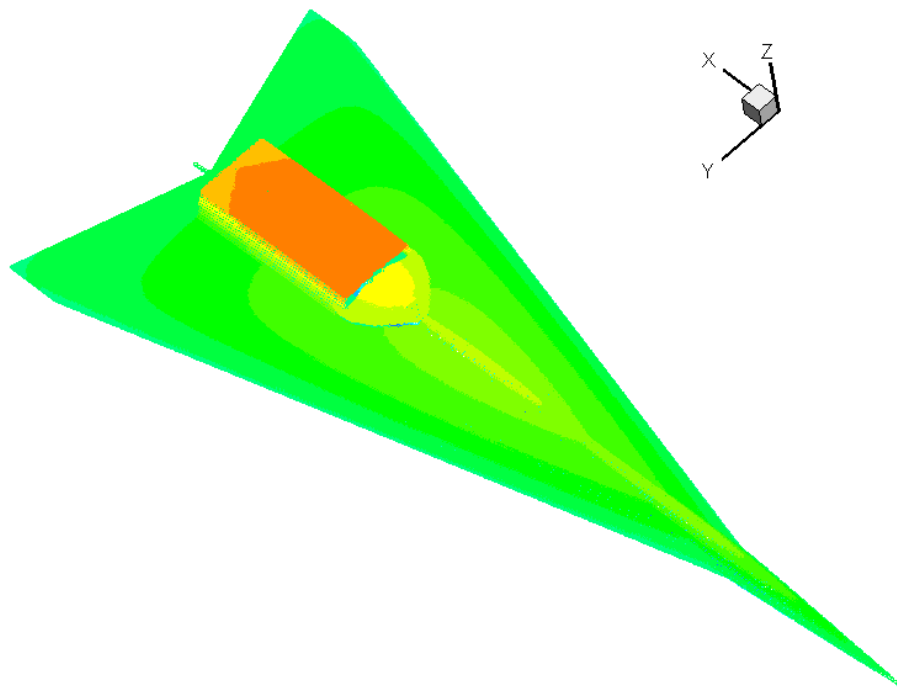


Figure 6.3-2. Best arrow wing design found in the second higher-fidelity ESAV MDO NSGA-II optimization run.

Table 6.3-4. Higher-Fidelity Optimal Design Found Using the Successive NSGA-II Method

Design Variable	Units	Value
1) Mission Maximum Cruise Altitude	ft	56,653
2) Mission Maximum Cruise Mach Number	N/A	2.47
3) Cruise Thrust	lbf	17,192
4) Maximum Bypass Ratio	N/A	0.13
5) Fan Pressure Ratio	N/A	3.08
6) Low Pressure Compressor Pressure Ratio	N/A	3.23
7) High Pressure Compressor Pressure Ratio	N/A	3.38
8) High Mach Number Bypass Ratio Cutoff	N/A	2.15
9) Low Mach Number Bypass Ratio Cutoff	N/A	1.01
10) Wing Semi-Span	ft	30.15
11) Wing Root Chord Length	ft	101.18
12) Wing Tip Chord Length	ft	11.52
13) Wing Leading Edge Sweep Angle	deg	55.96
14) Wing First-Order Shear (Dihedral)	ft	0.90
15) Wing Second-Order Shear	ft	-0.65
16) Wing Zero-Order Twist	deg	-0.65
17) Wing First-Order Twist	deg	-0.27
18) Wing Second-Order Twist	deg	-0.39
19) Fuselage Length	ft	133.42
20) Fuselage Maximum Width	ft	7.09
21) Fuselage Upper Height	ft	4.45
22) Fuselage Bottom Height	ft	6.06
23) Fuselage Starting X Coordinate	ft	-29.91
24) Fuselage Area Ruling Location	N/A	0.19
25) Fuselage Area Ruling Magnitude	N/A	0.46
26) Fuselage Area Ruling Width	N/A	0.41
TOGW	lbs.	181,599

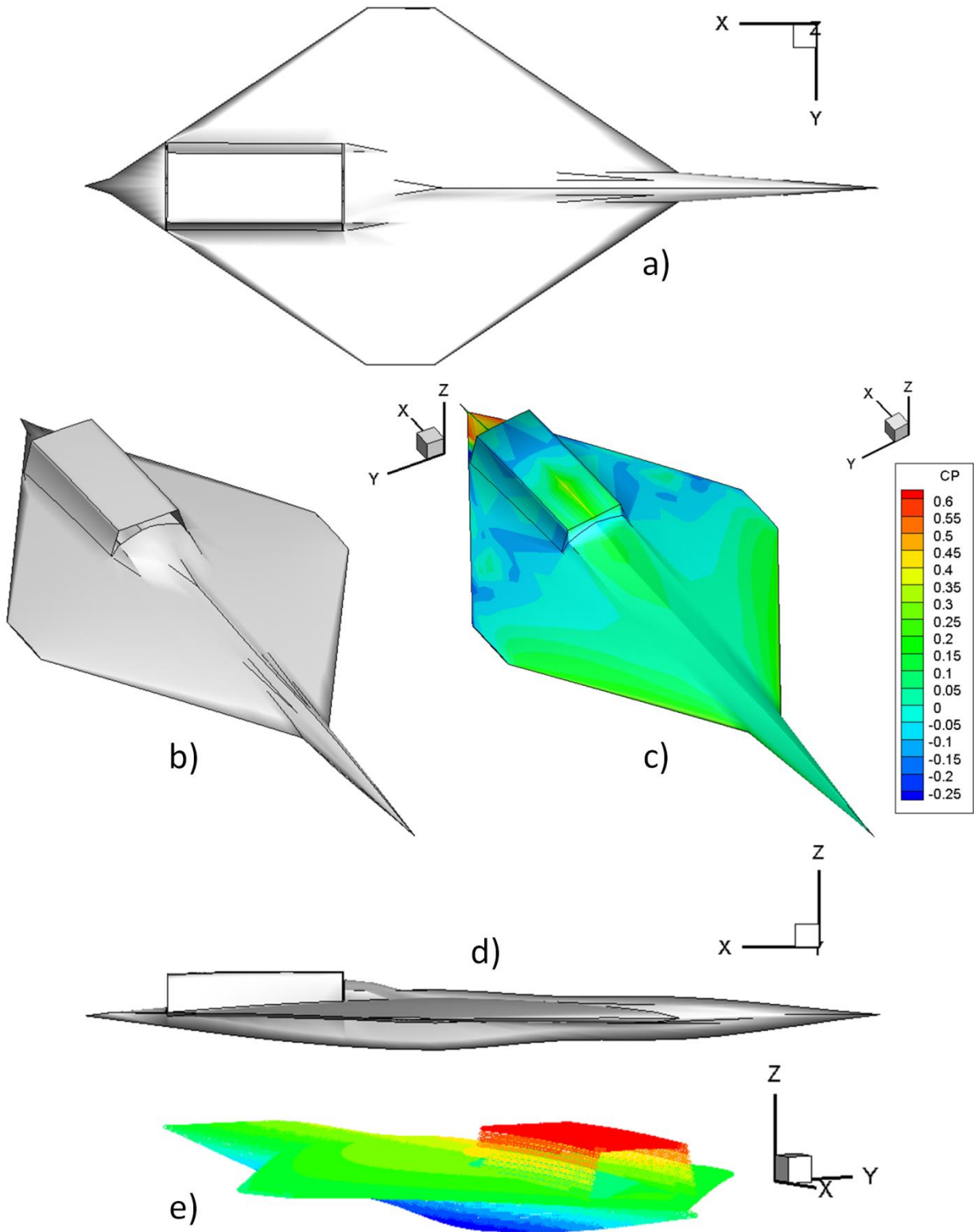


Figure 6.3-3. Final higher-fidelity ESAV MDO optimal configuration. From a) to e) it is the top view, *.stl ISO, ZONAIR M=2.2 C_p ISO, side view, and back-angled view.

6.4 Analysis of Final Optimal Higher-Fidelity ESAV MDO Configuration

6.4.1 Other Trapezoidal Wing Designs

The trapezoidal wing is not a new configuration for high speed aircraft. A recent example of a modern supersonic aircraft that has employed this type of wing design is the YF-23. The YF-23's first flight was in 1990. Its top speed is $M=2.2+$, and its wing loading is almost the same as the optimal ESAV configuration at 54 lbs./ft². A picture of this aircraft is shown in Fig. 6.4-1. Not only is the trapezoidal planform similar to the optimal ESAV found here, but its area ruling coke bottling is similar as well (see Fig. 6.3-3d).



Figure 6.4-1. Northrop YF-23 side view and top view showing trapezoidal planform and area ruling features.

A more recent concept by Boeing for the so-called F/A-XX aircraft (or sixth-generation air dominance fighter) also has a trapezoidal planform. This aircraft, like the ESAV, is a tailless design [119]. Although it has never been built or flown, it is encouraging that similar aircraft to that identified by the higher-fidelity ESAV MDO successive NSGA-II optimization are being considered for future military aircraft. This tailless, high-speed aircraft concept is shown in Fig. 6.4-2.



Figure 6.4-2. A Boeing concept for the F/A-XX aircraft reported by Aviation Week & Space Technology, April 1, 2013 on page 13.

6.4.2 Other Optimal ESAV Characteristics

Several other aspects of this final configuration are of interest besides the TOGW. For instance, its RCS and noise generation levels are important for survivability issues. Some of these other

outputs are shown in Table 6.4-1. Interestingly, the lowest RCS design from the second NSGA-II run was a trapezoidal wing with slightly different sweep angle (about one degree different), and was heavier than the optimal by about 40,000 lbs. The RCS value for this configuration cuts the maximum detection range by almost half. The design with the lowest combined noise generation is an arrow wing that flies more slowly ($M=2.24$), with a longer fuselage, and a shorter wing span (higher wing loading). Of course, these are not optimized for a smaller RCS or combined noise value, but are merely chosen from the 785 designs from the second NSGA-II optimization run.

Table 6.4-1. Other Parameters of Interest for Final Higher-Fidelity ESAV Design

Output Parameter	Units	Value
Flyover Noise Over FAR 36 Levels	dB	23.02
Sideline Noise Over FAR 36 Levels	dB	33.62
Combined Noise Over FAR 36 Levels	dB	28.32
Wing Area	ft ²	3398.1
Thrust to Weight Ratio	N/A	0.41
Wing Loading	lbs/ft ²	53.43
Flight Time	Minutes	203.8
Stall Speed Take-off	Knots	140.45
Stall Speed Landing	Knots	124.65
Critical Field Length	ft.	4260.96
Landing Distance	ft.	4994.71
C _{D0}	N/A	0.00321

6.4.3 Low Cost Range Extension of ESAV Design

The major cost to finding the higher-fidelity ESAV MDO optimal was the aerodynamics code, ZONAIR. The second most expensive analysis is NPSS/WATE++, although this is much less expensive than ZONAIR. Since the higher-fidelity optimal ESAV is found using a genetic

algorithm, there is no guarantee that it is at even a local optimum. However, due to the large computational cost of performing a SQP optimization starting from the optimal design, it is not desired to do this.

If the engine size and OML do not change in any way, the aerodynamics simulation already performed for that fixed design throughout the flight envelope (different Mach numbers and altitudes) would not have to be rerun. In other words, two of the design variables, maximum cruise Mach number and cruise altitude (or in other words the mission design variables) could be varied by an optimizer with no need to run the higher-fidelity aerodynamics and propulsion models.

A small study, that required about 30 minutes in total, was undertaken to see if a longer range can be obtained for the optimal ESAV by varying the Mach number and altitude, while leaving everything else fixed. Therefore, the objective function is changed from minimizing the TOGW to maximizing the range. The TOGW of the higher-fidelity successive NSGA-II method is fixed for this proposed range study. The 2.5g turn and the payload are still assumed part of the mission. If a true global optimum was found previously, then the maximum range would be 4000 n.mi.

The optimization procedure for this small study was to run a NSGA-II optimization. The resulting optimum was then to be used as the starting point for a following SQP gradient optimization method so that the stationary point could be found that corresponds to the NSGA-II optimum. With the objective function being the maximization of the range for a fixed TOGW, the cruise Mach number ends up being just greater than 2.2, the cruise altitude is 51,374 ft. This produces a maximum range of 4269 n.mi. These results are shown in Table 6.4-2.

Table 6.4-2. New Optimum Resulting from Small Thirty Minute Mission Optimization

Design Variable	Units	Old Value	New Value
Maximum Cruise Mach Number	N/A	2.47	2.20
Cruise Altitude	ft.	56,653	51,374
Range	n.mi.	4000	4269

In this case, the NSGA-II method found a stationary point, because the SQP method found that the optimum was its starting point. This small study shows that the optimum found from the higher-fidelity ESAV MDO successive NSGA-II method is really not a global optimum after all, but this is not surprising. However, with the current lack of a true global optimization method, strategies such as above, which only cost about 30 minutes, can be very valuable in understanding the design space and operational behavior of an aircraft design better.

6.5 Higher-Fidelity ESAV MDO Sensitivities

The ANOVA method was once again employed to determine the sensitivities of the design variables on the ESAV characteristics within a $\pm 2\%$ ball of the optimal configuration. As before, if the ball exceeds the global design space bounds, these are truncated to be within the global bounds. Owing to the fact that a higher-fidelity structural model is still being developed in another effort for the higher-fidelity ESAV MDO framework, an additional variable is used in the sensitivity study. This variable is called “WingWeightFactor.” As the name suggests, it is a multiplier factor used to adjust the wing weight calculated by the FLOPS weights module by $\pm 2\%$. In this way, the sensitivity of output characteristics, like TOGW, to the wing weight can be analyzed using the ANOVA method.

The ANOVA method sensitivities are calculated by creating a surrogate Kriging model within the $\pm 2\%$ ball of the neighborhood around the optimal ESAV configuration identified

previously. This Kriging model is then used by the ANOVA method to determine which input (independent) design variables, and combinations of design variables, have the greatest influence on the output (dependent) ESAV parameters. In this way, insight is hoped to be gained into what the most important factors are in designing this aircraft. Any input variable that has less than a 1% sensitivity indice is deemed to be statistically insignificant, and *vice versa*. The higher the sensitivity indice, the more influential it is on the output parameter. The design variable names found in the charts were given previously. The sensitivity indices are in percent on the horizontal axis and the vertical axis are a list of the independent variables on all sensitivity charts.

A good test to see if the sensitivities being predicted by the ANOVA analysis make sense is to take an aircraft output parameter that can only be attributed to known effects and see if those are the only effects being shown. Here, the aspect ratio was used for this purpose. Its sensitivity plot is shown in Fig. 6.5-1. Indeed, the chord and span variables are alone on the aspect ratio sensitivity plot, with a definitive zero for “Other” showing the validity of ANOVA.

The first chart to be analyzed is the TOGW sensitivity plot since this is the objective function. Figure 6.5-2 shows that many of the first-order and second-order sensitivity indices have a statistically significant effect on the TOGW. The three highest parameters are the root chord, fuselage width, and tip chord respectively. Only 13 of the possible 27 variables are statistically significant in either a first- or second-order sensitivity indice. A look at the individual weight components that make up the TOGW can clarify where these indices are coming from.

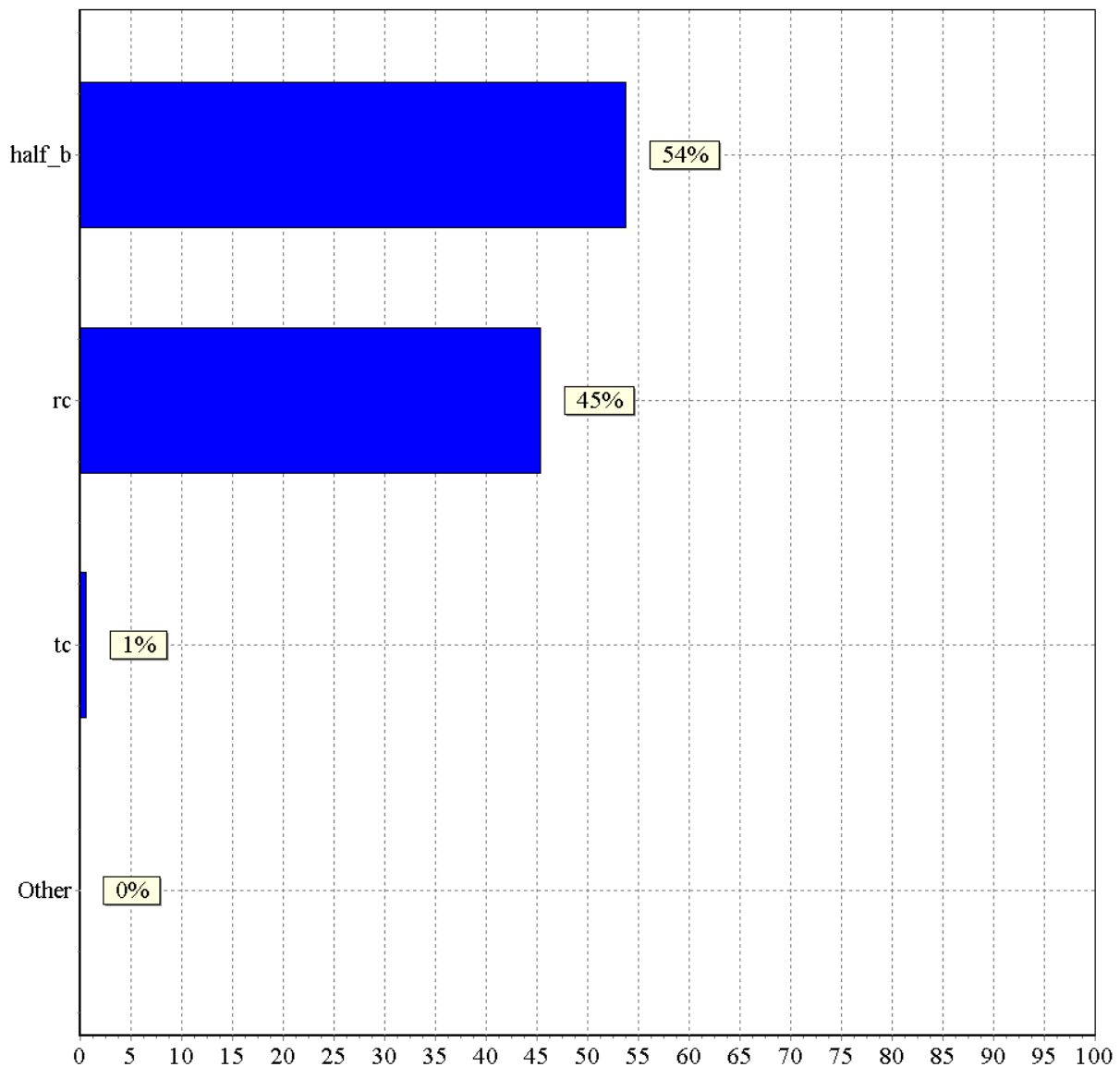


Figure 6.5-1. Higher-fidelity ESAV MDO framework aspect ratio sensitivity chart.

The fuel weight sensitivities look almost identical to the TOGW sensitivities as seen in Fig. 6.5-3. Since the main effects are tied to the wing and fuselage shape, drag seems to be the most likely cause for the fuel weight sensitivity plot makeup. However, this must be induced drag and wave drag since: 1) the C_{D0} drag sensitivities do not match the fuel weight sensitivities as seen in Fig. 6.5-4, and 2) the variables have to do with the wing shape and fuselage shape,

which can easily be tied to induced drag and wave drag. One thing to note is that most of the C_{D0} influence comes from the many design variables that make up the “Other” category.

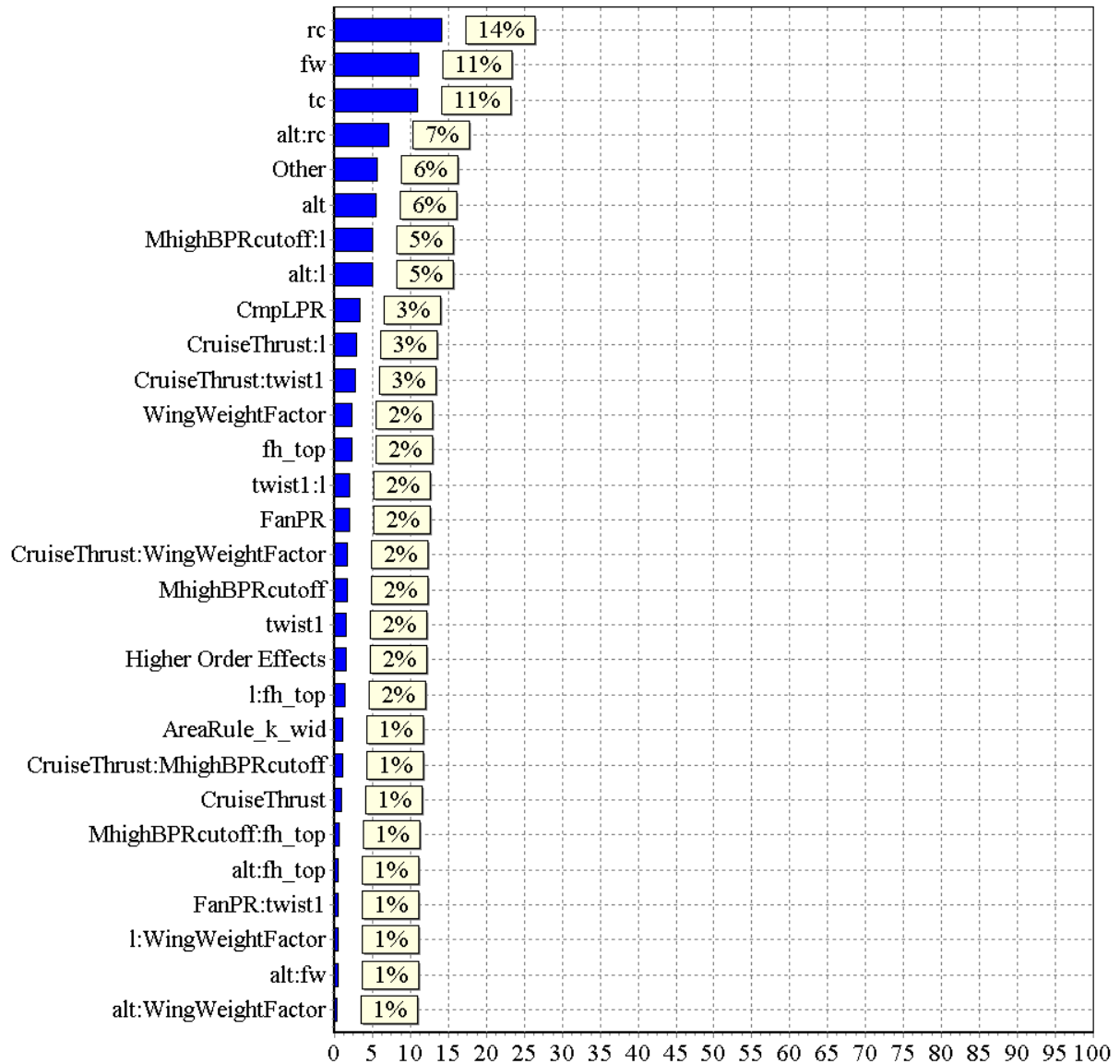


Figure 6.5-2. Higher-fidelity ESAV MDO framework TOGW sensitivity chart.

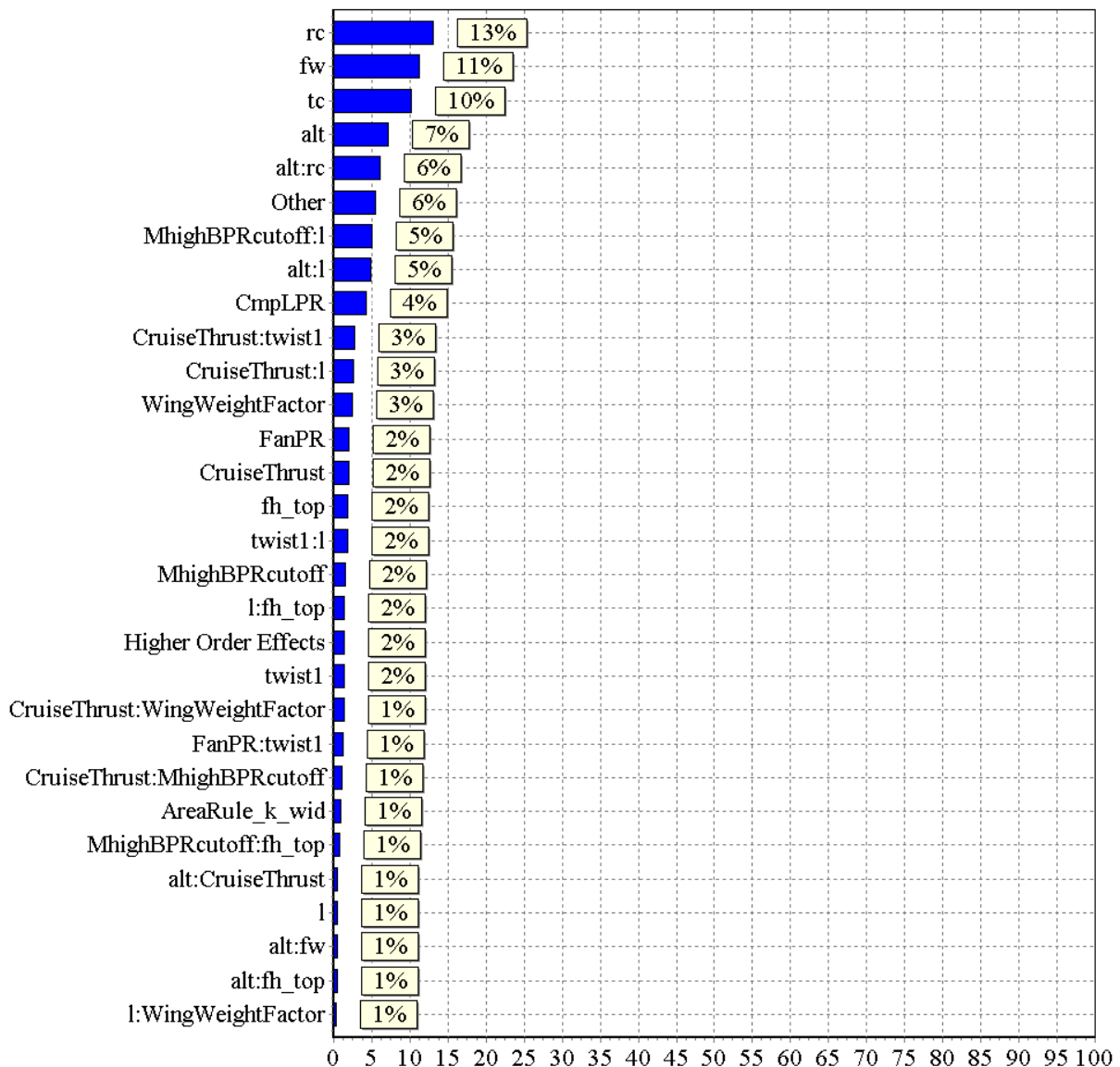


Figure 6.5-3. Higher-fidelity ESAV MDO framework fuel weight sensitivity chart.

Looking at the structural weight sensitivities in Fig. 6.5-5 shows the dry weight effect on the fuel weight is also a reason for the sensitivities. Out of the top four design variables in the structural weight sensitivities plot, three are the same (and in the same order) as both the fuel weight and the TOGW. The heavier the aircraft, the more fuel is required, and this relationship is evident here. The wing weight factor has a small effect on the structural weight.

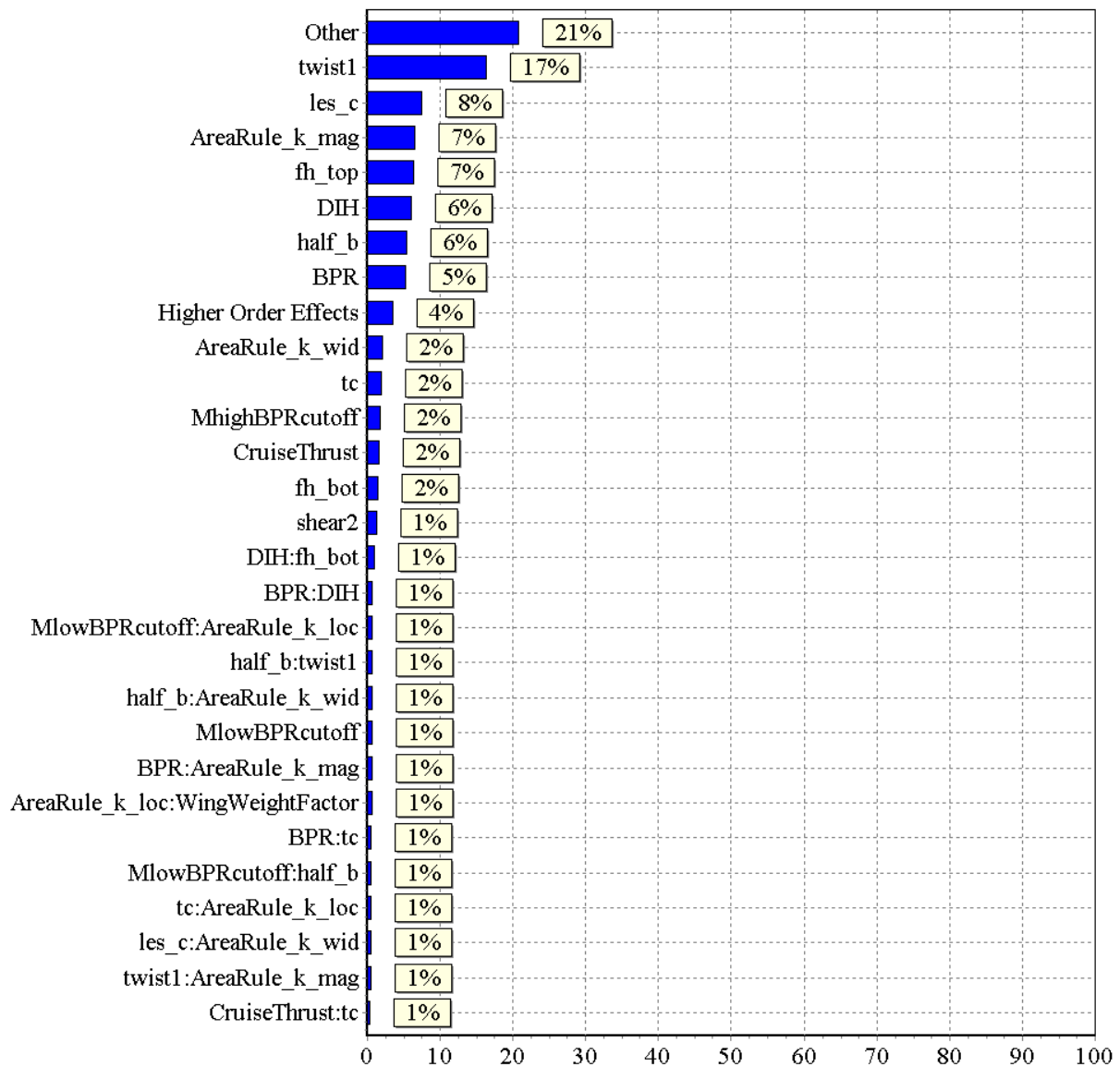


Figure 6.5-4. Higher-fidelity ESAV MDO framework C_{D0} sensitivity chart.

The altitude has a significant effect on the TOGW as well, not only as a first-order indice, but in several of the second-order sensitivity indices too. This can be explained by the propulsion weight sensitivity plot seen in Fig. 6.5-5 where the cruise altitude is the dominant design variable. The cruise condition is the engine design point, and thus the cruise altitude has a large

effect on the engine's size (weight) and SFC. Of course, the cruise thrust and Mach number are also significant.

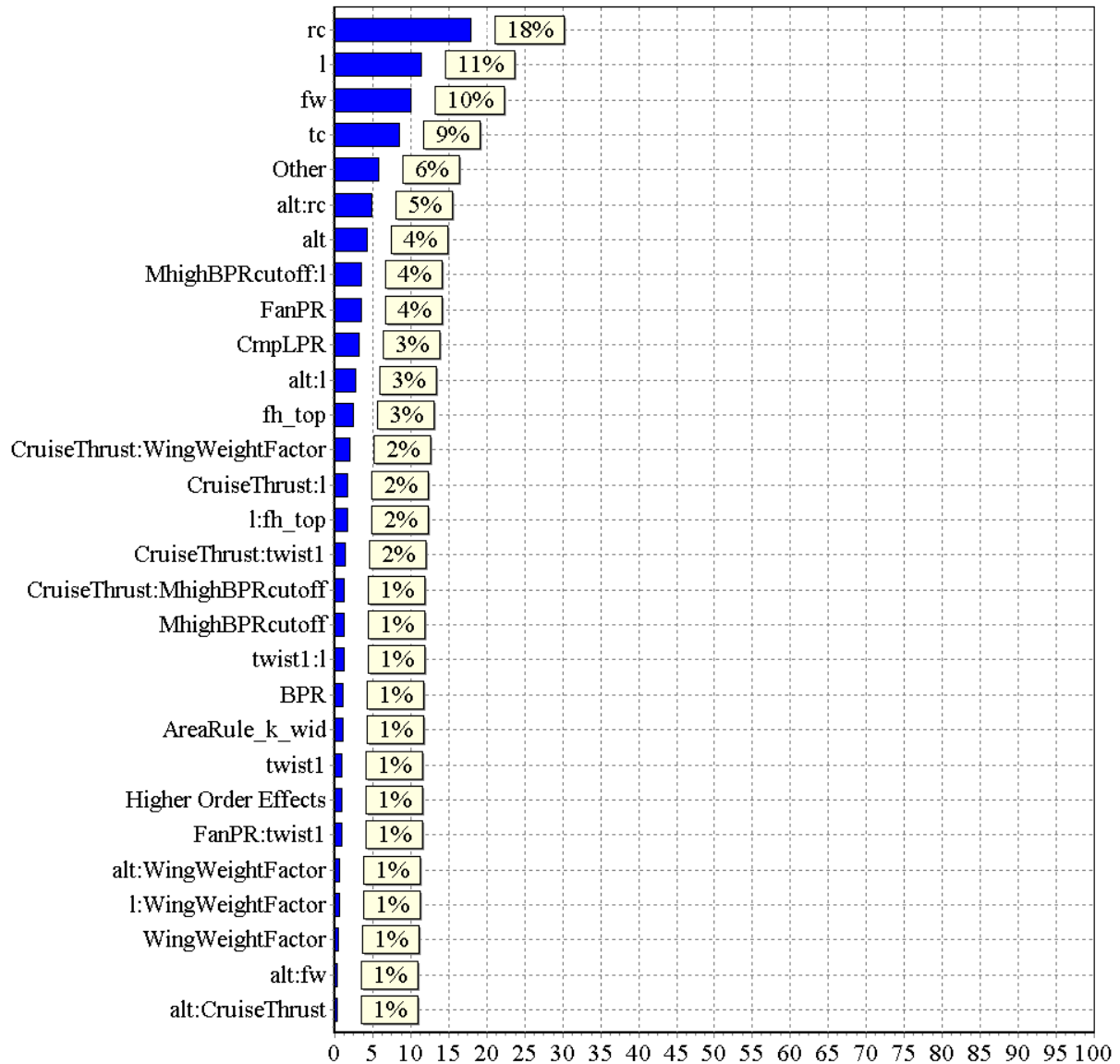


Figure 6.5-5. Higher-fidelity ESAV MDO framework structural weight sensitivity chart.

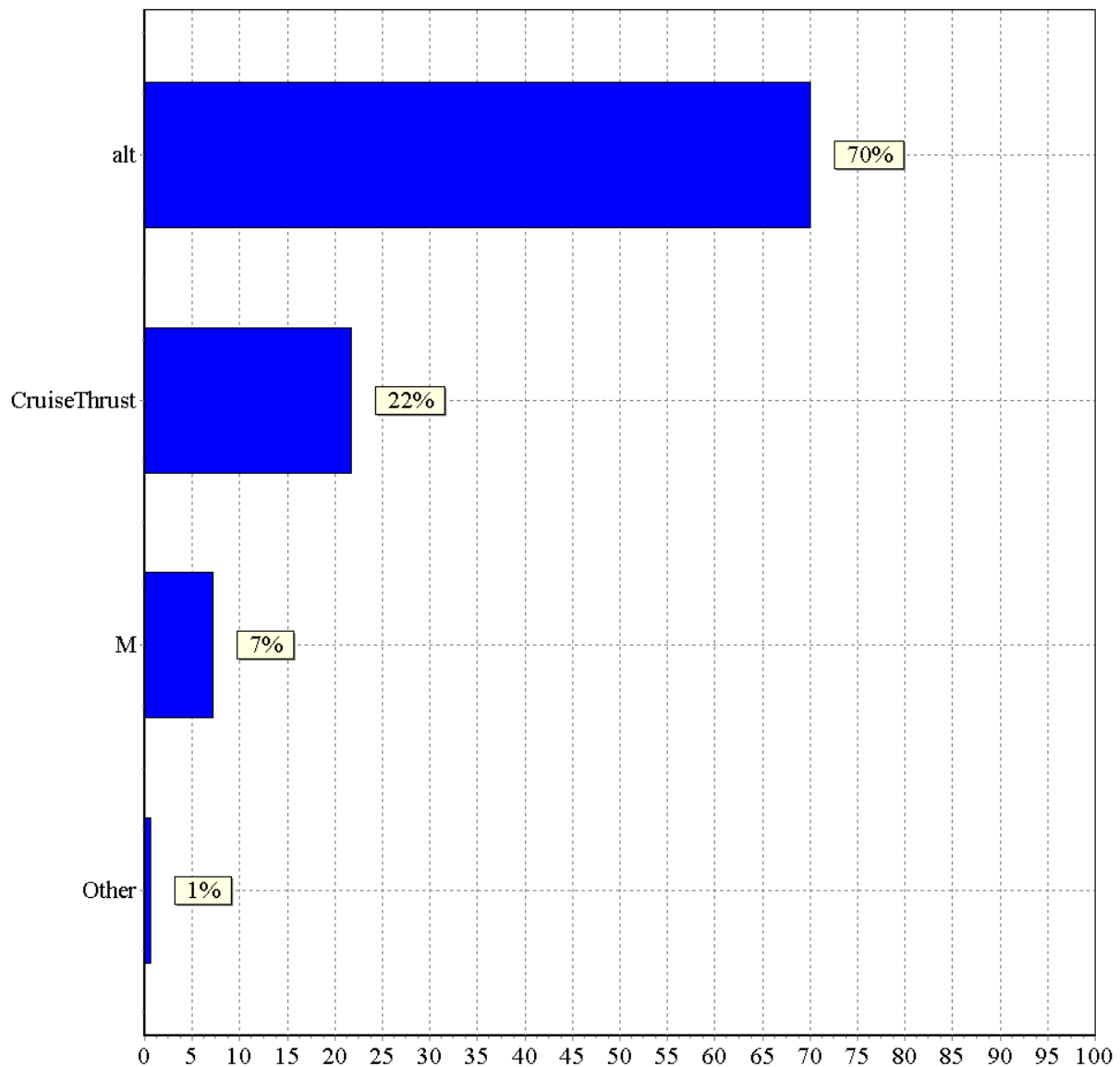


Figure 6.5-6. Higher-fidelity ESAV MDO framework propulsion weight sensitivity chart.

Other aircraft characteristics are also included in the ANOVA analysis. Specifically, the RCS, noise, thrust-to-weight ratio, and wing loading. The RCS, shown in Fig. 6.5-7, is sensitive to almost all of the design variables, including several engine variables. This can be accounted for by the fact that almost all parameters affect the aircraft OML in some way. This makes sense, because radar is reflected by the OML.

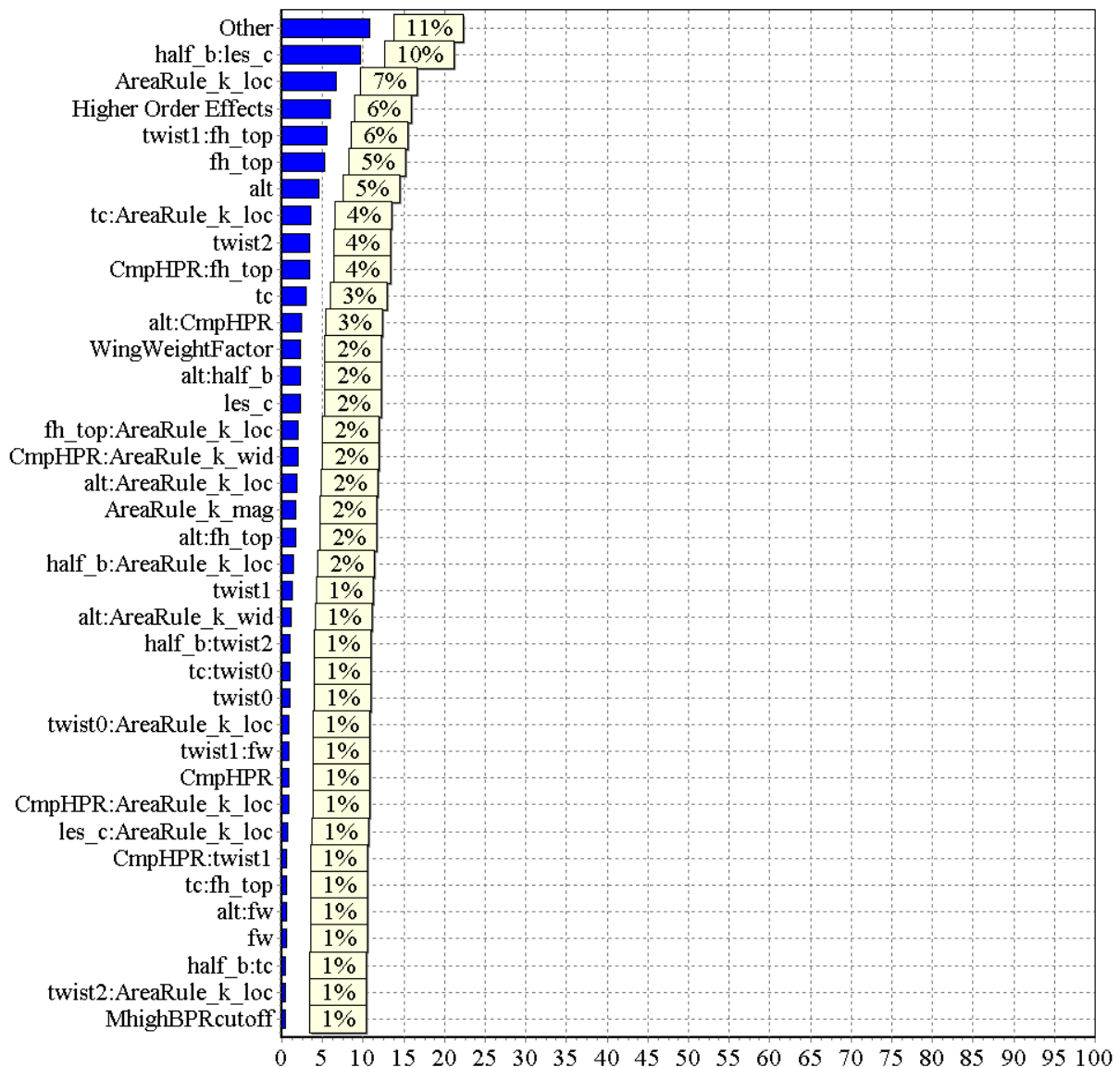


Figure 6.5-7. Higher-fidelity ESAV MDO framework RCS sensitivity chart.

The noise generated by the ESAV is most sensitive to variables that affect the engine design. This is to be expected because the largest contributor to noise is the engine exhaust velocity. For the low-fidelity ESAV MDO optimal design (arrow wing), the noise was shown to be sensitive to the tip chord. However, for this trapezoidal wing, the tip chord has a very small

effect on the aircraft's ability to shield the engine noise, and it therefore only has a 1% sensitivity indice.

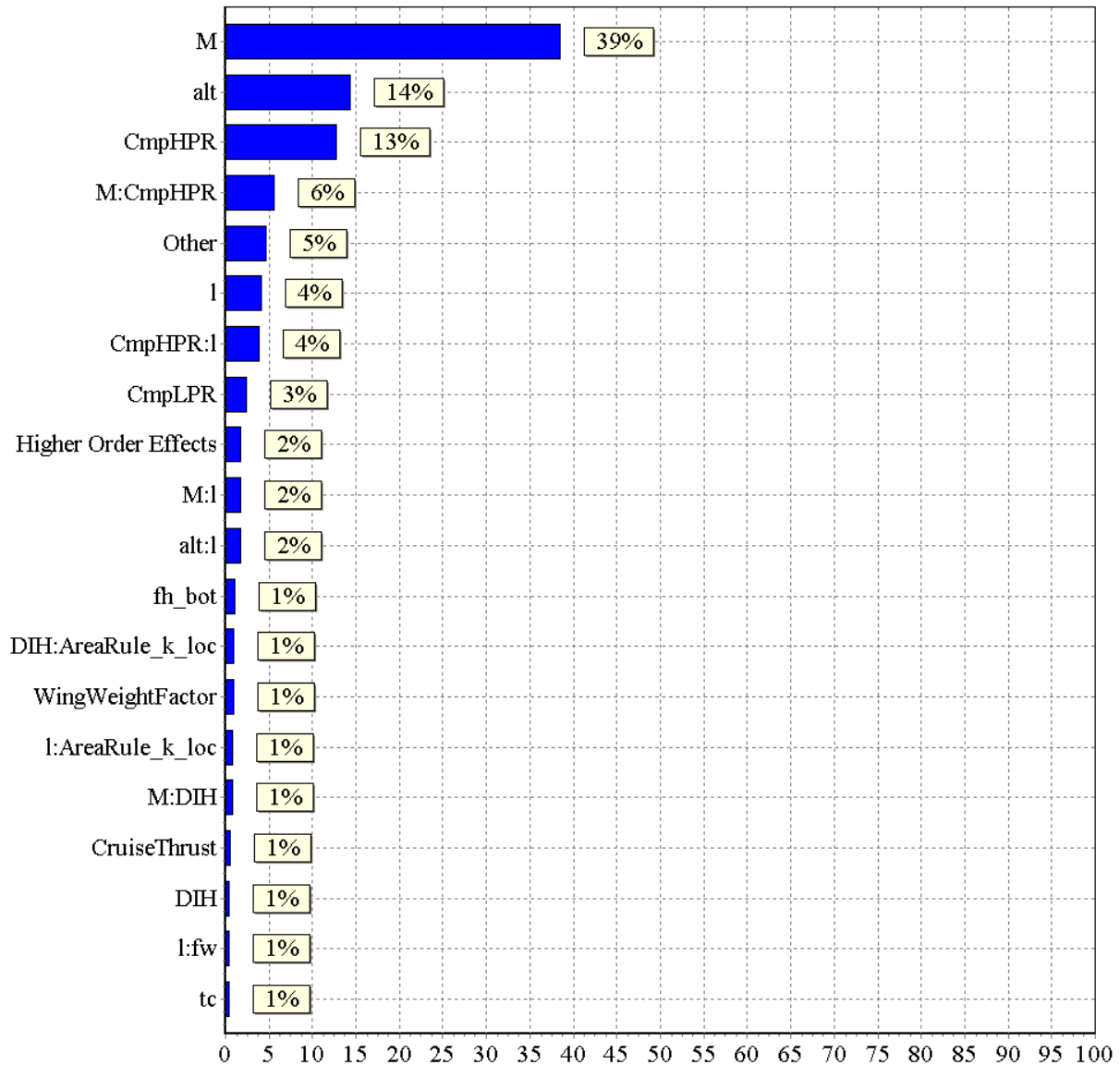


Figure 6.5-8. Higher-fidelity ESAV MDO framework noise sensitivity chart.

The thrust-to-weight (T/W) sensitivity plot, shown in Fig. 6.5-9, shows a large dependence on the design variables that affect the engine size, rather than on those that affect the weight. Likewise, the wing loading, W/S, sensitivity plot favors those variables that have the

greatest effect on the weight before those that affect the wing area. This can be seen in Fig. 6.5-10. This trend was previously seen in the low-fidelity T/W and W/S sensitivity plots, where the most influential design variables are those related to the numerator rather than the denominator.

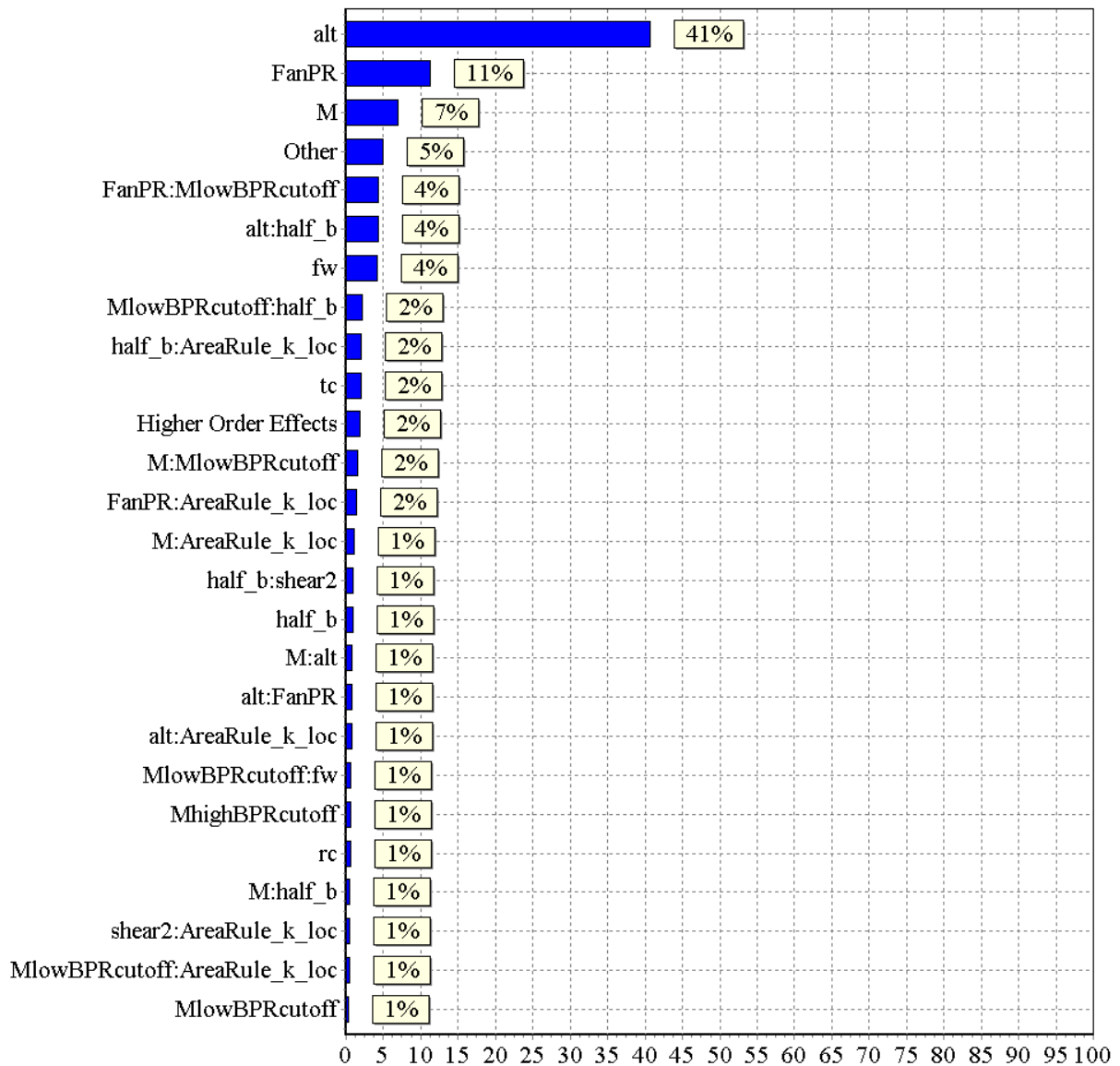


Figure 6.5-9. Higher-fidelity ESAV MDO framework thrust-to-weight ratio sensitivity chart.

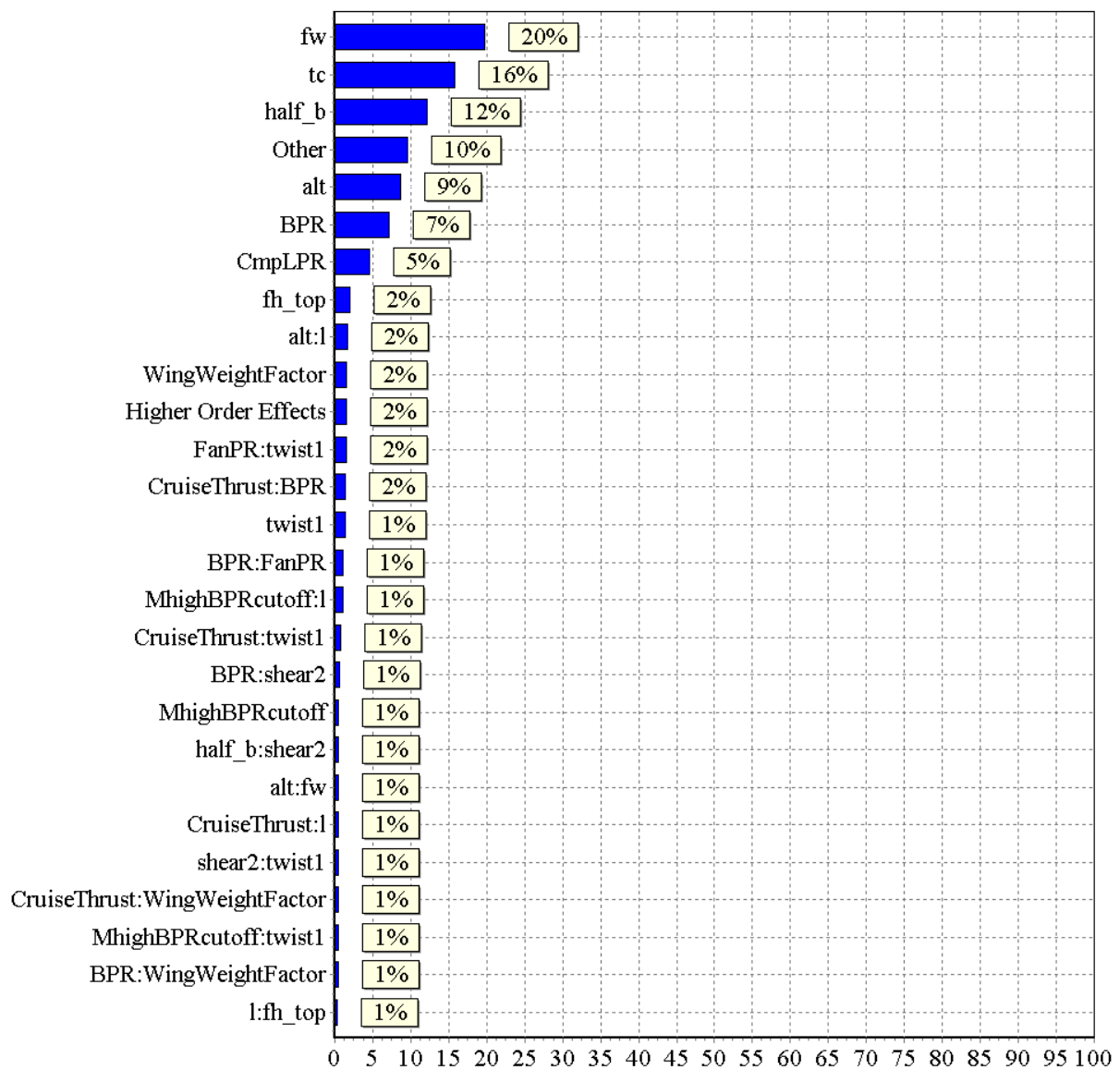


Figure 6.5-10. Higher-fidelity ESAV MDO framework wing loading sensitivity chart.

7. Conclusions

7.1 Conclusions

The focus of this work is a particular aircraft called an Efficient Supersonic Air Vehicle (ESAV). This is a long-range military bomber class aircraft has been designed for high speed (supersonic) flight and survivability. The design metric used here to differentiate designs (i.e. decide if one design is better than another) is the take-off gross weight (TOGW). That is, out of all possible ESAV designs in its design space, the one with the smallest TOGW that is still feasible is the optimal.

In order to find the optimal ESAV configuration, modular, multifidelity, and multidisciplinary analysis and optimization framework tools have been constructed. A key part of these frameworks is the object-oriented, multifidelity, VT-CST geometry model that offers quick runtimes, a watertight analytical representation of the ESAV geometry, and the ability to represent the various required geometry formats from all disciplines from the same single geometry representation.

Two ESAV MDO frameworks were built at two different fidelity levels. The first was a low-fidelity suite of codes, and the second upgrades a few of the key disciplines to higher-fidelity models and adds others not included in the low-fidelity framework. Specific challenges to designing an ESAV are addressed with this tool. For example, not only have several disciplines been included in the higher-fidelity framework that would normally be required for supersonic aircraft design, but the framework also includes disciplines specifically required by this particular advanced military aircraft. The analysis codes in the framework were validated by comparison with the B-58 supersonic bomber aircraft.

The worth of considering the whole aircraft system together at once in the MDO framework rather than the compartmentalized discipline design approach was clearly shown. That is, changing one or more design variables in one discipline was seen to non-trivially affect one or more other disciplines.

Given the very large design space, not every possible ESAV design can be analyzed. Two optimization methods were developed in an attempt to find a global ESAV optimum. The first uses statistical methods tools available from the field of the design of computer experiments (DOE) and was called the DOE method. The second method is an extrapolation of the current MDO state-of-the-art using the NSGA-II optimization algorithm. This method is called the successive NSGA-II method.

In general, if the computational resources of an optimization problem allow the more methodical DOE method, and its much higher computational cost, it should be undertaken. However, if computational resources are limited then the successive NSGA-II method can find a good design at a cost an order-of-magnitude less than the DOE method.

The low-fidelity ESAV MDO framework offers a design tool that has a low computational cost; therefore it can be used with the computationally expensive DOE method. The DOE method is a systematic process for exploring, understanding, and reducing the complete ESAV design space, and was formulated and exercised using the low-fidelity ESAV MDO framework. The largest computational expense in this process is an initial, large DOE exploration of the design space to understand a promising design space area to conduct the optimization steps in. The low-fidelity minimum TOGW feasible ESAV design found using the DOE method was a clipped arrow wing ESAV design. Relationships and trends between the

chosen design variables and aircraft characteristics are established using an analysis of variance (ANOVA) method.

A higher-fidelity ESAV MDO framework that included new, upgraded, and some of the same analysis models from the low-fidelity ESAV MDO framework was constructed. A detailed description of the process to choose the higher-fidelity propulsion model was given. Supersonic flight requires a very good estimate of propulsion performance in and of itself because the fuel weight is often a very large percentage of the total aircraft gross weight. In addition, the embedded engines required on the ESAV demanded the higher-fidelity propulsion model to account for the associated complex physical effects. The low-fidelity linear aerodynamics model was deemed inappropriate for high-curvature conical camber supersonic wings (e.g. the validation case showed that the model could not capture conical camber aerodynamic effects) and the need for an accurate drag prediction for the unknown ESAV configurations. The ZONAIR aerodynamics model, upgraded the low-fidelity counterpart. The TOGW was indeed sensitive to the drag, so the higher-fidelity aerodynamics model was required, even with its significantly higher computational cost.

Two new models introduced in the higher-fidelity ESAV MDO framework were needed for the specific challenges associated with the military application of the ESAV. These two disciplines are the engine exhaust-washed structures (EEWS) and the radar cross section (RCS) models. The EEWS discipline was to capture the detrimental effects of hot engine exhaust gases on the aircraft aft structures due to the ESAV requirement of embedded engines. The RCS discipline was needed to understand the ESAV radar survivability characteristics.

The less expensive computational effort of the low-fidelity effort was hoped to be used to guide the higher-fidelity ESAV MDO framework optimization process, saving higher-fidelity

effort. This multifidelity approach did not work for this particular application, because the higher-fidelity ESAV MDO framework could not verify the low-fidelity ESAV results – including the low-fidelity optimal ESAV. This result showed that the low-fidelity framework would misguide the higher-fidelity framework if used in that capacity. The mismatching was generally due to the drag prediction difference between the low-fidelity, linear aerodynamics and the higher-fidelity ZONAIR aerodynamics.

The large DOE method is not applicable to the higher-fidelity ESAV MDO framework directly because of the very high computational cost. However, the successive NSGA-II method requires about an order-of-magnitude fewer objective function evaluations. The successive NSGA-II method had been developed and tested previously using the low-fidelity framework. This method introduced an approach to use the higher-fidelity ESAV MDO framework directly to find the optimum ESAV configuration. Even with the order-of-magnitude reduction in required runs, the higher-fidelity ESAV MDO framework still took just over a month of computational time on a single, eight-core, 2.27 GHz, desktop computer to find the optimum, trapezoidal wing, ESAV configuration.

The low-fidelity optimum was analyzed using the higher-fidelity framework and *vice versa*. The arrow wing, low-fidelity, ESAV optimum was not feasible according to the higher-fidelity framework due to drag. Similarly, the trapezoidal wing, higher-fidelity, ESAV optimum was judged to be not feasible by the lower-fidelity framework because the cruise thrust was predicted to be insufficient at the cruise altitude. This shows that the lower-fidelity framework would not have been able to guide the higher-fidelity optimization process to the trapezoidal wing optimum, because this area of the low-fidelity design space is infeasible.

The low-fidelity mismatch and the subsequent inability to use low-fidelity results to guide the higher-fidelity analyses portray an important conclusion from this effort. That is, low-fidelity MDO models must be able to faithfully compute aircraft characteristics in all disciplines important to the objective function and constraints in order to be useful to a higher-fidelity MDO model. In the ESAV case, the low drag predictions led to a smaller engine. This, coupled with the large supersonic growth factor, caused the optimizer to find an area of the low-fidelity design space that was infeasible according to the higher-fidelity ESAV MDO framework.

A related conclusion to the drag prediction mismatching is that there is a need to develop a better medium-fidelity, aerodynamics code more appropriate to solve the ESAV MDO problem. It is suggested that this code should be a higher-order panel method that can handle

- the transonic regime,
- viscosity effects,
- compute stability derivatives, and
- be able to run on a massively parallel compute infrastructure.

The eight-core machine that ran the parallel ZONAIR model did make it tractable, but just barely. A massively parallel computing scheme would have allowed even the DOE optimization method to be used directly with the higher-fidelity ESAV MDO framework.

The optimal ESAV found with the higher-fidelity ESAV MDO framework is a trapezoidal wing planform that did not resemble the low-fidelity clipped arrow wing design. As the higher-fidelity optimization progressed during the second NSGA-II run, the optimizer considered both arrow and trapezoidal wing designs. In later final generations of the NSGA-II optimization, the algorithm focused on the trapezoidal planform and discarded the arrow wing

design. The trapezoidal wing result is not surprising given that similar aircraft, like the YF-23 and Boeing's concept for the F/A-XX aircraft also have trapezoidal wings.

7.2 Future Work

The most immediate needs to take the next step in the development of the ESAV MDO framework is to add a stability and control model and a higher-order structures model. The tailless design specification of the ESAV makes this aircraft particularly difficult to control, especially during low-speed flight and in yaw. A stability and control model is currently being developed in another effort, so this framework deficiency will hopefully be resolved shortly. A structures model will add the ability to obtain physics-based estimates of the structural weight of the aircraft. The sensitivity studies showed that the structural weight was often very closely tied to the TOGW.

The wing weight parameter in the higher-fidelity ESAV MDO framework sensitivity study was shown to have a statistically significant effect on the TOGW, but it was small compared to effects with the larger sensitivity indices. However, through numerous examples of exercising the low-fidelity and higher-fidelity ESAV MDO frameworks, the optimizer can sometimes exploit the lack of a physics-based structural weight model and get stuck in an area of the design space that has a very low drag, but *would* have a very high structural weight not detected by the low-fidelity (empirical) structural weight model. The stability and control model is expected to have a similar beneficial effect in keeping the optimizer away from design space areas that have good characteristics, but otherwise are actually infeasible due to being uncontrollable.

Once the framework has been enhanced with these two additions, a larger design space can be explored. An immediate option is to include planforms that have trailing and leading edge

breaks. The objective function may also be varied to other possible measures of merit, or a multiobjective function could be constructed to allow simultaneous minimization of the TOGW, RCS, and noise.

Another design space study might be to try three engines rather than four to understand the tradeoff between fewer engines more engines. Fewer engines would have less secondary weight (a certain amount of weight is required for each engine to be integrated onto the aircraft), but each engine itself would be larger. Therefore, the overall propulsion weight for three engines may be smaller, but the larger size may have unintended consequences, like increased drag. This trade space should be studied for any future ESAV design.

The current $M=2.5$ upper limit on the ESAV is in place to avoid high-speed heating effects that begin to emerge in a higher Mach number range. Future ESAV MDO framework efforts might address this challenge and provide methods to accurately analyze aerodynamic heating and the corresponding design challenges.

The current higher-fidelity ESAV MDO optimal design identified in this work, even with its limited Mach number range, still represents a capability that the USAF does not currently have. That is, an optionally-manned, low-observable aircraft that can deliver military payload over a long range in anti-access and area-denial situations – a scenario that becomes more probable in today's changing world.

Bibliography

1. **Vanderplaats, Garret N.** *Numerical Optimization Techniques for Engineering Design*. New York : McGraw-Hill Inc., 1984.
2. **Mangasarian, O. L.** *Nonlinear Programming*. New York : McGraw-Hill Inc., 1969. 070398852.
3. *Role of Figures of Merit in Design Optimization and Technology Assessment*. **Jensen, S. C., Rettie, I. H., and Barber, E. A.** 2, s.l. : American Institute of Aeronautics and Astronautics, 1981, Journal of Aircraft, Vol. 18. 0021-8669.
4. **Nicolai, L. M., and Carichner, G. E.** *Fundamentals of Aircraft and Airship Design: Volume I - Aircraft Design*. [ed.] J. A. Schetz. Reston : American Institute of Aeronautics and Astronautics, Inc., 2010. Vol. 1. 9781600867514.
5. **Progress in Astronautics and Aeronautics.** *Advances in Collaborative Civil Aeronautical Multidisciplinary Design Optimization*. [ed.] E. Kessler, M. D. Guenov and F. K. Lu. Reston, VA : American Institute of Aeronautics and Astronautics, Inc., 2010. Vol. 233. 9781600867255.
6. *Aerodynamic Design via Control Theory*. **Jameson, Antony.** 3, s.l. : Journal of Scientific Computing, 1988, Vol. 3.
7. **Ferran, Lee.** Obama: Hey Iran, Can We Get Our Drone Back? *ABC - Investigative Unit*. [Online] ABCNEWS, 12 12, 2011. [Cited: 10 5, 2012.] <http://abcnews.go.com/Blotter/obama-asks-iran-rq-170-sentinel-drone-back/story?id=15140133#.UG8UElGHNPE>.
8. *Air-Combat Strategy Using Approximate Dynamic Programming*. **McGrew, J. S., How, J. P., Williams, B., and Roy, N.** 5, s.l. : American Institute of Aeronautics and Astronautics, 2010, Journal of Guidance, Control, and Dynamics, Vol. 33.

9. **Krepinevich, A. Watts, B. and Work, R.** *Meeting the Anti-Access and Area-Denial Challenge*. Washington, D.C. : Center for Strategic and Budgetary Assessments, 2003.
10. **Thomas, J.** *China's Active Defense Strategy and its Regional Implications*. Washington, D.C. : Center for Strategic and Budgetary Assessments, 2011. Testimony before the U.S.-China Economic and Security Review Commission.
11. **Majumdar, D.** Anti-access/area denial challenges give manned aircraft edge over UAVs. *Flightglobal*. [Online] 7 25, 2012. [Cited: 10 5, 2012.]
<http://www.flightglobal.com/news/articles/anti-accessarea-denial-challenges-give-manned-aircraft-edge-over-uavs-374745>.
12. **Moran, M. J., and Shapiro, H. N.** *Fundamentals of Engineering Thermodynamics, 4th Ed.* New York : John Wiley & Sons, Inc., 1999. 0471317136.
13. *Proof of the Double Bubble Conjecture*. **Hutchings, M., Morgan, F., Ritore, M., and Ros, A.** pp. 45-49, s.l. : Electron. Res. Announc. Amer. Math. Soc., July, 2000, Vol. 6.
14. **Burns, J. A.** Notes on Calculus of Variations With Modern Applications to Control Theory, Numerical Methods, and Differential Equations. *www.math.vt.edu*. [Online] [Cited: Jan. 3, 2012.] http://www.math.vt.edu/people/jaburns/class_policy/Lecture_notes.pdf.
15. **Ewing, G. M.** *Calculus of Variations with Applications*. New York : Dover Publications, Inc., 1985. 0486648567.
16. **Luenberger, D. G., and Ye, Y.** *Linear and Nonlinear Programming*. New York : Springer, 2008. 9780387745022.
17. **Nocedal, J., and Wright, S. J.** *Numerical Optimization, 2nd Ed.* New York : Springer, 2006. 9780387303031.

18. Boeing 777 Facts. *Boeing*. [Online] [Cited: April 08, 2012.]
http://www.boeing.com/boeing/commercial/777family/pf/pf_facts.page.
19. Airplane Timeline. *National Academy of Engineering - Great Achievements*. [Online] [Cited: April 08, 2013.] <http://www.greatachievements.org/?id=3728>.
20. **Hicks, R. M., Murman, E. M., and Vanderplaats, G. N.** *An Assesment of Airfoil Design by Numerical Optimization*. NASA Ames, Moffet Field : National Aeronautics and Space Administration, 1974. NASA-TM-X-3092.
21. *Optimal Shape Design of Supersonic Bypass Inlet*. **Kim, H. J., Kumano, T., Liou, M. S., and Povinelli, L. A.** Fort Worth : American Institute of Aeronautics and Astronautics, 2010. 10th AIAA Aviation Technology, Integration, and Operations Conference. AIAA-2010-9171.
22. **Giunta, A. A.** *Aircraft Multidisciplinary Design Optimization Using Design of Experiments Theory and Response Surface Modeling Methods*. Blacksburg : Virginia Tech, 1997. PhD Dissertation.
23. *An MDO Investigation of the Impact of Practical Constraints on an HSCT Configuration (Multidisciplinary Design Optimization)*. **MacMillin, P. E., Golovidov, O. B., Mason, W. H., Grossman, B., and Haftka, R. T.** Reno, NV : AIAA 35th Aerospace Sciences Meeting, 1997. AIAA-1997-98.
24. *A Multilevel Decomposition Procedure For Efficient Design of High Speed Civil Transport*. **Chattopadhyay, A., and Pagaldipti, N.** Reno, NV : AIAA 32nd Aerospace Sciences Meeting, 1994. AIAA-1994-97.
25. *Integration of Multifidelity Multidisciplinary Computer Codes for Design and Analysis of Supersonic Aircraft*. **Geiselhart, K. A., Ozoroski, L. P., Fenbert, J. W., Shields, E. W., and Li, W.** Orlando : 49th AIAA Aerospace Sciences Meeting, 2011. AIAA-2011-0465.

26. **Ozoroski, L. P., Geiselhart, K. A., Padula, S. L., Li, W., Olson, E. D., et. al.** *Initial Multidisciplinary Design and Analysis Framework*. Hampton : National Aeronautics and Space Administration, 2010. NASA-TM-2010-216711.
27. *New Approaches to High Speed Civil Transport Multidisciplinary Design and Optimization*. **Zink, P.S., et al., et al.** Big Sky, MT : Aerospace Conference Proceedings, 2000 IEEE, 2000. doi: 10.1109/AERO.2000.879412.
28. *High Fidelity Multidisciplinary Optimization (HFMDO)*. **Alston, K., Doyle, S., Winter, T., Kim, H., and Ragon, S.** Orlando, FL : 48th AIAA Aerospace Sciences Meeting, 2010. AIAA-2010-1314.
29. *Multidisciplinary High-Fidelity Analysis and Optimization of Aerospace Vehicles, Part 1: Formulation*. **Walsh, J. L., et al., et al.** Reno, NV : AIAA 38th Aerospace Sciences Meeting, 2000. AIAA-2000-418.
30. *Multidisciplinary High-Fidelity Analysis and Optimization of Aerospace Vehicles, Part 2: Preliminary Results*. **Walsh, J. L., et al., et al.** Reno, NV : 38th AIAA Aerospace Sciences Meeting, 2000. AIAA-2000-419.
31. *A Mixed-Fidelity Approach for Design of Low-Boom Supersonic Aircraft*. **Li, W., Shields, E., and Geiselhart, K.** Orlando : 48th AIAA Aerospace Sciences Meeting, 2012. AIAA-2010-845.
32. *High-Fidelity Aerostructural Design Optimization of a Supersonic Business Jet*. **Martins, J. R. R. A., Alonso, J. J. and J., and Reuther J.** 3, s.l. : Journal of Aircraft, 2004, Vol. 41.
33. *Multifidelity Design Optimization of Low-Boom Supersonic Jets*. **Choi, S., et al., et al.** 1, s.l. : Journal of Aircraft, 2008, Vol. 45.

34. *Multilevel Optimization of a Supersonic Aircraft*. **Vazquez, M., Dervieux, A. and Koobus, B.** 2101-2124, s.l. : Elsevier Finite Elements in Analysis and Design, 2004, Vol. 40.
doi: 10.1016/j.finel.2004.01.010.
35. *Two-Level Multifidelity Design Optimization Studies for Supersonic Jets*. **Choi, S., Alonso, J. J. and Kroo, I. M.** 3, s.l. : Journal of Aircraft, 2009, Vol. 46.
36. *Simultaneous Airframe and Propulsion Cycle Optimization for Supersonic Aircraft Design*. **Rallabhandi, S. K. and Mavris, D. N.** 1, s.l. : Journal of Aircraft, 2008, Vol. 45.
37. **Kroo, I., et al., et al.** *Multifidelity Analysis and Optimization for Supersonic Design*.
Hampton, VA : National Aeronautics and Space Administration, 2010. NASA-CR-2010-216874.
38. **Adams, B.M., Bohnhoff, W.J., Dalbey, K.R., Eddy, J.P., Eldred, M.S., Gay, D.M., Haskell, K., Hough, P.D., and Swiler, L.P.** *DAKOTA, A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis: Version 5.0 User's Manual*. s.l. : Sandia Technical Report, December 2009.
Updated December 2010 (Version 5.1) Updated November 2011 (Version 5.2). SAND2010-2183.
39. *pyMDO: An Object-Oriented Framework for Multidisciplinary Design Optimization*.
Martins, J. R. R. A., Marriage, C., and Tedford, N. P. 4, s.l. : ACM Transactions on Mathematical Software, August 2009, Vol. 36.
40. Isight & the SIMULA Execution Engine. *Dassault Systemes*. [Online] [Cited: Aug. 16, 2012.] <http://www.3ds.com/products/simulia/portfolio/isight-simulia-execution-engine>.
41. Phx ModelCenter. *Phoenix Integration*. [Online] [Cited: Aug. 15, 2012.]
<http://www.phoenix-int.com/software/phx-modelcenter.php>.

42. *Multidisciplinary Environments: A History of Engineering Framework Development*. **Padula, S. L., and Gillian, R. E.** Portsmouth, VA : 11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, 2006. AIAA-2006-7083.
43. **Raymer, D. P.** *Aircraft Design: A Conceptual Approach*. Reston, VA : American Institute of Aeronautics and Astronautics, Inc., 1999.
44. *Universal Parametric Geometry Representation Method*. **Kulfan, B. M.** 1, s.l. : American Institute of Aeronautics and Astronautics, 2008, Journal of Aircraft, Vol. 45. 0021-8669.
45. **Farin, G.** *Curves and Surfaces for CAGD: A Practical Guide*. 5th. San Francisco : Academic Press, 2002. 1558607374.
46. **Antcliff, K.** *Vehicle Sketch Pad User Manual*. Hampton, VA : NASA Langley Research Center, 2012. Version 1.7.92.
47. *Knowledge-Based Engineering Approach to Support Aircraft Multidisciplinary Design and Optimization*. **La Rocca, G., and van Tooren, M. J. L.** 6, s.l. : American Institute of Aeronautics and Astronautics, 2009, Journal of Aircraft, Vol. 46. 0021-8669.
48. *Dynamic CAD Objects for Structural Optimization in Preliminary Aircraft Design*. **Ledermann, C., Ermanni, P., and Kelm, R.** 7, s.l. : Elsevier, 2006, Aerospace Science and Technology, Vol. 10.
49. *Parametric Geometry Model for Multidisciplinary Design Optimization of Tailless Supersonic Aircraft*. **Morris, C. C., Allison, D. L., Schetz, J. A., and Kapania, R. K.** Minneapolis, MN : AIAA Modeling and Simulation Technologies Conference, 2012. AIAA-2012-4580.
50. *Aerodynamic Shape Parameterisation and Optimization of Novel Configurations*. **Straathof, M. H., van Tooren, M. J., and Koren, B.** London : Royal Aeronautical Society, 2008.

Proceedings of the RAeS Aerodynamic Shape Parameterisation and Optimization of Novel Configurations Conference.

51. **Craidon, C. B.** *Description of a Digital Computer Program for Airplane Configuration Plots.* s.l. : NASA, 1970. NASA TM X-2074.

52. **McCullers, L. A.** *Flight Optimization System User's Guide.* s.l. : NASA Langley Research Center, Feb. 2011. Release 8.20.

53. —. *FLOPS Program Distribution Documentation - WateDoc.txt.* s.l. : NASA Langley Research Center, 2010.

54. **Richard C. Feagin, William D. Morrison Jr.** *Delta Method, An Empirical Drag Buildup Technique.* Dec. 1978. NASA CR-151971.

55. **McCullers, L. A.** Personal Email Communication to Craig Morris. 2010.

56. *The N2 Chart.* **Lano, R.** 1977, TRW Informational Report.

57. **McCullers, L. A.** Program WINGDES - Release 2001.1 User's Manual. *University of Virginia - Mechanical and Aerospace Engineering Department.* [Online] Feb. 29, 2008. [Cited: June 13, 2011.] <http://www.mae.virginia.edu/meclab/images/WdesMan.pdf>.

58. —. *AWAVE- User's Guide for the Revised Wave Drag Analysis Program.* Hampton, VA : National Aeronautics and Space Administration, 1992.

59. **Mason, W. H.** *FRICITION User's Guide.* Blacksburg, VA : Department of Aerospace and Ocean Engineering - Virginia Tech, 2006.

60. **Harry W. Carlson, Julio Chu, Lori P. Ozoroski, L. A. McCullers.** *Guide to AERO2S and WINGDES Computer Codes for Prediction and Minimization of Drag Due to Lift.* Hampton : Langley Research Center, Nov. 1997. NASA TP-3637.

61. **Jr., Roy V. Harris.** *An Analysis and Correlation of Aircraft Wave Drag*. Hampton : NASA, Mar. 1964. NASA TM X-947.
62. **McCullers, L. A.** Software for Aerodynamics and Aircraft Design (W.H. Mason, Virginia Tech). *Mason Info for AOE 4065-4066 Design (Aircraft)*. [Online] Sept. 9, 2010. [Cited: June 13, 2011.] http://www.dept.aoe.vt.edu/~mason/Mason_f/MRsoft.html#AWAVE.
63. **Mason, W. H.** http://www.dept.aoe.vt.edu/~mason/Mason_f/SD1.html. *Virginia Tech Aerodynamics and Design Software Collection*. [Online] Feb. 8, 2011. [Cited: June 13, 2011.] http://www.dept.aoe.vt.edu/~mason/Mason_f/FRICTman.pdf.
64. *Engineering Relations for Heat Transfer and Friction in High-Velocity Laminar and Turbulent Flow over Surfaces with Constant Pressure and Temperature*. **Eckert, E. R. G.** pp. 1273-1283, s.l. : Trans. ASME, 1956, Vol. 78.
65. **Schetz, Joseph A.** *Boundary Layer Analysis*. Englewood Cliffs, NJ : Prentice-Hall, Inc., 1993. 0-13-086885-X.
66. **Karl A. Geiselhart, Michael J. Caddy, Shelby J. Morris Jr.** *Computer Program for Estimating Performance of Air-Breathing Aircraft Engines*. May 1991. NASA TM-4254.
67. **Michael J. Caddy, Stanley R. Shapiro.** *NEPCOMP – The Navy Engine Performance Computer Program, Version I*. April 1975. NADC-74045-30.
68. **Fishbach, L. H., and Caddy, M. J.** *NNEP - The Navy NASA Engine Program*. Cleveland, OH : National Aeronautics and Space Administration, 1975. NASA TM X-71857.
69. *A Two-Dimensional Interpolation Function for Irregularly-Spaced Data*. **Shepard, D.** New York : Association for Computing Machinery, 1968. 23rd ACM National Conference. pp. 517-524.

70. *Algorithm 905: SHEPPACK: Modified Shepard Algorithm for Interpolation Shepard Algorithm for Interpolation*. **Thacker, W. I., Zhang, J., Watson, L. T., Birch, J. B., Iyer, M. A., and Berry, M. W.** 3, s.l. : ACM Trans. Math. Software, 2010, Vol. 37. Article 34.
71. **Thacker, W. I., Zhang, J., Watson, L. T., Birch, J. B., Iyer, M. A., and Berry, M. W.** toms. *Algorithm 905 {SHEPPACK}*. [Online] Sept. 2010. [Cited: Feb. 21, 2013.]
<http://www.netlib.org/toms/905>.
72. **Corporation, LTV Aerospace.** *Advanced Supersonic Technology Concept Study Reference Characteristics*. Hampton, VA : National Aeronautics and Space Administration, 1973. NASA-CR-132374.
73. **Clark, Bruce J.** *Computer Program To Predict Aircraft Noise Levels*. Sept. 1981. NASA TP-1913.
74. **Gillian, R.** *Aircraft Noise Prediction Program User's Manual*. Langley, VA : National Aeronautics and Space Administration, 1982. NASA-TM-84486.
75. FAR Part 36 - Noise Standards: Aircraft Type and Airworthiness Certification. *U.S. Government Printing Office*. [Online] 2011. [Cited: 10 10, 2012.]
<http://www.gpo.gov/fdsys/pkg/CFR-2011-title14-vol1/pdf/CFR-2011-title14-vol1-part36.pdf>.
76. Bird Problem (Constrained). *Phoenix Integration*. [Online] [Cited: Aug. 15, 2012.]
http://www.phoenix-int.com/software/benchmark_report/bird_constrained.php.
77. **Venkataraman, P.** *Applied Optimization with MATLAB Programming*. Hoboken, NJ : John Wiley & Sons, Inc., 2009. 9780470084885.
78. **Santner, T. J., Williams, B. J., and Notz, W. I.** *The Design and Analysis of Computer Experiments*. New York : Springer-Verlag, 2003. 0387954201.

79. **Fang, K. T., Runze, L., and Sudjianto, A.** *Design and Modeling for Computer Experiments*. Boca Raton, FL : Chapman & Hall/CRC Taylor & Francis Group, 2006. 9781584885467.
80. **Montgomery, D. C.** *Design and Analysis of Experiments*. 7th. Castleton, NY : John Wiley & Sons, Inc., 2009. p. 461. 9780470128664.
81. **Vanderplaats, G. N.** *CONMIN - A FORTRAN Program for Constrained Function Minimization: User's Manual*. Moffet Field, CA : National Aeronautics and Space Administration, 1973. See also Addendum to Technical Memorandum, 1978.. NASA-TM-X-62282.
82. *A Fast Elitist Multi-Objective Genetic Algorithm: NSGA-II.* **Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T.** 2, s.l. : IEEE Computational Intelligence Society, 2002, IEEE Transactions on Evolutionary Computation, Vol. 6, pp. 182-197. 1089-778X.
83. **Rangaiah, G. P., and Bonilla-Petriciolet, A., [ed.].** *Multi-Objective Optimization in Chemical Engineering: Developments and Applications*. West Sussex, UK : John Wiley & Sons Ltd., 2013. Section 16.5. 9781118341667.
84. *The Arrow Wing: Its Potentialities and Drawbacks With Regard to In-Flight Aerodynamic Research.* **Chacksfield, J. E.** 8, August 1997, Aircraft Engineering and Aerospace Technology, Vol. 49. 0002-2667.
85. **Rech, J., and Leyman, C. S.** *A Case Study By British Aerospatiale and British Aerospace on the Concorde*. s.l. : British Aerospace - Aerospatiale, 1980. TL 685.7 R4 1980.
86. *ACSynt - A Standards-Based System for Parametric Computer Aided Conceptual Design of Aircraft.* **Jayaram, S., Myklebust, A., and Gelhausen, P.** Irvine, CA : American Institute of Aeronautics and Astronautics, 1992. Aerospace Design Conference,. AIAA-1992-1268.

87. **Nichols, L. D., and Chamis, C. C.** *Numerical Propulsion System Simulation: An Interdisciplinary Approach*. Cleveland, OH : National Aeronautics and Space Administration, 1991. NASA-TM-105181.
88. **Plenener, R. M., and Snyder, C. A.** *The Navy/NASA Engine Program (NNEP89) – A User's Manual*. Cleveland, OH : National Aeronautics and Space Administration, 1991. NASA-TM-105186.
89. **Mason, W. H.** ACSYNT Program Overview/User Orientation. *W. H. Mason's Info. for Aircraft Design*. [Online] [Cited: Nov. 22, 2010.]
http://www.aoe.vt.edu/~mason/Mason_f/ACSYNT.pdf.
90. **Hill, P., and Peterson, C.** *Mechanics and Thermodynamics of Propulsion*. 2nd. Reading : Addison-Wesley Publishing Company, Inc., 1992.
91. **Technology, National Institute of Standards and.** NIST-JANAF Thermochemical Tables. *NIST Chemical Kinetics Database*. [Online] [Cited: Aug. 30, 2012.] <http://kinetics.nist.gov/janaf>.
92. **Seddon, J., and Goldsmith, E. L.** *Intake Aerodynamics*. 2nd. Reston, VA : American Institute of Aeronautics and Astronautics, 1999.
93. **Consortium, Numerical Propulsion System Simulation.** NPSS User Guide, Software Release: NPSS 2.3.0, Revision 2. *User Documentation*. [Online] July 7, 2010. [Cited: Aug. 29, 2012.] <http://www.wolverine-ventures.com/userdocs/UserGuide.pdf>.
94. *Preliminary Design for Embedded Engine Systems*. **Florea, R. V., Reba, R., Vanslooten. P. R., Sharma, O., Stucky, M., O'Brien, W., and Arend, D.** Orlando, FL : American Institute for Aeronautics and Astronautics, 2009. 47th AIAA Aerospace Sciences Meeting. AIAA-2009-1131.

95. *Optimization of Bleed-Flow-Control for an Aggressive Serpentine Duct*. **Florea, R., Haas, M., Hardin, L. W., Lents, C. E., and Stucky, M. B.** Reno, NV : American Institute of Aeronautics and Astronautics, 2005. 43rd AIAA Aerospace Sciences Meeting. AIAA-2005-1205.
96. *CFD Models of a Serpentine Inlet, Fan, and Nozzle*. **Chima, R. V., Arend, D. J., Castner, R. S., Slater, J. W., and Truax, P. P.** Orlando, FL : American Institute of Aeronautics and Astronautics, 2010. 48th AIAA Aerospace Sciences Meeting. AIAA-2010-0033.
97. *Reduced-Order Modeling of a High-Fidelity Propulsion System Simulation*. **Lee, K., Nam, T., Perullo, C., and Mavris, D. N.** 8, s.l. : American Institute of Aeronautics and Astronautics, Inc., Aug. 2011, AIAA Journal, Vol. 49.
98. **Taylor, J. W. R., [ed.]**. *Jane's All The World's Aircraft, 1963-1964*. New York : The McGraw-Hill Book Company, Inc., 1963. pp. 482-483.
99. *Towards Flying Qualities Constraints in the Multidisciplinary Design Optimization of a Supersonic Tailless Aircraft*. **Morris, C. C., Sultan, C., Allison, D. L., Schetz, J. A., and Kapania, R. K.** Indianapolis, IN : American Institute of Aeronautics and Astronautics, 2012. 14th AIAA/ISSM Multidisciplinary Analysis and Optimization Conference. AIAA-2012-5517.
100. *Effects of Inlet Flow Distortion on the Performance of Aircraft Gas Turbines*. **Kurzke, J.** July, s.l. : ASME, 2008, J. of Eng. for Gas Turbines and Power, Vol. 130.
101. *Boundary-Layer-Ingesting Inlet Flow Control*. **Owens, L. R., Allan, B. G., and Gorton, S. A.** 4, s.l. : American Institute of Aeronautics and Astronautics, July-August 2008, Journal of Aircraft, Vol. 45.
102. Zonair Users Manual. *Zona Technology Inc.* [Online] [Cited: Jan. 17, 2013.] http://www.zonatech.com/Documentation/ZONAIR_USERS_MANUAL_4.4.pdf.

103. **Haney, M. A.** *Topology Optimization of Engine Exhaust-Washed Structures*. Department of Mechanical and Materials Engineering. Dayton, OH : Wright State University, 2006. Dissertation.
104. *Design of Engine Exhaust-Washed Structures for an Efficient Supersonic Air Vehicle MDO Application*. **Deaton, J. D., and Grandhi, R. V.** Indianapolis, IN : American Institute for Aeronautics and Astronautics, 2012. 14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference. AIAA-2012-5493.
105. *Thermal-Structural Design and Optimization of Engine Exhaust-Washed Structures*. **Deaton, J. D., and Grandhi, R. V.** Denver, CO : American Institute of Aeronautics and Astronautics, 2011. 52nd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference. AIAA-2011-1903.
106. **Jenn, D. C.** Matlab Software. *NPS Electrical & Computer Engineering*. [Online] [Cited: 10 8, 2012.] <http://faculty.nps.edu/jenn>.
107. **Chatzigeorgiadis, F.** *Development of Code for a Physical Optics RADAR Cross Section Prediction and Analysis Application*. Monterey, CA : Naval Postgraduate School, 2004. Masters' Thesis.
108. *Overview of Low Observable Technology and Its Effects on Combat Aircraft Survivability*. **Paterson, J. 2, s.l.** : American Institute of Aeronautics and Astronautics, 1999, Journal of Aircraft, Vol. 36.
109. **Jenn, D. C.** *Radar and Laser Cross Section Engineering*. Reston : American Institute of Aeronautics and Astronautics, 2005. 1563477025.

110. Appendix D: What Every Computer Scientist Should Know About Floating-Point Arithmetic. *Numerical Computation Guide*. [Online] Sun Microsystems, Inc. [Cited: 10 8, 2012.] http://docs.oracle.com/cd/E19957-01/806-3568/ncg_goldberg.html#674.
111. **Boyd, J. W., Migotsky, E., and Wetzel, B. E.** *A Study of Conical Camber for Triangular and Sweptback Wings*. Moffett Field, CA : National Advisory Committee for Aeronautics, 1955. NACA-RM-A55G19.
112. **Phelps, E. R.** *Pressure Distributions at Mach Numbers of 1.6 and 1.9 of a Conically Cambered Wing of Triangular Plan Form With and Without Pylon-Mounted Engine Nacelles*. Moffett Field, CA : National Advisory Committee for Aeronautics, 1956. NACA-RM-A56B03.
113. **Paulson, J. W.** *Low-Speed Measurements of Oscillatory Lateral Stability Derivatives of a Model of a 60 deg. Delta-Wing Bomber*. Langley Field, VA : National Aeronautics and Space Administration, 1959. NASA-TM-X-13.
114. *A Multidisciplinary Design Optimization Framework for Design Studies of an Efficient Supersonic Air Vehicle*. **Allison, D. L., Morris, C. C., Deaton, J. D., Schetz, J. A., Kapania, R. K., et. al.** Indianapolis, IN : American Institute of Aeronautics and Astronautics, 2012. 14th AIAA/ISSM Multidisciplinary Analysis and Optimization Conference. AIAA-2012-5492.
115. SAE International Standards. *Gas Turbine Jet Exhaust Noise Prediction*. [Online] SAE, 7 2012. [Cited: 10 9, 2012.] <http://standards.sae.org/arp876e/>.
116. **Zorumski, W. E.** *Aircraft Noise Prediction Program Theoretical Manual*. Hampton, VA : National Aeronautics and Space Administration, 1982. Revised Nov. 1993. NASA-TM-83199 Parts 1 and 2.
117. **Brewer, D. G.** *Hydrogen Aircraft Technology*. Boca Raton, FL : CRC Press Inc., 1991. p. 269. 0849358388.

118. *STOVL Aircraft For Shipboard Operation*. **Yackle, A. R.** Hilton Head, SC : American Institute of Aeronautics and Astronautics, 1992. p. 4. AIAA-1992-4215.
119. What will replace the F-15E Strike Eagle? *Flightglobal/Blogs*. [Online] December 17, 2012. [Cited: April 3, 2013.] <http://www.flightglobal.com/blogs/the-dewline/2012/12/what-will-replace-the-f-15e-st.html>.
120. **Strang, G.** *Calculus*. Wellesley, MA : Wellesley-Cambridge Press, 1991. 0961408820.
121. **Geiselhart, K. A., Caddy, M. J., and Morris, S. J.** *Computer Program for Estimating Performance of Air-Breathing Aircraft Engines*. Hampton, VA : National Aeronautics and Space Administration, 1991. NASA-TM-4254.
122. **Caddy, M. J., and Shapiro, S. R.** *NEPCOMP - The Navy Engine Performance Computer Program, Version I*. Warminster, PA : Naval Air Development Center, 1975. NADC-74045-30.
123. **Geiselhart, K. A.** *FLOPS Program Distribution Documentation - Engine Cycle Analysis Program*. Hampton, VA : NASA Langley Research Center, 2009.
124. **Morris, S. J., Nelms, W. P., and Bailey, R. O.** *A Simplified Analysis of Propulsion Installation Losses for Computerized Aircraft Design*. Moffet Field, CA : National Aeronautics and Space Administration, 1976. NASA-TM-X-73136.
125. *High-Speed Civil Transport Study Using ACSYNT*. **Malone, B.** Monterey, CA : American Institute of Aeronautics and Astronautics, 1993. Aircraft Design, Systems and Operations Meeting. AIAA-1993-4006.
126. *A Method for the Identification and Assessment of Critical Technologies Needed for an Economically Viable HSCT*. **Mavris, D. N., Bandte, O., and Brewer, J. T.** Los Angeles, CA : American Institute of Aeronautics and Astronautics, 1995. Aircraft Engineering, Technology, and Operations Congress. AIAA-1995-3887.

127. **Gordon, S.** *The Navy/NASA Engine Program (NNEP89) – Interfacing the Program for the Calculation of Complex Chemical Equilibrium Compositions.* Cleveland, OH : National Aeronautics and Space Administration, 1991. NASA-CR-187208.
128. *A Multidisciplinary Design Optimization Approach To Sizing Stopped Rotor Configurations Utilizing Reaction Drive and Circulation Control.* **Mavris, D. N., Tai, J., and Schrage, D. P.** Panama City Beach, FL : American Institute of Aeronautics and Astronautics, 1994. 5th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization. AIAA-1994-4296.
129. **Squire, D.** Afterbody Drag Prediction for Conceptual Aircraft Design. *Masters Thesis.* Blacksburg, VA : Virginia Polytechnic Institute and State University, 1992.
130. **Ball, W. H., and Atkins, R. A.** *Rapid Evaluation of Propulsion System Effects Vol. II – PIPSI Users Manual.* Air Force Flight Dynamics Laboratory and Air Force Wright Aeronautical Laboratories, Air Force Systems Command. Dayton, OH : Wright-Patterson Air Force Base, 1978. AFFDL-TR-78-91, Volume II.
131. **Kurzke, J.** *GasTurb 12 Users Manual: Design and Off-Design Performance of Gas Turbines.* Germany : Joachim Kurzke, 2012.
132. *Numerical Propulsion System Simulation.* **Claus, R. W., Evans, A. L., Lylte, J. K., and Nichols, L. D.** 4, Great Britain : Pergamon Press, 1991, Computing Systems in Eng., Vol. 2. 0956-0521/91.
133. *An Object Oriented Extensible Architecture for Affordable Aerospace Propulsion Systems.* **Follen, G. J.** Paris, France : RTO-MP-089, 2002. RTO A VT Symposium on Reduction of Military Vehicle Acquisition Time and Cost through Advanced Modelling and Virtual Simulation.

134. **Parker, K. I., Felder, J. L., Lavelle, T. M., Withrow, C. A., et. al.** *Integrated Control Modeling for Propulsion Systems using NPSS*. Cleveland, OH : National Aeronautics and Space Administration, 2004. NASA-TM-2004-212945.
135. *Flow Simulation of Supersonic Inlet with Bypass Annular Duct*. **Kim, H. J., Kumano, T., Liou, M. S., Povinelli, L. A., and Corners, T. R.** 1, s.l. : American Institute of Aeronautics and Astronautics, 2011, J. of Propulsion and Power, Vol. 27.
136. *Performance, Usage, and Turbofan Transient Simulation Comparisons Between Three Commercial Simulation Tools*. **Gomes, K. J., Masiulaniec, K. C., and Afjeh, A. A.** 2, s.l. : American Institute of Aeronautics and Astronautics, 2009, Journal of Aircraft, Vol. 46.
137. **Onat, E., and Klees, G. W.** *A Method to Estimate Weight and Dimensions of Large and Small Gas Turbine Engines*. Cleveland, OH : National Aeronautics and Space Administration, 1979. NASA CR-159481.
138. **Tong, M. T., and Naylor, B. A.** *An Object-Oriented Computer Code for Aircraft Engine Weight Estimation*. Cleveland, OH : National Aeronautics and Space Administration, 2009. NASA TM-2009-215656.
139. *A Computer Code for Gas Turbine Engine Weight and Disk Life Estimation*. **Tong, M. T., Halliwell, I., and Ghosn, L. J.** 2, s.l. : Journal of Engineering for Gas Turbines and Power, 2004, Vol. 126.

A. Appendix Optimization Background

1.A Basic Optimization Concepts

Optimization is a subject studied in a first calculus course. The introductory problem usually consists of taking the derivative of a single-variable function like $f(x) = x^2 - 2x + 3$, for instance, and then looking for the point where its slope is zero [120]. This point is called a stationary point and the test is a necessary condition for an extremum (maximum or minimum). However, since this is just a necessary condition, a stationary point can be either an extremum or not an extremum (e.g. saddle point). For this example, the stationary point can be simply found by solving for the derivative with respect to x being equal to zero as,

$$\frac{df}{dx} = 0 = 2x - 2 \quad (A1)$$

$$x = 1. \quad (A2)$$

The function value at the stationary point is $f(1) = 2$. Two intuitive methods can be employed to determine if this stationary point is a maximum or a minimum. The first is a graphical method. By plotting the function, the point at $x = 1$ is clearly shown to be a minimum with a value of 2 as seen in Fig. A-1.

Besides graphing this smooth function, a basic test to find out if the stationary point at $x = 1$ is a maximum or a minimum, one can evaluate the function at a value slightly larger than $x = 1$ and slightly smaller. If these evaluations both result in larger function values, then $x = 1$ can be declared a local minimum and if both are smaller, then $x = 1$ can be declared a local maximum. To continue this example we can solve for two adjacent points to the stationary point, $f(0) = 3$, $f(1) = 2$, and $f(2) = 3$, making $x = 1$ a minimum (both adjacent points are larger

than the extremum point). For this elementary, unconstrained, one-dimensional, convex case this extremum is both a local and a global minimum.

The usual method in calculus to determine if a stationary point is a maximum or a minimum is by determining the sign of the second derivative. It is a maximum if $f''(x) < 0$ and a minimum if $f''(x) > 0$. When $f''(x) = 0$, it is an inflection point, but if it is also a stationary point then this is a saddle point. For this example, the second derivative is, $f''(x) = 2 > 0$ so this is shown to be a minimum mathematically.

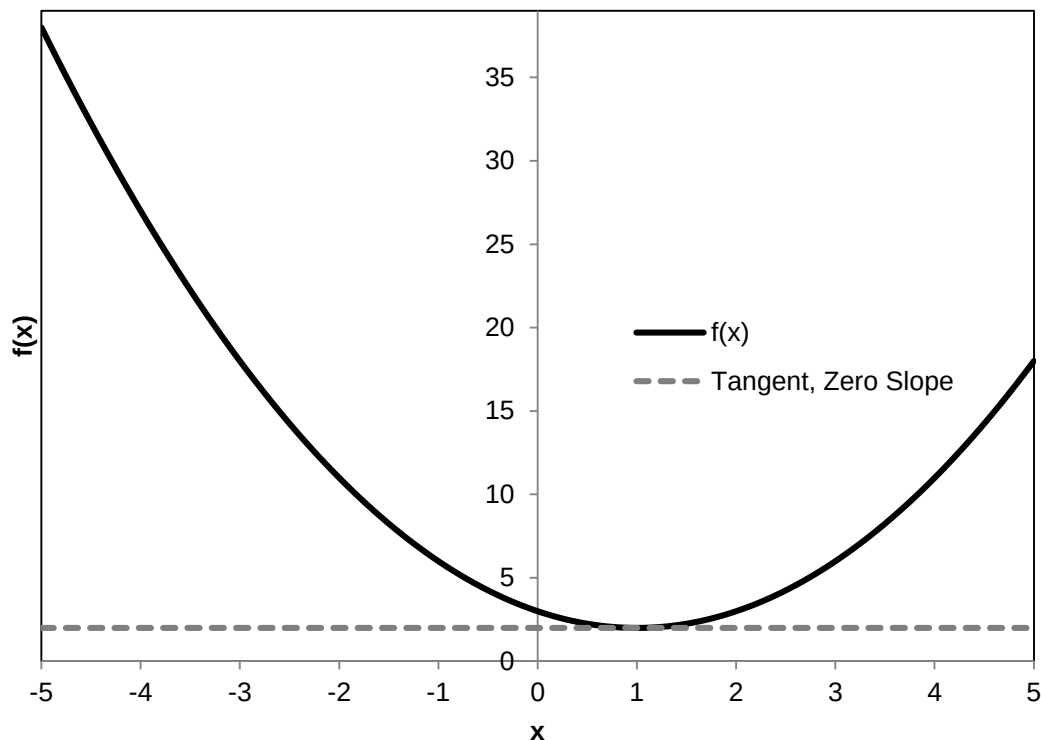


Figure A-1. Plot of $f(x) = x^2 - 2x + 3$ to graphically determine the extremum point, its function value, and whether it is a maximum or minimum.

2.A Multi-Variable, Nonlinear, and Constrained Optimization

A later course in calculus introduces tools for handling more realistic problems that include multi-variables and constraints. These optimization tools are the gradient and Lagrange multipliers respectively. The gradient is a multi-variable extension of the first derivative while the Lagrange multiplier is a new concept which handles constraints.

In order to find the extremum point(s), the gradient of a function, $f(x, y)$, is solved after being set equal to a vector of zeros (if a solution exists). For the two variable Cartesian case, the gradient being set to the homogeneous solution is expressed as,

$$\nabla f = \frac{\partial f}{\partial x} \hat{x} + \frac{\partial f}{\partial y} \hat{y} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \quad (\text{A3})$$

The solution to Eq. A3, $(\tilde{x}, \tilde{y})$, determines the stationary point(s). In order to determine whether the extremum point(s) is a maximum, minimum, or saddle point, the second derivative is again employed. For the two-dimensional case, the tangent line of slope zero is now a tangent plane and the three second derivatives, f_{xx} , f_{xy} , and f_{yy} are required.

A somewhat involved test case that highlights the difficulties that often arise during optimization is the following problem given in Eq. A4.

$$\min_{x \in E^2} f(\vec{x}) = 10\cos(x_1)\sin(x_1) + \frac{3}{2}x_1^2 + \frac{3}{2}x_2^2 \quad (\text{A4a})$$

$$\text{subject to: } -10 \leq x_1 \leq 10, \quad (\text{A4b})$$

$$-10 \leq x_2 \leq 10, \quad (\text{A4c})$$

$$3x_1 + 10x_2 = 5. \quad (\text{A4d})$$

This optimization problem can be characterized as being multi-dimensional with inequality constraints and one equality constraint. The vector of variables is defined in the Euclidean space of dimension two. There are four inequality constraints that essentially define the bounds of a

square box that the solution must reside in. The equality constraint defines a line, and specifies that the function extremum must be on this line. A plot of this problem showing the constraints and function is given in Fig. A-2.

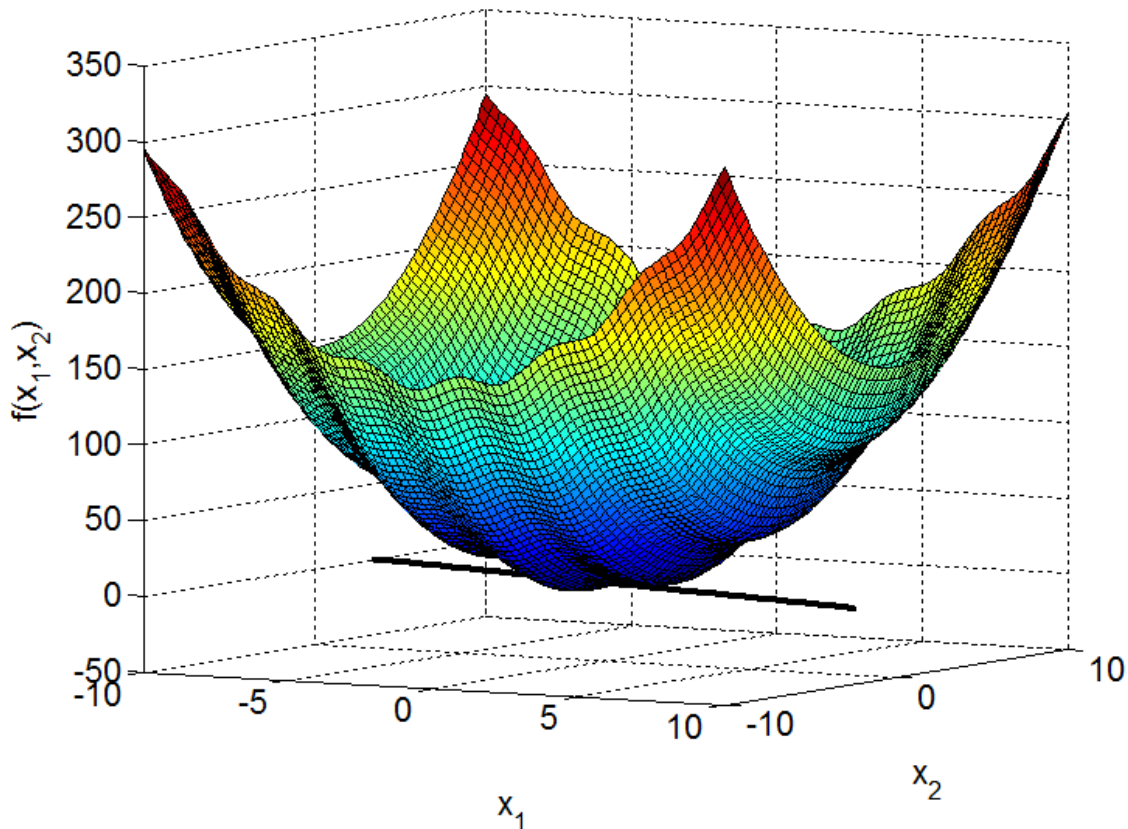


Figure A-2. Plot of Eq. A4 that shows the function to be optimized, the problem's inequality constraints (x_1 and x_2 axes limits), and the equality (intersecting line) constraint.

3.A Unconstrained Optimization

First, it is instructive to solve the unconstrained problem. Setting the gradient to the null vector and solving yields the stationary points. This is the necessary condition. The sufficient condition is that the Hessian matrix, $\mathbf{H}$, be positive definite for all possible values of the design variables. In other words, this means all eigenvalues are positive, or, for any arbitrary, real-valued, vector

$\vec{x}$, of the correct dimension, the expression $\vec{x}^T \mathbf{H} \vec{x} > 0$ is true. It should be noted that the sufficient condition is not usually practical in real-world design applications. The gradient and Hessian for this two-dimensional problem can be written down as,

$$\nabla f(x_1, x_2) = \begin{bmatrix} 10\cos^2(x_1) - 10\sin^2(x_1) + 3x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \quad (\text{A5})$$

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 f(x_1, x_2)}{\partial x_1^2} & \frac{\partial^2 f(x_1, x_2)}{\partial x_1 \partial x_2} \\ \frac{\partial^2 f(x_1, x_2)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x_1, x_2)}{\partial x_2^2} \end{bmatrix} = \begin{bmatrix} -40\sin(x_1)\cos(x_1) + 3 & 0 \\ 0 & 3 \end{bmatrix} \quad (\text{A6})$$

There are a finite number of solutions, and these are solved for using numerical techniques and are shown in Table A-1. The fact that this problem needs to be solved numerically, although the function and derivatives exist analytically, highlights a general result for multidisciplinary design optimization of aircraft or other real systems; that is, there are almost no analytically-determined optimal solutions to these systems. The two questions that now need to be answered are:

1. Which of these stationary points are minimums?
2. Which of the minimums is the solution to the problem?

Table A-1. Rounded Numerical Stationary Point Solutions to Eq. A5

x_1	x_2
-3.25216	0
-2.87683	0
-0.68232	0
0.92618	0
2.02899	0

The second derivative and function evaluations are used to answer these questions. The second derivatives pulled from the Hessian are,

$$f_{x_1x_1} = -40\sin(x_1)\cos(x_1) + 3, \quad (\text{A7})$$

$$f_{x_2x_2} = 3, \quad (\text{A8})$$

$$f_{x_1x_2} = 0. \quad (\text{A9})$$

The second derivative with respect to x_2 is always positive and the mixed derivative is always zero. This just leaves the second derivative with respect to x_1 which can be solved for at each stationary point listed in Table A-1. The results of this exercise are in Table A-2 and show that there are three minimums and two maximums. This can also be graphically verified in Fig. A-3. Further, based on the function values, it is easily determined that the minimum value of -4.19578 occurs at $\hat{x}_1 = -0.68232$ and $\hat{x}_2 = 0$ and, in this case, is unique.

Table A-2. Rounded Numerical Second Derivatives of Eq. A7 and Function Values of Eq. A4a using Stationary Point Inputs

x_1	x_2	$f_{x_1x_1}$	Max./Min.	$f(x_1, x_2)$
-3.25216	0	7.38680	Min.	14.76813
-2.87683	0	-7.10237	Max.	14.93984
-0.68232	0	22.57651	Min.	-4.19578
0.92618	0	-16.21247	Max.	6.08982
2.02899	0	18.86807	Min.	2.20816

In proceeding to solve the full constrained problem of Eq. A4, Lagrange multipliers will be employed since the first derivatives of the function lose their meaning. This is because the stationary points are not guaranteed to be extremum points for the constrained system. Therefore the problem changes with constraints so as one is to find the “best possible” [120] solution rather than the global unconstrained optimum of the function only.

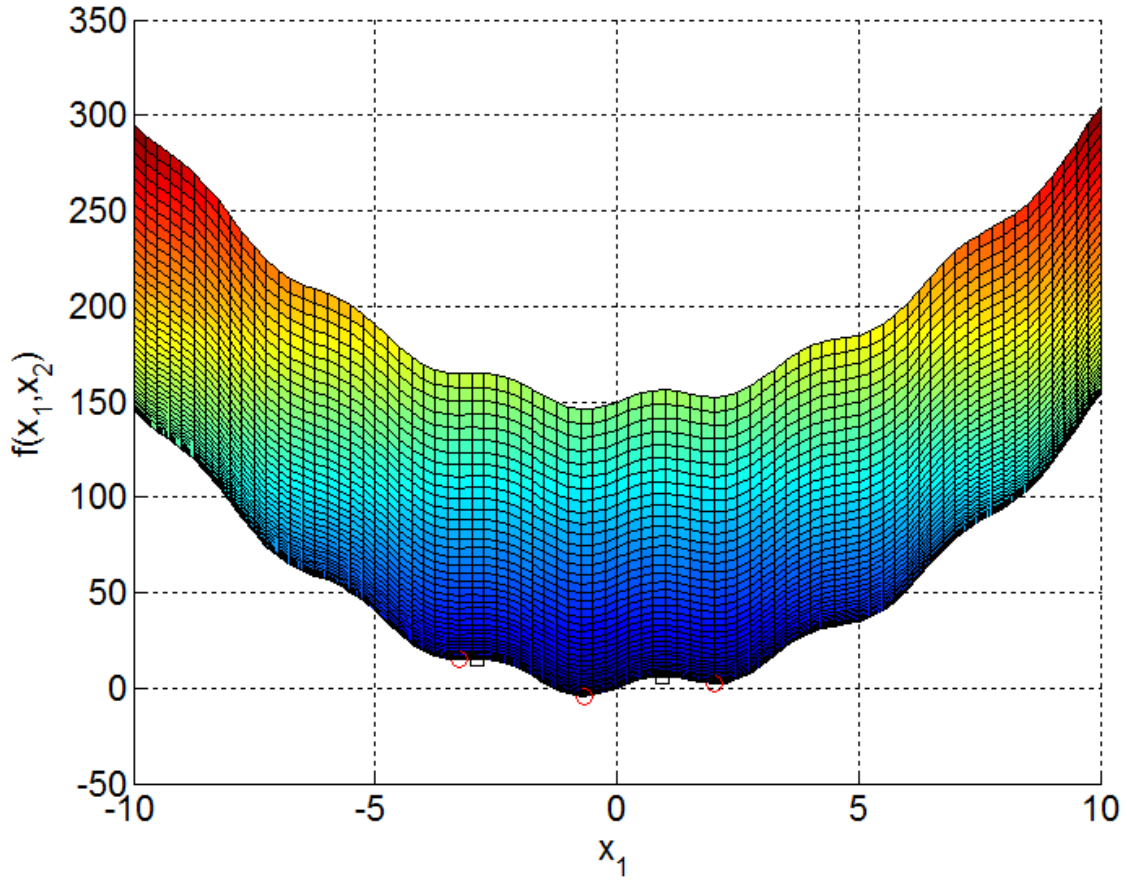


Figure A-3. Unconstrained function optimization plot showing the three local minima points (red circles) and two local maxima points (black squares). The unique global unconstrained minimum point is the center circle.

4.A Constrained Optimization

Applying the KKT conditions to the example problem (Eq. A4), the seven resulting partial derivatives are,

$$L_{x_1} = 10\cos^2(x_1) - 10\sin^2(x_1) + 3x_1 - \mu_1 + \mu_2 - 3\lambda = 0, \quad (\text{A10a})$$

$$L_{x_2} = 3x_2 - \mu_3 + \mu_4 - 10\lambda = 0, \quad (\text{A10b})$$

$$L_{\mu_1} = -x_1 - 10 \leq 0, \quad (\text{A10c})$$

$$L_{\mu_2} = x_1 - 10 \leq 0, \quad (\text{A10d})$$

$$L_{\mu_3} = -x_2 - 10 \leq 0, \quad (\text{A10e})$$

$$L_{\mu_4} = x_2 - 10 \leq 0, \quad (\text{A10f})$$

$$L_\lambda = -3x_1 - 10x_2 + 5 = 0. \quad (\text{A10g})$$

Figure A-2 clearly showed that all of the inequality constraints are not active at the minimum and that the minimum is therefore an interior point in the bounded region. Therefore, by the complementarity condition, all those Lagrange multipliers are zero ($\hat{\mu}_i = 0$). The problem is therefore reduced to three equations and three unknowns,

$$10\cos^2(x_1) - 10\sin^2(x_1) + 3x_1 - 3\lambda = 0, \quad (\text{A11a})$$

$$3x_2 - 10\lambda = 0, \quad (\text{A11b})$$

$$-3x_1 - 10x_2 + 5 = 0. \quad (\text{A11c})$$

Substitution yields one equation of one unknown,

$$10\cos^2(x_1) - 10\sin^2(x_1) + 3x_1 + \frac{27}{100}x_1 - \frac{9}{20} = 0. \quad (\text{A12})$$

All stationary points of the constrained problem can now be found by solving Eq. A12 for all roots and then substituting back in to Eq. A11 to find the corresponding values of x_2 and λ . The results of this exercise are given in Table A-3.

Table A-3. Rounded Numerical Stationary Point Solutions to Eq. A11 and Function Values at These Points

x_1	x_2	λ	$f(x_1, x_2)$
-0.65441	0.69632	0.20890	-3.45972
0.91369	0.22589	0.06777	6.16510
2.02386	-0.10716	-0.03215	2.22563

The equality constraint is shown cut across the function in Fig. A-4 along with the three stationary points listed in Table A-3 (magenta star marker is the global optimum for the constrained problem). It is noticed that the minimum value is not as small as the minimum value in the unconstrained case (see Table A-2). However, it is close to the unconstrained value for this case because the equality constraint curve (black line in Fig. A-4) cuts close to the unconstrained minimum. The Lagrange multipliers can be thought of as sensitivities of the objective function to the constraints [17]. For instance, the inequality constraints' Lagrange multipliers are all zero which implies that these constraints do not affect the problem, which was shown to be true.

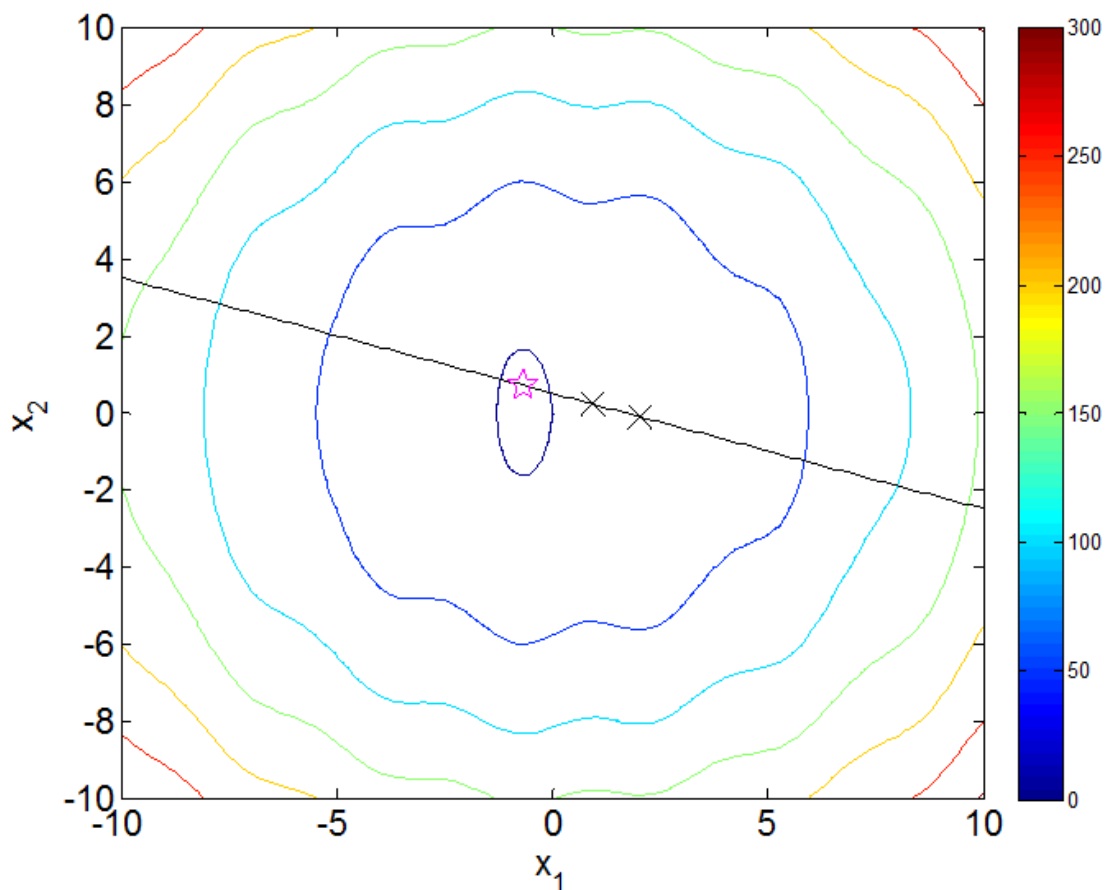


Figure A-4. Contour plot of the example problem showing stationary points along the inequality constraint and the constrained minimum solution (magenta star)

B. Appendix Background Information on Four Propulsion Model

Candidates

1.B FLOPS Propulsion Module

The FLOPS propulsion model is based on QNEP [121], which is a simplified version of the Navy Engine Performance Computer Program (NEPCOMP) [122]. Its propulsion analysis begins with the generation of an input deck in an ASCII text file containing data in a fixed-width table format with the columns [52]

- Mach number,
- altitude,
- power code or engine power setting,
- gross thrust,
- ram drag,
- fuel flow,
- NO_x emissions index or grams of NO_x per kilogram of fuel (not required), and
- nozzle exit area (required only if nozzle installation drag is to be scaled).

The user has the option of generating an input deck, or using FLOPS to generate an input deck. For the internally-generated engine deck, multiplying factors can be applied to change engine parameters, if desired. The fuel flow is calculated by FLOPS at idle, subsonic, and supersonic points. The fuel flow is scaled either linearly or nonlinearly with engine-rated thrust depending on the user's choice. The nonlinear scaling equation is included in the user's manual [52] if the user chooses this option. A boost engine may also be added for take-off and climb. For an

externally-created engine deck, the nozzle drag must be accounted for and input. The afterbody drag could also be added. A maximum nozzle area and length as well as a reference nozzle area and length can also be specified.

The engine deck has one entry per every Mach number/altitude/thrust combination. There can be anywhere from two to twenty Mach numbers, each with two to twenty altitudes. Up to sixteen thrust levels can be input for each Mach number/altitude point. However, there is a limit of 3000 entries allowed in an input deck (not the expected $20 \times 20 \times 16 = 6400$). If there are duplicate entries in the engine deck, these are ignored. A Mach number of 5 is the highest possible for the input deck, and at least one part-power cruise must be input for a Mach number/altitude point.

In order to have FLOPS generate an engine input deck, the engine cycle desired must be defined. There are several engine cycles available in FLOPS:

- turbojet,
- separate flow turbofan (2 compressor components),
- mixed flow turbofan (2 compressor components),
- turbine bypass,
- separate flow turbofan (3 compressor components),
- mixed flow turbofan (3 compressor components),
- 3 spool separate flow turbofan (3 compressor components),
- 2 spool turbojet, or
- a user-defined engine cycle.

With all of these different cycle choices, it is feasible that the ESAV MDO framework could include the propulsion cycle as a design variable choice using the FLOPS propulsion module.

These various cycles require component maps. FLOPS comes with a default component data map called ENGTAB. There are many options available including afterburner throttle settings (the nozzle area is variable with an afterburner), jet noise calculation, specified bleed flow and power extraction by the user, fuel choice through specifying the fuel heating value, and the technology year which changes the compressor's efficiency (polytropic).

The installation drag can be included. These drag components are nozzle boattail drag calculated using a simple empirical estimate based on the nozzle area ratio (assuming the engine is in a traditional nacelle, which does not apply to embedded engines), and spillage drag including bleed flow drag. The user must be careful in the model input so as to maintain feasibility, since there are no internal limits placed on discharge temperature and pressure, the jet velocity, and the bypass ratio.

The engine cycle can be either one of these built-in cycles or it can be a user-specified cycle. A user-defined cycle must include the components of the engine, the characteristics of the components (and thermodynamic station numbers corresponding to the components – see Fig. B-1 for a typical example), and flow path choices.

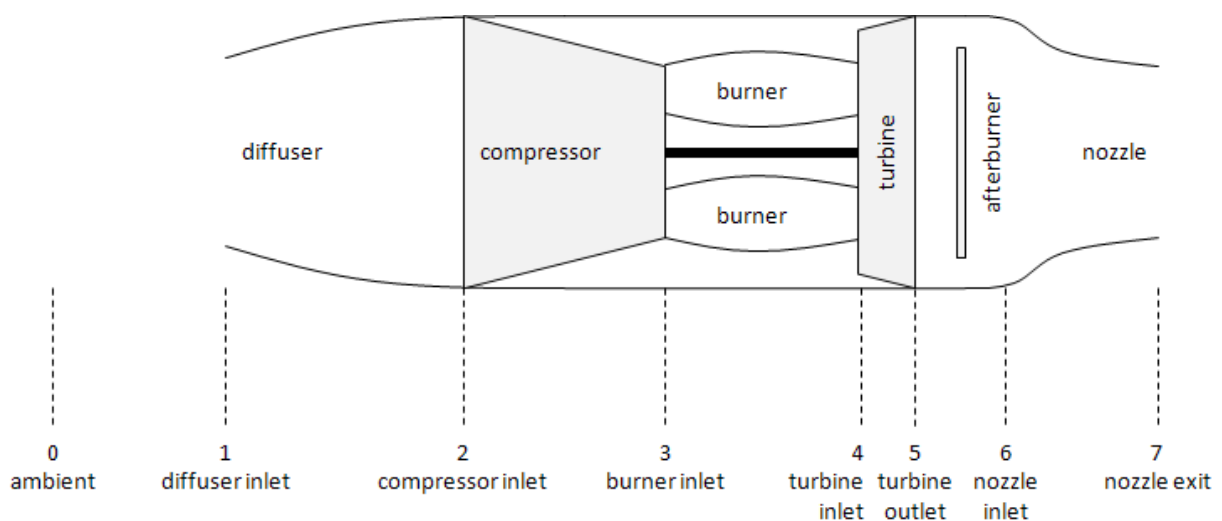


Figure B-1. Thermodynamic station numbers of a typical turbojet engine.

The component characteristics are not trivial or easy to obtain. Thus, choosing to create a user-defined cycle is beyond the scope of a conceptual design or maybe even preliminary design. To properly characterize the inlet, the mass flow, temperature, Mach number, static pressure and drag must be specified. Ducts and burners are very similar, except a duct merely passes the fluid flow through it while a burner encloses a combustion flow. Ducts or burners need the pressure drop, efficiency, NO_x levels produced (not needed for duct), and the fuel heating value (not needed for duct). Compressor data requirements are the compressor inlet flow conditions, the adiabatic efficiency, the pressure ratio, stator angles (variable geometry can be included with this option), the bleed power loss, and the polytropic efficiency (changing ratio of specific heats). A turbine needs the inlet pressure ratio, the bleed flow entering the turbine, the corrected flow properties at the turbine inlet, the adiabatic efficiency, the stator angles (variable geometry can be included with this option), and the power split factor if more than one turbine is driving the shaft. For the heat exchanger the user must choose the pressure drop, a guess for the temperature rise ratio, and a heat exchanger effectiveness value (ratio of actual heat transferred to the maximum possible amount of heat transfer given an infinite area). A splitter component requires the bypass ratio and the pressure drop. A mixer needs the stream flows entrance area, the pressure ratio or Mach number, and the ratio of actual to ideal velocity of mixed flow at the exit. Finally, a nozzle needs the throat area, the flow coefficient ($\dot{m}_{actual}/\dot{m}_{ideal}$, where $\dot{m}$ is the maximum mass flow rate), the boattail drag, the exit static temperature and pressure, the exit velocity, and the type of nozzle (converging/diverging or convergent). For an afterburner, the nozzle area is computed otherwise it is input.

A table of loads demanded by aircraft operations may also be input for the user-defined cycle. For off-design calculations, controls/limits are generally required on the component. These

controls can be imposed on several properties at each thermodynamic station. The performance properties can then be obtained from the off-design points. Both the controls and the performance properties are shown in Table B-1 [123].

Table B-1. FLOPS Off-Design Calculation Controls and Performance Properties

Controls/Limits	Performance Properties
Mass flow	Net thrust
Total pressure and temperature	Air flow
Fuel-to-air ratio	Fuel flow
Corrected flow*	Thrust Specific Fuel Consumption (TSFC)
Mach number	Thrust to airflow ratio, $T/\dot{m}_a$
Static Pressure	Inlet drag
Interface relative flow error	

* Mass flow per unit area based on standard conditions which is a function of Mach number only

The built-in FLOPS cycle analysis uses component maps to create the engine input deck. As mentioned above, the default FLOPS map is a file called ENGTAB. However, there are still some user-defined inputs available with this option. For instance, an afterburner can be included. Mission limits can be defined including the maximum Mach number and dynamic pressure. Engine noise can be calculated or ignored. Most importantly, the design point conditions are input. These are the net thrust, Mach number, altitude, overall pressure ratio (also fan pressure ratio for turbofans), bypass ratio if applicable, turbine entry temperature, and the throttle ratio (maximum allowed turbine inlet temperature to design point temperature). Depending on the built-in engine cycle chosen, extra parameters must also be input (e.g. duct burner efficiency, bleed fractions, turbine cooling flow, include spillage drag, etc.). If desired, the engine weight can be calculated using the cycle analysis. This weight can be specified to be the engine by itself,

or some combination of engine, inlet, nacelle, and nozzle weights. In order to calculate the weight using this method, however, engine geometry and engine material detail must be known *a priori* and cannot be designed.

Once the engine input deck has been created, FLOPS runs the propulsion analysis. This analysis includes checking for obvious errors in the input data. Extrapolation or interpolation is employed where necessary when part-power data is missing. Most interpolation is linear or bilinear between table data. The propulsion installation drag is calculated, but this is under the assumption of a traditional engine-nacelle configuration. The limiting flight altitude for the propulsion module is 291,152 ft. because FLOPS uses the U.S. Standard Atmosphere model.

2.B ACSYNT Propulsion Module

ACSYNT is similar to FLOPS in that it provides all of the usual aircraft conceptual design tools in one code, including propulsion. Originally, the propulsion model was written by Morris et. al. [124]. This code included the calculation of the thrust, the specific fuel consumption, the inlet size, the pressure recovery of the engine, and the installation drag. The drag included the additive, subsonic spillage, bleed, bypass, auxiliary air systems, boundary-layer diverter, nozzle boattail, and interference drag on the region adjacent to multiple nozzle installations. It was found that errors for net thrust and SFC when compared to another model based on F-5A test data were generally in the 10% range, but errors did reach into the high teens for some cases [124].

Eventually, the ACSYNT propulsion model was replaced by the Navy/NASA Engine Program (NNEP) [86], and then still later was improved to the NNEP89 version [125]. This program performed the one-dimensional cycle analysis in steady state, or a table look-up of this data could also be used. The form of the look-up tables [126] is a function of the expected

variables such as altitude, Mach number, power codes, etc. Both the design point and off-design points could be calculated for turbojets and turbofans.

NASA Glenn issued a technical memorandum [68] about the NNEP software, and the following observations were made. First, the motivation for the creation of the code was from a need to be able to calculate performance of new engine concepts throughout the flight envelope. Therefore, the code is capable of an arbitrary engine configuration made up of selected component combinations. Another feature was that the engine configuration could be changed during code runs through an optimizer trying to minimize SFC for a given thrust. Therefore, the code capabilities are to

- have the ability to perform an optimization,
- process stacked component maps for various cycle engine operation,
- handle multiple configurations in order to simulate flowpath switching, and
- have power code dependent inlet and boattail drag calculations.

Two goals included in NNEP were to

1. include the ability to create a computer-generated engine schematic and to
2. simplify the format of the input data.

The stacked maps are used to handle components with various geometries. Therefore, the maps must represent the component performance as a function of variable parameters. Hence, a stacked map is a regular two-dimensional map extended to three-dimensions by the addition of more two-dimensional maps, differing in the third variable. The possible components that can be included (to a maximum of sixty) are similar to those defined in the FLOPS user-defined cycle option:

- Inlets

- Ducts/burners
- Compressors
- Turbines
- Mixers
- Heat exchangers
- Splitters
- Nozzles
- Loads – or the power requirements demanded from engine by aircraft

No more than 24 of each different component can be included (e.g. could not have 25 compressor stages).

Installation drag, like FLOPS, includes boattail and inlet drag. However, the program assumes an isolated nacelle engine installation and has no option for an embedded engine. The inlet drag is calculated by combining both empirical and theoretical relations. The boattail drag uses an empirical drag coefficient based on the boattail slope (angle between the converging nozzle and the horizontal). The program assures flow continuity between components and that an energy balance is achieved between the various components.

As mentioned previously, NNEP was later extended into the NNEP89 version in ACSYNT. One significant contribution was to include the general equilibrium method [127] so that any engine propellant could be included and dissociation effects would not have to be excluded. In addition to this feature, other desirable enhancements were also made [88]. Like its predecessor, NNEP89 performs one-dimensional steady state analysis of turbine engine cycles, but NNEP89 also performs a thermodynamic analysis. The ability to simulate almost any engine configuration with the code also remained. This, of course, includes the ability to simulate an

afterburner. Linear extrapolation is employed if the data requested by the program is outside of a component map range. The off-design performance is calculated by use of the component performance maps with scaling. Therefore, it is clear that component map data input is critical to the accuracy of the code. NNEP89 output includes the engine's altitude, Mach number, inlet recovery, total airflow, gross and net thrust, fuel flow, and the TSFC.

NNEP89 includes nozzle performance modeling. Turbine blade cooling is also included through the specification of the blade cooling bleed flow. The engine weight and size (dimensions) are calculated by NNEP89 by the specifications of the individual components included in the cycle definition and some other user input. The user can choose the engine design variables by specifying which are free and which are not. Free variables are changed by the program until the correct physical flow is obtained. Some of these possible design variables are the

- inlet airflow,
- “R” value of compressor or fan,
- third dimension value on stacked maps for compressor and turbine,
- pressure ratio,
- burner outlet temperature,
- primary or secondary flow area,
- throat area,
- shaft rotational speed,
- primary flow temperature change,
- bypass ratio, and
- horsepower extracted as an aircraft load.

As explained in References [88] and [128], “R” values correspond to arbitrary lines that are drawn by the user onto the map in the general shape of the surge line (top boundary of map). The lines do not cross or intersect each other (but cut across other values) and are assigned an arbitrary value that increases as they move away from the surge line. Usually the surge line is defined as having $R=1$. The advantage to the “R” value is that the three map properties (pressure ratio, corrected flow, and efficiency) can be defined in a tabular format as a function of only the corrected speed, stator angle, and the “R” value. The design point then needs to be defined with the R value, the corrected rotational speed, and the stator angle. In comparison, for the map without the “R” lines, if just given the two values of corrected rotational speed and stator angle, then a whole family of possible values for compressor pressure ratio, corrected mass flow, and adiabatic efficiency needs to be deciphered.

Installation drag can be calculated by NNEP89. This includes spillage drag, bleed drag, and boattail drag. Any viscous drag losses must be supplied by the user. In addition, there is no separation criterion for over-expanded nozzles which can cause errors in the calculation of pressure drag. An afterbody drag prediction analysis was undertaken using ACSYNT [129]. It was found that propulsion boattail drag was only reliable for axisymmetric nozzles in ACSYNT, but that it was able to perform a nozzle interference drag correction. Also, it was noted that ACSYNT under-predicted boattail drag in the supersonic regime but did well in the subsonic regime. An alternative to ACSYNT for the prediction of boattail drag was the PIPSI code [130]. However, although faster to execute, PIPSI relies on accurate drag tables, which generally do not exist during the conceptual design stage of an aircraft.

3.B GasTurb Propulsion Model

The next propulsion analysis code to be considered is called GasTurb 11 or GasTurb 12. The User's Manual goes into great detail about this program [131]. GasTurb is able, for the most common gas turbine architectures, to perform a thermodynamic cycle analysis for both the design point and off-design conditions. It is also able to calculate the geometrical design or dimensions of the engine. There are three different modes of analysis called the

- basic mode,
- performance mode, and the
- more mode.

The “basic” mode controls basic parameters like the pressure ratio, the turbine inlet temperature, and component efficiencies. The more detailed parameters are set to defaults. This is a valid assumption, because during conceptual design, more detailed design parameters of an engine are left as default values based on historical/existing engines.

The “performance” mode gives the user more data input options, like the simulation of the internal air system, and the specification of turbine cooling. The basic dimensions of the engine, like the nozzle area, are required as input. Calculation options are also added in this mode like cycle optimization and Monte Carlo analyses (used to analyze the effects of manufacturing tolerances).

The “more” mode calculates the flow area at all thermodynamic stations and the static parameters for the design point and off-design conditions. There is a promise with this mode that preliminary design results will be improved and some insight into the interaction between thermodynamics, aerodynamics at the component level, and the actual physical design of the engine will be gained [131]. This mode also includes the ability to design the compressor and

turbine disks – a task that is beyond the scope of conceptual and preliminary design. In addition, the dimensions of the flow annulus are derived and an engine cross section is output.

The pertinent engine models for an ESAV vehicle that are available for analysis in GasTurb are the turbojet, turbofan, and variable cycle engines. The turbojet and turbofan have different options available as seen in Table B-2. Selected cycles can have changes made to them by the user including choosing the nozzle to be convergent or converging/diverging or whether there is an afterburner or not. There are other cycles that are also available not relevant to an ESAV concept like turboshafts and turboprops.

Table B-2. Engine Cycles Available in GasTurb

Turbojet	Turbofan	Variable Cycle
Single spool	Two spool (mixed and unmixed)	Two bypass door locations
Two compressor	Three spool (mixed and unmixed)	
Two spool	One speed mixed	

There is a two-level internal error checking feature in GasTurb. This is based on ranges set on input variables by the programmer. The first level generates a warning message. To trigger the warning message an input variable cannot be too far from the defined range. The second level generates an error message which is sent to the user if the input variable is far out of the defined range. The default ranges can be changed by the user, however, if unusual propulsion cycles are required by the user.

Output data meant for other disciplines' input can be in the form of either built-in variables or composed variables. Composed variables are user-defined variables built with mathematical relationships between the built-in variables. Constraints, in the form of maximums and minimums, can be placed on these variables. The constraints can then be used during any

internal optimization of the engine. For example, the user could specify that the fan pressure ratio be equal to 3.0, which effectively puts an equality constraint onto the system solution. Sensitivities of parameters and composed variables to the propulsion cycles can be calculated by GasTurb. If manufacturing tolerances are known for components, a Monte Carlo simulation can be performed to get better estimates for the real component dimensional spread.

Variable geometry components (blade angles) can be handled by one component map in GasTurb by employing the assumption that guide vane angles will all change simultaneously, given they are all connected to the same actuator controlled by the corrected compressor speed. However, if this assumption is not valid, more complex variable geometry is also available through the use of multiple maps. Like the previous propulsion models, GasTurb comes with predefined ASCII file component maps that can be modified by an advanced user.

GasTurb has the option of automatically using bleed flow to control an engine and keep it from surging (allow bleed flow from the engine core to reduce the back pressure when near engine-surge conditions). Of course, by using this option, an engine may end up being feasible but have its performance much reduced. This automatic bleed flow handling option can either be an inter-stage bleed or downstream of the compressor.

An aircraft mission can be defined using an option called a design table. This table includes constraints. The engine is then simulated within the mission parameters at all defined mission points without restarting GasTurb, and detailed results can then be exported as an ASCII file or a spreadsheet file. The off-design points are calculated with the Newton-Raphson iteration method. This off-design option is very attractive from the viewpoint of an aircraft MDO framework in that all propulsion performance at mission points can easily be found from a pre-

defined input table to GasTurb. Sensitivities at these off-design points can also be calculated by GasTurb.

Since a large difference may occur between off-design and design-point sensitivities, this information may be very helpful in understanding the characteristics of a given engine design. Composed values can be defined using the MissionSum and MissionAvg functions. The operation performed by these two functions is self-explanatory. For example, to calculate the total fuel used in a mission, a variable can be defined which calculates the fuel used in each segment and then the MissionSum function can be used to calculate the total fuel used during the mission. Another similar option available in GasTurb is the definition of a flight envelope which is a list of air speeds and altitudes, starting at sea-level, that the engine will operate in.

Transient analyses are available with GasTurb (not available in FLOPS or ACSYNT). These analyses are computed using Euler integration (uses the Newton-Raphson iteration method at each time step). Before a transient analysis can be computed, a starting point is defined which requires the calculation of a reference operating line. For example, if the engine acceleration transient were to be calculated, then the idle operating line would have to be first calculated. The fuel flow calculations are not accurate during transient analyses. Transient calculations are limited in GasTurb to those properties dependent on the polar moment of inertia of the compressor and turbines. Other transient behaviors, like heat transfer and blade tip clearance, are not able to be calculated by GasTurb.

The “more” mode allows for more detailed analyses of components and engine internal flow that can be generally considered beyond the scope of conceptual design, but this is required to calculate the design point engine dimensions. Here, the design Mach number or design flow area is specified. The geometry of the engine can be calculated with this mode, but it takes more

computational resources to do so. GasTurb can also be used in conjunction with experimental testing of engines but this is beyond the scope of ESAV conceptual design.

For bypass engines (turbofans or variable cycle engines) the mass flow can either be calculated or the core flow can be specified. However, if the core flow is specified then the required bypass fan flow will be calculated or the compressor design will be altered to fulfill physical requirements.

An afterburner can be specified as a reheat temperature (temperature obtained by the flow from the additional afterburner combustion process) for off-design calculations. The nozzle geometry will be automatically changed to compensate for the activation of the afterburner, and this is done assuming the action of a single actuator. Alternatively, the user can build a custom nozzle model using composed variables. The afterburning effects are calculated in two steps. The first is to specify the dry (no afterburning) operating point at steady state. This will allow the calculation of the inlet conditions for the afterburning system. Next, the necessary afterburner fuel flow that is required to reach the specified reheat exit temperature is calculated. This step, will in turn, cause a new nozzle throat area to be calculated.

GasTurb is capable of performing cycle optimizations internally. Two methods are available to do this, a gradient search, and an adaptive random search. A global optimum is attempted by using a multiple start location strategy. When an optimum is found from a certain starting point, a new starting location is found and another optimization occurs. This is done until the same optimum is reached from different starting points or until the user stops the optimization. Up to seven design variables can be optimized in the same cost function with up to seven constraints. The constraints may apply to both the design point and off-design points. If a

mission is defined with one single off-design point, then an optimization can be done at that off-design point.

If the GasTurb user is not satisfied with the standard component maps that come with the software, then custom maps may be used. The standard maps have been taken from the open literature, however, if the case arises that an engine design deviates too far from these maps (e.g. if the compressor pressure ratio is too high) then special maps should be considered. Compressor and turbine maps are based on Mach number similarity. Therefore, for a given point, all Mach numbers are implicitly fixed, which means that all losses that are a function of Mach number are also fixed. The viscous losses depend on the Reynolds number, therefore for smaller Reynolds numbers, a correction factor may need to be applied as the viscous losses will have an effect on the engine performance.

GasTurb can handle inlet flow distortion into the engine using the parallel compressor theory [100]. This theory allows it to handle both pressure and temperature distortion. This is done by splitting the actual flow into two equivalent flows. Each flow has uniform, but different, total properties. These two streams enter two respective imaginary compressors working in parallel that both have the same component map. Both compressors discharge to the same static flow properties downstream that allow the calculation of operating point positions on the map. If either of these two points reaches the surge line, the stability boundary is reached and the engine is predicted to surge (even if the mean operating point is far from the surge line). GasTurb does not predict the effects of temperature distortion as accurately as pressure distortion. For cases where compressors have more than one stage with vanes or struts in between, then the assumption that the parallel compressors discharge to the same static flow properties is no longer

valid. GasTurb allows for this situation by simply setting the “no mass flow between sectors” option to “on” for the compressor.

The engine control system can be simulated in GasTurb by the use of constraints. Set properly, not all constraints will be active at all flight regimes in a given mission but some may be active at some points of the mission. This will realistically constrain the engine to operate in acceptable limits much like a control system would work. Also, as mentioned previously, automatic bleed may be used to protect against surge, which is another function of a control system. A control schedule may be defined which makes maximum constraints of built-in parameters and composed parameters a function of flight conditions. GasTurb includes a PID control option for fuel flow where the user sets the gains.

The gas properties in GasTurb are “half-ideal” in that the specific heat, enthalpy, entropy, and gas constant are only functions of temperature and the gas composition. Pressure is assumed to not affect these properties in GasTurb. The combustion temperature is a function of initial temperature, the fuel-to-air ratio, the water-air-ratio, and pressure. Various commercially-available jet fuels are predefined models in GasTurb and can be selected easily from the file in which they reside. In addition, custom fuels can be added to this file.

An international standard atmosphere model is employed by GasTurb and allows for offsets (like a hot day temperature offset). Fogging and wet compression effects can be included in the engine analysis. Emissions from the engine can be calculated including CO₂, NO_x, CO, and unburned hydrocarbons.

GasTurb under-predicts its calculations of engine mass [131], because several real, but small, details of the engine are not modeled. These details may be added by carefully choosing thicknesses for casing material; however this is generally beyond the scope of the conceptual

design level. The mass of inter-duct points assumes they were manufactured by casting. The struts are assumed to be hollow and to have the same thickness as the inner and outer walls of the flow channel. Again, the engine dimensions are calculated with the “more” mode.

For supersonic engine design, which is pertinent to the ESAV, the standard map scaling procedure cannot be applied, and significant problems may arise. The User’s Manual [131] does provide a section with suggestions on how to perform supersonic analyses.

4.B NPSS and WATE++ Models

4.B-1 NPSS

The Numerical Propulsion System Simulation code (NPSS) is a propulsion simulation code developed by a collaborative partnership between NASA Glenn, the U.S. aero-propulsion industry, and the U.S. Department of Defense [93]. It supports detailed aero-thermo-mechanical computer simulations of complete aircraft engines and realistically models the physical interactions that occur throughout an aircraft engine. One-, two-, and three-dimensional models of the internal and external engine flow can be incorporated into an NPSS engine model. However, the core of NPSS is an engineering model built up with individual engine components and their physical interactions. NPSS is able to reconcile these higher-dimensional models with the zero-dimensional component-based models by its object-oriented design.

NPSS is a component-based object-oriented engine cycle simulator that is designed to perform many common tasks related to a propulsion system. It can perform engine cycle design, steady-state and transient off-design performance prediction, test data matching and many other traditional tasks of engine cycle simulation codes. This includes calculating the correct fuel flow to the engine. Similar to the other propulsion codes considered previously, its engine model is

assembled from a collection of interconnected components controlled through the implementation of an appropriate solution algorithm. NPSS is currently being developed and improved, and one improvement is the addition of more sophisticated component models than in previous versions. NPSS can operate with either physics-based models or simple thermodynamic models based on component efficiencies and engine configuration [132].

The working gas that makes up the flow can be modeled using several different existing tools. Each is a National Institute of Standards and Technology (NIST) compliant gas property table. In fact, thermodynamic packages must be specified before any engine elements or subelements are instantiated by NPSS. The available models are

- CEA (NASA chemical equilibrium code),
- JANAF (implementation of NIST gas properties prepared by Honeywell),
- GasTbl (based on Therm with additional humidity calculations and some chemical equilibrium capabilities developed by Pratt & Whitney),
- AllFuel (package containing gas and fuel properties prepared by General Electric), and
- FPT (package used by NPSS tables and/or functions that describe the thermodynamic properties of the fluid).

Each package is different and must be used with attention towards each packages' specific details.

NPSS has many convenient built-in features. The top level solver in NPSS is set up automatically, can be supplied constraints by the user, and can handle discontinuities that may arise during a solve computation. Both steady-state and transient analyses can be performed. The user has the flexibility to define custom functions and components using the NPSS programming

language (very similar to C++). The program also offers support for test data matching and analysis. Design points and off-design points are specified by the user.

An engine model is defined in NPSS by the use of input files, and can therefore be easily automated in the ESAV MDO framework. One or more input files may be used, and in order to keep functionality compartmentalized and organized, several files are usually used to define the NPSS model. The first step is to define the gas-property model of the fuel. Next, the model's components need to be defined along with the linkages between components. There are many standard components available that can be used to build up an engine model. Table B-3 lists these along with their description.

Table B-3. NPSS Components Used for Engine Model Build-up

Engine Component Name	Description
Element	Major engine components and their attributes
Socket	Software method to add needed calculations to an element
Subelement	Like an element but designed to support the object to which they are attached. Passes information via sockets.
Function	Perform special calculations required by other objects
Table	Special functions that perform table lookups
Port	How elements communicate their input and output
Link	Manage the data flow through ports
Assembly	Connected elements
Executive	Account for and run defined executable tasks in an assembly
Solver	Drive model to a physically-valid solution

The user then specifies details of the solver constraints and what output to provide from the converged engine simulation. A cautionary note is that flow of data between objects can be complex [132]. Finally, the solution can be initiated for the specified cases and inputs. The output data format is defined by the user through several data viewing options. This allows for easy integration of the NPSS model with external models required by the user, and the larger propulsion model in an MDO framework. Using these basic NPSS functionalities and tools, any gas turbine engine configuration may be built with NPSS.

When the abstract NPSS elements (components) ports are linked, thermodynamic stations are created. NPSS has internal checks to ensure that input and output are properly connected. Every port must be linked to another port, but a restriction is imposed in that a port can be linked to one, and only one, other port. A subelement is considered a child of its element or even other subelements that call it. A subelement will usually perform major calculations in behalf of its parent (e.g. compressor or turbine maps) and are designed to fit into sockets. A simple turbojet engine model cycle to be analyzed by NPSS is shown in Figure B-2. This flow chart is based on a similar model found in Reference [93]. This flow chart shows examples of an assembly, several elements, a subelement and socket, bleed flow, different input and output port types, links throughout, a top-level solver, and data viewers.

Several other levels of solvers can exist in addition to an overall, or top level solver. If the top level solver is used, it is responsible for the time step accounting. The top-level solver object is automatically instantiated, but it still must be configured by the user to work properly unless all elements use their default independent and dependent variables. An example of using lower-level solvers is when solvers for compressor and/or turbine assemblies are defined. These solvers converge before the top level solver for the entire engine assembly is executed and can

speed the top-level engine solution. The order of execution of objects within an assembly must be defined by the user through the executive.

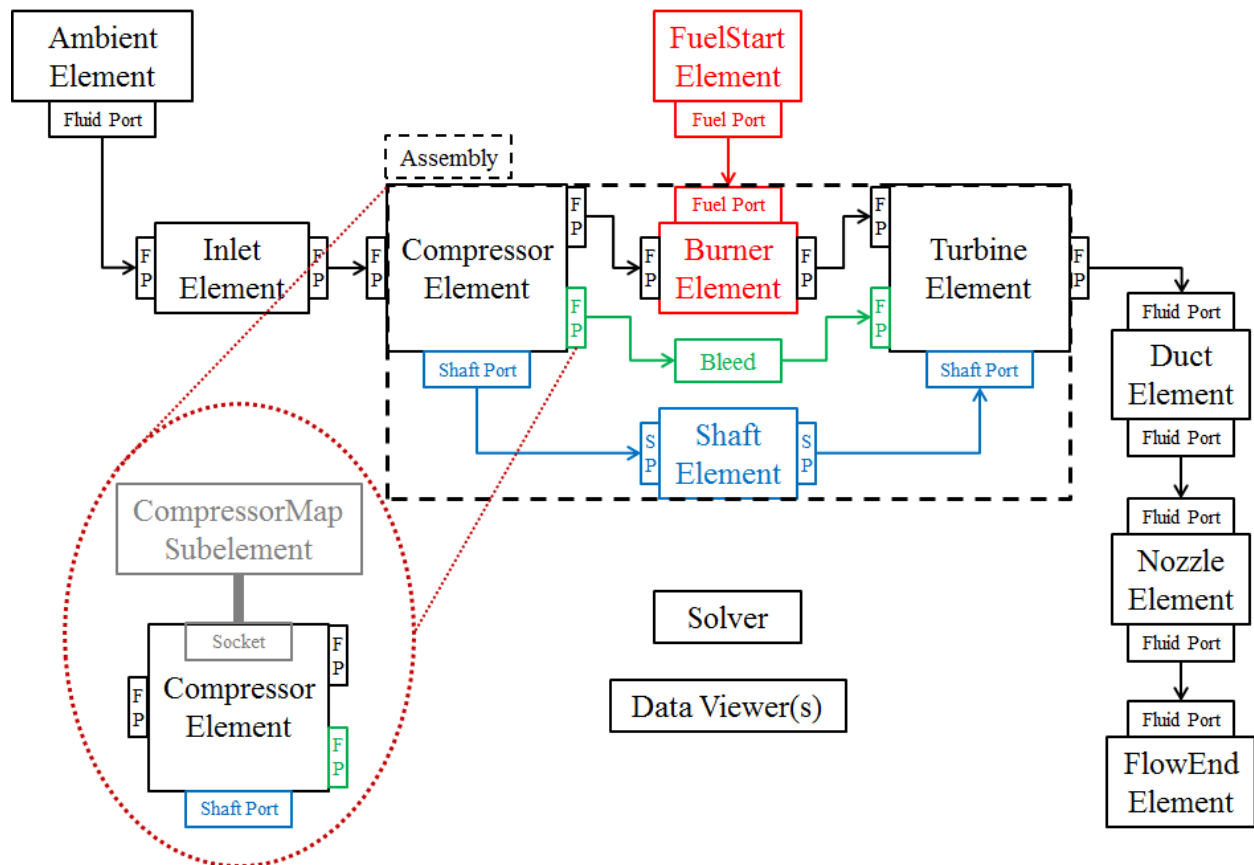


Figure B-2. Simple turbojet engine model with top-level solver and dataviewer in NPSS showing a bleed, elements and ports and a blow-out view showing a subelement and socket.

Other codes can be integrated into NPSS by use of the Common Object Request Broker Architecture (CORBA). This allows an NPSS simulation to be run on different computers, each of a different architecture. For example, an NPSS model running on a UNIX workstation could obtain its high-pressure compressor definition from a multi-stage three-dimensional computational fluid dynamics program running on a supercomputer. This feature allows parallel computing to be used for more complex models which can speed execution time. In addition to flow codes, NPSS can link engine analysis with heat transfer and structural models [132] for

complete propulsion integration design. The NPSS models can be run either in batch mode, which requires the input and output to be contained in files, or in an interactive mode where the user inputs commands using the command line format. The interactive mode is mostly for debugging purposes but is not limited to that. Data is input using the NPSS programming language format.

NPSS software development has gone through, and continues to go through, a sophisticated process as the U.S. propulsion industry employs it for real-world applications [133]. Specifically, validation was a concern very early in the process [132]. The process has adhered to identifiable and recognizable software development processes from its start. This includes

- software development plans,
- configuration management plans,
- verification and validation plans, and
- design plans.

An incremental release process is followed which includes bug fixes and any urgently needed enhancements without waiting for major releases.

An engine control system model can be integrated with the engine model in NPSS [134]. Two possible methods for modeling such a system are to have a

1. generic controller interface for communication with control development software, and
2. linear model generator which creates linear models from nonlinear NPSS simulations.

The generic controller interface gives the user the flexibility to use a control model which exists in some other software. The linear model generator is useful in that it can transform the nonlinear

engine simulation into a usable general linear form with Eq. B1a being the state equation and Eq. B1b being the output equation.

$$\dot{x} = Ax + Bu \quad (\text{B1a})$$

$$y = Cx + Du \quad (\text{B1b})$$

Here, x is the state vector, u is the control vector, y is the output vector, A is the state matrix, B is the input matrix, C is the output matrix, and D is the feedthrough matrix. The linear model is created about a base engine operation point, like the design point, and the four Jacobian matrices A , B , C , and D are generated with small-perturbation theory. Sensor models can be “placed” on the engine at various locations to measure engine conditions in order to provide input to the controller.

NPSS has been used in previous optimization studies. For instance, one pertinent study to the ESAV is that NPSS was part of a process to optimally design a relaxed compression supersonic inlet [21]. This can produce smaller cowl drag, although at a cost of a decrease in the pressure recovery and an increase in flow distortion. In a supersonic inlet, the oblique shock train associated with the inlet is eventually terminated by a normal shock to reduce the flow to the subsonic flow conditions required by the inlet. Viscosity and the inlet geometry have a great effect on inlet performance and so must be included in the analysis for accurate results; otherwise significant degradation of inlet performance will not be captured. Reference [135] describes a coupling approach between the inlet fluid simulation and the NPSS engine model. In addition, other external codes can be integrated in an inlet simulation, like optimization and sensitivities. The inlet design is completed by varying the nozzle throat area of the engine to complete the converged NPSS solution.

A few other features of NPSS are noteworthy. For instance, NPSS is capable of representing variable geometry of the blades in the compressor and turbine [136]. During transient simulations, fluid momentum and rotor inertia effects can also be modeled. NPSS was previously compared with two other codes in a transient analysis. One of these was a code written with the Modelica computer language used in conjunction with the graphical environment, Dymola, and the other was written in Matlab/Simulink. The NPSS model was found to have nearly identical transient results, although it was the slowest of the three simulations [136]. The installation effect of ram drag is calculated with NPSS elements PerfNASA or EngPerf, or possibly by some other user-defined element [93].

4.B-2 Weight and Dimensions

The Numerical Propulsion System Simulation software by itself does not produce the dimensions and weights of a given engine model. Its purpose is to complete the thermodynamic analysis of the fluid as it would behave within a given engine model, thus allowing for the prediction of engine performance. The low-level details required to get an accurate weight of the engine model being simulated by NPSS requires a code that can take these details into consideration. Some of these possible details are the required connecting hardware, material selection, the kind and number of bearings, the design choice for the compressor and turbine disks, the design of the corresponding blades, the number of these blades, etc.

The time-honored Weight Analysis of Turbine Engines (WATE) code requires high level engine details [137], which resulted in an engine weight accuracy of $\pm 10\%$ after comparison of results with nine different engines. An empirical database strategy was employed that consisted of component weight data from military and commercial engines. There were a total of twenty-nine engines used that included both turbofans and turbojets, dry and augmented. Some of these

engines had been actually manufactured and the rest were proposed (this work was done in the late 1970's). Both the subsonic and supersonic flight regimes were also represented.

The WATE program was developed based on correlations of component weight and physical characteristics rather than on major cycle characteristics like bypass ratio and overall pressure ratio [137]. In this way, the weight of the engine was built up component by component, and was thought to add flexibility and accuracy. However, large scatter in the data from these twenty-nine engines precluded the use of trend lines to estimate component weights. This is because each engine that was part of the database was originally designed with different design criteria causing significant weight scatter. The different design criteria identified which caused the large scatter were

- material properties,
- blade geometry,
- stage loading,
- shaft speed, and
- design life.

The WATE software assumed that the thermodynamic characteristics were known from a given engine model. During the late 1970's when WATE was created, the Navy-NASA Engine Program (NNEP) [68] was the state-of-the-art engine simulation program, however, today that program is NPSS. Researchers at NASA have integrated the object-oriented NPSS code and an object-oriented version of WATE called WATE++ [138]. An example turbofan engine design pictorial output from WATE++ is shown in Fig. B-3. More engine detail is output into the WATE++ output file than is shown in the graphical output, facilitating MDO framework automation.

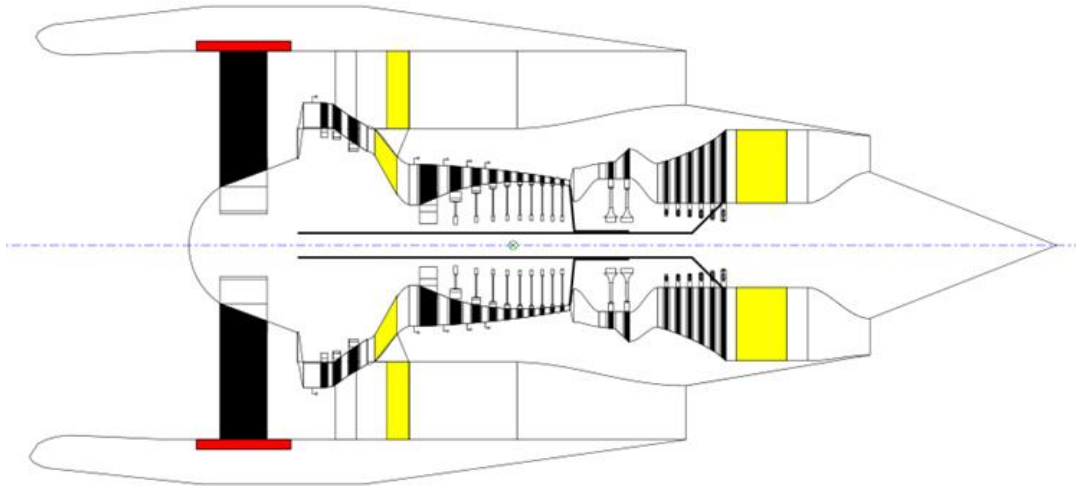


Figure B-3. Example turbfan WATE++ engine design.

The integration of NPSS and WATE++ allows for seamless data transfer between thermodynamic cycle solutions from NPSS to the component weight programs in WATE++. The reason for wanting the close data relationship can be seen with the example of how WATE calculates the weight of axial flow fans and compressors.

4.B-2.1 WATE Compressor Example

An example of a major engine component, the compressor, effectively introduces the software's capability and use. The required data input for WATE to correctly analyze a compressor [137] is

- allowable pressure ratio – first stage,
- entrance and exit Mach number,
- hub-to-tip diameter,
- compressor geometry design mode,
- blade material effective density (total blade weight / total volume),
- maximum inlet & exit temperatures,
- first & last stage blade aspect ratios,
- overspeed factor (maximum rotation speed / design rotation speed),

- blade solidity,
- density of the disc material,
- blade taper ratio, and
- blade volume factor (total volume / blade volume).

The shaft speed is found using a curve fit to data. Several of these data are obtained from the NPSS thermodynamic simulation. Other data must be generated from material assumptions and the choice of engine design parameters. From a conceptual design perspective, the default values are usually used, unless there is a reason to deviate to a better value for a given application. The materials selected are likewise the same as those used in the example engine models that came with the NPSS/WATE++ code maintained by NASA.

To continue with the example of the axial flow compressor [137], the first stage flow area is determined by the specified inlet Mach number and by the corrected airflow from the cycle data. The specified radius ratio is used to find the inside and outside diameters of the flow path. The shaft speed (tip speed) and stage radius are used to calculate the compressor angular velocity in units of RPM. Moving into the compressor, the stages behind the first stage are found by the compressor geometry design mode chosen by the user and the values calculated for the first stage. These downstream stage dimensions along with the shaft speed are then used to find the tip speeds (important parameter to determine weight). The blade aspect ratio for these downstream stages is found by assuming a proportional change for each stage (if entry and exit aspect ratios are different). The volume of the metal in the stages is then calculated, and drawing upon the properties of the material selected (specifically density), a weight is assigned. The total stage weight requires the blade solidity factor input by the user which finds the number of blades for each stage (rounded down between integers).

The length of the stages uses the blade chord, c_{blade} , found from the height of the blade, h_{blade} , divided by the aspect ratio, AR ,

$$c_{blade} = \frac{h_{blade}}{AR}. \quad (B2)$$

Examination of the database of engines used as the foundation to WATE revealed a general trend. This was that the stator length was equal to the rotor length (or c_{blade}) with 17% for clearance between stators and rotors. Therefore, using this observed relationship, the sum of all stages' c_{blade} gives the total compressor length.

The compressor discs are a major weight contributor to both the compressor weight and the total engine weight. The maximum blade pull stress at the blade root is an important value in determining the weight of the compressor via these disk weights. It is a function of the tip speed, blade height, material density, design rotation speed, and the overspeed factor. An inversely-proportional relationship exists between the disc thickness and the product of the blade pull stress and disc diameter (blade load per unit thickness) which allows the calculation of the compressor disc weight [137].

The connecting hardware weight is calculated using an equation that assumes the metal to be steel and a fixed thickness dimension for the spacer (0.075 inches thick). The disk radius, stage length, and material density are parameters in this equation [137]. The outer case is also done stage-by-stage and is assumed to be made out of the same material as the stage disk. A thickness is supplied by the user (default value is 0.1 *in*), and the weight calculation is performed by WATE.

The total stage weight is the simple sum of the blades, stators, disk, connecting hardware, and case weights. Each stage weight is then summed to give the total compressor weight. Thus

the diameter, length, and weight of the compressor can be found. The rotational inertias are also found in a similar stage-by-stage fashion.

4.B-2.2 Other Components

The weight for an axial turbine follows a very similar process as seen for the compressor.

However, for the turbine case, the shaft speed is transferred from the corresponding compressor.

There are two ways to solve for the dimensions and weight. One method is to specify the first stage's maximum diameter and then the number of other stages required from the work loading parameter (loading required on stages to obtain the required work from the turbine to drive the compressor). The other method is to specify the number of stages and then back out the first stage's diameter using the work loading parameter. Also, a cooling indicator can be input to account for cooling holes in the blades. A turbine stage is considered a stator-rotor pair rather than the rotor-stator pair defined for the compressor.

Duct weight analysis assumes that the internal pressure is the major structural load and that the duct can be modeled as a thin-walled cylinder subjected to an internal pressure. Using the specified duct material properties, the resulting inner and outer diameters, thickness, and length the weight falls out with a simple calculation.

Burners are very similar to ducts but there are wall liners, a burner dome, a fuel manifold, and fuel nozzles added. Primary burners also have the associated frame weight. The liner assumes a default thickness of 0.055 in.

A shaft requires connected component details, the material of the shaft, and the radius ratio of the inner shaft (if it is concentric). The shear stress is found from the torque load which sets the required shaft diameter. The diameter is then used along with material properties and the length to calculate the weight. Concentric shaft weights are done in a similar manner except that

there is an iteration performed to solve for the separate diameters. A 0.2 inch clearance is assumed between the concentric shafts. Due to other detailed considerations (like bearing placement) the shaft weight calculated by WATE should be considered the lower bound on the actual weight. The rotational inertia is assumed to be negligible compared to the compressors and turbines.

Frames are generally used to support bearings. Frame weights are based on a correlation of the total frame-projected area and local diameter. Nozzle weights are also based on past trends and can include multiple-stream, fixed-geometry, and variable-geometry types. The nozzle length needs to be specified and the diameter is taken from the connecting component. The wall thickness is assumed to be 0.1 *in* for fixed nozzles and 0.275 *in* for variable-geometry nozzles.

Other components are also part of the WATE code but are not used this effort (like centrifugal compressors typically used in small gas turbine engines), and therefore no explanations of these are included here. For more information on WATE's abilities and example engine models the reader is referred to Reference [137].

4.B-2.3 Improvements to WATE

Since the late 1970's, when WATE was initially completed, several improvements to the code have been made. Four relevant improvements are discussed here. These include the incorporation of improved weight-calculation routines for compressor and turbine disks, a common aerospace materials database, the incorporation of object-oriented programming principles, and the integration with the current state-of-the-art NPSS engine thermodynamic code. The resulting code, WATE++, which benefits from each of these enhancements, is the version of WATE being utilized for the present analysis.

Improved weight-calculation routines for compressor and turbine disks were incorporated into WATE for two main reasons. The first is due to the criticality of reliable engine weight estimation in the conceptual design stage [139]. The second was the ability to design an engine that met safety demands (e.g. the most catastrophic failure of an engine is potentially a disk burst [139]). The safety demands are difficult to enforce because modern designs are pushing the limits of tip speeds and turbine inlet temperatures in order to increase engine performance; both of which increase the risk of having a disk burst. The ESAV will be no exception from this trend.

Related to the weight estimation of the compressor and turbine disks is the selection of the appropriate material from which the disk will be made. The proper selection of disk material will affect both the feasibility of the design and its weight. WATE++ was improved by incorporating a common aerospace materials database. Each component of the engine can have its material specified using the available database materials. Then the materials database is used to look up that component's density, modulus, yield strength, ultimate strength, Poisson's ratio, and some fatigue data. A list of these available materials is included in Reference [139].

To improve the maintainability and extensibility of the original FORTRAN version of WATE, it was converted into an object-oriented version [138] called WATE++. WATE++ has its own ports, elements, and subelements much like NPSS. Likewise, input/output is handled with text files making it ideal for automation in an MDO framework. It calculates the weight and dimensions of individual engine components with analytical calculations replacing many of the original empirical relationships. These component results are then performed for the complete engine. The usual way to calculate the weight of a component is to calculate the material volume and then multiply that by the material density [138].

WATE++ is extensible and includes inheritance, polymorphism, and encapsulation object-oriented concepts in its design. Code duplication is minimized by the use of abstract component classes and inheritance. Readability and maintainability are improved by the use of encapsulation. Polymorphism allows single identifiers to work with any object class which simplifies cross-component analyses, like calculating the component geometries to draw the complete engine. One identifier is used in this way to work with all of the different engine component geometries to draw the whole engine as designed (see Fig. B-3).

NPSS is itself an object-oriented code, and the shift to an object-oriented WATE code facilitated the implementation of WATE++ seamlessly into the NPSS framework. In this way, WATE++ calculations have direct access to the thermodynamic design of the engine and all of the accompanying data. This data is critical to accurately simulate the mechanical design of the engine components, and assign the corresponding weight and dimensions. Thus, for this effort, both the thermodynamic propulsion design and the corresponding weight and dimensions of the resulting physical engine are calculated and input with the use of just one executable code.