MULTIOBJECTIVE DESIGN OPTIMIZATION OF ROCKETS AND MISSILES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MUSTAFA YAVUZ ÖZTÜRK

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
AEROSPACE ENGINEERING

MARCH 2009

Approval of the thesis:

MULTIOBJECTIVE DESIGN OPTIMIZATION OF ROCKETS AND MISSILES

submitted by **MUSTAFA YAVUZ ÖZTÜRK** in partial fulfillment of the requirements for the degree of **Master of Science in Aerospace Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. İsmail Hakkı Tuncer _____
Head of Department, **Aerospace Engineering**

Prof. Dr. Ozan Tekinalp _____
Supervisor, **Aerospace Engineering Dept., METU**

Examining Committee Members:

Dr. Güçlü Seber _____
Aerospace Engineering Dept., METU

Prof. Dr. Ozan Tekinalp _____
Aerospace Engineering Dept., METU

Dr. İlkay Yavrucuk _____
Aerospace Engineering Dept., METU

Dr. Dilek Funda Kurtuluş _____
Aerospace Engineering Dept., METU

Dr. Tayfun Çimen _____
Systems Engineer, ROKETSAN

Date: 27.03.2009

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Mustafa Yavuz Öztürk

Signature :

ABSTRACT

MULTIOBJECTIVE DESIGN OPTIMIZATION OF
ROCKETS AND MISSILES

Öztürk, Mustafa Yavuz

M.Sc., Department of Aerospace Engineering

Supervisor: Prof.Dr. Ozan Tekinalp

March 2009, 100 pages

Multidisciplinary design optimization of aerospace vehicles has attracted interest of many researchers. Well known aerospace companies are developing tools for the mutlidisciplinary design optimization. However, the multiobjective optimization of the design is a new and important area investigated very little by the researchers. This thesis will examine the approaches to the multiobjective and mutlidisciplinary design optimization of rockets and missiles. In the study, multiobjective optimization method called MC-MOSA will be used.

Keywords: Design Optimization, Missile Conceptual Design, Simulated Annealing

ÖZ

ROKET VE FÜZE TASARIMINDA ÇOK AMAÇLI OPTİMİZASYON

Öztürk, Mustafa Yavuz
Yüksek Lisans, Havacılık ve Uzay Mühendisliği Bölümü
Tez Yöneticisi: Prof.Dr. Ozan Tekinalp

Mart 2009, 100 sayfa

Hava uzay araçlarının çok disiplinli tasarım optimizasyonu pek çok araştırmanın konusu olmuştur. Tanınmış havacılık ve uzay şirketleri çok disiplinli tasarım optimizasyonu için araçlar geliştirmektedirler. Fakat tasarımın çok amaçlı olarak optimizasyonu yeni bir yaklaşım olduğundan bu alanda pek bir çalışma yapılmamıştır. Bu tezde füze ve roketlerin çok amaçlı ve çok disiplinli tasarım optimizasyonu çalışılacaktır. Çalışmada daha önce geliştirilen ve MC-MOSA adı verilen çok amaçlı optimizasyon yöntemi kullanılacaktır.

Anahtar Kelimeler: Tasarım Optimizasyonu, Füze Kavramsal Tasarımı, Benzetilmiş Tavlama

TABLE OF CONTENTS

APPENDICES

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF SYMBOLS

$\delta\bar{f}^{+}$	Initial average objective increase
$f_i^{0,\max}$	Maximum reference values for each objective function
$f_i^{0,\min}$	Minimum reference values for each objective function
$\tilde{F}_j^{*}$	Minimum semi-major estimation
$\tilde{f}_i(x)$	Normalized objective functions for each objective
σ_0	Variation in the objective function during initial search
$\chi_d^2(1-p)$	$100(1-p)$ percentile point of the chi-square distribution with d degrees of freedom
$\tilde{F}_j^{best}$, $\tilde{F}_j^{2^{nd}best}$	Current best and 2nd best objective function values
$[x_b,z_b],[x_E,z_E],[x_s,z_s]$	Body, earth and stability axes
A	Accuracy
a	Semi-major axes of ellipses
A, B	Number of inequality and equality constraints on design variables
a_{3d}	Semi-major axes of ellipsoids
A_b	Area of propellant burning surface
ALPHA, MACH	Angle of attack and Mach number vectors
a_{sound}	Speed of sound
A_t , A_e	Nozzle throat and exit areas
a_T , a_N	Tangential and normal accelerations
$b_{i,j}$	Span lengths of finset elements, measured from root
B_{nose}	Nose bluntness radius
C,D	Number of inequality and equality constraints on objectives

C_F	Thrust coefficient
$c_{i,j}$	Chord lengths of finset elements
$\mathbf{C_{L,1}}$, $\mathbf{C_{D,1}}$	Lift and drag coefficient databases corresponding to before-burnout and initial cg
$\mathbf{C_{L,2}}$, $\mathbf{C_{D,2}}$	Lift and drag coefficient databases corresponding to before-burnout and final cg
$\mathbf{C_{L,3}}$, $\mathbf{C_{D,3}}$	Lift and drag coefficient databases corresponding to after-burnout and final cg
$C_{L,trim}$, $C_{D,trim}$	Trim lift and drag coefficients
CLμ	Cluster ratio
COμ	Number of cells occupied by non-dominated points
d	Design space dimension
D_{aft}	End diameter of missile aft section
D_{body}	Diameter of missile body section
D_{exit}	Nozzle exit diameter
D_{nose}	Diameter of missile nose section
D_p	Propellant diameter
DPR	Design pressure ratio
e	Eccentricity of ellipses/ellipsoids
$E(\mathbf{r})$	Energy of the configuration
E_{max}	Maximum energy of missile
$E_{max}-E_{tot}$	Current consumed total energy of missile
e_{per}	Energy percentage parameter
E_{tot}	Current total energy of missile
FEN	Allowed number of function evaluations
$\text{FEN}_{\text{inner}}$	Number of inner loops
FFN	Number of fitness functions
$f_i(\mathbf{x})$	Objective functions
f_i^0	Reference values for each objective function
f_{int}	Linear interpolation function
f_{int2}	Bilinear interpolation function

f_{int3}	Trilinear interpolation function
g	Gravitational constant
g_a , h_b	Inequality and equality constraint function(s) on design variables
HD	Hyper area difference
I	Number of objectives
I_{sp}	Specific impulse
I_{tot}	Total impulse
J	Number of design variables
k	Specific heat ratio of combustion products
k_B	Boltzmann constant
k_{vl}	Volumetric loading parameter
L, D	Lift and drag forces
L_{aft}	Length of missile aft section
L_{body}	Length of missile constant radius section
$l_{flat,i}$	Length of constant thickness portion of the chord
l_{inlet} , l_{exit}	Length of nozzle inlet and exit sections
l_{max} / c	l_{max} to chord ratio
$l_{max,i}$	Distance from leading edge to maximum thickness point on the chord
l_{motor}	Rocket motor length
L_{nose}	Length of missile nose section
l_{nozzle}	Length of nozzle
l_{other}	Length of missile front section
l_p	Propellant length
l_{tot}	Missile total length
l_{wh}	Warhead length
M	Mach number
m	Missile current mass
m_0 , m_{fin}	Missile initial and final mass
m_c , n_d	Inequality and equality constraint function(s) on objectives

m_{case}	Rocket motor case mass
m_{con}	Control section mass
$m_{finsets}$	Total mass of all finest elements
m_{liner}	Liner mass
m_{motor}	Rocket motor mass
m_{other}	Mass of other sections of missile
m_p	Propellant mass
m_{wh}	Warhead mass
n	Pressure exponent of propellant
NDP	Number of non-dominated points
$node\#_1$	Number of nodes in a.o.a. command profile before burnout
$node\#_2$	Number of nodes in a.o.a. command profile after burnout
$n_{panel,i}$	Number of elements in finsets
OS	Overall spread
p	Probability that the real minimum is larger than its estimate
P	Probability function
P_a	Ambient pressure
P_c	Combustion chamber pressure
P_e	Nozzle exit pressure
P_{ref}	Propellant reference pressure
Q	Rotation matrix for ellipses
Q_{3d}	Rotation matrix for ellipsoids
R	Instantaneous radius of curvature of flight path
R	Range
RC	Restart coefficient
r_{ref}	Propellant reference burn rate
r_t , r_e	Nozzle throat and exit radii
S	Design space
s_{max}	Maximum step size
S_{ref}	Missile reference area
t	Current flight time

t/c	Thickness to chord ratio
T	Current thrust
T	Temperature
t_0	Initial simulation time
T_0	Initial temperature
t_b	Propellant burning time
t_{case}	Rocket motor case thickness
T_d	Design thrust
t_i	Thicknesses of finset elements
t_{liner}	Liner thickness
V	Current velocity of missile
V_0	Initial velocity of missile
V_∞	Free-stream velocity
V_{hit}	Hit velocity
W	Weight force
x	Current test point
$\mathbf{x}$	Set of independent design variables
x_0	Initial range
x_{cg}	Distance of the current cg of missile from nose
$x_{cg,0}$	Distance of the initial cg of missile from nose
$x_{cg,final}$	Distance of the final cg of missile from nose
$x_{le,i}$	Distances of root chord leading edges of finsets from nose
x_{lower}, x_{upper}	Lower and upper bounds of design variables
X_s , Z_s	Total forces along stability axes
y	New test point
z	Current altitude
z_0	Initial altitude
z_{hit}	Altitude at hit phase initiation
α	Angle of attack
α_0	Initial angle of attack
α_{com}	Current angle of attack command

α_{hit}	Hit phase angle of attack
$\boldsymbol{\alpha}_{\mathbf{node,1}}$	Angle of attack node array corresponding to before-burnout phase
$\boldsymbol{\alpha}_{\mathbf{node,2}}$	Angle of attack node array corresponding to after-burnout phase
$\alpha_{node,i}$	Angle of attack values at nodes
β	Scalar objective function parameter (with $1 \leq \beta \leq \infty$)
β_{inlet} , β_{exit}	Nozzle inlet and exit angles
γ	Current flight path angle
γ_0	Initial flight path angle
ΔE	Change in the energy of a system
ΔL	Distance between initial and final cg positions
δ_{lim}	Flap deflection limit
Δs	Infinitesimal displacement along the missile flight path
Δt	time step of one simulation loop
$\Delta\gamma$	Infinitesimal change in flight path angle
η_a	Penalty coefficients for inequality constraint functions
θ	Current pitch angle
θ_{hit}	Hit angle
θ_i	Random unit vectors
Λ_i	Random step sizes
$\Lambda_{le,i}$, $\Lambda_{te,i}$	Leading and trailing edge sweep angles of finset elements, positive backwards
v_i	Weights for each objective function
ξ_b	Penalty coefficients for equality constraint functions
ρ_{air}	Air density
ρ_{case}	Rocket motor case material density
$\rho_{finsets}$	Finset element material densities
ρ_{liner}	Liner material density
ρ_{nozzle}	Nozzle material density
ρ_p	Propellant density

σ_{case}	Yield strength of rocket motor case material
τ	Time constant
$\Phi_{i,j}$	Roll angles of finset elements, measured clockwise from top vertical center
Ω	Total constraint function due to inequality and equality constraints

LIST OF ABBREVIATIONS

a.o.a. :	Angle of attack
Av. :	Average
cg :	Center of gravity
cp :	Center of pressure
CS :	Case study
DOF :	Degrees of freedom
FE :	Function evaluation
FF :	Fitness function
MC-MOSA :	Multiple Cooling Multiobjective Simulated Annealing
RKF45 :	Runge-Kutta-Fehlberg integration method
SA :	Simulated annealing
SD :	Standard deviation

CHAPTER 1

INTRODUCTION

Design optimization can be defined as finding the maximum or minimum value of an objective function by systematically choosing the values of design variables from within an allowed set. In this definition, design variables refer to the specifications of the system that can be controlled by the designer during system design (e.g. diameter and wing span values of a missile). Design objectives are the outputs of the system, which are functions of design variables and are tried to be maximized or minimized by changing the values of the design variables (e.g. range and mass of a missile). They are represented by objective functions. Defining the design variables set and evaluating it during optimization process generally involve multidisciplinary studies since they represent the characteristics of the system in terms of many different disciplines such as aerodynamics, flight mechanics, propulsion, automatic control and structural mechanics. Moreover, many aerospace systems are actually "system of systems", that is, they involve several subsystems, each of which performs certain functions to satisfy the requirements allocated to it, so that the whole system satisfies the system requirements. This design approach actually emerges from the fact that modern aerospace systems are generally required to accomplish more than one and generally conflicting objectives as a result of increasing operational needs. Because of these reasons, multidisciplinary and multiobjective optimization is inevitable for many aerospace systems.

Irish statistician and economist Francis Ysidro Edgeworth [7] was the first to propose the notion of multiobjective optimization. Italian economist Vilfredo Pareto generalized the multiobjective optimization in his book first published in 1896 [19]. His name is given to the hyper surface (or the curve, in two-objective problems), containing the optimum solutions of multiple objectives (Pareto front).

Optimization algorithms are either deterministic, or stochastic. The deterministic ones require function derivatives, or gradient information. Some gradient-based methods are Newton's method, steepest descent and conjugate gradient. These algorithms are well-developed in the literature and have been adapted for multiobjective optimization. The usual approach is to use a scalarizing function to convert a vector of objectives to a scalar objective function [24]. A comprehensive review of various continuous multiobjective methods that use gradient information may be found in [18]. However, gradient-based algorithms converge to local optimum. Since most engineering problems are modeled with nonlinear, multimodal and even discontinuous functions, gradient-based methods may fail to provide even a feasible solution. To overcome this problem, several stochastic methods that do not require gradient information have been applied developed such as random search, grid search, evolutionary algorithms and simulated annealing. Among them, evolutionary algorithms and Simulated Annealing (SA) have been most successful [9]. They have also been adapted for multiobjective optimization. A detailed description of SA algorithms can be found in Chapter 3.

A missile is a guided weapon system that is propelled by a certain type of engine, follows a certain trajectory in the air and uses aerodynamic or thrust forces for steering. Launch vehicles used in space missions and unguided missiles are generally called rockets. Some characteristics that define the performance of a typical rocket or missile are range, accuracy, hit effectiveness and flight time. In addition to these, an effectively designed rocket/missile system has some other capabilities such as high reliability, maintainability, endurance against environmental conditions, and of course, low cost. A typical missile system is comprised of several subsystems such as launcher, engine, guidance-control units

and warhead. In addition, depending on the operational concept and requirements, some other subsystems may be implemented into a missile such as seeker and data link.

Missiles are generally classified according to mission specific properties such as launch platform location, launch platform type, trajectory shape, target location, target type, speed and guidance method. One example classification is given in Table 1-1 [33].

Table 1-1 Classification of Missiles

Platform/Target Location	**Trajectory/ Target Type**	**Example System**
Surface to Surface/ Air to Surface	Ballistic	ATACMS (U.S.A.)
	Cruise	Tomahawk (U.S.A.)
	Anti-ship	Exocet (France)
	Anti-tank	Hellfire (U.S.A.)
Surface to Air	Anti-aircraft	S-75 Dvina (Russia)
	Anti-ballistic	Arrow (Israel)
Air-to-air		AIM-9 Sidewinder (U.S.A.)

From the initial formulation of the concept for a new rocket/missile system to the end of the life cycle, there is a continuous need to predict the performance of alternative designs that satisfy changing operational requirements and to introduce improvements against changing threat. Design engineers use several methods to

obtain information on the performance of alternative rocket/missile configurations. Among these are analytical estimates, computer simulations, laboratory tests, and flight tests. Simple analytical techniques provide estimates of rocket/missile performance characteristics, such as maximum range and flight time, in the very beginning of the design cycle, but they do not allow evaluation of complex interactions of subsystems. The most accurate method is flight testing, but it is also the most costly. Laboratory testing also provides accurate information, but it is restricted mainly to subsystem evaluation. A compromise between these methods is flight simulation. The major system performance measures can be predicted by modeling rocket/missile flight under dynamic and environmental conditions. It allows the mathematical analysis of flight which is complex and involves nonlinearities, logic sequences, singular events, and interactions among multiple subsystems [17].

The various levels of simulations needed for rocket/missile development and the diversity of simulations used for related analyses require a wide range of simulation approaches. In the case of missiles, these approaches differ in terms of the degree of fidelity in simulating the target track sensor and in simulating the transient control and maneuver response of the missile. For example, methods used to simulate target sensors range from the very simple assumption that tracking is performed perfectly to the use of real-time simulations using actual flight hardware seekers that view simulated target scenes that radiate electromagnetic energy. Also methods used to simulate missile motion range from the use of simple two-degree-of-freedom (2-DOF) formulations to very sophisticated six-degree-of-freedom (6-DOF) models. Also a major consideration is whether the simulation should be designed to operate in real time. Certain simulated events, such as control-surface deflections and seeker signal processing, contain high-frequency spectral components that require very small computational time steps to simulate. Because of the small time steps, the time to calculate these events may be considerably longer than the time it would take for them to happen in actual missile flight. When actual missile hardware components are substituted for some of the mathematical equations in the

simulation, it is necessary that the model be constrained to run in real time to mesh with the real-time operation of the hardware. The level of detail to be included in a simulation very much depends on the objectives of the user. Including unnecessary detail increases the chances for subtle program errors that may never be found and that could mask important simulation results, decreases the general visibility of the interactions within the simulation program and complicates the interpretation of simulation results, and increases development time and computer setup and run times. Conversely, omitting detail that is important to the objectives of the user may invalidate the simulation for its intended purpose and lead users to wrong conclusions. [17]

In this thesis, Multiple Cooling Multiobjective Simulated Annealing (MC-MOSA) algorithm [24] is modified and applied to several missile design optimization problems. The aim was to develop a tool which runs together with a simulation algorithm so that it allows for evaluation of many design parameters and outputs the Pareto-optimal solutions and the corresponding design variables. This tool also accepts geometric variables into the optimization loop as well as the other variables, hence eliminates the need for separate runs for different aerodynamic configurations. The functional flow of the tool can be summarized as in Figure 1-1.

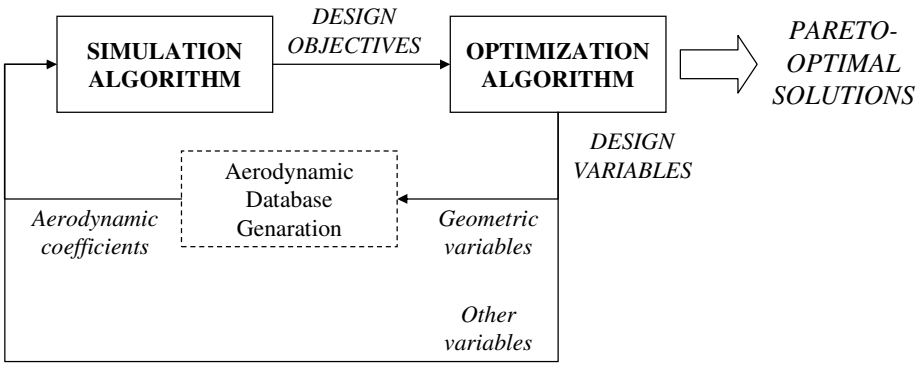


Figure 1-1 Functional Flow of the Optimization Tool

This tool can be used in missile or rocket conceptual design studies, for which assumptions of Section 2.2 apply, to find out the optimum configurations as close to design targets as possible. A surface to surface / air to surface anti-tank missile is modeled for this purpose. The simulation is intended to provide flyout characteristics e.g. range as a function of time for different configurations which can be obtained using a 2-DOF simulation, but not intended to provide more detailed performance metrics such as target coverage diagrams, performance against maneuvering targets, rotational response characteristics or detailed guidance/autopilot/seeker performance, which requires at least 3-DOF. [17] As a result, a modified 2-DOF simulation is developed for the purposes of this thesis. The tool is prepared in FORTRAN programming language.

This thesis report contains 5 chapters. Chapter 2 and 3 give detailed information about the simulation part and optimization part of the modified MC-MOSA algorithm, respectively, Chapter 4 presents the results of the case studies performed, and Chapter 5 finalizes the report with conclusions about this study.

CHAPTER 2

SIMULATION ALGORITHM

As stated in Chapter 1, a 2-DOF simulation algorithm was developed for simulating the flight of a surface to surface/air to surface anti-tank missile. It is actually a "modified" 2-DOF simulation because the missile is not assumed to be purely point mass, geometric dimensions of the missile are also included in the calculations. Following paragraphs give information about configuration of the missile modeled, assumptions used throughout the simulation, functional architecture, description of the algorithm and some sample results.

2.1 Missile Configuration

A baseline missile configuration was determined for simulation studies, the main geometric properties of which are based on AGM-114 Hellfire missile [30, 31, 32]. The general view of this configuration can be seen in Figure 2-1.

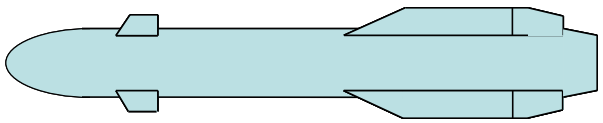


Figure 2-1 Baseline Missile

The general properties of the baseline missile are summarized below (these properties are fixed and was not altered during optimization process, in other words, they are not included in the design variables set):

Launch platform: The missile can be launched from land platforms or helicopters.

Propulsion: The missile has a solid propellant, single stage rocket motor.

Wing configuration: In the front section there are 4 small fixed wings (called "strakes") with X configuration, that is, they make 45° angles with horizontal plane during flight. Since they are located in front of the center of gravity (cg), they increase maneuverability. In the aft section wings and flaps are located, also with X configuration. Wings provide lifting surfaces. Since they are located behind cg, they increase stability. Flaps just behind them are moving surfaces which are used for control.

Control: The missile performs angle of attack control by moving flaps in accordance with angle of attack command sent from missile computer.

The parameters defining the geometry are shown in Figure 2-2, Figure 2-3, Figure 2-4, Figure 2-5 and Figure 2-6. Nose section shown in Figure 2-3 has "ogive" geometry.

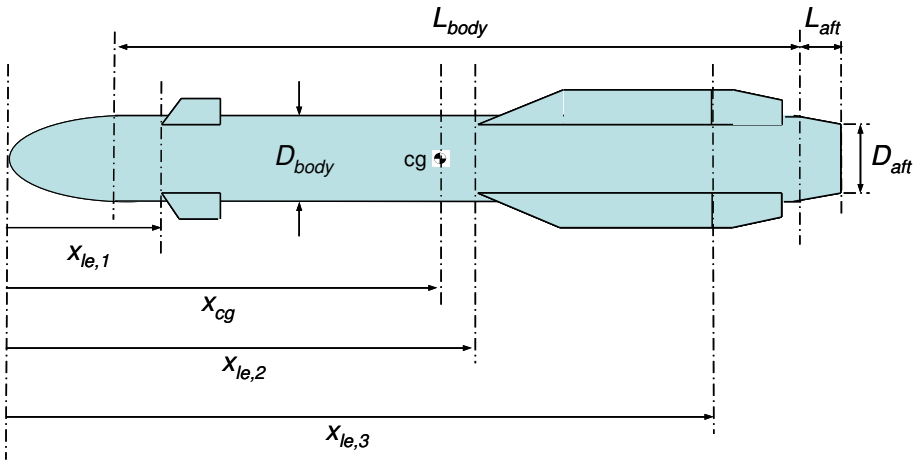


Figure 2-2 Missile Geometry Parameters

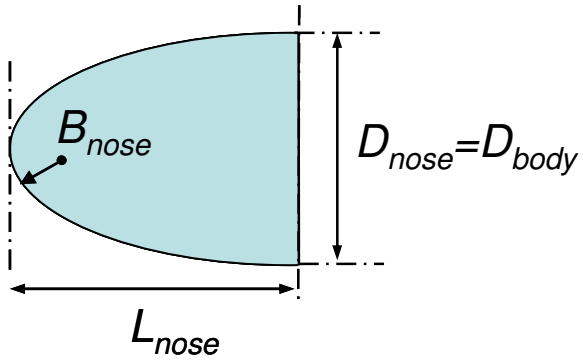


Figure 2-3 Nose Geometry Parameters

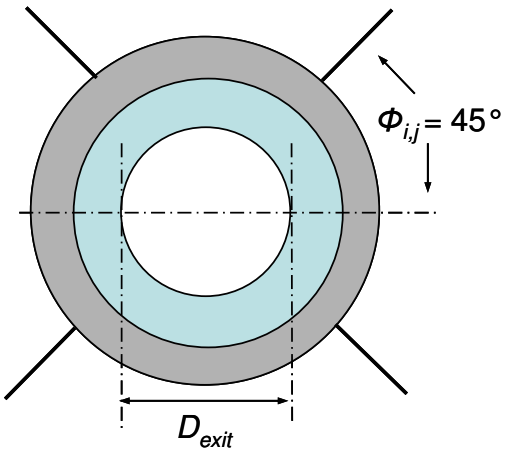


Figure 2-4 Engine Exit and Finset Roll Angle Parameters (looking from behind)

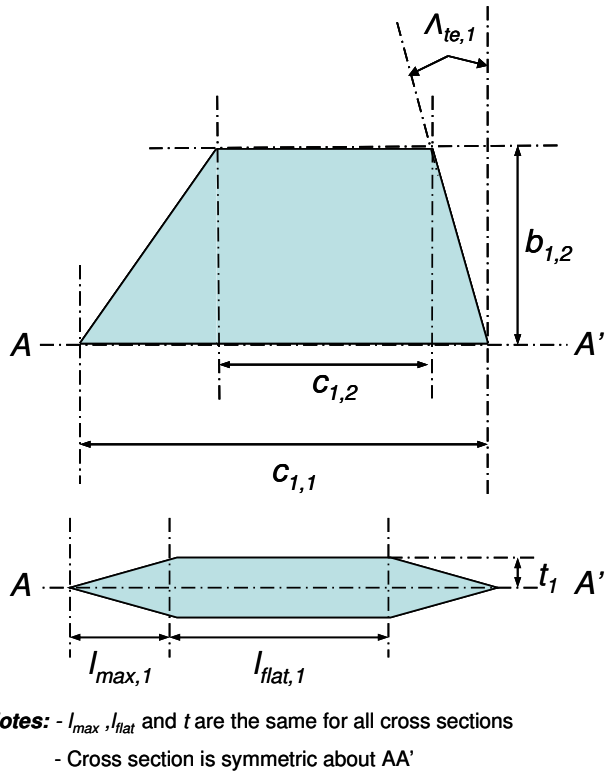
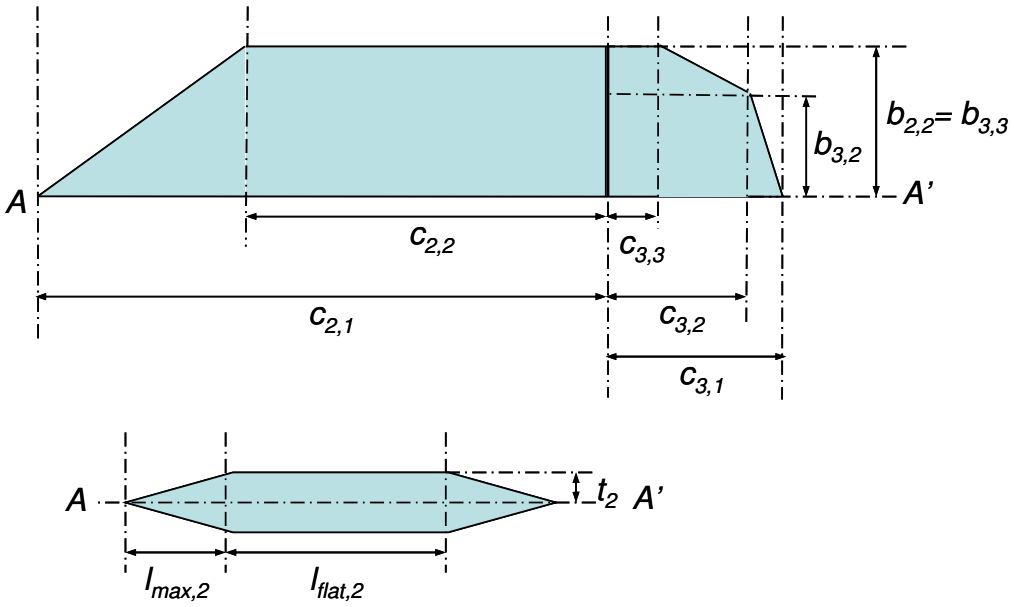


Figure 2-5 Strake Geometry Parameters



Notes: - l_{max} ,l_{flat} and *t* are the same for wings and flaps, and for all cross sections

- Cross sections are symmetric about AA'

Figure 2-6 Wing and Flap Geometry Parameters

2.2 Assumptions

The assumptions used throughout the simulation are as follows:

- The inertial reference frame and convention for the coordinate system assumed is as shown in Figure 2-7.

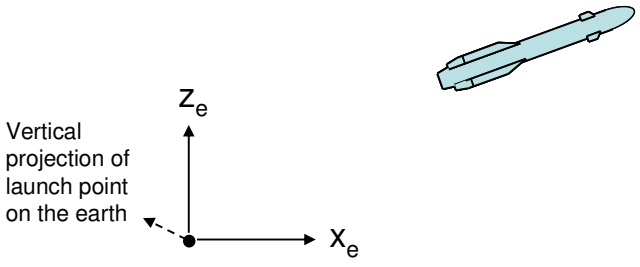


Figure 2-7 Reference Frame

- Flat and non-rotating earth is assumed since the feasible range (~1.5-10 km) and maximum altitude (~500-1000 m) of this baseline missile is relatively low. Hence a non-translating and non-rotating reference frame whose origin is at the projection of launching point on earth surface is used.

- The dynamics of the missile motion in launcher, which is constrained in radial direction, are neglected.

- In case of helicopter launch, the downwash effects on the missile aerodynamics during launch period are neglected.

- A modified 2-DOF simulation is used, that is, the missile flies in x_e-z_e plane as a point mass, with zero lateral force and zero yaw and roll moments and in X configuration. However, angle of attack and cg position are considerations since horizontal and vertical forces on the point mass are calculated accordingly.

- The missile flies always in trim condition, that is, in a condition where pitch moment is always zero. This is done by calculating flap deflection angles required to trim the missile at the current angle of attack command in each time step and assuming that a perfect control system deflects the flaps to these angles instantaneously.

- Response of the missile to the angle of attack command is modeled by a first order transfer function with a certain time constant.

- US Standard Atmosphere 1976 [34] is assumed for atmospheric calculations.

- The changes in aerodynamic coefficients due to altitude are neglected. However, changes in speed of sound and air density due to altitude are taken into account.

- Engine and mass models with assumptions described in Section 2.3.2 are used.

- Mass and cg position change linearly until propellant burnout time. Also, these changes are caused only by propellant burning.
- No effects of thrust misalignments (linear or angular) are included.
- No wind or other atmospheric disturbances are included.

2.3 Algorithm Definition

The simulation algorithm is designed as a subroutine of the modified-MC-MOSA algorithm. Every time this simulation subroutine is called by the main program, the required parameter set is transferred to the main program and a different missile configuration is simulated. The parameters transferred can be seen in Table 2-4. The simulation results (range, flight time, hit angle, hit velocity and aerodynamic coefficients for a specific flight condition) are used as objectives in the case studies of Chapter 4; hence they are transferred to the main program after simulation loop ends. More information about these objectives is given in Chapter 4.

The steps of the algorithm are as follows:

STEP 1 PERFORM INITIAL STEPS

Step 1.1 Calculate aerodynamic coefficients based on the design variables set transferred from main program and generate aerodynamic databases for each flight phase.

Step 1.2 Calculate initial and final values of mass and cg position of the missile.

Step 1.3 Initialize range, altitude, velocity, flight path angle and angle of attack.

Step 1.4 Calculate aerodynamic coefficients corresponding to a specific flight condition (described in Chapter 4).

STEP 2 START SIMULATION LOOP

Step 2.1 Calculate speed of sound, air pressure and air density corresponding to the current altitude.

Step 2.2 Perform interpolations based on current flight phase

- Interpolate for mass and cg position using initial and final values.
- Interpolate for aerodynamic coefficients using current Mach number, angle of attack and cg position.

Step 2.3 Calculate angle of attack command.

Step 2.4 Calculate thrust force, aerodynamic forces and total forces along stability axes.

Step 2.5 Integrate range, altitude, velocity, flight path angle and angle of attack.

Step 2.6 Calculate the current total energy.

Step 2.7 Go to **Step 2.1** until altitude reaches zero.

STEP 2′ END SIMULATION LOOP

STEP 3 Transfer the objective values (range, flight time, hit angle, hit velocity and aerodynamic coefficients for a specific flight condition) to the main program.

STEP 4 Return to the main program.

Some detailed explanations of these steps are explained in the following paragraphs.

2.3.1 Calculation of Aerodynamic Databases

Lift and drag coefficients for trim ($C_{L,trim}$ and $C_{D,trim}$) are assumed to be functions of angle of attack, Mach number and missile geometry. They are calculated using Missile DATCOM prediction tool, using *TRIM* control card. This tool accepts an input set which contains an angle of attack vector, a Mach vector and parameters defining the whole geometry of the missile. The parameters in the input set are listed in Table 2-1. Definitions of these parameters are given in Section 2.1.

Table 2-1 Missile DATCOM Input Parameter Set

B_{nose}	$l_{max,i} / c_{i,j}$	$x_{le,i}$	$n_{panel,i}$
D_{nose}	$l_{flat,i} / c_{i,j}$	$t_i / c_{i,j}$	δ_{lim}
L_{aft}	$b_{i,j}$	$\Lambda_{te,i}$	***ALPHA*** vector
D_{aft}	$c_{i,j}$	$\Phi_{i,j}$	***MACH*** vector

Every time the simulation subroutine is called, a new DATCOM input set based on the new parameter set transferred from main program is generated, DATCOM is run and the resulting $C_{L,trim}$ and $C_{D,trim}$ values are stored in arrays as a database. This database tabulates $C_{L,trim}$ and $C_{D,trim}$ values corresponding to each pair of elements of angle of attack and Mach number vectors. The database contains three parts:

- First one belongs to before-burnout phase and based on initial cg position,
- Second one belongs to before-burnout phase and based on final cg position,
- Third one belongs to after-burnout phase and based on final cg position,

This partition is necessary because of two reasons: First, DATCOM computes $C_{D,trim}$ by taking base-jet plume interaction effects into account, and these effects are implemented into calculations by a corresponding input file parameter. This parameter is changed to zero for the case of after-burnout indicating no plume effects. Second, before-burnout database is needed to be further divided into two, differing by cg position inputs since cg position affects $C_{L,trim}$ and $C_{D,trim}$. This database generation is performed once before simulation loop starts, and resulting database is used in interpolation described in 2.3.4. The angle of attack and Mach number vectors are fixed (not taken as optimization parameters) and given in (2.1) and (2.2).

$$ALPHA = [\ 0, 1, 2, 3, 4\ ] \quad \text{(deg)} \tag{2.1}$$

$$MACH = [\ 0.4, 0.5, 0.6, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95, 1.05, 1.1, 1.15, 1.2, 1.3, 1.4, 1.5, 1.6\ ] \tag{2.2}$$

The bounds of these vectors had to be kept tight because of unexpected results resulting from DATCOM's internal calculation methods when these bounds are exceeded. If some angle of attack and Mach values beyond these bounds occur in the simulation loop, the algorithm extrapolates for $C_{L,trim}$ and $C_{D,trim}$.

Graphical demonstrations of sample aerodynamic databases using the geometric parameters given in Table 2-4 are shown in Figure 2-8, Figure 2-9, Figure 2-10, Figure 2-11, Figure 2-12 and Figure 2-13.

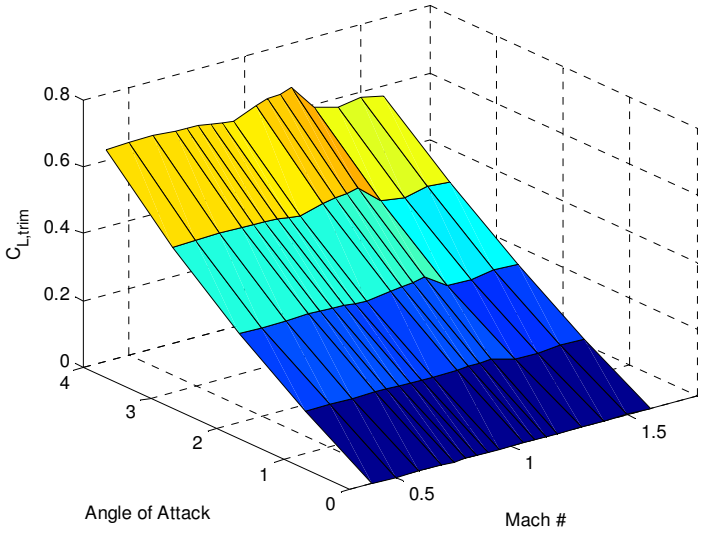


Figure 2-8 Trim Lift Coefficient vs. [Angle of Attack, Mach Number] (before burnout , $x_{cg} = x_{cg,0}$)

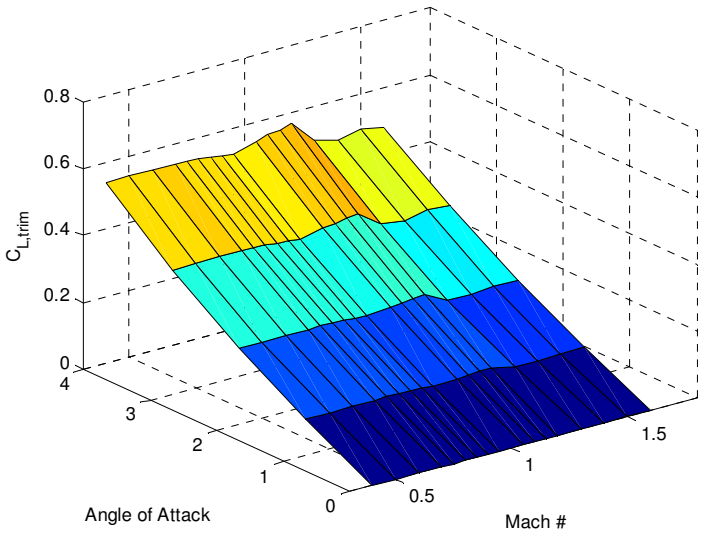


Figure 2-9 Trim Lift Coefficient vs. [Angle of Attack, Mach Number] (before burnout , $x_{cg} = x_{cg,final}$)

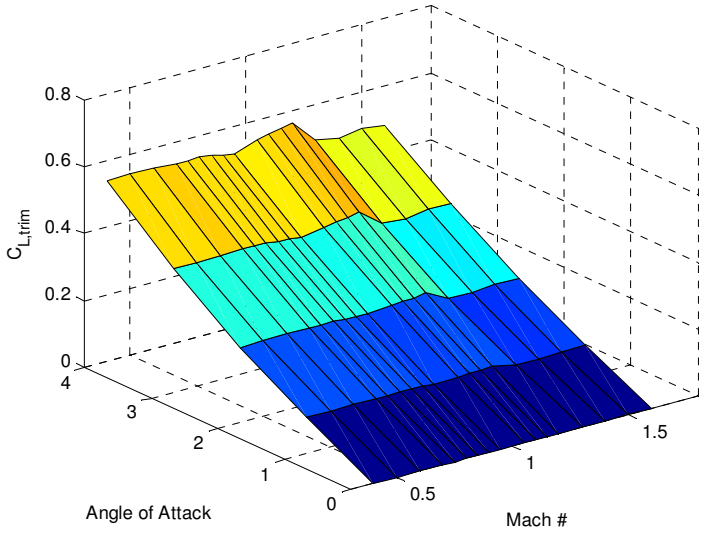


Figure 2-10 Trim Lift Coefficient vs. [Angle of Attack, Mach Number] (after burnout , $x_{cg} = x_{cg,final}$)

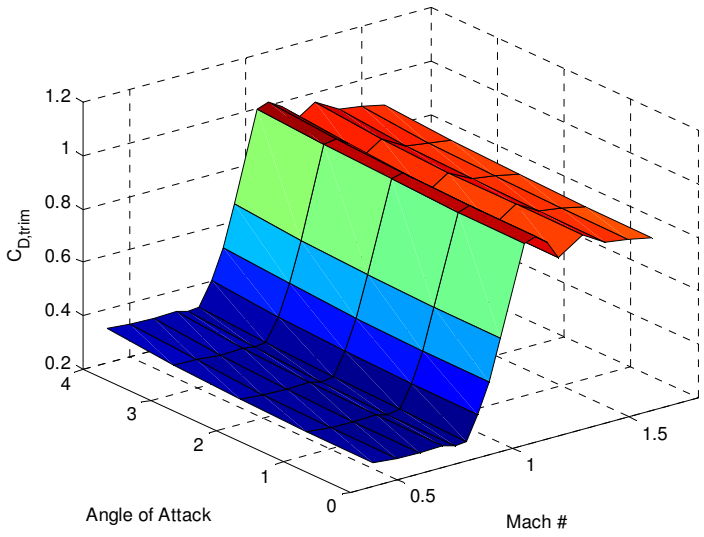


Figure 2-11 Trim Drag Coefficient vs. [Angle of Attack, Mach Number] (before burnout , $x_{cg} = x_{cg,0}$)

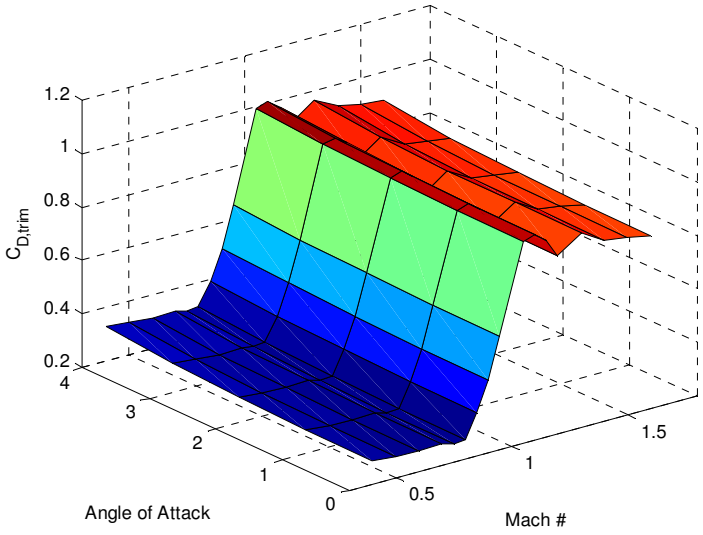


Figure 2-12 Trim Drag Coefficient vs. [Angle of Attack, Mach Number] (before burnout , $x_{cg} = x_{cg,final}$)

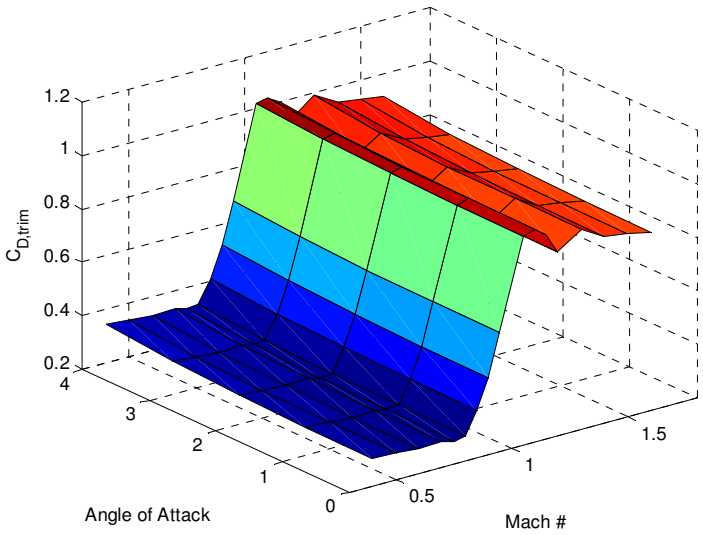


Figure 2-13 Trim Drag Coefficient vs. [Angle of Attack, Mach Number] (after burnout , $x_{cg} = x_{cg,final}$)

Some interpretations of these figures are as follows:

- All the figures show that $C_{L,trim}$ changes with angle of attack almost linearly and $C_{D,trim}$ changes very little, ceteris paribus. These are because angle of attack range is relatively tight and close to zero (linear region).

- From Figure 2-8 and Figure 2-9 it is seen that as cg shifts forward, slopes of $C_{L,trim}$ - α lines decrease. As stated in Section 2.1, wings and control surfaces are located towards the aft of the missile and initial cg position is near the middle. Since strake surface areas are smaller in magnitude, the center of pressure (cp), on which total aerodynamic forces acts, is behind cg. A shift-forward in cg elongates the moment arm of total lift force assuming that cp does not change. This means that the flap deflections required to trim the missile at a specific angle of attack and Mach number become smaller, which in turn decreases $C_{L,trim}$. This can also be proven analytically by writing moment equations for the two cg positions shown in Figure 2-14, using $\Delta L>0$ and assuming small angle of attack.

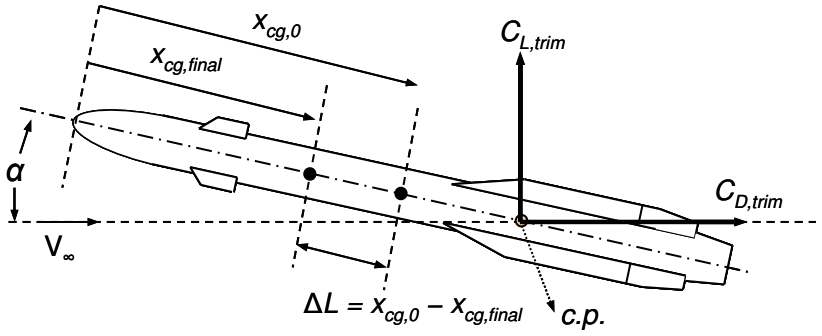


Figure 2-14 Effect of cg Shift on Trim Aerodynamic Coefficients

- Figure 2-11 and Figure 2-12 show that $C_{D,trim}$ remains almost unchanged as cg shifts forward, ceteris paribus. This is expected because of small angles of attack and can also be proven by the moment equations of Figure 2-14.

- Figure 2-9 and Figure 2-10 show that $C_{L,trim}$ values before and after burnout are almost the same, ceteris paribus. As stated previously in this section, it is base drag calculations that differ in these phases, and they have seemingly no contribution to $C_{L,trim}$.

- Figure 2-12 and Figure 2-13 show that $C_{D,trim}$ surface slightly shifts up as burnout is completed. This is because as base-plume interaction effects disappear, a full base drag instead of a partial one is induced at the aft, adding slightly to $C_{D,trim}$.

These aerodynamic databases are generated assuming sea level density in DATCOM. Figure 2-15 and Figure 2-16 demonstrates the results of a DATCOM run showing the changes of $C_{L,trim}$ - M and $C_{D,trim}$ – M curves with altitude for different a.o.a. values. For $C_{D,trim}$ (Figure 2-16) these curves actually close to each other, hence not to deteriorate the visualization of altitude errors, only one curve is shown in this graph. Actually it corresponds to the maximum altitude error. Then, taking into account that maximum altitude of the baseline missile of this study is ~500-1000 m, the errors due to altitude are very small as seen in this two figures, hence the assumption is justified.

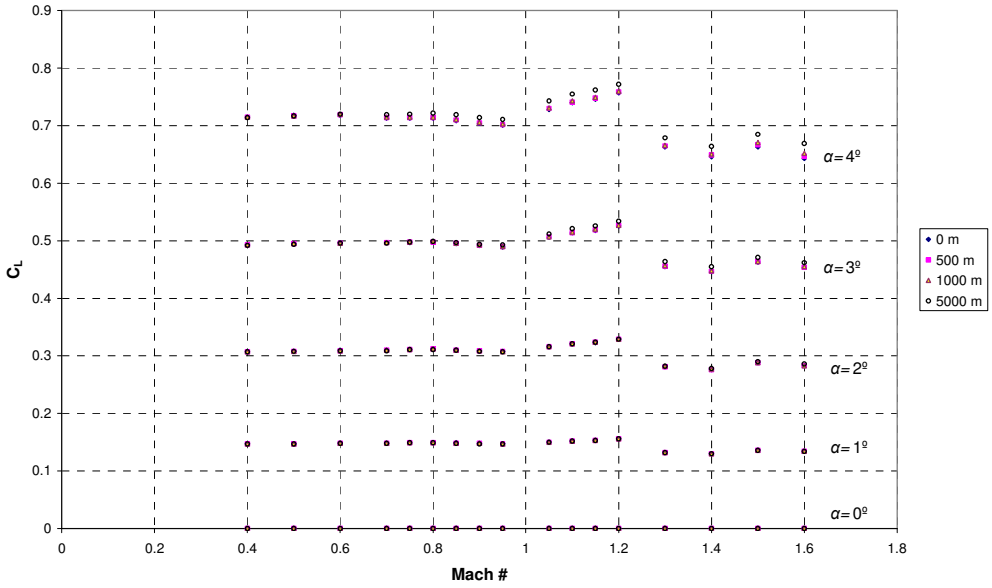


Figure 2-15 Change of $C_{L,trim}$ with Altitude

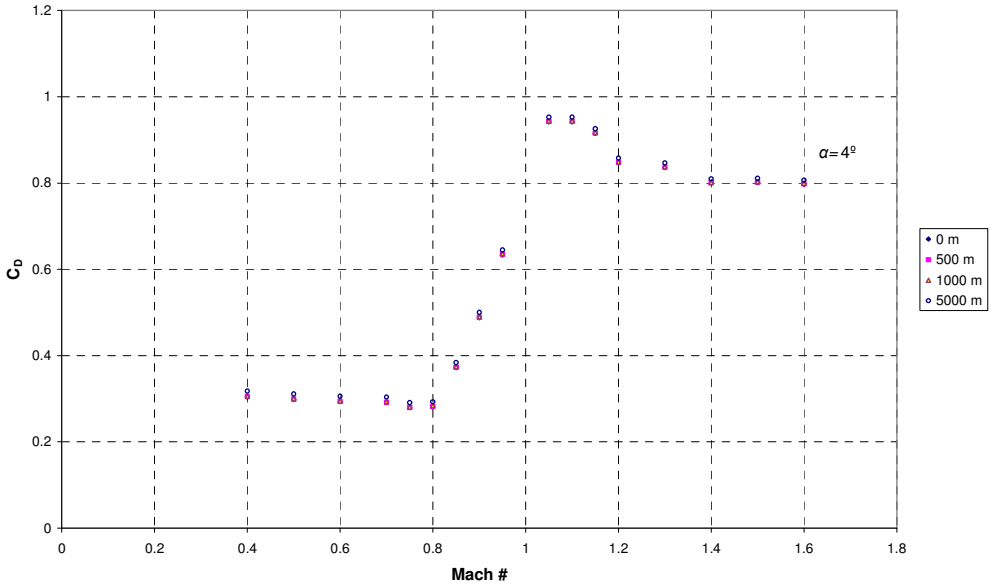


Figure 2-16 Change of $C_{D,trim}$ with Altitude

The effects of base drag do not change $C_{L,trim}$, hence second and third $C_{L,trim}$ databases are the same. Moreover, it is observed from the data of Figure 2-12 and Figure 2-13 that the maximum increase in $C_{D,trim}$ is 1.10 times and the average increase is 1.06 times. Since these are relatively small differences, to improve the algorithm run time, no separate DATCOM run is performed to obtain the third database. Instead, the $C_{D,trim}$ elements of second database is multiplied by the average increase (1.06) to get the third database. This greatly improves the total run time of the optimization algorithm since DATCOM spends more than 90 percent of the run time of one optimization loop.

2.3.2 Propulsion and Mass Models

Propulsion system and warhead are the sections responsible for most of the mass of a typical rocket or missile. Propulsion system is modeled as a single stage, solid propellant and neutral burning (constant thrust) rocket motor. Total impulse, which is a measure of momentum released by burning of propellant and defined as the total area under thrust-time curve, dictates the mass and geometry of the propellant. In this study, the design thrust and burning time are selected as design variables.

The configuration of rocket motor can be seen in Figure 2-17.

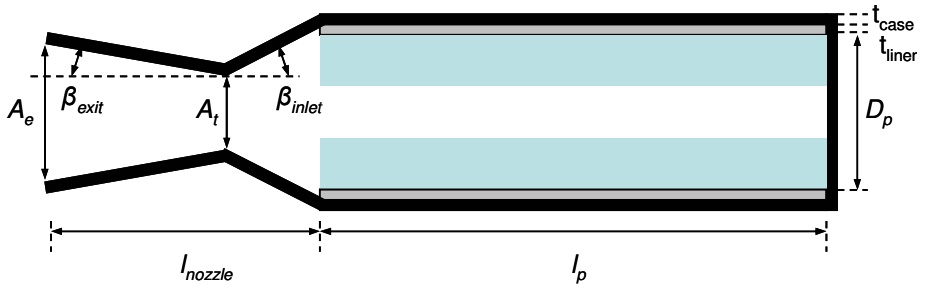


Figure 2-17 Rocket Motor Parameters

Various constants related to propulsion and mass models are selected and used in the calculations described throughout this section. The values and references for these constants are given in Table 2-2.

Table 2-2 Constants Used in Propulsion and Mass Models

	Value	**Unit**	**Reference**
Propellant	CMDB+HMX	**-**	[22]
r_{ref}	0.02	m/s	[22]
P_{ref}	6.89×10^6	Pa	[22]
n	0.49	-	[22]
I_{sp}	220	s	[22]
ρ_p	1800	kg/m^3	[22]
C_F	1.3	-	[22]
k	1.25	-	[3]
k_{vl}	0.7	-	[3]
σ_{case}	4.35×10^8	Pa	Al7075-T73
P_e	101325	Pa	-
t_{liner}	0.002	m	-
D_{body}	0.178	m	-
DPR	10	-	-
β_{inlet}	45	deg	[23]
β_{exit}	15	deg	[23]
ρ_{case}	2823	kg/m^3	Al7075-T73
ρ_{liner}	860	kg/m^3	EPDM
ρ_{nozzle}	1570	kg/m^3	Cellulose phenolic
$\rho_{finsets}$	2700	kg/m^3	Al
m_{con}	3.5	kg	-
m_{wh}	10	kg	[30, 31, 32]
m_{other}	17.6	kg	[30, 31, 32]
l_{wh}	0.6	m	[30, 31, 32]
l_{other}	0.57	m	[30, 31, 32]

For a single stage neutral burning rocket motor, total impulse and mass of the propellant is given by [3]

$$I_{tot} = T_d \cdot t_b,\ m_p = \frac{I_{tot}}{I_{sp} g} \tag{2.3}$$

To have a constant thrust, a star-shaped cross section is chosen for propellant grain. Figure 2-18 shows the successive burning contours, which demonstrates the growth of internal cavity [22]. The lengths of these contour lines are roughly the same, which means that burning area is roughly constant.

Figure 2-18 Successive Burning Contours of Star-Shaped Propellant Grain

For the sake of simplicity, the outer diameter of the propellant grain is assumed equal to missile diameter, since case thickness that will be calculated below and liner thickness assumed in Table 2-2 are small compared to missile diameter. Length of the propellant grain can be calculated using the volume formula for the grain configuration and calculated propellant mass [3]

$$l_p = \frac{4 m_p}{\pi D_p^2 \rho_p k_{vl}} \tag{2.4}$$

Here, k_{vl} is the volumetric loading, which is defined as the ratio of propellant volume to combustion chamber volume. It depends on the geometry of star-shaped cavity. A detailed design of this shape is beyond the scope of the context and k_{vl} is assumed as 0.7 in this study.

Assuming a neutral burning means that area of the burning surface remains constant during propellant burning. At the end of burning, the area is assumed equal to the area of propellant outer surface. Hence, the area of burning surface at all instants of burning is given by

$$A_b = \pi D_p l_p \tag{2.5}$$

Assuming that combustion product is calorically perfect, expansion in the nozzle is isentropic, and maximum flow rate through the nozzle is achieved, propellant mass can also be calculated by the following expression [23]

$$m_p = A_b \rho_p r_{ref} \left(\frac{P_c}{P_{ref}} \right)^n t_b \tag{2.6}$$

The combustion chamber pressure P_c can be found from this equation by substituting the other known variables. Required case thickness to endure this pressure can be calculated using hoop stress formula, [23]

$$t_{case} = \frac{P_c D_{body} DPR}{2\sigma_{case}} \tag{2.7}$$

Nozzle throat and exit areas can be found from [3]

$$A_t = \frac{T_d}{C_F P_c} \tag{2.8}$$

$$\frac{A_t}{A_e} = \left(\frac{k+1}{2} \right)^{\frac{1}{k-1}} \left(\frac{P_e}{P_c} \right)^{\frac{1}{k}} \sqrt{\frac{k+1}{k-1} \left(1 - \left(\frac{P_e}{P_c} \right)^{\frac{k-1}{k}} \right)} \tag{2.9}$$

Nozzle exit diameter D_{exit}, which will be an input for aerodynamic calculations, can be found from A_e. The lengths of convergent (inlet) and divergent (exit) sections of nozzle can be found using the geometry shown in Figure 2-17,

$$l_{inlet} = \left(\frac{D_p}{2} - r_t \right) \cot \beta_{inlet} \tag{2.10}$$

$$l_{exit} = \left(r_e - r_t \right) \cot \beta_{exit} \tag{2.11}$$

Then, the total lengths of nozzle and motor are,

$$l_{nozzle} = l_{inlet} + l_{exit} \tag{2.12}$$

$$l_{motor} = l_p + l_{nozzle} \tag{2.13}$$

Thus, the whole geometry of rocket motor shown in Figure 2-17 has been defined. By calculating the volumes of individual components (propellant, case, liner and nozzle) and using the assumed material density values given in Table 2-2, the masses of each component and hence the total mass of rocket motor can be found,

$$m_{motor} = m_p + m_{nozzle} + m_{case} + m_{liner} \tag{2.14}$$

To calculate the masses of finset elements (strakes, wings and flaps), the volumes should be calculated. All the elements are solid hexagonal prismatoids, they can be assumed to be made of aluminum and masses can be calculated accordingly.

Thus, the total initial mass and length of the missile can be found using the assumed values of Table 2-2,

$$m_0 = m_{motor} + m_{wh} + m_{con} + m_{finsets} + m_{other} \tag{2.15}$$

$$l_{tot} = l_{motor} + l_{wh} + l_{other} \tag{2.16}$$

The mass of missile after burnout is given by

$$m_{fin} = m_0 - m_p \tag{2.17}$$

The centers of mass of control, nozzle and propellant sections are assumed to be on the middle of each section. The centers of mass of warhead and section in front of warhead are assumed to be on 2/3 towards the aft of each section. The centers of mass of finset elements are assumed to be on the middle of average horizontal lengths of trapezoidal planform areas, to simplify the calculation. Then; summing the moments about the foremost point of missile nose, the initial and final cg positions of the missile can be calculated.

2.3.3 Atmospheric Calculations

Simulation loop starts with calculations of atmospheric parameters. Speed of sound, air density and pressure are assumed to be the functions of altitude only, and in each loop they are calculated. An atmospheric model based on U.S. Standard Atmosphere, 1976 [34] is employed.

After that, Mach number is calculated by

$$M = \frac{V}{a_{sound}} \tag{2.18}$$

2.3.4 Interpolations

To find the corresponding $C_{L,trim}$ and $C_{D,trim}$ values at the current Mach number, angle of attack and cg position at a time step of the simulation loop, trilinear and bilinear interpolations are used depending on the current flight phase. Angle of attack and Mach number vectors used are defined by (2.1) and (2.2).

- In before-burnout phase, a trilinear interpolation given in (2.19) is carried out to find the corresponding $C_{L,trim}$ and $C_{D,trim}$. Aerodynamic databases corresponding to initial cg and final cg are used as interpolation arrays.

- In after-burnout phase, a bilinear interpolation given in (2.19) is carried out to find the corresponding $C_{L,trim}$ and $C_{D,trim}$. Aerodynamic database corresponding to final cg is used as interpolation array.

$$C_{L,\,trim} = \begin{cases} f_{int3}\,(\mathbf{C_{L,1}},\ \mathbf{C_{L,2}},\ M,\ \alpha,\ x_{cg})\,, & t \le t_b \\ f_{int2}\,(\mathbf{C_{L,3}},\ M,\ \alpha)\,, & t > t_b \end{cases} \tag{2.19}$$

$$C_{D,trim} = \begin{cases} f_{int3}\,(\mathbf{C_{D,1}},\ \mathbf{C_{D,2}}\ M,\ \alpha,\ x_{cg})\,, & t \le t_b \\ f_{int2}\,(\mathbf{C_{D,3}},\ M,\ \alpha)\,, & t > t_b \end{cases}$$

In a similar fashion, current mass and cg position are calculated hereafter, performing a linear interpolation given by

$$m = \begin{cases} f_{int}\,(\,[t_0\ \ t_b\,;\ m_0\ \ m_{final}]\,,\,t\,)\,, & t \le t_b \\ m_{final}\,, & t > t_b \end{cases} \tag{2.20}$$

$$x_{cg} = \begin{cases} f_{int}\,(\,[t_0\ \ t_b\,;\ x_{cg,0}\ \ x_{cg,final}]\,,\,t\,)\,, & t \le t_b \\ x_{cg,final}\,, & t > t_b \end{cases} \tag{2.21}$$

2.3.5 Calculation of Angle of Attack Command

An angle of attack command profile is determined using nodes equally spaced in time until burnout and equally spaced in energy consumption after burnout. This approach has been shown to be superior to using all nodes equally spaced in time [23]. An additional hit phase is added to the end of the profile, which is initiated when missile reaches to a pre-determined distance to the ground. In this phase, a negative angle of attack command is produced to hit the target with a steep hit angle. A sample profile is given in Figure 2-19.

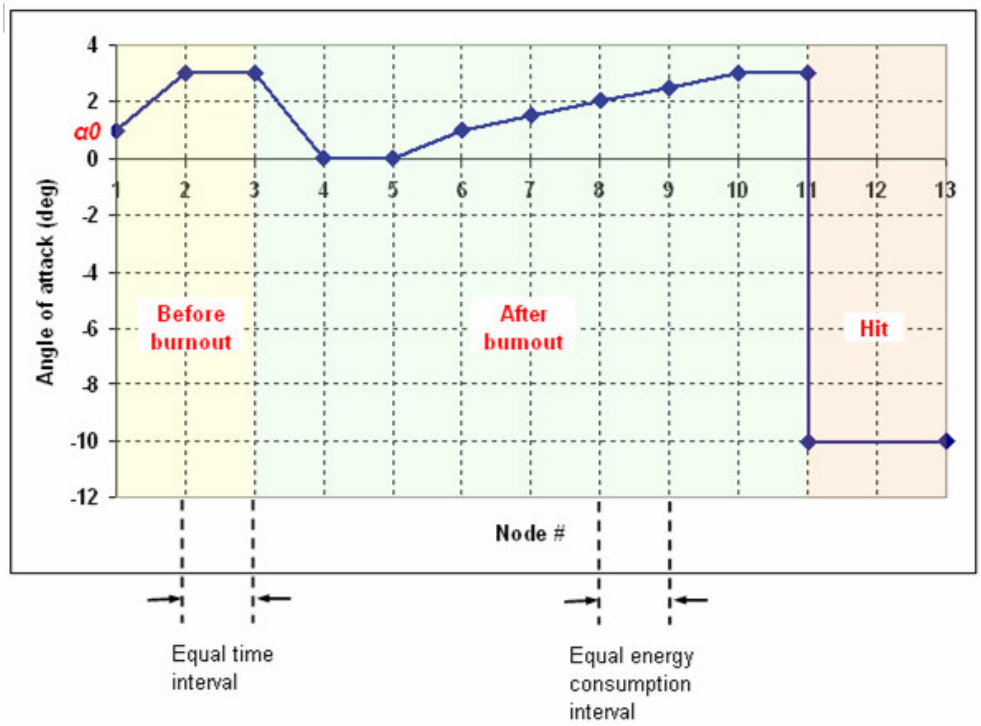


Figure 2-19 Sample Angle of Attack Command Profile

The angle of attack values in each node are actually optimization variables and they are given some upper and lower bounds in optimization algorithm. The values shown in Figure 2-19 are samples. The first node is the initial angle of attack given to the missile, which is also an optimization variable. The angle of attack command for each simulation loop is found by linear interpolation with time until burnout and with consumed total energy after burnout. Hit phase angle of attack command is a constant negative value, as stated above. The angle of attack command is calculated by

$$\alpha_{com} = \begin{cases} f_{int}\left(\boldsymbol{\alpha}_{\mathbf{node,1}}, t\right), & t \le t_b \\ f_{int}\left(\boldsymbol{\alpha}_{\mathbf{node,2}}, E_{max} - E_{tot}\right), & t > t_b \text{ and } z \ge z_{hit} \\ \alpha_{hit}, & t > t_b \text{ and } z < z_{hit} \end{cases} \tag{2.22}$$

After-burnout phase is the region where nodes are placed in equal energy consumption intervals. The beginning point is where the missile has its maximum energy, which is the burnout point, and a portion of this energy will have been consumed at the end of flight. The total energy of the missile at any instant is calculated by

$$E_{tot} = m \cdot g \cdot z + \frac{1}{2} \cdot m \cdot V^2 \tag{2.23}$$

Specifically, the maximum total energy (E_{max}) can be found by inserting the burnout mass, corresponding altitude and speed at burnout, into (2.23).

Before-burnout phase can be divided into equal time intervals since the burnout time is a known parameter before simulation starts. The total energy at burnout is also calculated when after-burnout phase is initiated. However, the total energy at the end of after-burnout phase is unknown at that instant, hence a known parameter at that node is needed to divide after-burnout phase into equal energy consumption intervals. A parameter e_{per} is introduced, which is the predicted percentage of the maximum energy consumed in after-burnout phase. This parameter is chosen as an optimization variable. Then the missile energy at the end of after-burnout phase becomes a known parameter and equals E_{max} .(1- e_{per}) . The node at that point is shown in Figure 2-19 as coincident with hit phase initiation point, but in general this is not the case. To find the angle of attack command at that point, the algorithm either interpolates using the value of last node or extrapolates beyond, which is determined by e_{per} and z_{hit}.

2.3.6 Calculation of Forces and Derivatives

Figure 2-20 shows the forces on the missile at any instant of flight, which are thrust (T), gravitational force (W), lift (L) and drag (D). The angles α, γ and θ are angle of attack, flight path angle and pitch angle, respectively. $[x_s,z_s]$, $[x_b,z_b]$ and $[x_E,z_E]$ defines stability, body and earth axes, respectively. **V** is the total velocity vector [1].

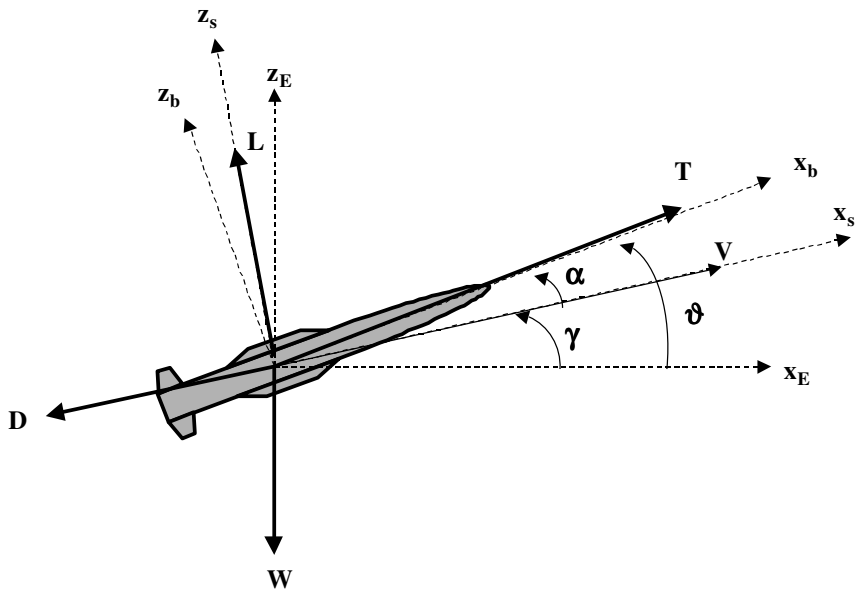


Figure 2-20 Forces and Angles on Body, Stability and Earth Axes

Thrust force varies with altitude and this change is given by the following thrust equation [23], in which P_e - P_a stands for the pressure difference between nozzle design exit pressure and ambient pressure.

$$T = T_d + (P_e - P_a)A_e \tag{2.24.}$$

Resultant forces in stability axes are given by

$$X_S = T \cdot \cos(\alpha) - D - W \cdot \sin(\gamma) \tag{2.25}$$

$$Z_S = T \cdot \sin(\alpha) + L - W \cdot \cos(\gamma) \tag{2.26}$$

where

$$L = \frac{1}{2} \cdot \rho_{air} \cdot V^2 \cdot S_{ref} \cdot C_{L,trim} \tag{2.27}$$

$$D = \frac{1}{2} \cdot \rho_{air} \cdot V^2 \cdot S_{ref} \cdot C_{D,trim} \tag{2.28}$$

S_{ref} is the reference area which is taken as the maximum cross sectional are of the missile body. $C_{L,trim}$ and $C_{D,trim}$ are lift and drag coefficients for trim described in Section 2.3.4.

To find the velocity and flight path angle derivatives, tangential and normal acceleration components will be used [1]. They are shown in Figure 2-21 and given by

$$a_T = \frac{X_S}{m} \tag{2.29}$$

$$a_N = \frac{Z_S}{m} \tag{2.30}$$

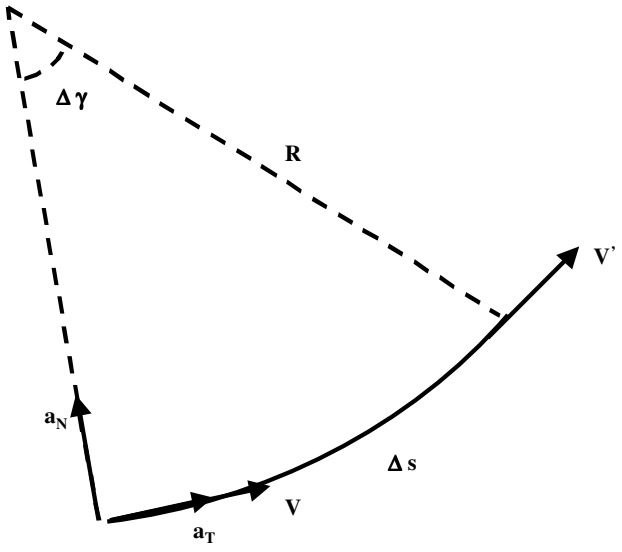


Figure 2-21 Tangential and Normal Acceleration

where

Δs : Infinitesimal displacement along the missile flight path

$\Delta \gamma$: Infinitesimal change in flight path angle

R : Instantaneous radius of curvature of flight path

Acceleration components can also be written as

$$a_T = \frac{\partial V}{\partial t} = \dot{V} \tag{2.31}$$

$$a_N = \frac{V^2}{R} \tag{2.32}$$

From (2.29) and (2.31), velocity derivative can be obtained as

$$\dot{V} = \frac{X_s}{m} \tag{2.33}$$

Figure 2-21 indicates that

$$\Delta\gamma = \frac{\Delta s}{R} \tag{2.34}$$

$$\Delta s = V \cdot \Delta t \tag{2.35}$$

From (2.34) and (2.35) it can be shown that

$$\lim_{\Delta t \to 0} \frac{\Delta\gamma}{\Delta t} = \frac{\partial\gamma}{\partial t} = \dot{\gamma} = \frac{V}{R} \tag{2.36}$$

Using (2.30), (2.32) and (2.36), flight path angle derivative can be obtained as

$$\dot{\gamma} = \frac{Z_s}{m \cdot V} \tag{2.37}$$

Besides, from Figure 2-20 it is seen that

$$\frac{\partial x_E}{\partial t} = V \cdot \cos(\gamma) = \dot{x}_E \tag{2.38}$$

$$\frac{\partial z_E}{\partial t} = V \cdot \sin(\gamma) = \dot{z}_E \tag{2.39}$$

(2.33), (2.37), (2.38) and (2.39) are the equations defining the motion of the missile and are integrated to obtain the instantaneous velocity, flight path angle and earth coordinates of the missile.

$$V_{i+1} = V_i + \dot{V} \cdot \Delta t \tag{2.40}$$

$$\gamma_{i+1} = \gamma_i + \dot{\gamma} \cdot \Delta t \tag{2.41}$$

$$x_{i+1} = x_i + \dot{x} \cdot \Delta t \tag{2.42}$$

$$z_{i+1} = z_i + \dot{z} \cdot \Delta t \tag{2.43}$$

To find the angle of attack derivative, a first order transfer function given by the following is assumed.

$$\frac{\alpha(s)}{\alpha_{com}(s)} = \frac{1}{\tau \cdot s + 1} \tag{2.44}$$

Taking the inverse Laplace,

$$\dot{\alpha} = \frac{1}{\tau}(\alpha_{com} - \alpha) \tag{2.45}$$

Then, the angle of attack in each loop is found by integrating this equation

$$\alpha_{i+1} = \alpha_i + \dot{\alpha} \cdot \Delta t \tag{2.46}$$

Finally, from Figure 2-20, the pitch angle θ is found by

$$\theta_{i+1} = \gamma_{i+1} + \alpha_{i+1} \tag{2.47}$$

Since this is a 2-DOF simulation with trim flight, pitch angle in each loop corresponds to the case where pitch moment is always zero.

The simulation loop is executed until missile hits the ground, that is, z becomes less than zero. The results (objectives) are returned to the main program thereafter.

2.4 Sample Results

In this section, the results of a simulation run with a sample design variable set are presented. In addition to the constants given in Table 2-2, parameters given in Table 2-3 are also taken as constants. They are also employed in the case studies of Chapter 4. The design variable set used in this sample simulation run is given in Table 2-4.

Table 2-3 Constants Used in Simulation

	Value	**Unit**
ALPHA	*see Eq. (2.1)*	deg
MACH	*see Eq. (2.2)*	-
D_{nose}	17.8	cm
D_{aft}	16	cm
$n_{panel,1}$	4	-
$\Phi_{1,j}$	45	deg
$\Lambda_{te,2}$	0	deg
$n_{panel,2}$	4	-
$\Phi_{2,j}$	45	deg
$\Lambda_{le,3}$	0	deg
$n_{panel,3}$	4	-
$\Phi_{3,j}$	45	deg
$x_{le,3}$	$x_{le,2} + c_{2,1}$	cm
x_0	0	m
t_0	0	s
$node\#_1$	2	-
$node\#_2$	8	-

Table 2-4 Design Variable Set of Sample Simulation Run

	Value	Unit
t_b	3	s
T_d	7000	N
B_{nose}	7	cm
L_{aft}	5	cm
$b_{1,2}$	5	cm
l_{max1} / c	0.2	-
$c_{1,1}$	10	cm
$c_{1,2}$	7	cm
t_1 / c	0.02	-
$\Lambda_{te,1}$	0	deg
$x_{le,1}$	30	cm
$b_{2,2}$	8	cm
l_{max2} / c	0.2	-
$c_{2,1}$	50	cm
$c_{2,2}$	35	cm
t_2 / c	0.02	-
$x_{le,2}$	110	cm
$b_{3,2}$	6	cm
$c_{3,1}$	8	cm
$c_{3,2}$	8	cm

	Value	Unit
$c_{3,3}$	5	cm
δ_{lim}	25	deg
z_0	10	m
V_0	10	m/s
γ_0	5	deg
α_0	2	deg
τ	0.3	s
e_{per}	0.9	-
α_{hit}	-15	deg
z_{hit}	100	m
$\alpha_{node,2}$	3	deg
$\alpha_{node,3}$	3	deg
$\alpha_{node,4}$	0	deg
$\alpha_{node,5}$	0	deg
$\alpha_{node,6}$	1	deg
$\alpha_{node,7}$	1.5	deg
$\alpha_{node,8}$	2	deg
$\alpha_{node,9}$	2.5	deg
$\alpha_{node,10}$	3	deg
$\alpha_{node,11}$	3	deg

The following figures show the results of the simulation run with parameter set given in Table 2-4.

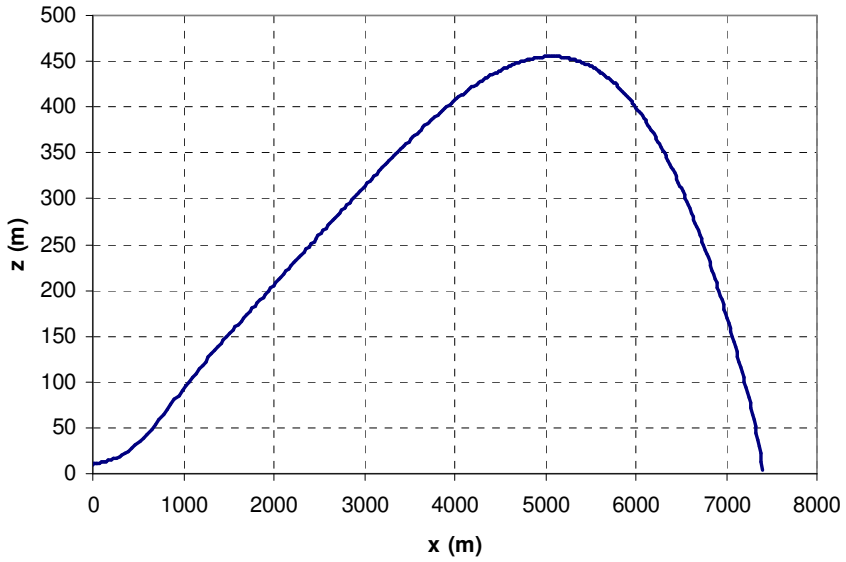


Figure 2-22 Altitude vs. Range

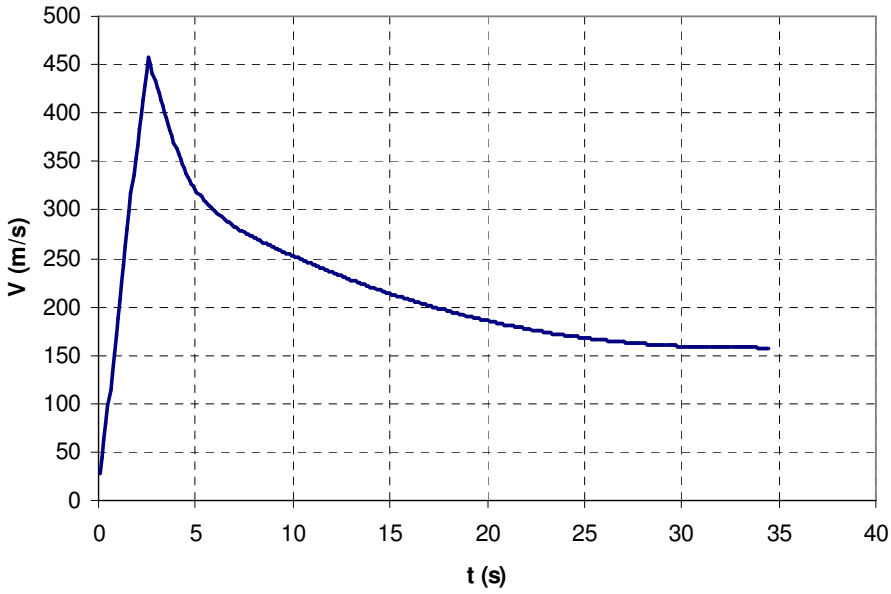


Figure 2-23 Speed vs. Time

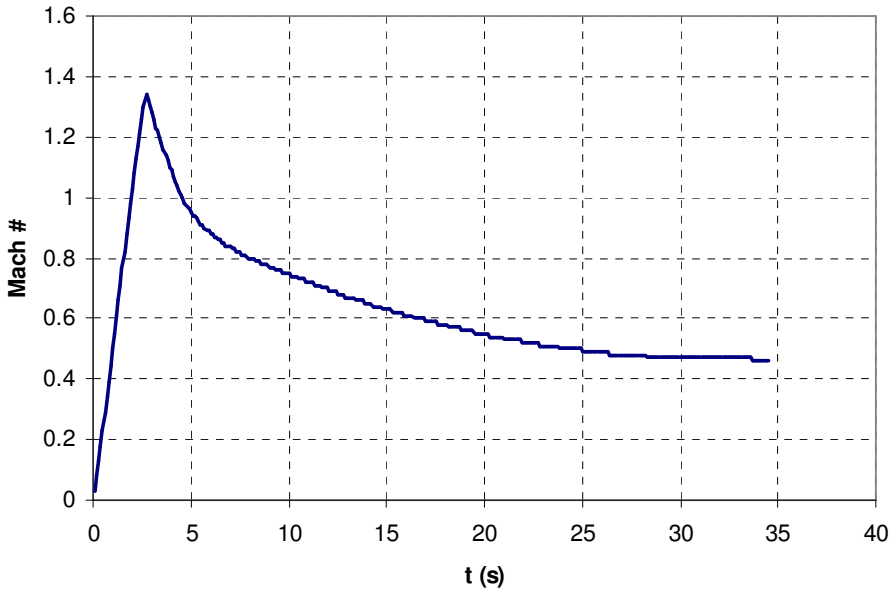


Figure 2-24 Mach Number vs. Time

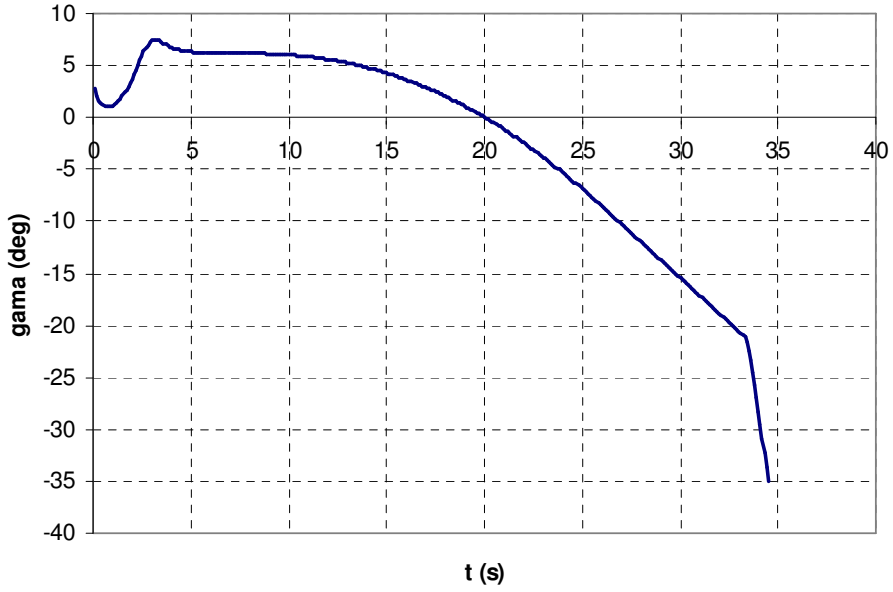


Figure 2-25 Flight Path Angle vs. Time

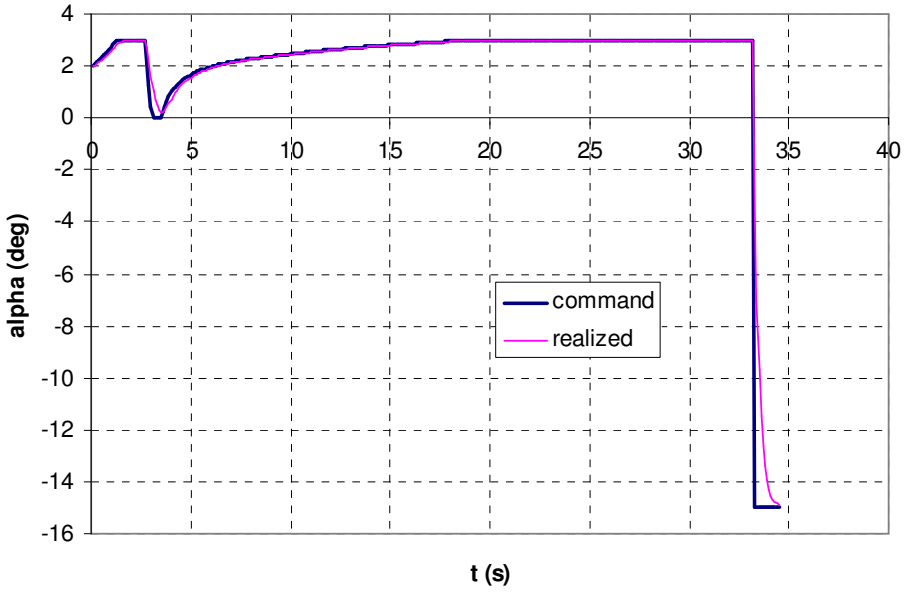


Figure 2-26 Angle of Attack vs. Time

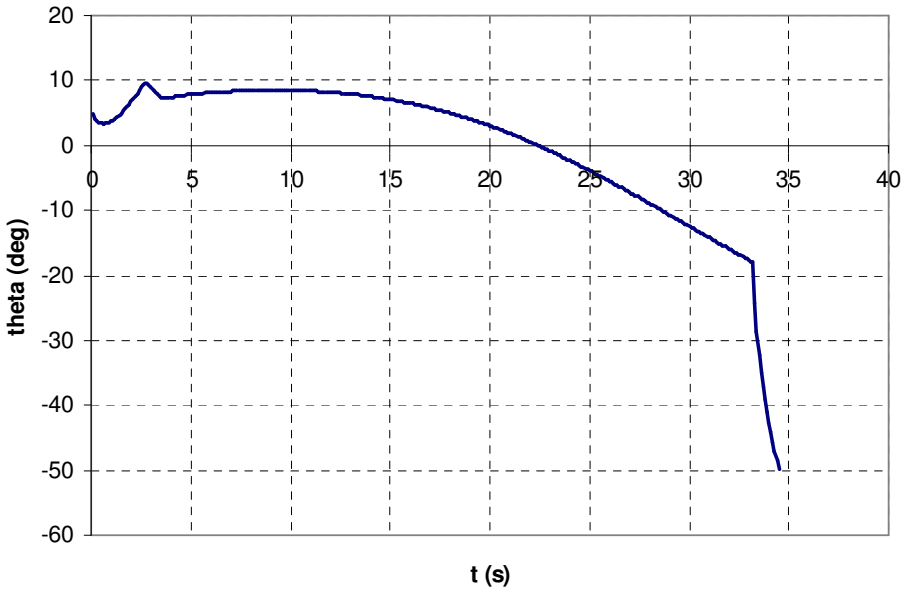


Figure 2-27 Pitch Angle vs. Time

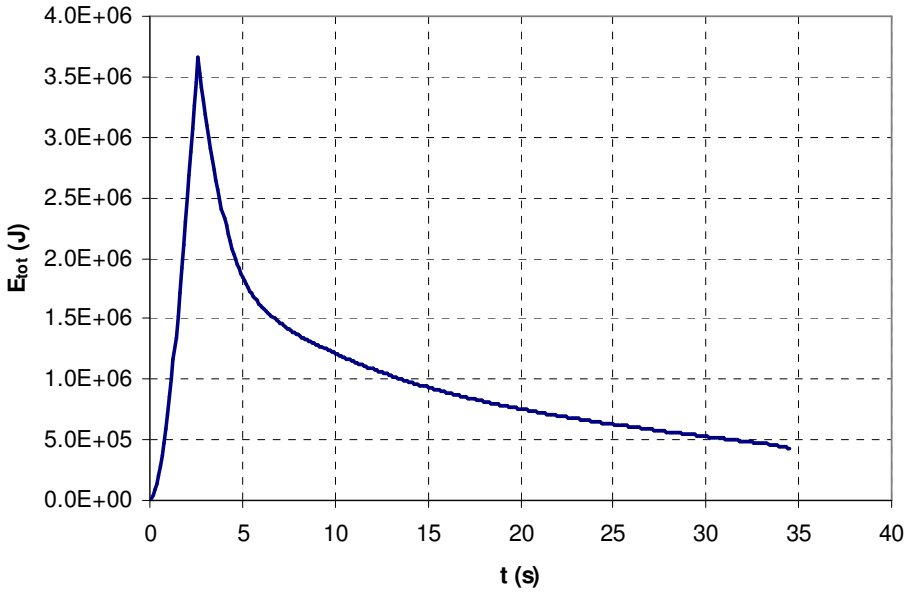


Figure 2-28 Total Energy vs. Time

The detailed interpretation of the results of that sample simulation run will not be given here since this run is for demonstration purposes only. Instead, some comments regarding to the simulation and optimization parts of the modified MC-MOSA in the light of these results are made below:

- The trajectory shape in Figure 2-22 is mainly determined by the angle of attack commands in the nodes. Optimization algorithm will try to reach to an optimum angle of attack command profile that will generate a trajectory which satisfies the objectives simultaneously.

- Speed, Mach number and total energy assumes similar profiles, whose maxima are mainly determined by rocket motor parameters. Again, the optimization algorithm will search for the optimum thrust parameters which as well satisfy the total impulse constraint imposed on thrust parameters.

- For an anti tank missile, a hit angle as steep as possible is required in terms of warhead effectiveness. Hence, the last portion of the angle of attack command profile shown in Figure 2-26 is employed to have a steep hit angle. Hit angle will be chosen as an optimization objective in some of the case studies.

- For an anti tank missile, hit angle is more important than hit velocity in terms of warhead effectiveness. However, a high hit velocity may be also important in terms of survivability and endurance against counter-measures for certain operational conditions. Hit velocity will also be used as an optimization objective in some of the case studies.

The integration method used, which is Euler method, and integration time step Δt have obviously an effect on the accuracy of the results. A study was carried out to investigate these effects. The time step used in the above study was Δt = 0.1 s. Separate studies were performed for Δt = 0.01 s and Δt = 0.001 s. Moreover, an integration algorithm using Runge-Kutta-Fehlberg Method (RKF45) described in [35] was prepared and applied. It has an adaptive procedure to determine if the proper time step is being used. At each integration step, two different Runge-Kutta approximations having errors in the order of 4 and 5 are made and compared. If the two answers are in close agreement, the approximation is accepted. If the two answers do not agree to a specified accuracy, the step size is reduced. If the answers agree to more significant digits than required, the step size is increased.

In these runs, a parameter set different from Table 2-4 is used. The comparative results are given in the following figures. Although only four graphs are presented, the other simulation profiles; namely; Mach number, flight path angle, angle of attack command and total energy showed similar behavior. The only remarkable difference is observed in the trajectory shape seen in Figure 2-29. The objectives of the case studies of Chapter 4, namely; range, flight time, hit angle and hit velocity are not affected remarkably by the change of integration method and time step. Hence, to benefit from simulation run times, the integration method for the case studies was chosen as Euler method with time step Δt = 0.1 s.

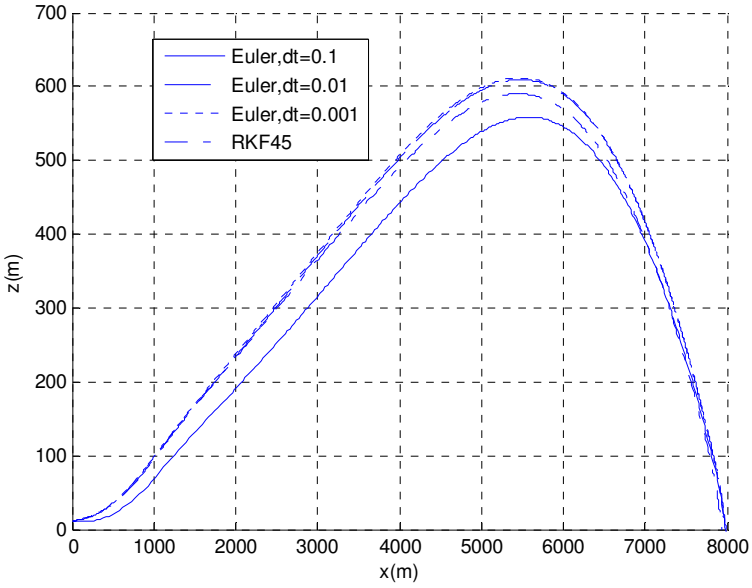


Figure 2-29 Altitude vs. Range (comparative study)

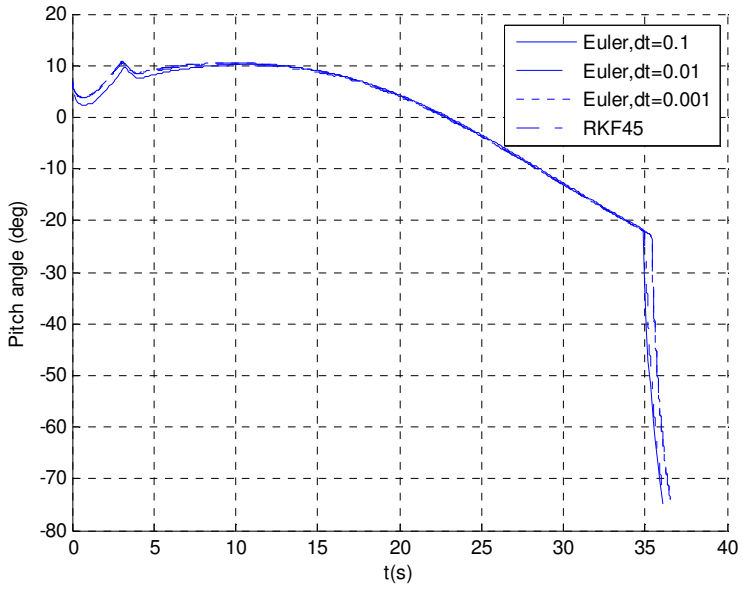


Figure 2-30 Pitch Angle vs. Time (comparative study)

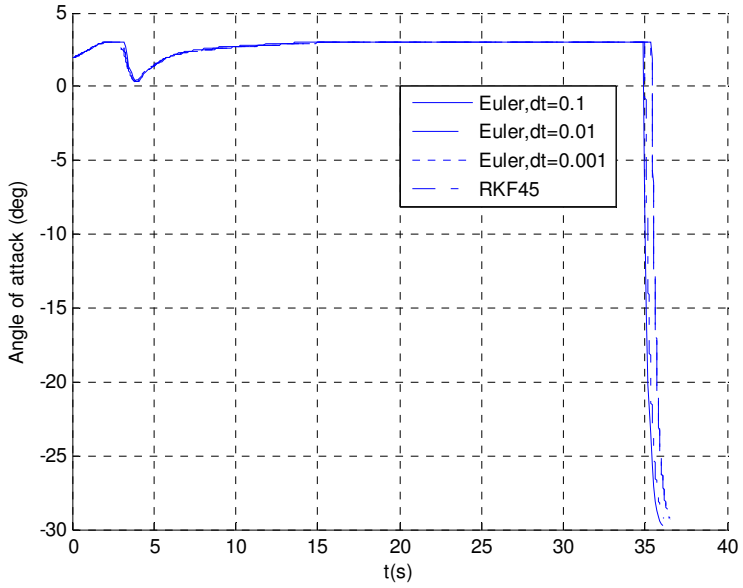


Figure 2-31 Angle of Attack vs. Time (comparative study)

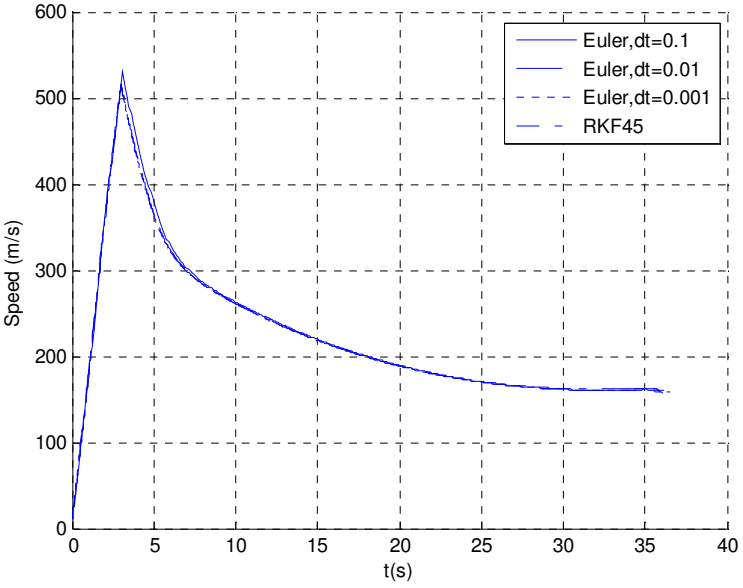


Figure 2-32 Speed vs. Time (comparative study)

CHAPTER 3

OPTIMIZATION ALGORITHM

Simulated annealing (SA) is an optimization method that simulates the physical annealing process. The concept is based on the manner in which liquids freeze or metals recrystallize in the process of annealing. The molecules of hot glass or metal are free to move about. Temperature is an average of the thermal energy in each molecule of an object. If the temperature drops quickly, these molecules solidify into a complex structure. However, if the temperature drops slowly, they form a highly ordered crystal. The molecules of a crystal solidify into a minimal energy state. In an annealing process, a melt, initially at high temperature and disordered, is slowly cooled so that the system at any time is approximately in thermodynamic equilibrium. As cooling proceeds, the system becomes more ordered and approaches a "frozen" ground state at zero temperature. If the initial temperature of the system is too low or cooling is done insufficiently slowly, the system may become quenched forming defects or freezing out in metastable states (i.e. trapped in a local minimum energy state).

For a given temperature T, the probability of a system to be in state **r** may be found from the Boltzmann distribution $\exp\left(\frac{E(r)}{k_B T}\right)$, where $E(\mathbf{r})$ is the energy of the configuration, and k_B is the Boltzmann constant [11]. To simulate the annealing process, Metropolis criterion may be used. For this purpose, the change in the energy of a system with the movement of an atom is calculated (ΔE). If the

movement lowers the energy of the system, it is accepted ($\Delta E \leq 0$). Otherwise it is accepted with probability of $P(\Delta E) = \exp\left(\frac{-\Delta E}{T}\right)$ (Accepting some movements that do not lower the energy prevents getting stuck in local optima). In SA, the objective function replaces the energy of the system, and the design variables represent the atoms. This idea was first used by Kirkpatrick et al. to solve discrete combinatorial optimization problems [11]. The technique was later extended to the optimization of functions of continuous variables [2, 4, 8, 20, 21, 26].

In SA, the success of the algorithm to find the global optimum by the fewest number of function evaluations is closely related to the method used in selecting the next candidate point. Another important aspect is the cooling scheme used. For these purposes, various methods are proposed. Some of them are summarized in [24].

For the optimization problems subject to this thesis, MC-MOSA [24], with some modifications, is used as the optimization algorithm. It is based on Hide-and-Seek, which is a continuous SA algorithm that uses pure random walk, where both the search direction and step size of test points are taken from uncorrelated uniform distributions, and an adaptive cooling schedule [2, 20]. In addition to these, MC-MOSA uses special elliptic and ellipsoid fitness functions as objective functions. Each function has its own adaptive cooling schedule.

Following paragraphs give information about the formulation, definition and some sample results of the modified MC-MOSA algorithm.

3.1 Formulation of the Optimization Problem

A constrained multiobjective optimization problem may be stated as follows:

$$\text{minimize } f_i(\mathbf{x}), \qquad i = 1, \ldots, I \tag{3.1}$$

$$\text{with respect to } \mathbf{x} = x_j, \qquad j = 1, \ldots, J$$

$$\text{subject to } g_a(\mathbf{x}) \geq 0, \qquad a = 1, \ldots, A$$

$$h_b(\mathbf{x}) = 0, \qquad b = 1, \ldots, B$$

$$m_c(f_i(\mathbf{x})) \geq 0, \qquad c = 1, \ldots, C$$

$$n_d(f_i(\mathbf{x})) \geq 0, \qquad d = 1, \ldots, D$$

(3.1) defines a minimizing problem. Objectives to be maximized can be included in this definition by taking the negatives of their objective functions. The design variables are usually given some upper and lower bounds, these bounds can be treated as inequality constraints and augmented into (3.1). Objective functions may also be constrained with some constraint functions or given some upper and lower bounds, which defines the feasible solution space. In this case, the optimal solutions are sought only within this space.

The vector objective function of (3.1) may be converted into a scalar one by using a scalarizing function with weights. A general one may be expressed as [5];

$$F = \left(\sum_{i=1}^{I} \nu_i \left| f_i(x) - f_i^{\,0} \right|^{\beta} \right)^{1/\beta} \tag{3.2}$$

To be able to assign weights to objective functions of different magnitudes, objective functions can be normalized with respect to some reference values:

$$\tilde{f}_i(x) = \frac{f_i(x) - f_i^{0,\min}}{f_i^{0,\max} - f_i^{0,\min}} \tag{3.3}$$

Here, the bounds of the feasible space are defined by the designer. They may result from the physical limitations known from the real experience (e.g. maximum or minimum range limits known for similar missile systems) or intolerability for some values beyond limits defined by design requirements (e.g. requirement for maximum flight time). The constraint functions and bounds for design variables are also derived by making similar judgments due to the nature of the problem (e.g. imposing constraints over propellant burn time and/or thrust levels to satisfy the maximum total impulse that can be achieved using a specific amount of propellant).

By choosing the reference values in (3.2) equal to $f_i^{0,\min}$, taking β=1 (weighted sum approach) and combining (3.2) with (3.3),

$$F = \sum_{i=1}^{I} w_i \tilde{f}_i(x) \tag{3.4}$$

Then, the optimization problem becomes minimizing F of (3.4) under conditions stated in (3.1).

By minimizing the single F, the designer will reach to a particular point on Pareto front. In MC-MOSA, the problem is further developed by introducing a population of scalar objective functions (F's) called fitness functions (FF's). In this approach, instead of summing up the objective values, some ellipses or ellipsoids (for 2 and 3 objective problems, respectively) are constructed and special points on them are used to generate an estimate for the whole Pareto front. Using a population of FF's rather than one enables the algorithm to move towards the whole Pareto front. Moreover, evaluating them together reduces the run times of the algorithm [24].

Two types of FF's that may be used in a two objective problem are linear and elliptic FF's. Linear FF's are especially suited for convex Pareto fronts. To have a

capability for solving non-convex problems, some special elliptic and ellipsoidal FF's are introduced in MC-MOSA.

The construction of elliptic FF's of MC-MOSA is shown in Figure 3-1 [24]

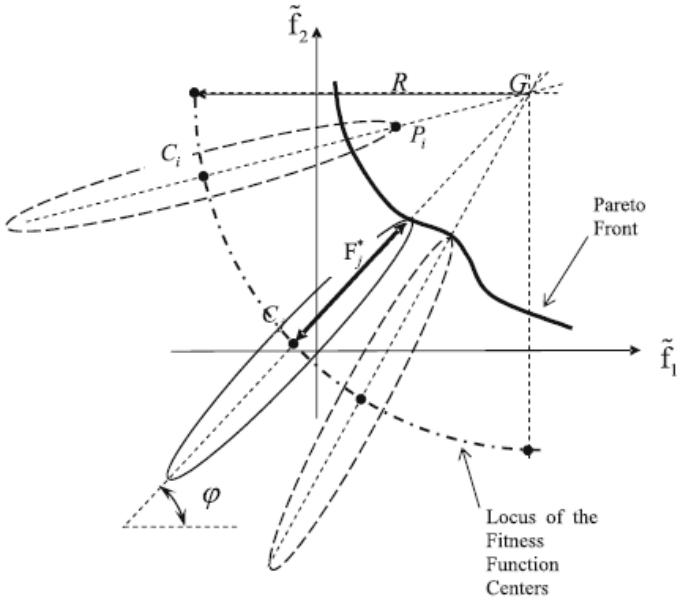


Figure 3-1 Construction of Elliptic Fitness Functions

In this figure, $\tilde{f}_1\tilde{f}_2$ defines the normalized objective space. A square area in this space is defined having corners at origin, *G*, (1,0) and (0,1). Origin corresponds to the normalized values of $f_i^{0,\min}$ which are zero, and (1,0) and (0,1) points corresponds to the normalized values of $f_i^{0,\max}$ which are one. This area covers all the possible solutions in or out of the feasible space. C_j are the centers of the ellipses placed on a quarter-circle with origin *G* and a pre-determined radius *R*. P_i is a particular solution corresponding to a design variable set $\boldsymbol{x_i}$. In each function evaluation *i*, the ellipses are constructed which are centered at C_j, passing through

point P_i and whose semi-major and semi-minor axes are aligned with the coordinate directions. Then, the semi-major axes are aligned with lines C_jG by performing a rotation. The length of the semi-major axis of such an ellipse can be written as

$$a = \sqrt{r^T \cdot Q^T \cdot \Lambda \cdot Q \cdot r} \tag{3.5}$$

where

$$r = \left\{ \tilde{f}_1^{\,P} - \tilde{f}_1^{\,C}, \tilde{f}_2^{\,P} - \tilde{f}_2^{\,C} \right\}^T, \; Q = \begin{bmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{bmatrix}, \; \Lambda = \begin{bmatrix} 1 & 0 \\ 0 & \kappa \end{bmatrix} \tag{3.6}$$

In (3.6), $\kappa = \dfrac{1}{1-e^2}$ and e is the eccentricity of the ellipses. φ is the angle between the horizontal axis and the line C_jG corresponding to each ellipse.

In this fashion, centers of ellipses are uniformly spread along the quarter-circle, ellipses are constructed as described and semi-major axes are calculated. Then, the optimization problem becomes minimizing the semi major axes of the ellipses (i.e., $F_j = a_j$). The minimum of each semi major axis corresponds to the point closest to a particular center.

The above procedure may be extended for a three-dimensional problem, constructing ellipsoids in this case. The centers are distributed on the surface of an eighth of a sphere almost uniformly, since providing a perfect uniformity is rather difficult and not necessary for the purposes of MC-MOSA. The rotation required to direct the semi-major axes towards G is defined as such: To align $\tilde{f}_1$ axis with the line connecting point C to the center of the sphere G, first rotate by an angle φ along $\tilde{f}_3$ axis, then rotate by an angle θ around the negative $\tilde{f}_2$ axis direction. The semi-major axis of such an ellipsoid may be written as

$$a_{3d} = \sqrt{r_{3d}{}^T \cdot Q_{3d}{}^T \cdot \Lambda_{3d} \cdot Q_{3d} \cdot r_{3d}} \tag{3.7}$$

where

$$r_{3d} = \left\{ \tilde{f}_1^{\,P} - \tilde{f}_1^{\,C}, \tilde{f}_2^{\,P} - \tilde{f}_2^{\,C}, \tilde{f}_3^{\,P} - \tilde{f}_3^{\,C} \right\}^T \tag{3.8}$$

$$Q_{3d} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \cdot \begin{bmatrix} \cos\varphi & \sin\varphi & 0 \\ -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.9}$$

$$\Lambda_{3d} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \kappa & 0 \\ 0 & 0 & \kappa \end{bmatrix} \tag{3.10}$$

3.2 Algorithm Definition

This section describes the optimization algorithm used in this thesis, which is MC-MOSA with some modifications. A two-loop optimization method, which was not present in the original MC-MOSA, is investigated in this thesis. It is applied in [23] for single objective optimization. In this thesis it is extended to multiobjective problems.

In two-loop approach, the optimization process is carried out in two concentric loops. In the inner loop, control parameters are changed while remaining parameters are kept constant. Geometry, thrust and initial condition parameters are changed in the outer loop more slowly. In the beginning of each outer loop, temperatures for each FF are reset. The parameter groups are given in Chapter 4.

As mentioned in the end of Section 2.3.1, DATCOM is the main time-consuming element of the algorithm. By applying two-loop optimization, aerodynamic database generation process is needed only in the outer loop, greatly improving the algorithm run times. If all the parameters are changed in one loop, for a total of 10k function evaluations, it takes roughly 12 hours (using a PC that has AMD-2800 processor and 512 MB of RAM) to complete one of the case studies of Chapter 4. In the two-loop approach; for the same number of loops and using the same

machine, this duration is reduced to only about 1.5 minutes, allowing much more loops to be applied and hence improving the accuracy of the results.

The steps of the modified MC-MOSA are as follows:

STEP 1 PERFORM INITIAL STEPS

Step 1.1 Define the numbers of design variables and FF's. Define allowed number of function evaluations (FE). Define eccentricity for FF's (ellipses/ellipsoids) and radius of the circle/sphere upon which FF centers are distributed. Define the probability value used in minimum semi-major estimation.

Step 1.2 Initialize the temperature array whose elements correspond to initial temperatures for the annealing schedules of each fitness function.

Step 1.3 Construct lower and upper bound vectors for design variables.

Step 1.4 Initialize random number generators and determine the percentile point of the chi-square distribution corresponding to the dimension of design variable space.

Step 1.5 Initialize test point vector to the mid-points between lower and upper bound vectors.

Step 1.6 Distribute centers of the FF's (ellipses/ellipsoids) on the surface of a circle/sphere and calculate rotation matrices Q_j for each FF.

Step 1.7 Define maximum and minimum reference values for each objective.

Step 1.8 Define boundaries of feasible solution space.

Step 1.9 Apply (3.3) to boundary points of feasible space and get normalized feasible solution vectors.

Step 1.10 Calculate semi-major lengths corresponding to boundary points of feasible space, assign maximum and minimum of them as upper and lower bounds for minimum semi-major estimation.

Step 1.11 Initialize counters for number of function evaluations, accepted function evaluations and records. Assign initial values of 2^{nd} best solutions array to all zero.

Step 1.12 Generate the new set of control variables (new test point) based on current point by moving a random direction and with a random step size. In the beginning of each outer loop, also change geometry, thrust and initial condition variables when generating new point. (Step size is bounded so that the upper/lower bounds on design variables are not violated)

Step 1.13 Evaluate the new test point (run the missile simulation) and obtain the values of objectives. Check for certain errors during evaluation, if encountered, break evaluation and go to **Step 1.12.**

Step 1.14 Normalize the objective values. For maximization problems apply a proper normalization such that it is converted to a minimization problem.

Step 1.15 Check whether the objective values are within the feasible space. If not, go to **Step 1.12.**

Step 1.16 Calculate semi-major lengths corresponding to the initial test point using (3.5) or (3.7) and assign them as the initial values of best solutions array.

STEP 2 START FUNCTION EVALUATION LOOP

Step 2.1 Repeat **Steps 1.12-1.15.**

STEP 3 START FITNESS FUNCTION EVALUATION LOOP

Step 3.1 Calculate semi-major lengths corresponding to the current test point using (3.5) or (3.7). If the current solution (semi-major length) is less than the best solution for the current FF,

- Assign it to the corresponding element of best solutions array. Assign the previous best solution to the corresponding element of 2^{nd} best solutions array.
- Estimate the minimum semi-major for the current FF.
- Update the estimation bounds found in **Step 1.10** so that current best solution is guaranteed to lie between them.
- Bound the estimation using these bounds.
- Update the corresponding element of the temperature array.
- Activate record flag indicating a record is found currently.

Step 3.2 If the current solution is not less than the best solution for the current FF, calculate acceptance probability.

Step 3.3 Go to **Step 3.1** until all FF's are evaluated.

STEP 3′ FINISH FITNESS FUNCTION EVALUATION LOOP

Step 2.2 If record flag is activated,

- Archive current test point and objective values as a record.
- Assign test point as the current point.
- Increase record and accepted function evaluation counters by 1 and deactivate the record flag.

Step 2.3 If record flag is not activated,

- Find the FF that has the highest acceptance probability.
- Accept the test point with the highest acceptance probability using a random number generator. If accepted, assign test point as the current point and increase accepted function evaluation counters by 1.

Step 2.4 Check restart condition. If satisfied, randomly select a test point among the records found so far and make this point as the current test point.

Step 2.5 Check whether an inner loop is completed. If so, reset each FF temperatures to initial temperature and go to **Step 2.1**.

Step 2.6 Go to **Step 2.1** until allowed number of function evaluations is reached.

STEP 3′ FINISH FUNCTION EVALUATION LOOP

STEP 4 Sort non-dominated solution points among the records to obtain the Pareto front.

Some remarks concerning the modified MC-MOSA algorithm defined above are given in the following sections.

3.2.1 Number, Sign Convention and Normalization of Objectives

The algorithm covers both two and three objective problems and the choice can be made by adjusting a single parameter before running the algorithm.

As stated earlier, negatives of the objectives that are to be maximized can be taken within the algorithm so that the problem becomes a whole minimization to avoid confusions among objectives. In the normalization equation (3.3), choosing reference values that has the same sign as the objective makes the normalized

values always positive, which prevents any sign confusion within the algorithm. Moreover, normalizing both objective values and feasible space boundary points with respect to the reference values (whose normalized values are 0 and 1) makes all the comparisons between feasible space and objectives consistent. Normalized solution space including feasible space is shown in Figure 3-2.

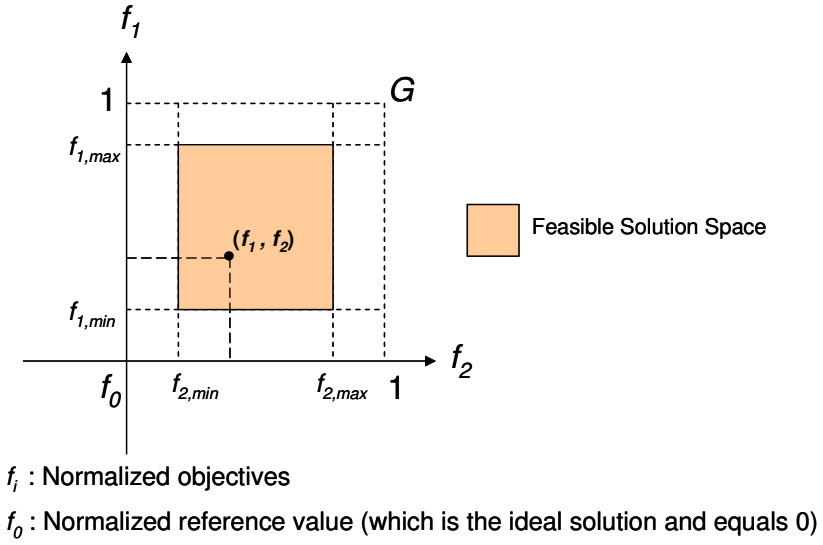


Figure 3-2 Normalized Solution Space

3.2.2 Handling of Constraints

The common approach in SA algorithms is to make the problem unconstrained by augmenting the design variable constraint functions of (3.1) to the objective function using penalty coefficients [16, 25]. This approach combines all the constraint functions into a single one, just like objective functions are combined into a scalar one given in (3.4). In [24], the following formulation is used for this purpose

$$\Omega = -\sum_{a=1}^{A} \eta_a \min(0, g_a(x)) + \sum_{b=1}^{B} \xi_b \left| h_b(x) \right| \tag{3.11}$$

To add constraint functions of different magnitudes, suitable penalty coefficients are required, just as weights for the case of objective functions. Then, this term is added to the scalar objective function so that the problem becomes minimizing this augmented objective function. If the problem includes lots of constraints, this method makes handling of them easier. However, it has some drawbacks:

- This method does not guarantee that the resulting Pareto solutions satisfy the constraints on design variables, just gives the solutions that have the minimum constraint violation.

- This approach requires some initial effort to find the accurate penalty coefficients which are related to the relative magnitudes of constraint functions. In a problem including many constraints, a considerable amount of initial trials may be required for that purpose.

Tournament selection method given in [5] may also be applied for constraint functions [6]. Among the current and newly generated test points, the one having the smallest overall constraint violation is chosen as the new test point. Both this and the augmentation method rely on minimization of objective and constraint functions both, but tournament selection method treats objective and constraint functions separately.

Another approach is to keep the problem constrained, track constrains as algorithm proceeds, if they are violated, reject the test point and generate a new one (rejection method) [21]. This approach eliminates the need for penalty coefficients. However, it also has some drawbacks:

- This method decreases the likelihood of evaluating the points on the constraint boundaries.

- If the constraint boundaries define a relatively narrow region in the design variables space, that is, considerably narrower than the region defined by upper and lower bounds, it may take long times for the algorithm to pull the test point into that region, making the optimization ineffective.

The constraint handling method used in modified MC-MOSA is described in connection to test point generation in the following section.

3.2.3 Generating New Test Point

Various methods are proposed for generating a new test point. Vanderbilt and Lougie [26] use random walk with a fixed maximum step size, where, for each variable, it is updated after a predetermined number of trials. The new set of step sizes is selected proportional to the inverse of the objective function's Hessian. Corona et al. on the other hand displaces one optimization variable at a time [4]. Siarry et al. also selects the next test point using random walk, but only displacing a randomly selected subset of optimization variables [21]. In [4] and [21], the step size is kept constant for a predetermined number of iterations.

Two additional methods are proposed in this thesis, which are closely related to constraint handling. These methods are intended to provide a general method for handling any type of constraints (linear, non-linear, discontinuous, piecewise, etc.) with any number of design variables, and especially for implicit constraint functions. For problems having lots of constraints, these methods become rather complicated and computationally intense, but they generalize all types of constraints and do not require explicit expressions for constraint functions.

First one is called squeeze-and-replace method and depicted in Figure 3-3. This figure shows a two dimensional design space of variables x_1, x_2 with upper and lower bounds (0.2, 1.0), However, this method can be applied for any design space dimension. Design variables are constrained into the shaded region. In this method, the region defined by upper and lower bounds is squeezed so that constraint

function has at least one real value for every variable in the squeezed region, as shown with dashed lines. When generating new test points, the step sizes are bounded within this region. If a generated point P_1 violates the constraint, one of the variables (which is previously chosen among the variables of this constraint) is moved to a point P_1* on the constraint curve while others are kept constant. Thus, new test point becomes P_1*. Squeezing guarantees that always a P_1* can be found on that region. An algorithm using bisection method can perform this operation: Initialize x_1 with its lower bound while keeping other variables as they are generated, increase x_1, seek for a sign change in the constraint function value while halving the search interval and changing the movement direction in each sign change occurrence, iterate this operation until a predetermined convergence is satisfied and repeat the steps for upper bound. This process can be repeated for other dimensions of the design space using the values found in the previous loop. The whole process can be repeated for other constraint functions, obtaining the squeezed region covering all the constraints. And finally, the step sizes for generating new test point are limited by this region. By this way, always a P_1* can be found corresponding to any variable.

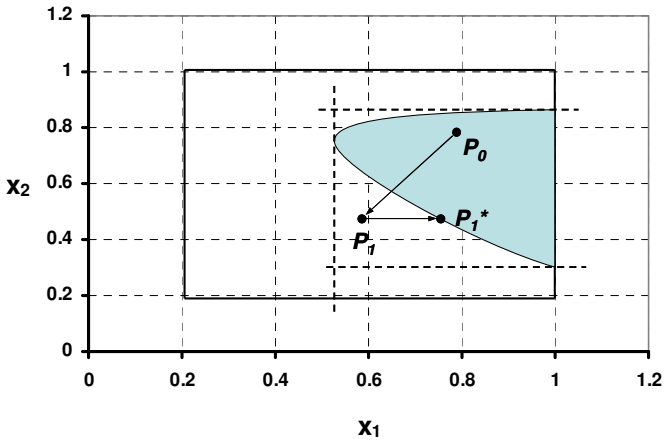


Figure 3-3 Squeeze-and-Replace Method

The second one is called nearest point method and depicted in Figure 3-4. If a newly generated point P_1 violates a constraint, the point on the constraint curve nearest to the test point (P_1*) is selected as the new test point. In this method, the constraint curve, bounded by design variable bounds, is defined as a vector function of the variables belonging to that constraint. By differantiating this vector function, a point-to-curve method may then be used to find the point corresponding to the nearest distance. The corresponding algorithm requires derivative information, hence one has use a numerical derivative technique such as finite difference method.

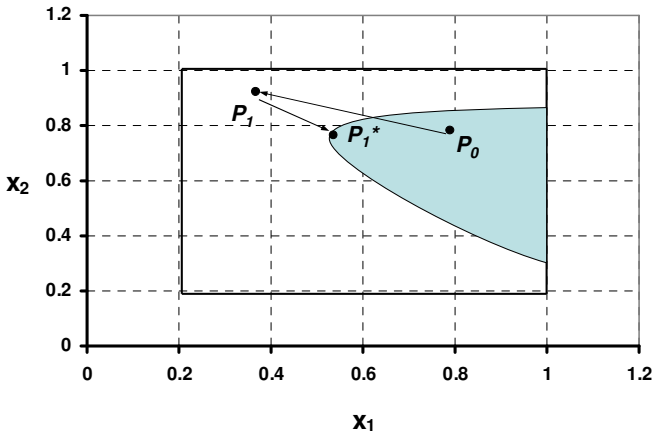


Figure 3-4 Nearest Point Method

Instead of these methods, a random point on the constraint curve may be selected as the new test point when constraints are violated. However, for implicit constraint functions this may not be an easy task and may require more computational power than the two methods.

The two methods proposed prevent evaluation of points beyond constraints and, although iterative, generate a replacement point in one-step instead of purely rejecting and generating a new one, probably many times.

In modified MC-MOSA, a simplified version of squeeze-and-replace method is employed. Since the constraint functions used in case studies subject to this thesis are readily squeezed as in Figure 3-5, one variable is chosen and simply replaced while keeping other variables constant.

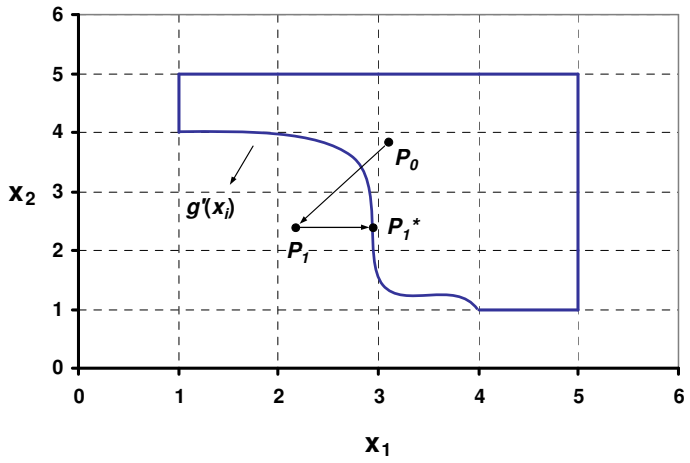


Figure 3-5 Replacement of Test Point

The variables chosen to be replaced for each constraint function are determined such that bounding that variable is anticipated to be more feasible than bounding others. For example, if a total impulse constraint is imposed on the thrust profile of a missile and if it is violated, it may be more feasible for engine designer to limit burning time instead of thrust levels since it can be adjusted by changing the burning surface geometry of the propellant. Or, if the root chord location of an aft wing from the nose is not to exceed the total length of the missile, changing the aft

wing location on the missile body may be more convenient than increasing length because an elongation of missile may be less tolerable in terms of design priorities.

The constraints on objective values (which define the feasible solution space) are handled simply by rejecting and regenerating the test point when these constraints are violated, after the function evaluation step.

For generating a new test point, the following steps are employed:

- Generate a unit vector θ originating at the current test point and directing towards a random direction in the design variables space.

- Choose random magnitudes Λ from uniform distribution such that

$$\Lambda = (\lambda \in (0,1); x + \lambda \cdot \theta \cdot s_{\max} \in S) \tag{3.12}$$

  where S is the design variables space.

- Set new test point y as

$$y = x + \Lambda \cdot \theta \cdot s_{\max} \tag{3.13}$$

  where s_{max} is the maximum allowed step size to stay in the upper/lower bounds. It is updated in each function evaluation loop by:

$$s_{max} = \begin{cases} x_i - x_{lower,i}, & \lambda_i \theta_i \le 0 \\ x_{upper,i} - x_i, & \lambda_i \theta_i > 0 \end{cases} \tag{3.14}$$

3.2.4 Acceptance of Test Points

In the process of SA algorithms, 3 cases may be encountered:

- If a test point improves all the objectives, that is, a design variable set lowers all the objective functions (FF semi-major lengths in the case of this thesis) with

respect to previous ones, that point is accepted and the new test point is generated based on this point (test points improving any FF is called "records" throughout this thesis).

- If the test point improves some of the objectives and deteriorates others, that point is accepted with a 1 or less probability depending on the method used.

- If a test point deteriorates all the objectives, it is accepted with a probability less than 1. After evaluating the probability if it is rejected, the new test point is generated based on the last record.

The acceptance probability may be calculated by various methods, the most common ones employed are presented in [24]. In MC-MOSA, if any of the FF semi-major lengths improve, then the test point is accepted with probability one. Otherwise the highest probability acceptance value is used. For each FF, the probability is calculated by

$$P = \min\left\{1, \exp\left(\frac{\Delta\tilde{F}}{T}\right)\right\},\ \Delta\tilde{F} = \tilde{F}(y) - \tilde{F}(x) \tag{3.15}$$

where x is the current test point and y is the candidate for current test point.

3.2.5 Annealing Schedule

In MC-MOSA, each FF has its own temperature parameter and cooled individually. Whenever a FF is improved, its temperature is updated as follows:

$$T_j = \frac{2(\tilde{F}_j(y) - \tilde{F}_j^{\,*})}{\chi_d^2(1-p)} \tag{3.16}$$

where $\chi_d^2(1-p)$ is 100(1 – *p*) percentile point of the chi-square distribution with *d* degrees of freedom. $\tilde{F}_j^*$ is the minimum semi-major estimation for that FF. It is calculated using the following heuristic estimator:

$$\tilde{F}_j^* = \tilde{F}_j^{best} + \frac{\tilde{F}_j^{best} - \tilde{F}_j^{2^{nd}best}}{(1-p)^{-d/2} - 1} \tag{3.17}$$

Parameter *p* corresponds to the probability that the real minimum is larger than its estimate $\tilde{F}_j^*$ and *d* is the number of design variables.

As shown in [23], using the estimator with upper and lower bounds results in a better estimator and improves the convergence rate. The upper and lower bounds in modified MC-MOSA are found by calculating semi-major values corresponding to the corner points of the feasible space, and taking the maximum and minimum of them as bounds. These bounds are further updated so that the current best solution lies between them. This improves the reliability of the estimation.

Kirkpatrick [10] suggested that a suitable initial temperature (T_0) is one that results in an average probability χ_0 of a solution that increases f being accepted of about 0.8. The value of T_0 will clearly depend on the scaling of f and, hence, be problem-specific. It can be estimated by conducting an initial search in which all increases in f are accepted and calculating the average objective increase observed $\delta\bar{f}^+$. It is then given by:

$$T_0 = \frac{\delta\bar{f}^+}{\chi_0} \tag{3.18}$$

Alternatively, if the standard deviation σ_0 of the variation in the objective function observed during this initial search is calculated, then the formulation of White [27] can be used:

$$T_0 = \sigma_0 \tag{3.19}$$

Since (3.16) defines an adaptive cooling, the initial temperature has minor effect in the course of function evaluation loops. Regardless, (3.18) is employed in modified MC-MOSA to initialize the temperature array for FF's. As mentioned in Section 3.2, FF temperature array is also initialized in the algorithm at the beginning of each outer loop.

3.2.6 Restart Condition

If the SA search is no longer making progress, i.e. many loops have passed since a new best solution was last found, a strategy by which the search is restarted from a random test point selected among the best solutions found thus far (while keeping the temperature unchanged) can prove effective. Such a restart strategy must be used with caution, since, if the conditions under which a restart is made are met too easily, only a limited part of the search space (possibly only a local minimum) will be explored.

In modified MC-MOSA, restart is applied when no record is found for a certain percent of maximum allowed number of function evaluations, which is called restart coefficient (RC). The effects of this parameter on the performance of the algorithm are investigated with some parametric studies presented in Section 3.3 and also in Chapter 4.

3.2.7 Selection of Non-Dominated Points

When all the function evaluation loops ends, the non-dominated points (or Pareto-optimal solutions) are selected among the records. To do this, objective pairs (or triples in 3-objective problem) are sorted with respect to first objective in ascending order. In this objective array, the rows where the second (or third) objective is descending are non-dominated points and they constitute the Pareto front. In formal words, a design variable set $x^* \in S^d$ corresponds to a Pareto-optimal solution if

there does not exist another $x \in S^d$ such that $f_i(x^*) \leq f_i(x)$ for all i; and $f_j(x^*) < f_j(x)$ for at least one j.

3.3 Sample Results

The modified MC-MOSA algorithm was tested against two simple multi-objective optimization problems with two and three objecives. The result are presented below.

The first problem is from [24]:

minimize $f_1 = x \ , \ f_2 = y$ (3.20)

subject to $5e^{-x} + 2e^{-0.5(x-3)^2} - y \leq 0$

$0 \leq x \leq 5, 0 \leq y \leq 5$

In Table 3-1, some input parameters and resulting number of non-dominated solutions of this sample run are given.

Table 3-1 Parameters of Sample Problem #1

FEN	**FFN**	***e***	***R***	***p***	**NDP**
5000	250	0.999	2	0.01	418

The resulting Pareto points are shown in Figure 3-6 along with the curve defined by constraint function. This curve actually defines the exact Pareto front for this problem. No Pareto points exist on the middle portion of the curve because of non-dominated selection.

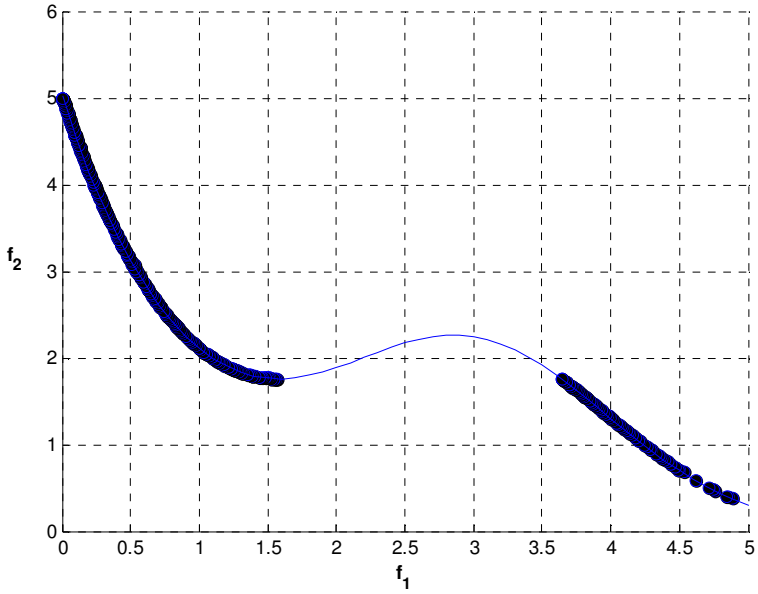


Figure 3-6 Pareto Points for Sample Problem #1

In [24], MC-MOSA is compared with NSGA-II for this sample problem, which is a famous genetic algorithm used in multiobjective optimization. The comparisons are made through some parametric studies.

The same studies were performed for modified MC-MOSA and the results are presented below along with the results of NSGA-II and original MC-MOSA. The quality metrics defined in Appendix B are used for comparison. The function evaluation number (FEN) is chosen the same as the study of [24], which is 10k. NSGA-II parameters in these studies are given as 100 population evaluated for 100 generations. Number of elliptic FF's (FFN) used MC-MOSA is given as 250, the same is utilized in modified MC-MOSA. For each algorithm 30 runs of this sample problem are performed, the resulting average values and standard deviations of the quality metrics are tabulated in Table 3-2.

Table 3-2 Comparison of Algorithms for Sample Problem #1

	NSGA-II		MC-MOSA		Modified MC-MOSA	
	Av.	**SD**	**Av.**	**SD**	**Av.**	**SD**
NDP	73.9	16.1	65.4	6.4	487.2	26.0
HD	0.390	0.012	0.386	0.004	0.383	0.006
A	37.9	27.5	66.1	15.4	42.1	21.0
OS	0.719	0.180	0.875	0.028	0.702	0.064
$CL_{1/25}$	3.29	1.04	2.20	0.19	15.82	1.13
$CL_{1/100}$	1.43	0.37	1.14	0.05	4.41	0.26
$CO_{1/25}$	23.9	7.1	29.7	1.7	30.9	2.1
$CO_{1/100}$	55.1	17.3	57.4	5.3	110.6	5.8

It may be observed from the table that modified MC-MOSA generated much more non-dominated points (NDP) than the other algorithms. It also performed slightly better than NSGA-II and nearly same as MC-MOSA in terms of HD. Its accuracy (A) metric lies in-between, nearer to NSGA-II. It has a worse average overall spread (OS) than MC-MOSA and NSGA-II, but observing standard deviations it can be said that OS performance is somewhat better than NSGA-II. Due to the definitions given in Appendix B, CL metrics are not conclusive here because NDP values of modified MC-MOSA are much more than the others. For low cluster step size ($CO_{1/25}$), performance of modified MC-MOSA is better than NSGA-II and nearly same as MC-MOSA. For low cluster step size ($CO_{1/100}$) it is quite better than the others. In general, modified MC-MOSA performed quite better than NSGA-II in terms of these quality metrics for this non-convex problem. It can be said that its general performance is also superior to original MC-MOSA due to much better NDP and better CO metric. It can be commented that modifications in constraint handling and test point generation methods may have a huge impact on these results.

Second problem is a 3-objective problem;

minimize $f_1 = \frac{10}{x},\ f_2 = -y,\ f_3 = 2x + y + 5$ (3.21)

subject to $2 \le f_1 \le 10,\quad -5 \le f_2 \le 0,\ \ 5 \le f_3 \le 20$

$0 \le x \le 5,\ 0 \le y \le 5$

In Table 3-3, some input parameters and the resulting number of NDP for this sample problem are given.

Table 3-3 Parameters of Sample Problem #2

FEN	**FFN**	***e***	***R***	***p***	**NDP**
20000	100 x 100	0.999	2	0.01	1444

The resulting Pareto points are shown in Figure 3-7. In order for a design problem to be a multiobjective optimization problem, it has to be formulated such that an improvement in one objective causes at least one deterioration in the other(s). This problem satisfies that property, and in accordance with the definition given in 3.2.7, each point on the Pareto surface of Figure 3-7 is an improvement over some other points, along at least one objective dimension.

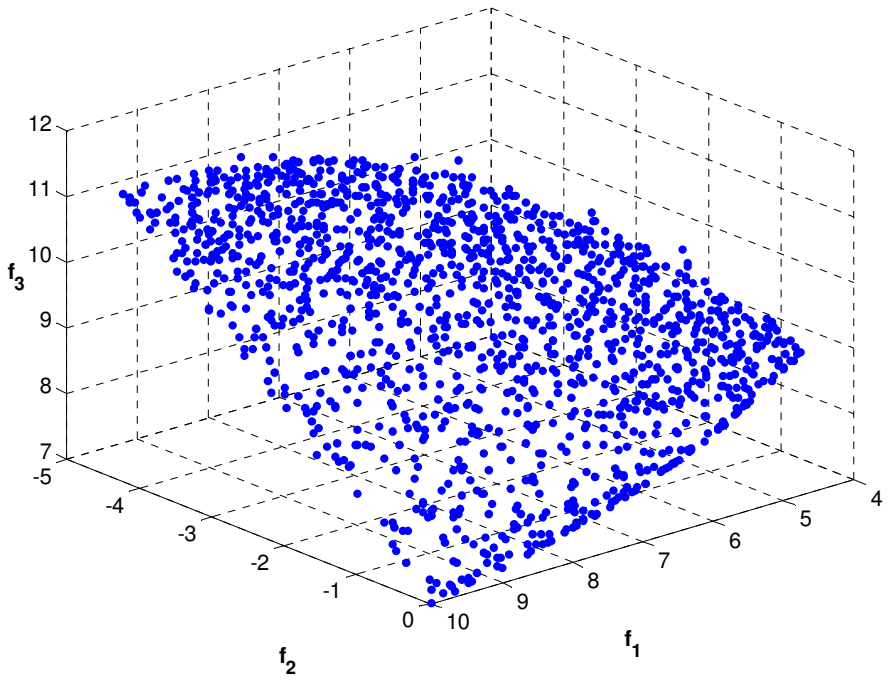


Figure 3-7 Pareto Points for Sample Problem #2

CHAPTER 4

CASE STUDIES

Modified MC-MOSA algorithm described in Chapter 3 is applied to 5 different missile multiobjective design optimization problems, utilizing the modified 2-DOF simulation described in Chapter 2. These are 2 and 3 objective problems. Some parametric studies are also carried out to investigate the performance of the algorithm for different input conditions. This chapter presents the results of these studies.

The optimization problems are as follows:

- Maximize range, minimize flight time.
- Maximize range and hit velocity.
- Maximize hit angle and hit velocity.
- Maximize trim lift coefficient, minimize trim drag coefficient for a specific flight condition.
- Maximize range and hit velocity, minimize flight time.

The specific flight condition 4th problem is defined by the angle of attack and Mach number as α=2°, M=0.65. These are some typical average values for after-burnout flight phase of the baseline missile of this thesis. Although the average values

apparently change for each design variable set, these values are assumed so that the resulting $C_{L,trim}$ and $C_{D,trim}$ and can be used as a measure of the aerodynamic effectiveness of the configuration for most of the flight.

The case studies are tabulated in Table 4-1. In accordance with the description given in Section 3.2.1, maximization problems are converted into minimization by properly normalizing the associated objective.

Table 4-1 Case Studies

Case Study #	**# of Objectives**	**Maximize**	**Minimize**
1	2	R	t
2	2	R, V_{hit}	
3	2	θ_{hit}, V_{hit}	
4	2	$C_{L,trim}$ (α=2°,M=0.65)	$C_{D,trim}$ (α=2°,M=0.65)
5	3	R, V_{hit}	t

The inequality and equality constaints imposed on the design variables are given in Table 4-2. Also given is the replaced variable for each constraint (see Section 3.2.3 for information on replaced variables). Some other equality constraints which are not included in this table are also employed, they are described in sections 2.1 and 2.2.

Table 4-2 Constraints on the Design Variables of Case Studies

#	Constraint Definition	Replaced Variable
1	$c_{1,2} - c_{1,1} \leq 0$	$c_{1,2}$
2	$L_{nose} - x_{le,1} \geq 0$	$x_{le,1}$
3	$c_{2,2} - c_{2,1} \leq 0$	$c_{2,2}$
4	$x_{le,2} - (L_{nose} + L_{body} - c_{2,1}) \leq 0$	$x_{le,2}$
5	$c_{3,2} - c_{3,1} \leq 0$	$c_{3,2}$
6	$c_{3,3} - c_{3,2} \leq 0$	$c_{3,3}$
7	$t_b - 30000 / T_d \leq 0$	t_b
8	$x_{le,3} - x_{le,2} - c_{2,1} = 0$	-
9	$\delta_{lim,max} + \delta_{lim,min} = 0$	-

As described in Section 3.2, a two-loop optimization method is used in these studies (except CS#4). The parameters changed in outer and inner loops and the upper and lower bounds on them are given in Table 4-3 and Table 4-4, respectively.

Table 4-3 Outer Loop Design Variable Bounds for Case Studies

		Lower	Upper	Unit
THRUST PARAMETERS	t_b	2	4	s
	T_d	5000	10000	N
GEOMETRY PARAMETERS	B_{nose}	5	10	cm
	L_{aft}	0	10	cm
	$b_{1,2}$	3	7	cm
	l_{max1} / c	0.1	0.2	-
	$c_{1,1}$	10	15	cm
	$c_{1,2}$	5	10	cm
	t_1 / c	0.01	0.02	-
	$\Lambda_{te,1}$	-10	5	deg
	$x_{le,1}$	25	35	cm
	$b_{2,2}$	5	10	cm
	l_{max2} / c	0.1	0.2	-
	$c_{2,1}$	40	60	cm
	$c_{2,2}$	28	38	cm
	t_2 / c	0.01	0.02	-
	$x_{le,2}$	80	110	cm
	$b_{3,2}$	4	7	cm
	$c_{3,1}$	6	9	cm
	$c_{3,2}$	6	9	cm
	$c_{3,3}$	0	9	cm
	δ_{lim}	-30	-20	deg
INITIAL CONDITION PARAMETERS	z_0	100	0	m
	V_0	0	20	m/s
	γ_0	5	10	deg
	α_0	-1	3	deg

Table 4-4 Inner Loop Design Variable Bounds for Case Studies

		Lower	Upper	Unit
CONTROL PARAMETERS	τ	0.2	0.7	s
	e_{per}	0.85	0.95	-
	α_{hit}	-25	-5	deg
	z_{hit}	300	50	m
	$\alpha_{node,i}$	0	5	deg

All the objectives are given some reference and feasible values. Reference values (f^{0}_{min} , f^{0}_{max}) are used in the normalization equation (3.3). They are chosen as zero for one extreme, and a high value for other extreme. The high values represent the ideal best or worst values anticipated for the objectives. Feasible values, on the other hand, are the practical best and worst values expected. They define the limits of objectives in optimization algorithm. Moreover, they are used in calculation of estimation bounds as described Section 3.2.5. Reference and objective values are given in Table 4-5.

Table 4-5 Reference and Feasible Values for the Objectives of Case Studies

	R (m)	t (s)	θ_{hit} (deg)	V_{hit} (m/s)	C_L (α=2°, M=0.65)	C_D (α=2°, M=0.65)
f^{0}_{min}	0	0	0	0	0	0
f^{0}_{max}	14000	80	90	600	2	2
f_{min}	2000	5	20	100	0.1	0.1
f_{max}	12000	60	90	500	0.5	0.5

In all of the following studies, eccentricity values for FF's are taken as e = 0.99, radii of circle/sphere on which FF centers are placed are taken as R = 2 and probability values used in minimum semi-major estimation are taken as p= 0.01.

4.1 Pareto-Optimal Solutions for Case Studies

This section covers the graphs showing the record points and the non-dominated points for each case study and comments on them. The feasible regions are also shown in each graph. Feasible region is one side of Pareto front that contains the point corresponding to the worst Pareto-optimal solution of each objective, which is also called "nadir point" [6].

Some intermediate Pareto-optimal solutions are chosen from each graph, these solutions and the corresponding design variables are presented in Appendix, Table A.1. The flight trajectories and angle of attack profiles for the two extreme Pareto-optimal solutions of each CS, as well as those for the intermediate solutions of design variable sets of Table A.1 are also given in the figures of Appendix.

All studies were performed utilizing a total of 100k function evaluations (total of inner and outer loops) and 1000 inner loops, with the exception of CS#4. This problem was a one-loop optimization as will be described below and in each loop DATCOM had to be run. Hence, the algorithm run time dramatically increased (11.3 hours for 10k function evaluations). Consequently, a total of 10k function evaluations were utilized for CS#4. The algorithm run times were approximately 7.5 minutes for CS#1, CS#2 and CS#3; and 14.6 minutes for CS#5.

In all problems, if no records are found for 0.05 percent of allowed number of function evaluations, restart process defined in Section 3.2.6 was applied. For 2-objective problems, 250 elliptic FF's are used whereas for 3-objective problem, 50x50=2500 ellipsoids are used.

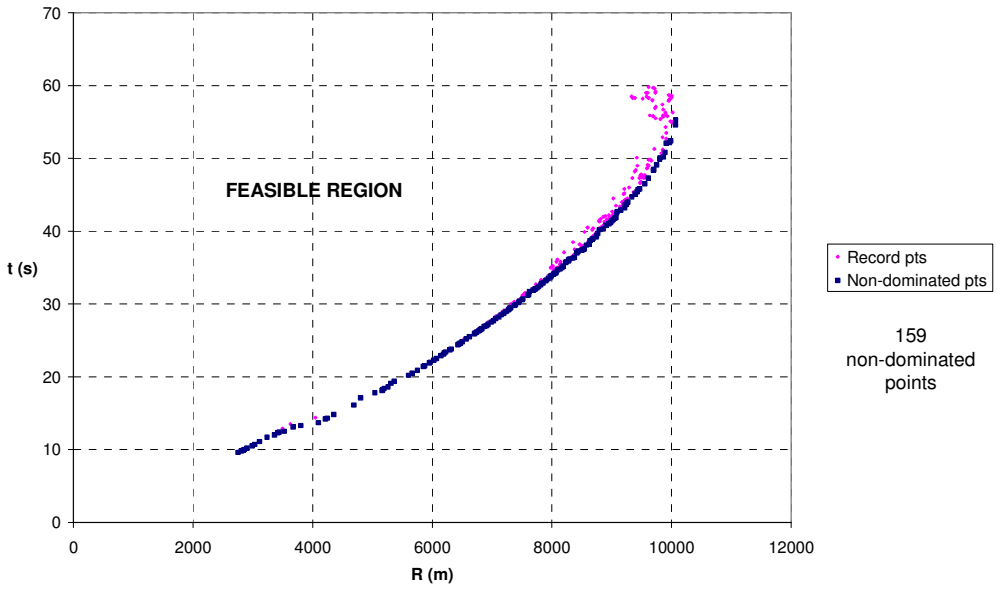


Figure 4-1 Pareto Points for CS#1: Range vs. Flight Time

Range vs. Flight Time turned out to be a convex problem and generated the most convex and clear front compared with the other case studies as seen in the figure above. Also it is seen that the ratio of NDP to number of records is quite higher for this problem than the others. It can be concluded that this is relatively the simplest multiobjective problem among other case studies.

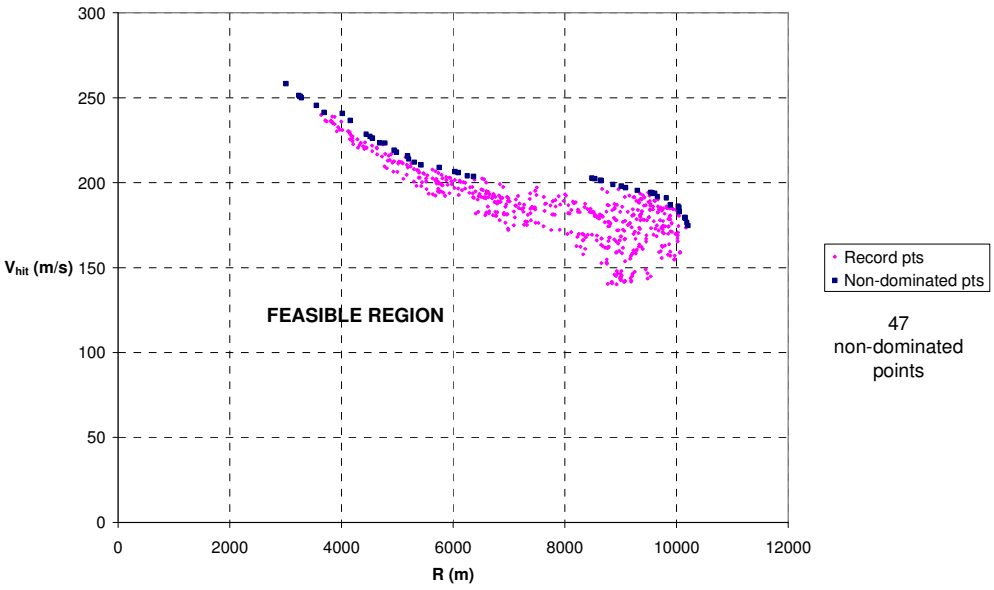


Figure 4-2 Pareto Points for CS#2: Range vs. Hit Velocity

Range vs Hit Velocity problem resulted in a non-convex front. The records between the ranges ~6200 m and 8200 m are not included in the Pareto front because of non-dominated selection. As proceeded from left to right in the figure, hit velocity decreases as range increases. However, after a point near 6500 m, velocity starts to increase for some time and then starts to decrease again at around 8200 m. This is because missile configurations of this portion gains so much altitude that the potential energy gains of them are enough to improve hit velocities as well as range (Altitude gain in this portion can be observed in Figure A.3). This portion is where range and hit velocity is non-conflicting for this baseline missile. It should also be noted that the border points between ~6200 m and ~6500 m are non-dominated by the left-side points but dominated by the right-side points of the front, hence they are excluded.

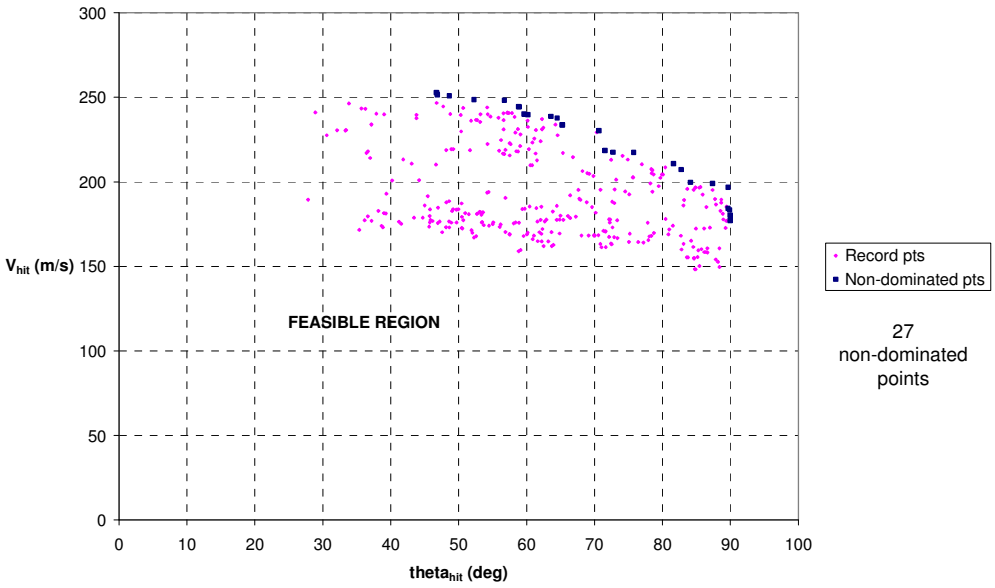


Figure 4-3 Pareto Points for CS#3: Hit Angle vs. Hit Velocity

Hit Angle vs Hit Velocity problem turned out to be convex, but not many non-dominated points were generated. Also the record points are more sparse and scattered. This problem can be interpreted as the most difficult one along with the 3-objective one.

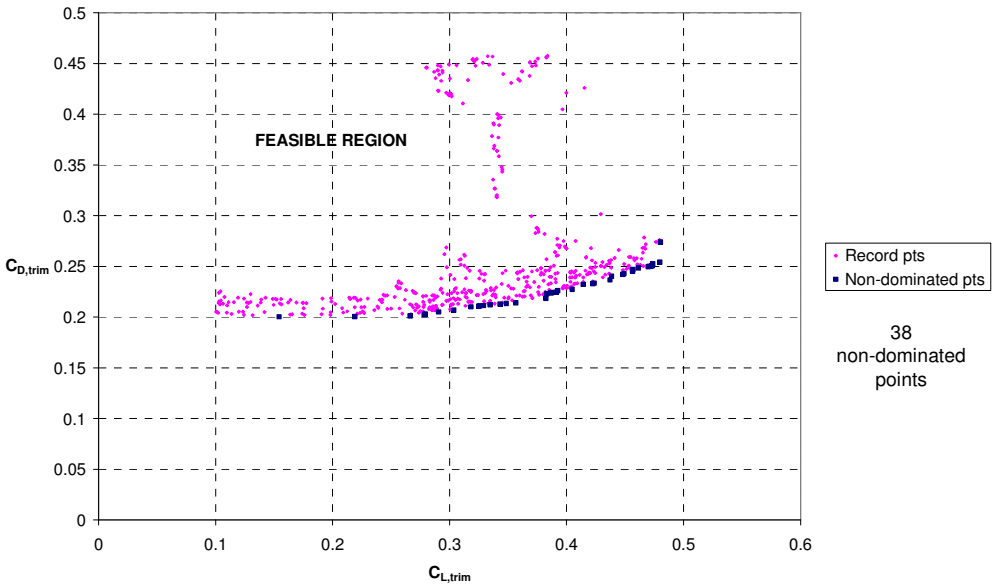


Figure 4-4 Pareto Points for CS#4: $C_{L,trim}$ vs. $C_{D,trim}$

For $C_{L,trim}$ vs. $C_{D,trim}$ problem; thrust, control and initial condition variables were not applicable because no flight simulation was needed, only DATCOM results corresponding to the specific flight condition (α=2°, M=0.65) were used. Consequently, a one loop optimization is performed using only geometry variables. This problem is also a convex one. Although non-dominated selection process caused Pareto points to be sparse for $C_{L,trim}$ values less than ~0.26, a fairly enough number of record points were generated in the neighbourhood of Pareto front so that it is well visualized. It should be noted again that these are the results of only 10k function evaluations, implying that this is also a relatively simple problem (except for the long run time) along with CS#1. It appeared that a smaller upper limit for $C_{D,trim}$ (e.g. 0.3 instead of 0.5) may have been used to eliminate the scattered points in the upper region of the graph.

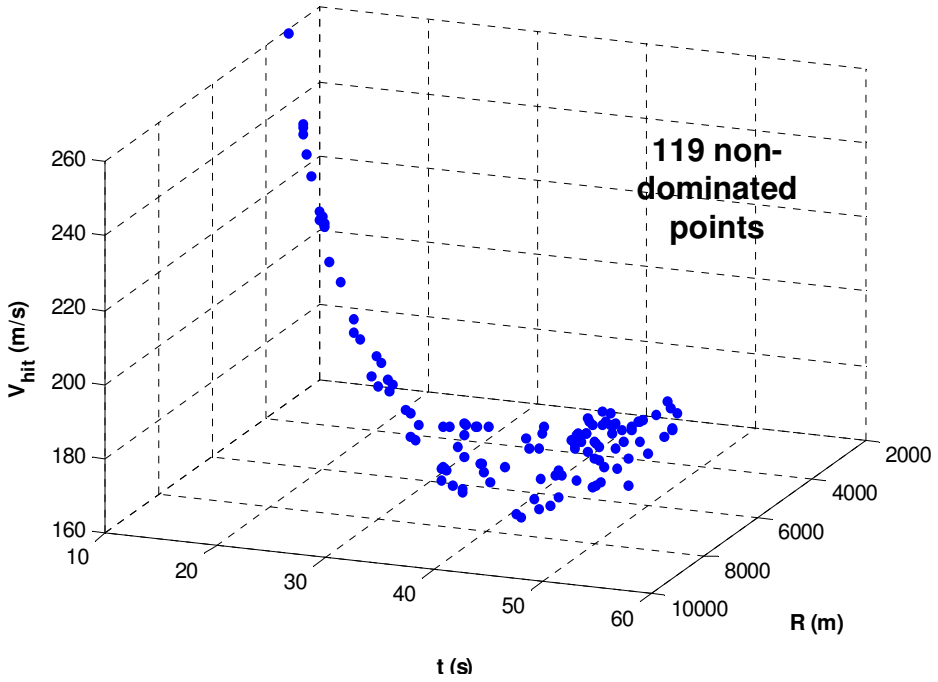


Figure 4-5 Pareto Points for CS#5: Range vs Flight Time vs Hit Velocity

A partially-curve and partially-surface like cluster of Pareto points occurred for the 3-objective problem of Range vs Flight Time vs Hit Velocity. Record points are not shown here not to deteriorate the visualization; however, it can be noted that 186 record points were generated, 119 of which are non-dominated points. Showing the feasible space on this figure is somewhat difficult, but it is on the side of the Pareto front including the point " R = 0 m, t = 100 s, V_{hit} = 0 m/s " (ideal worst values - nadir point).

4.2 Parametric Studies

The modified MC-MOSA algorithm has mainly four basic parameters: Total number of function evaluations, inner loop number, number of FF, and restart coefficient. In this section the effects of these parameters in obtaining Pareto front is

examined by the help of Range vs. Flight Time problem. For this purpose some quality metrics are used. Many different metrics are proposed in the literature [12, 28, 29]. The troubles associated with these metrics in evaluating the quality of the front have also been discussed in the literature [12, 29]. The quality metrics used in this study are taken from [24, 28] and also summarized in Appendix B.

Following tables compare the effects of four basic algorithm parameters. Average and standard deviation values of quality metrics are given for 30 runs. In all the following tables, the nominal values of algorithm parameters used are FEN=100k, FEN_{inner} =1000, FFN=250 and RC=0.005.

Table 4-6 Parametric Study for Function Evaluation Number (FEN_{inner} = 1000)

	FEN=10k		**FEN=100k**		**FEN=500k**	
	Av.	**SD**	**Av.**	**SD**	**Av.**	**SD**
NDP	124.1	21.4	114.0	18.0	117.3	15.2
HD	0.242	0.009	0.225	0.007	0.215	0.005
A	39.3	11.4	77.8	21.2	133.7	28.6
OS	0.234	0.036	0.318	0.032	0.386	0.028
$CL_{1/25}$	4.02	0.75	3.20	0.46	2.93	0.36
$CL_{1/100}$	1.46	0.15	1.31	0.09	1.26	0.06
$CO_{1/25}$	31.1	2.8	35.7	2.3	40.1	2.1
$CO_{1/100}$	84.7	8.6	86.8	8.8	93.1	8.9

Parametric study for increasing FEN (Table 4-6) is actually equivalent to a study for increasing outer loop number (i.e. evaluating more geometry, thrust and initial condition variables), since inner loop number is kept constant. NDP values for this problem do not show an apparent improvement or deterioration as FEN increases, the average and SD values are not much different. Both average and SD values improve for HD and OS. Accuracy also improves with increasing FEN since average A metric is improving together with a slight improvement in the ratio of SD

to average. Since NDP values are close, CL metrics can be compared. All the CL and CO metrics are improving too. Hence, more FEN apparently improves the quality of the front. However, it dramatically increases the computational cost because of the reason described in Section 3.2.

Table 4-7 Parametric Study for Inner Loop Number (FEN = 100k)

	FEN_{inner} = 200		**FEN_{inner} =1000**		**FEN_{inner} = 5000**	
	Av.	**SD**	**Av.**	**SD**	**Av.**	**SD**
NDP	98.3	10.1	114.0	18.0	188.0	43.8
HD	0.214	0.002	0.225	0.007	0.232	0.008
A	124.9	20.0	77.8	21.2	65.9	20.1
OS	0.392	0.021	0.318	0.032	0.295	0.038
$CL_{1/25}$	2.57	0.30	3.20	0.46	5.19	1.12
$CL_{1/100}$	1.20	0.06	1.31	0.09	1.71	0.27
$CO_{1/25}$	38.4	2.2	35.7	2.3	36.2	2.5
$CO_{1/100}$	82.0	6.4	86.8	8.8	108.6	10.7

Increasing inner loop number (Table 4-7) means that less geometry, thrust and initial condition parameters and more control parameters are evaluated in the course of algorithm. Average values of NDP improve as inner loop number increases, however, the ratio of SD to average also increases. Hence no significant difference is observed between 200 and 1000 inner loops but for 5000 inner loops the difference in NDP is obvious, the increase in SD to average ratio is inferior in this case. HD, A and OS deteriorate as inner loop number increases. Since NDP values for 200 and 1000 inner loops are close, CL metrics can be compared. More clustering (CL) is observed for 1000 inner loops compared to 200 inner loops. For 5000 loops NDP value is much different, hence CL metrics are not conclusive. Numbers of occupied cells for 25x25 grid are close for all the three inner loop numbers. For 100x100 grid they are close for 200 and 1000 inner loops, and they improve for 5000 inner loops. It should be noted here that less inner loops means

more outer loops for a constant FEN, and this leads to more algorithm run time because of the reason described in Section 3.2.

Table 4-8 Parametric Study for Fitness Function Number

	FFN=2		**FFN=50**		**FFN=250**	
	Av.	**SD**	**Av.**	**SD**	**Av.**	**SD**
NDP	43.4	12.5	79.0	11.3	114.0	18.0
HD	0.258	0.009	0.223	0.007	0.225	0.007
A	50.3	15.4	72.2	14.0	77.8	21.2
OS	0.339	0.037	0.333	0.033	0.318	0.032
$CL_{1/25}$	1.82	0.34	2.32	0.32	3.20	0.46
$CL_{1/100}$	1.18	0.10	1.17	0.06	1.31	0.09
$CO_{1/25}$	23.5	3.9	34.1	2.2	35.7	2.3
$CO_{1/100}$	36.5	8.9	67.2	7.0	86.8	8.8

NDP is improved as more fitness function is used (Table 4-8). HD is better when 50 FF's are used instead of 2, but no significant difference is observed when FFN is increased to 250. Actually, parametric studies presented in [24] showed that elliptic FF's may not be so successful in terms of HD for convex problems. Accuracy shows a similar behaviour with HD. It improves for 50 FF's, but even though the average value improves for 250 FF, the ratio of SD to average deteriorates, hence no significant accuracy difference occurs between 50 and 250 FF's. OS metric deteriorates as FFN increases. Since NDP values are significantly different, CL metrics are not conclusive. Number of occupied cells for both 25x25 grids and 100x100 grids improve, hence clustering is reduced with increasing FFN. In the light of these results and the parametric studies of [24], it may be more effective to use linear FF's instead of elliptic ones for this problem (Range vs. Flight Time) since it is a quite convex one.

Table 4-9 Parametric Study for Restart Coefficient

	RC=0.000025		**RC=0.005**		**RC=1**	
	Av.	**SD**	**Av.**	**SD**	**Av.**	**SD**
NDP	146.3	16.4	114.0	18.0	109.5	13.1
HD	0.204	0.006	0.225	0.007	0.217	0.003
A	220.8	25.0	77.8	21.2	120.9	30.5
OS	0.441	0.022	0.318	0.032	0.373	0.023
$CL_{1/25}$	3.42	0.48	3.20	0.46	2.84	0.32
$CL_{1/100}$	1.34	0.07	1.31	0.09	1.25	0.08
$CO_{1/25}$	42.9	2.0	35.7	2.3	38.6	2.0
$CO_{1/100}$	109.1	7.2	86.8	8.8	87.7	7.1

In Table 4-9, RC=0.000025 means restart process is applied if no record is found for only 3 funcion evaluations, and RC=1 means it is never applied. NDP does not change very much for RC=0.005 and RC=1 since the average decreases but ratio of SD to average decreases too. It improves when more frequent restart is applied. For intermediate RC; HD, A and OS take their worst values, very frequent restart is the best in terms of these metrics. Since NDP of RC=0.000025 is significantly different, CL metrics are not conclusive, but RC=0.005 and RC=1 can be compared. CL metrics both are reduced when no restart is applied. In terms of CO metrics, RC=0.005 and RC=1 are close to each other whereas RC=0.000025 is the best. It seems that very frequent restart generates the front that has the best quality in terms of most metrics.

CHAPTER 5

CONCLUSION

In this thesis, a recent SA algorithm called MC-MOSA was modified, applied to several multiobjective missile design optimization problems and the results are presented. Even though the baseline missile in this study was a surface to surface / air to surface anti-tank missile, the tool developed can be applied easily to any missile or rocket conceptual design optimization problem which aims to optimize basic performance parameters such as range, flight time and hit angle, and for which assumptions of 2.2 apply. As a future work, the simulation tool can be developed further by increasing the DOF and including seeker, guidance and target models, enabling the analysis of some other design objectives such as hit accuracy.

One of the major aims in the development of the optimization tool was to run it interactively with DATCOM aerodynamic prediction tool so that a new aerodynamic database is generated automatically in each function evaluation loop. The major difficulty in this approach was the resulting long run times of the algorithm. An investigation on a simple function evaluation loop (covering all simulation and fitness function evaluation loops) showed that more than 90 percent of a typical loop is spent by DATCOM. Two-loop optimization greatly improved the run times since DATCOM is called only in the outer loop and the resulting database is used in all of the corresponding inner loops. Also, based on an observation on DATCOM databases, 2 databases were generated instead of 3 by making a justified assumption. This further improved the run times. In future works,

a less time-consuming aerodynamic prediction subroutine may be developed. Some studies on rocket/missile aerodynamic prediction given in [13, 14, 15] may be used for this purpose.

The main highlights of the original MC-MOSA were carried to these studies, namely, the adaptive cooling schedule, elliptic/ellipsoid fitness functions and individual cooling schedules for FF's. Moreover, several modifications were made to lessen the initial effort prior to the algorithm runs and to have a more automated and reliable code:

- All maximization problems were converted into minimization problems and all objective values including feasible space boundaries were normalized with respect to reference solutions so that normalized solutions always lie in the interval [0,1]. This prevented any sign confusion during the development of the modified code.

- Estimation bounds were calculated within the code, using feasible space bounds. They were further updated to guarantee that current best semi-major value lies between these bounds.

- When no record is found for a certain percent of allowed number of function evaluations, restart was performed.

- Replacement method shown in Figure 3-5 were employed for constraint handling, which is an effective way for evaluation of constraint boundaries for problems in which design variable bounds are squeezed as described in Section 3.2.3. In other problems, however, one has to stick to constraint augmentation to the objective function with penalty coefficients, or develop algorithms based on methods proposed in Section 3.2.3, despite their disadvantages.

- An initial temperature prediction method was used, even though it has minor effect in the course of optimization thanks to adaptive cooling.

A total of 40 design variables were employed, which can be grouped as geometry, thrust, initial condition and control variables. For such a high dimensional design space and highly nonlinear optimization problems, the algorithm proved effective in generating Pareto fronts with fairly good number of points in most of the problems. The effectiveness of the algorithm was further investigated by means of parametric studies and the effects of 4 optimization-related parameters were demonstrated by means of some quality metrics.

REFERENCES

1. Avcioglu, H.T., "A Tool for Trajectory Planning and Performance Verification Of Cruise Missiles", MS Thesis, Middle East Technical University Libraries, Ankara, Turkey, 2000.

2. Belisle, C.J.P., Romeijin, H.E., Smith, R.L., "Hide-and-Seek: A Simulated Annealing Algorithm for Global Optimization" Technical Report, Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI, No: 90-25, 1990.

3. Chin, S.S., "Missile Configuration Design", McGraw-Hill, 1961

4. Corona, A., Marchesi, M., Martini, C., Ridella, S., "Minimizing Multimodal Functions of Continuous Variables with the Simulated Annealing Algorithm", ACM Trans. Math. Softw. 13(3), 262-280, 1987.

5. Deb, K., "Multiobjective Optimization Using Evolutionary Algorithms", Wiley, New York, 2001.

6. Deb, K., Chaudhuri, S., Miettinen, K., "Estimating Nadir Objective Vector Quickly Using Evolutionary Approaches", Genetic And Evolutionary Computation Conference, Seattle, Washington, USA, 643-650, 2006.

7. Edgeworth, F.Y., "Mathematical Psychics; an Essay on the Application of Mathematics to Moral Sciences", C.K. Paul and Co., London (Reprints of Economic Classics, A.M. Kelley, New York, 1961), 1891.

8. Hajek, B., "Cooling Schedules for Optimal Annealing", Math. Oper. Res. 13(2), 311-329, 1988.

9. Hajela, P., "Nongradient Methods in Multidisciplinary Design Optimization-Status and Potential", J. Aircraft. 36(1), 255-265, 1999.

10. Kirkpatrick, S., "Optimization by Simulated Annealing - Quantitative Studies", J. Stat. Phys. 34, 975-986, 1984.

11. Kirkpatrick, S., Gelatt, C.D., Vechi, M.P., "Optimization by Simulated Annealing", Science 220(4598), 671-680, 1983.

12. Knowles, J., Corne, D., “On Metrics Comparing Non-dominated Sets”, In: Proceedings of the 2002 Congress on Evolutionary Computation, pp. 711–716. IEEE Press, Piscataway, NJ, 2002.

13. Lesieutre, D., Love, J., Dillenius, M., “High Angle of Attack Missile Aerodynamics Including Rotational Rates - Program M3HAX”, Atmospheric Flight Mechanics Conference, San Diego, CA, AIAA 96-3392, Jul. 1996.

14. Lesieutre, D., Love, J., Dillenius, M., “Predicition of the Nonlinear Aerodynamic Characteristics of Tandem-Control and Rolling-Tail Missiles”, Atmospheric Flight Mechanics Conference, Monterey, CA, AIAA 2002-4511, Aug. 2002.

15. Lesieutre, D., Love, J., Dillenius, M., Blair, Jr., A.B., “Recent Applications and Improvements to the Engineering-Level Aerodynamic Prediction Software MISL3”, 40th Aerospace Sciences Meeting and Exhibit, Reno, NV, AIAA 2002-0275, Jan. 2002.

16. Lu, P., Khan, M. A., “Nonsmooth Trajectory Optimization: An Approach Using Continuous Simulated Annealing,” Journal of Guidance, Control, and Dynamics, Vol. 17, No. 4, pp. 685-691, 1994.

17. MIL-HDBK-1211(MI), “Missile Flight Simulation Part One: Surface to Air Missiles” 17 July 1995.

18. Miettinen, K., “Nonlinear Multiobjective Optimization”, Kluwer Academic Publishers, Boston, 1999.

19. Pareto, V., In: Schwier, A.S., Page, A.N. (eds.) Manual of Political Economy (translated from1927 French edition by Ann S. Schwier) Augustus M. Kelley Publishers, New York, 1971.

20. Romeijn, H.E., Smith, R.L., Simulated Annealing for Constrained Global Optimization”, J. Glob. Optim. 5, 101-126, 1994.

21. Siarry, P., Berthian, G., Durbin, F., Hamesy, J., “Enhanced Simulated Annealing for Globally Minimizing Functions of Many Continuous Variables”, ACM Trans. Math. Softw. 23(2), 209-228, 1997.

22. Sutton, G.P., Biblarz, O., “Rocket Propulsion Elements”, Edition: 7, Wiley-IEEE, 2000.

23. Tekinalp, O., Bingol, M., “Simulated Annealing for Missile Optimization: Developing Method and Formulation Techniques”, Journal of Guidance, Control and Dynamics, Vol. 27, No. 4, Jul-Aug 2004.

24. Tekinalp, O., Karsli, G., "A New Multiobjective Simulated Annealing Algorithm", Journal of Global Optimization, Vol. 39, No. 1, 49-77, Sep. 2007.

25. Tekinalp, O., Utalay, S., "Simulated Annealing for Missile Trajectory Planning and Multidisciplinary Missile Design Optimization", AIAA Paper 2000-0684, Jan. 2000.

26. Vanderbilt, D., Lougie, S.G., "A Monte Carlo Simulated Annealing Approach to Optimization over Continuous Variables", J. Comput. Phys. 56, 259-271, 1984.

27. White, S.R., "Concepts of Scale in Simulated Annealing", Proc. IEEE Int. Conf. Computer Design, 646-651, 1984.

28. Wu, J., Azarm, S., "Metrics for Quality Assesment of a Multiobjective Design Optimization Solution Set", ASME J. Mech. Des. 123, 18–25, 2001.

29. Zitzler, E., "Performance Assessment of Multiobjective Optimizers: An Analysis and Review", IEEE Trans. Evol. Comput. 7(2), 117–137, 2003.

30. "AGM-114 HELLFIRE - Jane's Air Forces News", http://www.janes.com/defence/air_forces/news/jalw/jalw001013_1_n.shtml, Last Update: 13.03.2009, Last Accessed: 13.03.2009

31. "AGM-114 Hellfire", http://www.fas.org/man/dod-101/sys/missile/agm-114.htm, Last Update: 30.05.2008, Last Accessed: 13.03.2009

32. "Boeing/Lockheed Martin AGM-114 Hellfire", http://www.designation-systems.net/dusrm/m-114.html, Last Update: 25.07.2007, Last Accessed: 13.03.2009

33. "Missile - Wikipedia, the free encyclopedia", http://en.wikipedia.org/wiki/Missile, Last Update: 12.03.2009, Last Accessed: 13.03.2009

34. "The Standard Atmosphere", http://www.atmosculator.com/The Standard Atmosphere.html, Last Update: 13.03.2009, Last Accessed: 13.03.2009

35. "Topic 14.5: Runge Kutta Fehlberg (Theory)", http://www.ece.uwaterloo.ca/~ece204/TheBook/14IVPs/rkf45/theory.html, Last Update: 13.03.2009, Last Accessed: 13.03.2009

APPENDIX A

SAMPLE MISSILE CONFIGURATIONS

Table A.1 Sample Pareto-Optimal Design Variables and Corresponding Solutions for Case Studies

	CS#1	CS#2	CS#3	CS#4	CS#5	Unit
t_b	3.05	3.01	3.03	-	2.82	s
T_d	7263.42	7508.99	7328.01	-	7706.84	N
B_{nose}	6.97	6.99	7.62	6.12	7.59	cm
L_{aft}	6.03	6.19	4.54	9.09	5.06	cm
$b_{1,2}$	4.98	5.39	5.00	4.84	5.37	cm
l_{max1} / c	0.153	0.144	0.148	0.167	0.148	-
$c_{1,1}$	12.93	12.80	12.51	12.66	12.56	cm
$c_{1,2}$	7.68	7.82	7.27	7.73	7.25	cm
t_1 / c	0.0146	0.0119	0.0153	0.0128	0.0155	-
$\Lambda_{te,1}$	-1.66	-2.74	-1.48	-4.745	-2.34	deg
$x_{le,1}$	30.94	29.33	29.89	31.31	29.83	cm
$b_{2,2}$	7.60	7.66	7.42	9.13	7.70	cm
l_{max2} / c	0.158	0.148	0.159	0.143	0.159	-
$c_{2,1}$	50.60	46.39	48.90	46.81	51.46	cm
$c_{2,2}$	33.84	32.19	33.54	33.31	31.89	cm
t_2 / c	0.0155	0.013	0.0142	0.0169	0.0158	-
$x_{le,2}$	92.70	90.80	96.93	92.78	95.86	cm
$b_{3,2}$	5.57	5.31	5.78	5.65	5.31	cm
$c_{3,1}$	7.65	7.95	7.69	7.96	7.71	cm
$c_{3,2}$	7.62	7.09	7.69	7.59	7.28	cm
$c_{3,3}$	5.79	4.28	4.38	4.82	5.08	cm
δ_{lim}	-24.80	-24.87	-24.87	-26.54	-25.24	deg
z_0	40.93	42.13	46.60	-	58.35	m
V_0	9.84	9.20	10.41	-	8.48	m/s
γ_0	7.39	7.17	7.34	-	7.79	deg
α_0	1.184	1.75	0.911	-	1.274	deg

Table A.1 (continued) Sample Pareto-Optimal Design Variables and Corresponding Solutions for Case Studies

	CS#1	CS#2	CS#3	CS#4	CS#5	Unit
τ	0.447	0.372	0.484	-	0.473	s
e_{per}	0.923	0.902	0.899	-	0.900	-
α_{hit}	-16.82	-7.80	-22.18	-	-17.01	deg
z_{hit}	120.01	110.02	185.06	-	187.16	m
$\alpha_{node,2}$	2.86	4.62	1.66	-	4.37	deg
$\alpha_{node,3}$	1.83	3.19	2.61	-	3.96	deg
$\alpha_{node,4}$	1.55	3.61	2.32	-	3.27	deg
$\alpha_{node,5}$	2.15	2.98	2.36	-	3.26	deg
$\alpha_{node,6}$	2.37	2.68	1.71	-	3.05	deg
$\alpha_{node,7}$	2.42	2.76	2.37	-	3.39	deg
$\alpha_{node,8}$	1.15	4.58	1.81	-	3.32	deg
$\alpha_{node,9}$	1.31	3.61	0.59	-	4.28	deg
$\alpha_{node,10}$	2.96	2.90	2.43	-	1.14	deg
$\alpha_{node,11}$	2.75	3.07	1.62	-	1.93	deg
R	8028.6	8483.2	-	-	8128.1	m
t	34.1	-	-	-	49.9	s
θ_{hit}	-	-	70.7	-	-	deg
V_{hit}	-	202.7	230.2	-	192.7	m/s
$C_{L,trim}$	-	-	-	0.405	-	-
$C_{D,trim}$	-	-	-	0.227	-	-

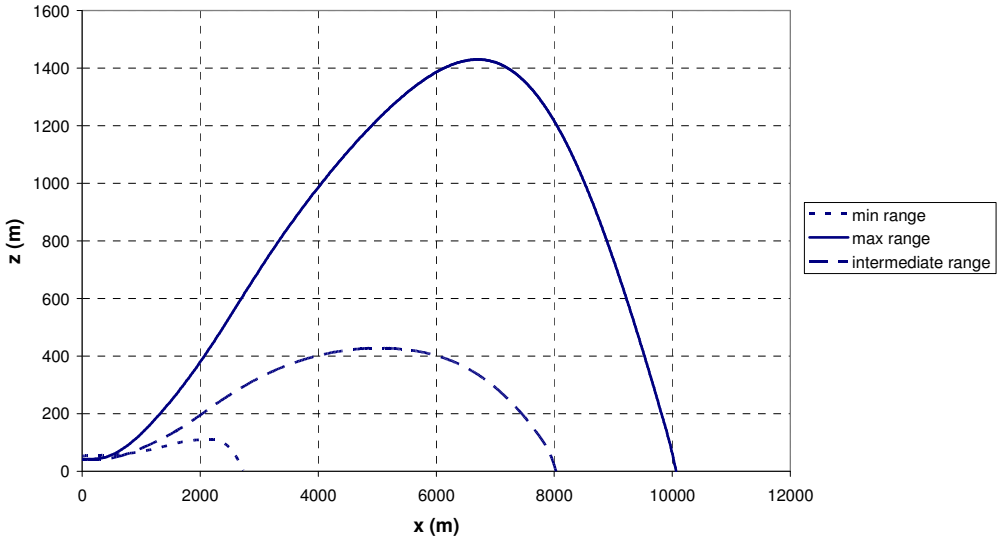


Figure A.1 Trajectories for Sample Missile Configurations of CS#1

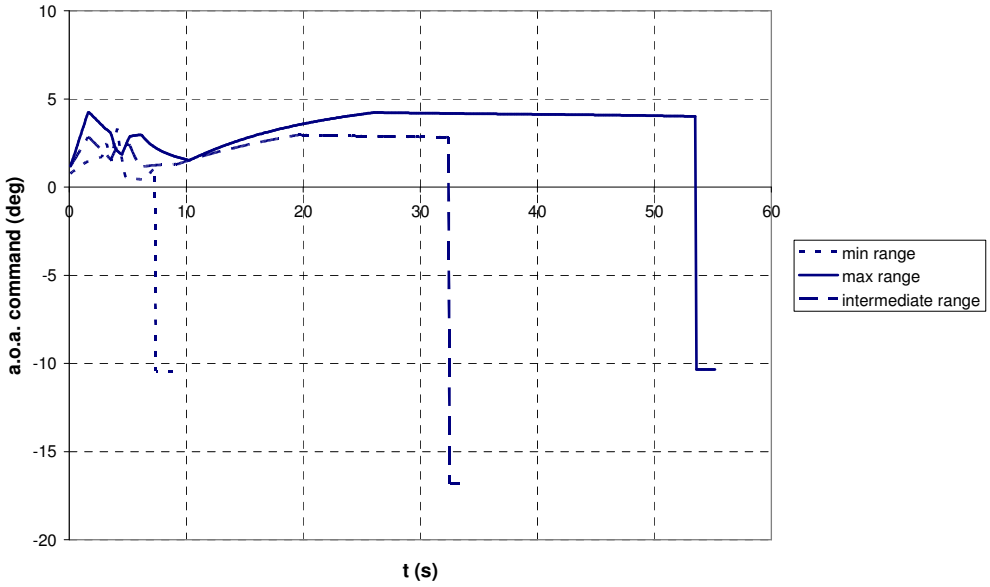


Figure A.2 A.o.a. Command Profiles for Sample Missile Configurations of CS#1

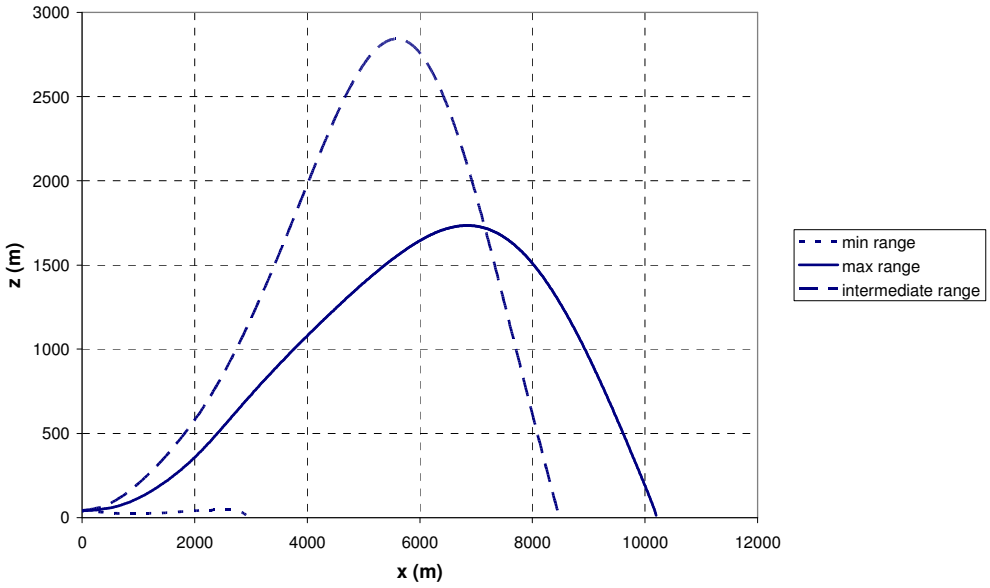


Figure A.3 Trajectories for Sample Missile Configurations of CS#2

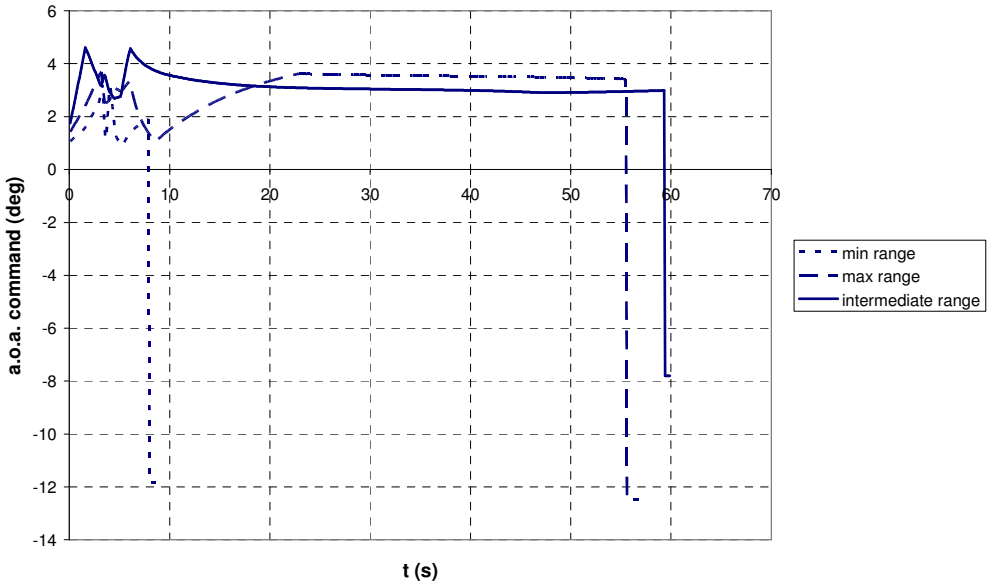


Figure A.4 A.o.a. Command Profiles for Sample Missile Configurations of CS#2

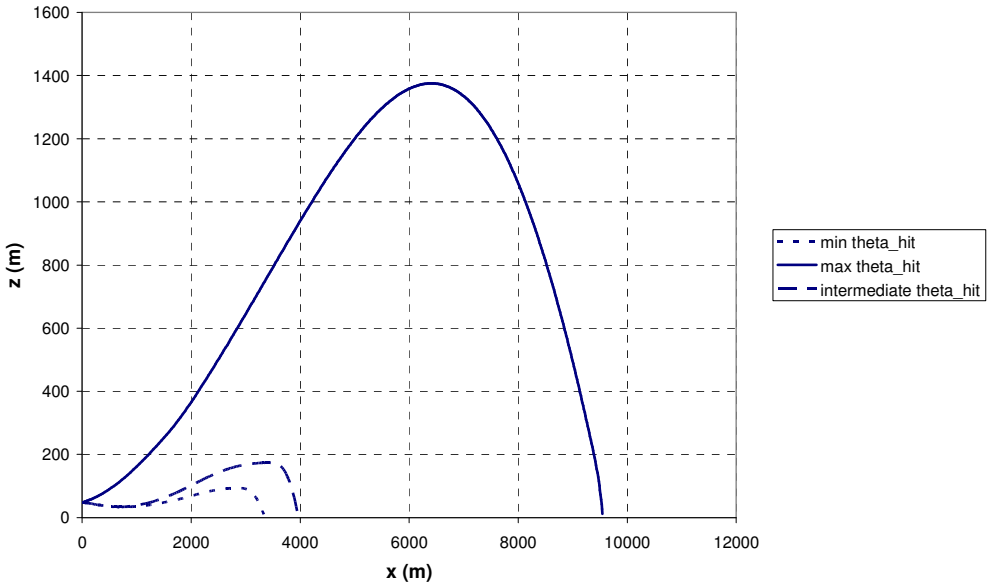


Figure A.5 Trajectories for Sample Missile Configurations of CS#3

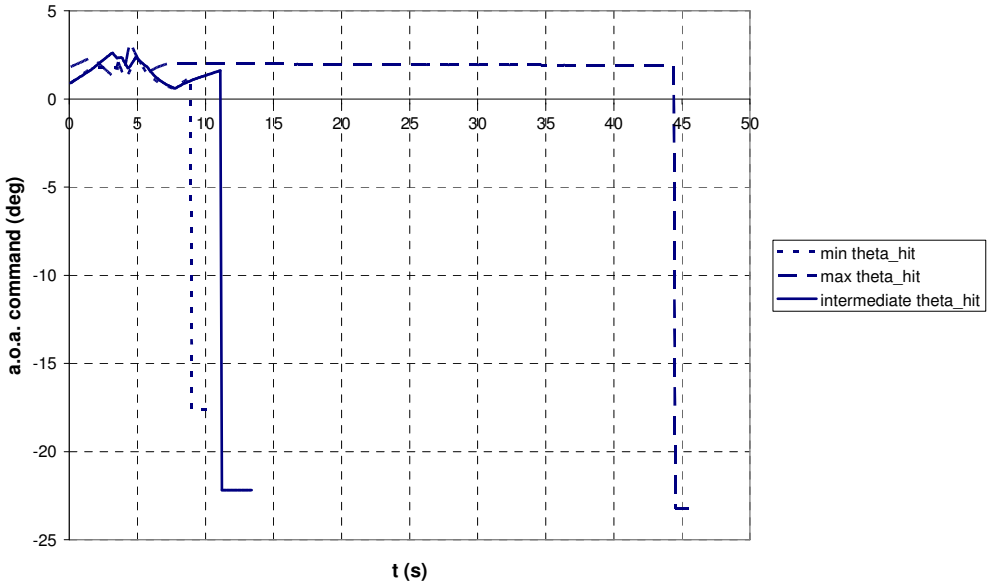


Figure A.6 A.o.a. Command Profiles for Sample Missile Configurations of CS#3

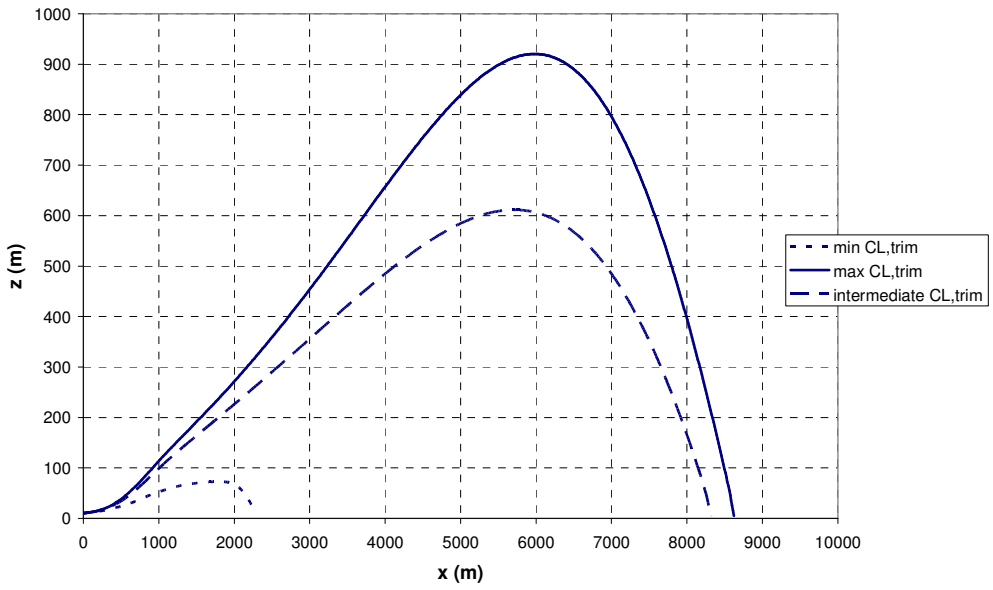


Figure A.7 Trajectories for Sample Missile Configurations of CS#4

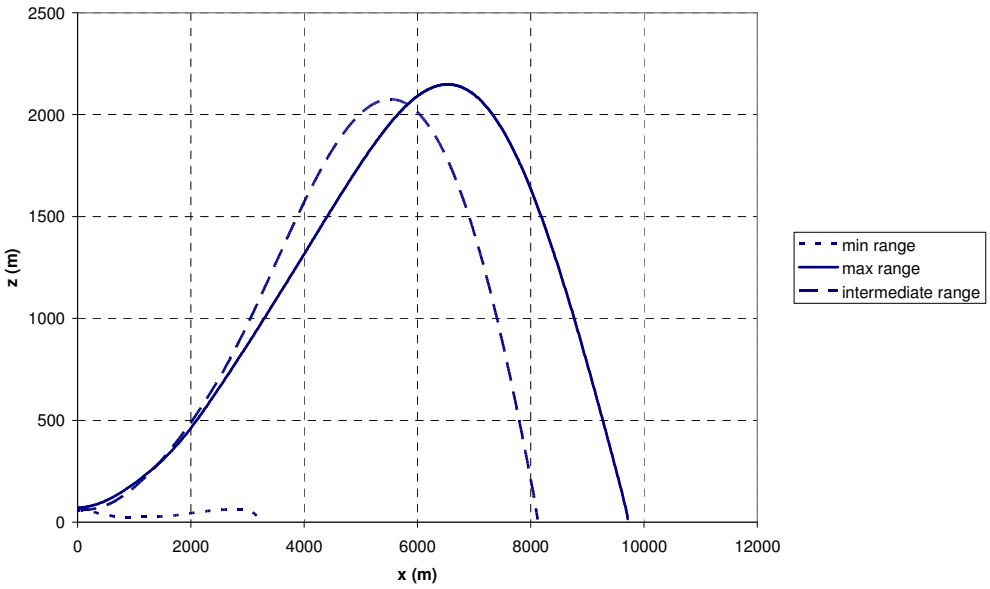


Figure A.8 Trajectories for Sample Missile Configurations of CS#5

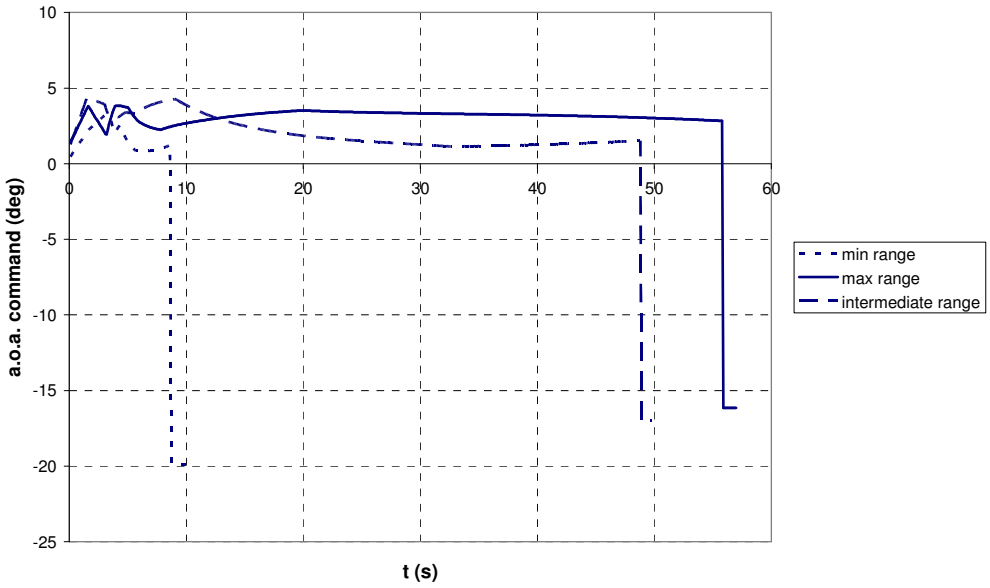


Figure A.9 A.o.a. Command Profiles for Sample Missile Configurations of CS#5

APPENDIX B

QUALITY METRICS

In this thesis, the quality assessment of the frontier obtained through multiobjective optimization is conducted using four metrics proposed in [24, 28]. They are: the hyper-area difference (HD), overall Pareto spread (OS), accuracy of the observed Pareto frontier (AC), cluster (CL).

The hyperarea difference is the area below the Pareto frontier, as shown in Figure B.1(a). Points A and B defines the bounding box around the Pareto front. Normally, HD shall be calculated using normalized objectives. Then, in general, it is claimed that the smaller the HD metric is, the better the observed Pareto solution set [28]. Overall Pareto spread (OS) is the area of the maximum rectangle constructed using the two extremes of the Pareto front (p_1 and p_2) as shown in Figure B.1(b). Again A solution set with the largest OS value is generally an indication that a particular front has spread to the extreme ends of the Pareto front, and consequently, it is comparatively better than a front with a smaller value. Accuracy (A) is a measure how smooth the observed front is. To obtain this metric, areas of the small rectangles constructed from neighboring solutions are summed up (Figure B.1(c)) to obtain a total area. The metric is defined as the inverse of this total area. If the solution set contains all the actual Pareto solutions (i.e., a continuous Pareto frontier), then the total area will be zero, causing the A metric to be infinite. Thus, a solution set with a large A value is better than the one with a smaller A value. It is desirable to have the solutions spread uniformly along the front. Clustering occurs, when too many solutions are found at certain parts of the front, while other parts are empty. To obtain the CLμ metric, the whole domain normalized objectives is

divided into square grids of size μ. Then, those rectangles occupied with a non-dominated solution are counted. The total number of non-dominated solutions in the set is divided to the number of occupied rectangles. Ideally, to have a good spread, each rectangle shall be occupied by a single solution giving a CLμ metric equal to one. For example, in Figure B.1(d) there are four solutions in the front, while only three grids are occupied (i.e., CLμ = 1.25). Similarly, of the two solution sets having almost equal number of solutions, the one with a smaller CLμ metric shall be preferred. However, this comparison may not be meaningful if each set contains very different number of solutions. For this reason, in this thesis the number of cells occupied by a non-dominated solution (COμ) is also used as a metric. In this case, the greater the COμ value, the better the solution is.

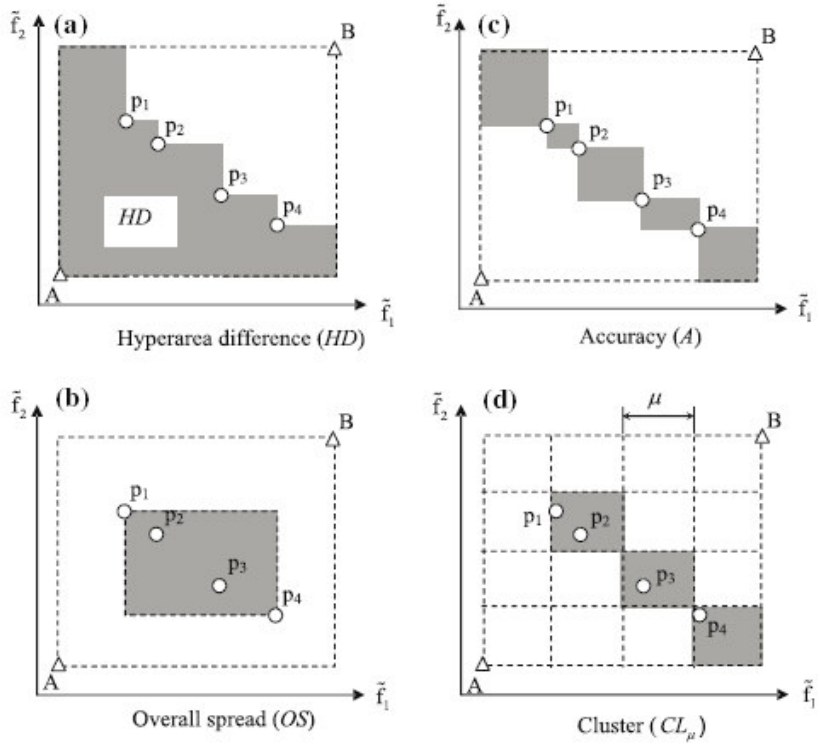


Figure B.1 Geometric Description of the Metrics Used in this Thesis [24, 28] with Normalized Objectives