Abstract

GAEBLER, JOHN ALEXANDER. Failure Bounding and Sensitivity Analysis Applied to Monte Carlo Entry, Descent, and Landing Simulations. (Under the direction of Dr. Robert Tolson).

In the study of entry, descent, and landing (EDL) scenarios, Monte Carlo sampling methods are often employed to study uncertainties in the designed trajectory. The large number of uncertain inputs and outputs, coupled with complicated non-linear models, can make interpretation of the results difficult. Often it is desirable to reduce the uncertainty of an output or to understand why a failure occurred. Methods are sought that can provide this information, thereby increasing the value of performing Monte Carlo analyses. The specific insights desired are the statistics of the inputs causing failure, the sensitivity of failed cases to input statistics, and the sensitivities of output statistics to input statistics.

Three methods that provide statistical insights were identified and applied to both a simple projectile trajectory simulation and an EDL simulation. The projectile trajectory was included to help understand and describe the methods. To test the merits of the methods a two dimensional ballistic EDL simulation was developed. The EDL simulation included a temperature varying atmosphere model, a rotating atmosphere with wind, and correlated entry states. This simulation had seventeen statistical inputs composed of initial states, atmospheric parameters, and vehicle properties.

The first technique studied was failure domain bounding. If during an analysis of a large sample set a case fails, i.e. by consuming all the propellant before engine shut down, it is imperative to understand the cause. This method identifies an upper bound on the failure region by utilizing an optimizer to locate the most probable failed case in the design space. With this knowledge, randomly generated cases within the complement safe region can be assumed successful, thus reducing the number of cases that need to be simulated when studying failure. This allows the generation of more failed cases for study which increases the accuracy of the failure probability approximation with less computational expense.

Next a global variance-based sensitivity analysis developed by I.M. Sobol was tested. The sensitivities provided are based on how the total variance of the output can be segmented into components due to individual or combinations of inputs. This knowledge allows an engineer to identify which input will have the greatest impact on reducing the variance, or

uncertainty, on an output. This method has the additional benefit of identifying which inputs are interacting, or coupling.

Finally a method that provides local probabilistic sensitivities was studied. These are sensitivities of an output mean or variance to an input mean or variance. The information provided is the same as approximating the partial derivative of the output statistics with respect to the input statistics by finite differencing. Instead Leibniz's rule is introduced to approximate certain partial derivatives with the benefit of requiring fewer simulations than finite differencing. These benefits are realized when calculating the sensitivities to inputs with infinite bounds on their probability density function.

The advantages and disadvantages of each method are discussed in terms of the insights gained versus the computational cost. Models with fewer input dimensions and small probabilities of failure can benefit from application of the failure domain bounding method. Variance-based sensitivities give many statistical insights, but potentially at high computational cost. Finally if the inputs are uncorrelated with probability density functions having infinite bounds, the probabilistic sensitivity analysis gives statistical insights without requiring many simulations.

Failure Bounding and Sensitivity Analysis Applied to Monte Carlo Entry, Descent, and Landing Simulations

by
John Alexander Gaebler

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the degree of
Master of Science

Aerospace Engineering

Raleigh, North Carolina

2009

APPROVED BY:

_____	_____
Dr. Fred DeJarnette	Dr. Andre Mazzoleni

_____
Dr. Robert Tolson
Committee Chair

Biography

John Alexander Gaebler was born in Glen Cove, New York and attended public schools in Deltona, Florida. He received a Bachelor of Science degree in Aerospace Engineering from the University of Florida (UF) in May 2007. At UF he participated in the Small Satellite Design Club, designing and testing a release mechanism for a CubeSat. In his last year he was an undergraduate research assistant for a doctoral candidate in the combustion laboratory. There he assisted in the experimental setup and imaging of supercritical jet interactions using planar laser induced fluorescence.

From May 2007 to November 2009, he was a graduate research assistant at the National Institute of Aerospace and NASA Langley Research Center in Hampton, Virginia. There he studied for a Master of Science degree in Aerospace Engineering from North Carolina State University. The first project undertaken was the study of inertial measurement unit data from the aero-breaking phase of the Mars Reconnaissance Orbiter to estimate atmospheric density at high altitudes. For a course project he helped design a reusable lunar transportation architecture utilizing orbital propellant depots. The design took first place in the graduate division of the Revolutionary Aerospace System Concepts-Academic Linkage (RASC-AL) 2009 design competition. His second research project involved identifying methods to gain statistical insights from Monte Carlo entry, descent, and landing simulations for the Atmospheric Flight and Entry Systems Branch at NASA Langley Research Center. Upon graduation he is taking a position as an aerospace engineer at NASA Goddard Space Flight Center in the Navigation and Mission Design Branch.

Acknowledgments

There are many individuals who deserve recognition for helping me in this research. First and foremost is my advisor Dr. Robert Tolson. I cannot overstate how thankful I am for the chance to draw from his knowledge and experience. Working under his tutelage at the National Institute of Aerospace has been a wonderful experience and a great stepping stone into the NASA community. There I was introduced to my NASA mentor Dr. Juan R. Cruz, whom I thank for his invaluable guidance and support. I would also like to thank the Atmospheric Flight and Entry Systems Branch for the opportunity to perform research that can help with future planetary exploration mission design.

I am grateful to Dr. Fred DeJarnette and Dr. Andre Mazzoleni for agreeing to serve on my committee. There are several engineers at NASA who deserve thanks for helping me along the way. I thank Dr. Lucas Horta for introducing me to sensitivity analysis. Finally many thanks to Dr. Luis Crespo, Dr. Daniel Giesy, and Dr. Sean Kenny for taking the time to discuss their work.

Table of Contents

List of Tables v
List of Figures vi
List of Symbols vii
List of Acronyms x
1 Introduction 1
1.1 Current Methods 1
1.2 Project Goals 2
1.3 Approach 3
2 Projectile Trajectory 4
2.1 Equations 4
2.2 Input Dispersions 5
2.3 Output Dispersions 5
2.4 Failure Definitions 6
3 EDL Simulation 8
3.1 Concept of Operation 8
3.2 Basic Assumptions 9
3.3 Equations of Motion 10
3.4 Models 11
3.5 Perturbed Inputs 12
3.6 Tracked Output 13
4 Failure Domain Bounding 17
4.1 Method Description 17
4.2 Application to Projectile Trajectory 20
4.3 Application to EDL Simulation 26
5 Global Variance-Based Sensitivities 31
5.1 Method Description 31
5.2 Application to Projectile Trajectory 35
5.3 Application to EDL Simulation 40
6 Probabilistic Sensitivity Analysis 44
6.1 Method Description 44
6.2 Application to Projectile Trajectory 46
6.3 Application to EDL Simulation 50
7 Conclusions 56
References 58
Appendices 60
Appendix A: Generating Correlated Entry States 61
Appendix B: Derivation of Sobol Global Sensitivities 64
Appendix C: Derivation of Probabilistic Sensitivities via Leibniz's Rule 70
Appendix D: Equations for Relevant Probability Density Functions 74

List of Tables

Table 3.1: EDL simulation events and corresponding triggers..... 9
Table 3.2: Perturbed inputs to the EDL simulation 13
Table 4.1: Hypersphere radius found for each constraint in the projectile trajectory..... 21
Table 4.2: Hypersphere radius found for each constraint in the EDL simulation 27
Table 5.1: Final position sensitivities by Sobol's method for projectile trajectory 39
Table 6.1: Sensitivities to projectile trajectory outputs via Leibniz's rule 47

List of Figures

Figure 2.1: Projectile trajectory 4
Figure 2.2: Projectile trajectory output histograms..... 6
Figure 2.3: Inputs causing failure: blue are successful, red are failed..... 7
Figure 3.1: EDL concept of operations..... 8
Figure 3.2: Initial EDL entry states..... 10
Figure 3.3: Nominal trajectory from EDL simulation 14
Figure 3.4: Nominal trajectory, velocity vs. altitude of final descent..... 14
Figure 3.5: Histograms of selected outputs with failure limits..... 15
Figure 4.1: Example of safe region defined by hypersphere radius 18
Figure 4.2: Failure domain bounding flow chart 19
Figure 4.3: Convergence comparison between MCS and FDB with 2000 run limit..... 22
Figure 4.4: Input histograms of cases causing γ_f failure 23
Figure 4.5: Input histogram of cases causing X_f failure..... 24
Figure 4.6: Input histograms of cases causing V_f failure 24
Figure 4.7: Convergence comparison between MCS and FDB for ~0.4% failures..... 25
Figure 4.8: Vector components to the CP of failure on parachute deployment altitude..... 28
Figure 4.9: Hypersphere radius with a non-influential variable 29
Figure 4.10: Convergence of EDL simulation failure probability for MCS vs. FDB 30
Figure 5.1: Visualization of Sobol's method, resampled sets have superscript R..... 33
Figure 5.2: Effect of resampling each input variable on final position 36
Figure 5.3: Residual variance of resampling each input variable on final position..... 37
Figure 5.4: Convergence analysis for sensitivities to main effects..... 38
Figure 5.5: Global sensitivities on parachute deployment altitude..... 40
Figure 5.6: Global sensitivities on percent propellant used..... 41
Figure 5.7: Parachute deployment altitude sensitivity convergence to surface temperature.. 42
Figure 6.1: Typical convergence of PSA sensitivities via Leibniz's rule..... 49
Figure 6.2: Sensitivity of parachute deployment altitude mean to each input mean 50
Figure 6.3: Sensitivity of parachute deployment altitude mean to each input variance 51
Figure 6.4: Sensitivity of parachute deployment altitude variance to each input mean 51
Figure 6.5: Sensitivity of parachute deployment altitude variance to each input variance 52
Figure 6.6: Sensitivity of final longitude mean to each input mean 53
Figure 6.7: Sensitivity of final longitude variance to each input mean 53
Figure 6.8: Sensitivity of final longitude variance to each input variance 54
Figure 6.9: Convergence of parachute deployment altitude to surface temperature 55

List of Symbols

a	sensed acceleration
a	lower bound on probability distribution
a_{deploy}	parachute deployment max acceleration
A	sample set (Sobol)
b	upper bound on probability distribution
B	sample set (Sobol)
$C_{D_{A,D}}$	aeroshell drag coefficient in the descent configuration
$C_{D_{A,E}}$	aeroshell drag coefficient in the entry configuration
C_{D_P}	parachute drag coefficient
C_X	parachute opening load factor
$E[\ ]$	expected value or mean
f_0	mean value of output function (Fourier-Haar series)
$f(\ )$	output function
$\overline{F}_g$	gravitational force
$F_{\parallel V_I}$	forces parallel to velocity vector
$F_{\perp V_I}$	forces perpendicular to velocity vector
$g(\ )$	constraint function
g	acceleration of gravity
G	gravitational constant
h	altitude
h_{const}	trigger height to start constant propulsive descent
h_{deploy}	parachute deployment altitude
$h_{V_{w,iso}}$	altitude at which the constant wind speed region begins
I_{SP}	specific impulse
k	constant in calculation of heat rate
m_E	entry mass
m_v	vehicle mass
M_p	planet mass
n	number of uncertain design inputs
N	number of samples
$\overline{p}$	uncertain design parameter
$p(\ )$	probability density function

$P_{propellant}$	percentage of propellant consumed
P_{rand}	array of independent random numbers
q_{conv}	convective heat load
q_{deploy}	parachute deployment dynamic pressure
$q_{estimate}$	estimated dynamic pressure
r	position magnitude
r_I	inertial radius
$r_{n,a}$	aeroshell nose radius
R_{CP}	hypersphere radius to Critical Point
$\{R\}$	vehicle position vector
S	global sensitivity
$\{S\}$	state vector
S_P	parachute reference area
t	time
T	number of terms sought (Sobol)
T_0	atmospheric surface temperature
T_{57}	atmospheric isothermal temperature
T_{calc}	calculated thrust required
T_{start}	trigger thrust value to start engine
u_i	hypersphere radius vector component
$\overline{u}$	hypersphere radius vector
U	number of inputs with uniform distributions (PSA)
v	velocity magnitude
v_I	inertial velocity
V	variance
V_0	initial velocity
V_f	final velocity
$\{V\}$	vehicle velocity vector
V_{const}	trigger velocity to start constant propulsive descent
V_{rel}	relative velocity
V_{w_o}	wind speed at the planet surface
$V_{w_{iso}}$	wind speed in the constant speed region
x	position along the x axis
x_i	general input variable
X_f	final distance

$\bar{X}$	matrix of input variables
y_0	initial height
y	position along the y axis
$y(\ )$	general output function
z	position along the z axis
γ	flight path angle
γ_0	initial flight path angle
γ_f	final flight path angle
γ_I	inertial flight path angle
$\mathbf{\Gamma}$	covariance matrix
Δ	step size
Δt_{HS}	trigger timer to release heatshield
Δt_{warmup}	trigger timer to warmup engines
θ	statistical parameter (mean or standard deviation)
λ	longitude of vehicle
λ_f	final longitude
λ_I	inertial longitude
μ	mean
ν	variance
ρ	linear correlation coefficient
ρ_0	atmospheric density at the planet surface
σ	standard deviation
φ	rotation matrix

List of Acronyms

2D	Two Dimensional
CP	Critical Point
EDL	Entry, Descent, and Landing
EOM	Equations Of Motion
FDB	Failure Domain Bounding
LaRC	Langley Research Center
MCS	Monte Carlo Sampling
NIA	National Institute of Aerospace
PDF	Probability Density Function
POST	Program to Optimize Simulated Trajectories
PSA	Probabilistic Sensitivity Analysis
RK4	Runge-Kutta 4^{th} order integration

1 Introduction

In the study of Entry, Descent, and Landing (EDL) scenarios, Monte Carlo sampling (MCS) methods are employed to study the effects of input variable uncertainties on trajectory output variables. The MCS method involves generating thousands of random input samples using their estimated probability distributions and assembling them into input cases. These input cases are then run through an EDL simulation yielding numerous output cases, one for each input case. Analyzing and gaining physical insight from these output cases is often an overwhelming task given that there are often many input variables.

Complications arise when there are failed cases that need to be understood. An example of such a failed case may be parachute deployment at too-low an altitude, which may lead to a reduced time between parachute deployment and landing, preventing successful completion of the EDL sequence. With such a large input set, along with the non-linear dynamics involved, it can sometimes prove difficult to identify the cause of a failed case, such as which input variable or combination resulted in failure.

Another situation that often arises is when there is a request for certain mission requirements, such as a smaller landing ellipse, so that a lander can land close to a location of high scientific value. In this case the engineers need to identify which inputs have the greatest influence on the landing ellipse if their uncertainty could be reduced. Running experiments to reduce the uncertainty in an input variable is often a costly endeavor. It is important to identify those input variables where the most benefit can be gained by reducing their uncertainty. Given these issues, methods are sought that can provide insights into the statistical sensitivities of simulation outputs with respect to inputs of the EDL trajectories.

1.1 Current Methods

Engineers at NASA Langley Research Center (LaRC) analyze EDL missions by running the Program to Optimize Simulated Trajectories (POST) [11] with uncertain input variables generated by MCS. These analyses often have hundreds of input variables. The simulation is executed thousands of times so that the uncertainty bounds on each perturbed

input will be well distributed, according to the Probability Density Function (PDF) assumed. Meeting with LaRC engineers active in MCS EDL studies shed light on the current methods used to understand the output uncertainties of the simulations. Mainly statistical plots are used to view the data, while relying on experience to understand the EDL system behavior. The following is a list of current methods for viewing the simulation outputs:

- Statistic quantities (e.g. mean, standard deviation, etc.)
- Histograms
- Percentiles
- Quantile-Quantile plots
- Scatter plots
- Cumulative distributions
- Confidence intervals
- Animations
- Detailed study of individual outlier cases
- Simultaneous animation of numerous trajectories

1.2 Project Goals

Relying on the data presented by the methods listed above is not always sufficient for understanding the simulation outputs. The engineers desire additional information to aid in in the decision making process. A common theme among the desired insights is a sense for the effect a particular input variable has on a particular output variable. Methods were sought that can provide insights into the following goals:

- Statistics of the input variables causing failure
- Sensitivity of failed cases to input variable statistics
- Sensitivity of output variable statistics to input variable statistics

1.3 Approach

First a literature search was conducted, identifying several possible methods to investigate. Methods were chosen for there ability to provide insight into the statistical output from a MCS analysis. Those considered here are Failure Domain Bounding (FDB) [4], Sobol's global variance-based sensitivities [15], and Probabilistic Sensitivity Analysis (PSA) [5]. The FDB method was chosen to generate additional failed cases with fewer simulations than typical MCS which provides an accurate estimate of the probability of failure while providing more failed cases to generate statistics of the inputs causing failure. Sobol's method was investigated to approximate global sensitivities of output variables with respect to input variables. Finally PSA was chosen to approximate local sensitivities of output variable statistics with respect to input variable statistics. A local sensitivity is the sensitivity of an output variable at a specific value (i.e. at a specific statistic such as the mean value), while a global sensitivity is the sensitivity of an output variable over the entire range of values.

First, to gain insight on the usefulness of the methods, a simple example based on a projectile fired from a cannon was tested. The projectile trajectory example has four inputs and three outputs with analytic equations of motion. Next a simplified EDL simulation was developed to test the methods. The simulation is based on the Viking lander mission concept of operations and inputs. It has 17 perturbed inputs and tracked 44 outputs.

This document is structured to first introduce the two example applications: the projectile trajectory, and the EDL simulation. Each method will be introduced, then results from testing on the projectile trajectory will be presented first, followed by the application to the EDL simulation.

2 Projectile Trajectory

A simple simulation was prepared to help understand the methods. The simple model is based on two dimensional (2D) Equations of Motion (EOM) for the flight of a projectile being fired from a cannon. The inputs are: the height of the cannon y_0, the initial velocity of the projectile V_0, the initial flight path angle γ_0, and the acceleration of gravity g. Outputs from the model are the final distance X_f, final flight path angle γ_f, and the final velocity V_f. The initial flight path angle is positive above the horizon as shown in Figure 2.1. This case is sufficiently nonlinear to provide a challenging, yet easily understood problem.

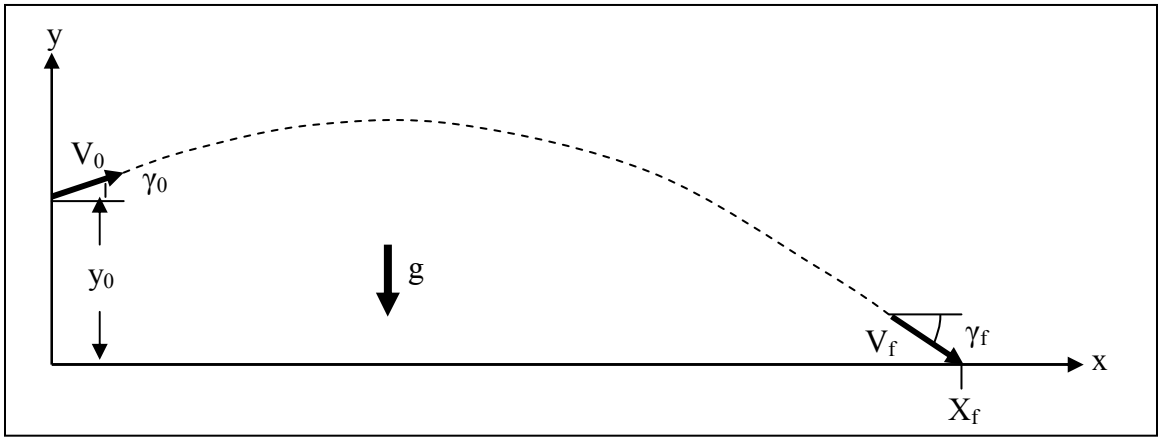


Figure 2.1: Projectile trajectory

2.1 Equations

The EOM for this model are easily developed in 2D as:

$$x(t) = V_0 \cos(\gamma_0) t \tag{2.1}$$

$$y(t) = -\frac{1}{2} g t^2 + V_0 \sin(\gamma_0) t - y_0 \tag{2.2}$$

Analytic solutions can be obtained for each of the outputs as follows:

$$X_f = \frac{V_0 \cos(\gamma_0)}{g}\left(V_0 \sin(\gamma_0) + \sqrt{(V_0 \sin(\gamma_0))^2 + 2gy_0}\right) \tag{2.3}$$

$$V_f = \sqrt{(V_0 \cos(\gamma_0))^2 + \left(-\left(V_0 \sin(\gamma_0) + \sqrt{(V_0 \sin(\gamma_0))^2 + 2gy_0}\right) + V_0 \sin(\gamma_0)\right)^2} \tag{2.4}$$

$$\gamma_f = \tan^{-1}\left(\frac{1}{V_0 \cos(\gamma_0)}\left(-\left(V_0 \sin(\gamma_0) + \sqrt{(V_0 \sin(\gamma_0))^2 + 2gy_0}\right) + V_0 \sin(\gamma_0)\right)\right) \tag{2.5}$$

2.2 Input Dispersions

Each input was assigned a PDF along with the statistics to define it. The acceleration of gravity was assumed to be 10 m/s^2 with a uniform distribution between 9 and 11. Initial height was set to 125 m with a uniform distribution between 120 and 130. The initial velocity was set to a nominal value of 1000 m/s normally distributed with a standard deviation of 100 m/s. Initial flight path angle was set to 0° normally distributed with a standard deviation of 5°. The distribution of g is unrealistic, but was chosen to increase the nonlinearity of the problem. Tight bounds were placed on initial height to make that variable insignificant. Finally the inputs were evenly split between uniform and normal distributions to cover the pertinent PDFs typically encountered in EDL simulations.

2.3 Output Dispersions

To investigate how these input distributions affect the output, 100,000 cases were generated through MCS to produce the histograms on the output variables in Figure 2.2. The top left plot shows that final position has a large number of cases where the projectile did not travel far, this occurs because any negative γ_0 causes the projectile to hit the ground quickly. A positive γ_0 allows the projectile to loft, achieving a large range and resulting in the long tail in the distribution. The top right plot shows that γ_f is one sided since it cannot be positive. Bottom left shows that V_f has a distribution similar to the distribution on the V_0, however it is offset slightly due to the extra height (~125 m) which adds additional energy to

the trajectory. Also in the figure are the nominal values obtained when the mean of each input is run through the simulation.

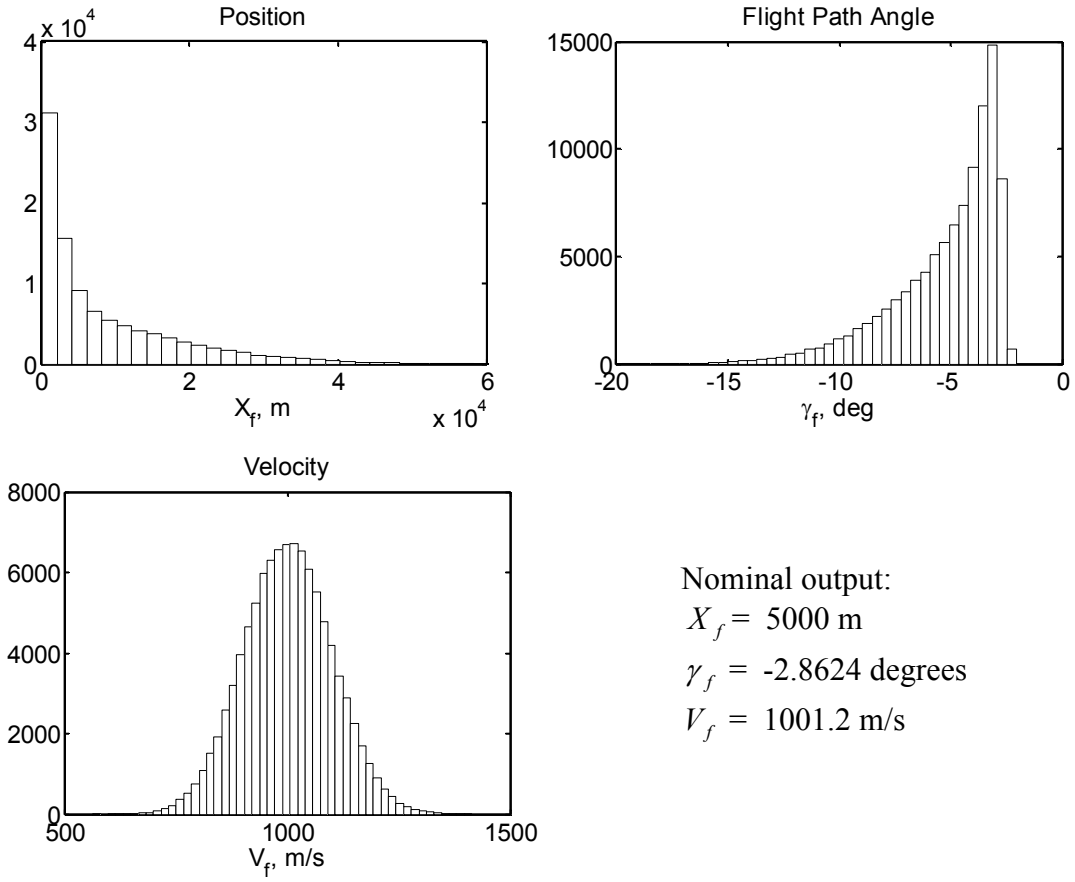


Figure 2.2: Projectile trajectory output histograms

2.4 Failure Definitions

Limits were set on each output beyond which the case is considered to have failed. These limits were arbitrarily set to allow for a failure probability (i.e., any one variable exceeding its limit) of approximately 1.7%. This required limiting the final position to be less than 47,000 m, final flight path angle greater than -14°, and a final velocity less than 1250 m/s. Figure 2.3 shows this failure region as a plot of initial velocity versus initial flight path angle. Successful combinations of these inputs are blue dots, while failures are plotted as red dots.

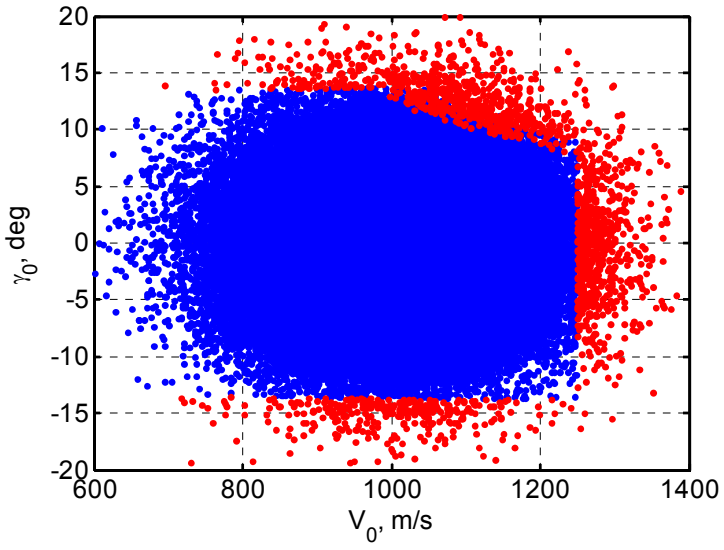


Figure 2.3: Inputs causing failure: blue are successful, red are failed

Analyzing the failure region from plots of the other possible input combinations indicates that y_0 and g have a minor effect on the failure region. Figure 2.3 shows each constraint clearly since the main contributors are γ_0 and V_0. Along the bottom of the plot are failures due to a γ_f less than -14°, along the right hand side are failures due to V_f greater than 1250 m/s. Along the top of the plot are two constraint violations, the horizontal line to the left of V_0 of 1000 m/s is a γ_f violation. To the right of this are failures from a X_f greater than 47,000 m, which is a nonlinear constraint dependent on V_0 and γ_0. Further investigation of this nonlinear constraint shows a dependency on g is present since a low value of g will let a projectile travel farther, increasing the chance of traveling too far.

3 EDL Simulation

A simulation was needed that would be representative of an EDL mission; however it also needed to be simplified so that the results of this study could be understood. The simulation created for the present study is a simplified model of the Viking lander missions encompassing all phases of EDL from entry to landing. The equations of motion were limited to 2D and integrated with a Runge-Kutta fourth order (RK4) algorithm. Any planet could be studied with this simulation; however as used in the present study it is configured for Mars trajectory analyses. The derivation of the equations of motion, the required models, and the definition of the example system were provided by Dr. Juan R. Cruz.

3.1 Concept of Operation

The EDL phases shown in Figure 3.1 are included in the simulation. The concept of operations starts with atmospheric interface and ends with touchdown.

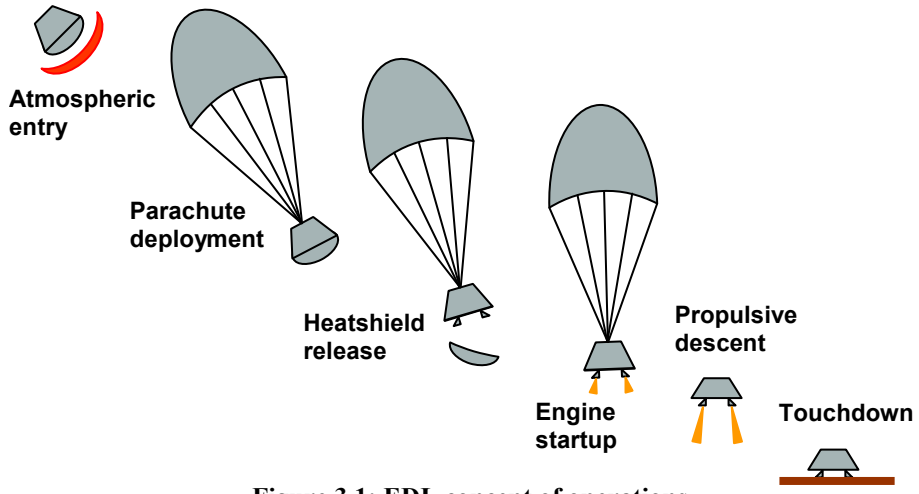


Figure 3.1: EDL concept of operations

Upon entry atmospheric drag slows the vehicle until the estimated dynamic pressure $q_{estimate}$ reaches a trigger value q_{deploy} where parachute deployment is initiated. A second trigger for parachute deployment is the change in sensed acceleration da/dt which must be decreasing to prevent an early parachute deployment. Once the parachute is deployed a timer

is started so that after a fixed time Δt_{HS} the heatshield is released. Next the engine ignites when the calculated thrust required T_{calc} is greater than a trigger value T_{start}. To prevent a premature ignition dT_{calc}/dt must be positive. Then the lander is released from the backshell and parachute after a fixed engine warmup time Δt_{warmup}. Once a specified altitude h_{const} or relative speed V_{const} is reached the constant propulsive descent stage begins. Finally the engine shuts off at touchdown. Table 3.1 shows each event with the associated trigger.

Table 3.1: EDL simulation events and corresponding triggers

Event	Trigger (to start event)	Values
Parachute Deployment	$q_{estimate} \le q_{deploy}$ or $da/dt < 0$	q_{deploy} = 525 Pa
Heat Shield Release	Δt_{HS} (fixed time)	Δt_{HS} = 7 sec
Engine Startup	$T_{calc} \ge T_{start}$ or $dT_{calc}/dt > 0$	T_{start} = 3859.6 N
Back Shell Release	Δt_{warmup} (fixed time)	Δt_{warmup} = 2 sec
Propulsive Descent	$h \le h_{const}$ or $V_{rel} \le V_{const}$	h_{const} = 16.5 m ; V_{const} = 2.44 m/s
Touchdown	$h = 0$	

3.2 Basic Assumptions

There are two degrees of freedom in the simulation restraining the motions to a planar trajectory. Motion is assumed to be in the equatorial plane so that down track distance is longitude. The planet is assumed to be spherical in shape and topography. Assuming that the planet's mass distribution is spherically homogeneous allows the use of a point-mass gravity model. The atmosphere is assumed to rotate with the planet. Superimposed upon the atmosphere is a wind model which ignores vertical motion. A ballistic entry trajectory is assumed with drag as the only aerodynamic force. Finally a gravity turn propulsive terminal descent technique is utilized.

3.3 Equations of Motion

The equations of motion are solved in terms of four state variables: r_I, λ_I, v_I, and γ_I (inertial radius, longitude, speed, and flight path angle, respectively). Figure 3.2 shows the definition of these state variables.

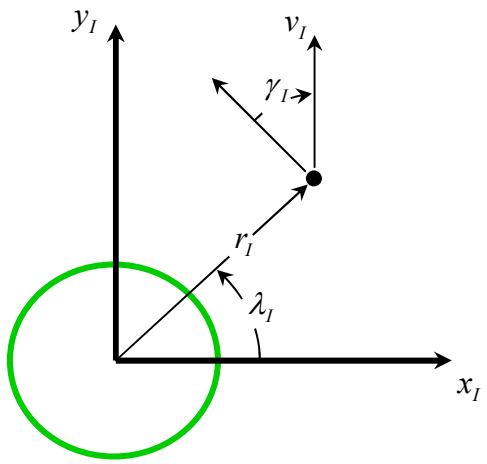


Figure 3.2: Initial EDL entry states

Forces arising from aerodynamics, acceleration of gravity, and propulsion are included in the simulation giving the governing EOM:

$$\frac{dv_I}{dt} = \frac{F_{\|V_I}}{m_v} \tag{3.1}$$

$$\frac{d\gamma_I}{dt} = \frac{v_I}{r_I}\cos\gamma_I - \frac{F_{\perp V_I}}{m_v v_I} \tag{3.2}$$

$$\frac{d\lambda_I}{dt} = \frac{v_I}{r_I}\cos\gamma_I \tag{3.3}$$

$$\frac{dr_I}{dt} = v_I \sin\gamma_I \tag{3.4}$$

Here $F_{\|V_I}$ is the sum of all forces acting on the vehicle parallel to the velocity vector, and $F_{\perp V_I}$ are those forces acting perpendicular to the velocity.

The initial entry states are modeled as correlated random variables using a covariance matrix provided by NASA[a]. Appendix A describes how to generate correlated random variables. Nominal initial conditions were chosen to be similar to Viking with a radius of 3637.2 km, velocity of 4.6 km/s, flight path angle of -16.9°, and longitude of 0°.

3.4 Models

Following is a brief explanation of the models incorporated into the simulation that are used to calculate forces acting on the vehicle. Assuming a spherical planet with uniform radial density distribution allows the planet to be treated as a point mass in the calculation of the gravitational force:

$$\left|\overline{F}_g\right| = \frac{GM_P m_v}{r_I^{\,2}} \tag{3.5}$$

A temperature model with a constant lapse rate from the surface to 57 km and an isothermal region above 57 km is used. Atmospheric density is calculated with the hydrostatic equation as a function of surface density and temperature. There are three perturbed variables involved: surface density, surface temperature, and isothermal temperature. Temperatures at the surface and at 57 km were given distributions that generate a density at 60 km consistent with published data [6],[14].

The wind model has a linearly increasing wind speed up to a perturbed altitude $h_{V_{w,iso}}$, where a constant wind speed region begins. Inputs related to wind speed are conservative estimates considering the variability of Mars wind. They were developed by considering average measurements at random landing sites from the Mars Regional Atmospheric Modeling System[b], as well as published data [9].

Drag coefficients for the aeroshell, parachute, and lander/backshell from Viking are used in this simulation [2],[7]. Equations were fit to published data as functions of Mach number.

[a] Personal communication with Jill Prince of NASA Langley Research Center
[b] Personal communication with Jamie Wilson of North Carolina State University

For propulsion, a variable thrust liquid propellant engine with a constant I_{SP} is assumed. To determine the propellant consumption the rocket equation was used.

Aerothermal considerations during entry are calculated at the nose of the aeroshell as a stagnation convective heat rate.

$$\dot{q}_{conv} = k\sqrt{\frac{\rho}{r_{n,A}}}V_{\infty}^{3} \tag{3.6}$$

Here ρ is the atmospheric density, $r_{n,A}$ is the aeroshell nose radius, k is a constant, and V_{∞} is the airspeed.

3.5 Perturbed Inputs

The perturbed input variables can be organized into three groups: initial states, atmospheric parameters, and vehicle properties. Initial states have correlated uncertainties making them the most complicated inputs from an analysis standpoint. Atmospheric variables will have the greatest uncertainties due to limited direct measurements of the Martian environment. Table 3.2 shows the nominal or mean values of each perturbed input with 3σ variance and PDF. Nominal initial states are given in section 3.3 having normal distributions.

Several sources were utilized to create Table 3.2. NASA experts were questioned to obtain dispersions for inputs with NASA as the source. Variables related to parachute modeling were provided by Dr. Juan R. Cruz at NASA LaRC.

Table 3.2: Perturbed inputs to the EDL simulation

Input	Nominal	3σ Variance	Distribution	Source
$C_{D_{A,D}}$		± 5 %	Normal	Ref. [6]
$C_{D_{A,E}}$		± 5 %	Normal	Ref. [6]
C_{D_P}		± 15 %	Uniform	J. R. Cruz
C_X	1.3	± 10 % (0.13)	Uniform	J. R. Cruz
$h_{V_{w,iso}}$	12 km	± 16.7 % (2 km)	Uniform	J. A. Gaebler
I_{SP}	205 s	± 10 % (20.5 s)	Normal	NASA
m_E	982.9 kg	± 0.2 % (2 kg)	Normal	NASA
S_P	204.96 m^2	± 3 % (6.14 m^2)	Uniform	J. R. Cruz
V_{w_o}	3 m/s	± 100% (3 m/s)	Uniform	J. R. Cruz
$V_{w_{iso}}$	20 m/s	± 60 % (12 m/s)	Uniform	J. R. Cruz
ρ_0	$1.56*10^{-2}$ kg/m^3	± 15 % ($2.34*10^{-3}$ kg/m^3)	Normal	Ref. [6]
T_0	225 K	± 6.7 % (15 K)	Normal	J. A. Gaebler
T_{57}	140 K	± 12.9 % (18 K)	Normal	J. A. Gaebler

3.6 Tracked Output

Outputs from the EDL simulation include the states, vehicle mass, heat rate, total heat load, parachute deployment altitude, parachute deployment acceleration, and propellant mass. To verify that the simulation was performing as intended, the history of the states is plotted in Figure 3.3 when run with the nominal inputs. As designed, the terminal altitude and velocity are zero. The relative flight path angle approaches -90° at touchdown. From entry to touchdown the vehicle crosses approximately 16° of longitude or 950 km across the Martian surface. Discontinuities in the plot of the vehicle mass are due to the separation phases (heatshield release, etc.). Once the engine is started the mass decreases with propellant consumption. The red lines in the plots show 3 events: parachute deployment, backshell separation, and the start of constant propulsive descent respectively.

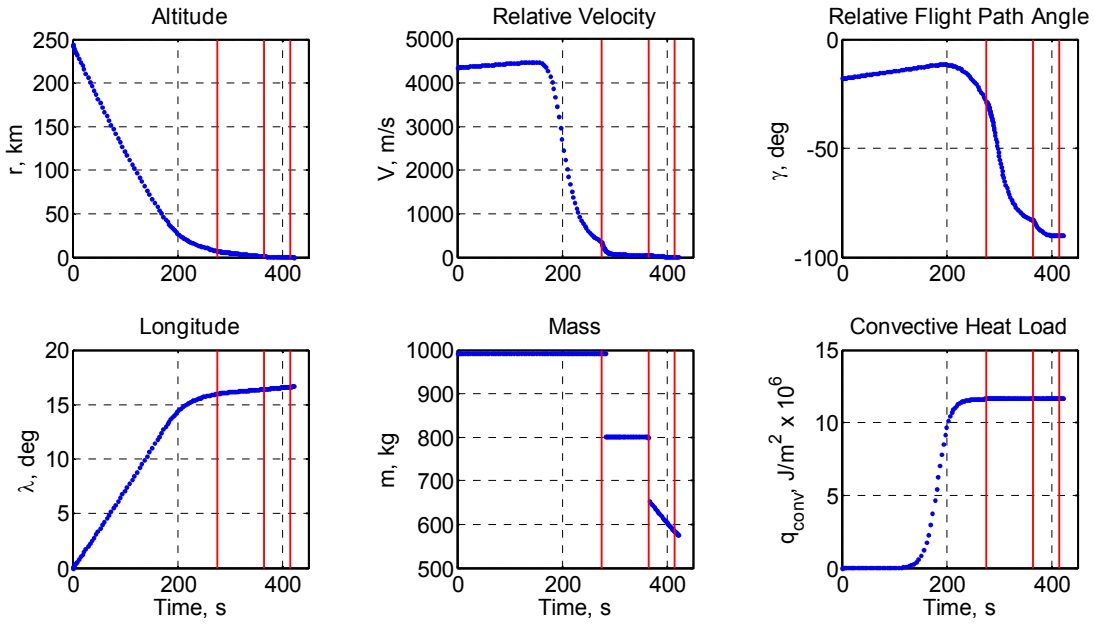


Figure 3.3: Nominal trajectory from EDL simulation

Figure 3.4 shows a plot of velocity versus altitude which can be compared to real mission responses [3].

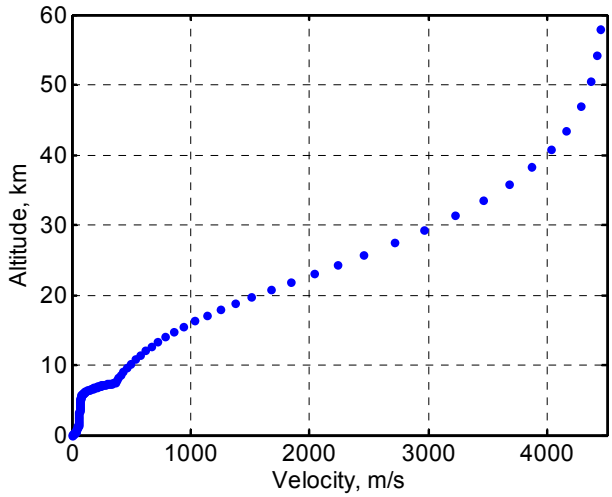


Figure 3.4: Nominal trajectory, velocity vs. altitude of final descent

Output statistics can be seen in Figure 3.5 displaying the histograms for select outputs with a red line at the failure limit. Note that the fourth plot for longitude has two lines since it has a bounded failure constraint.

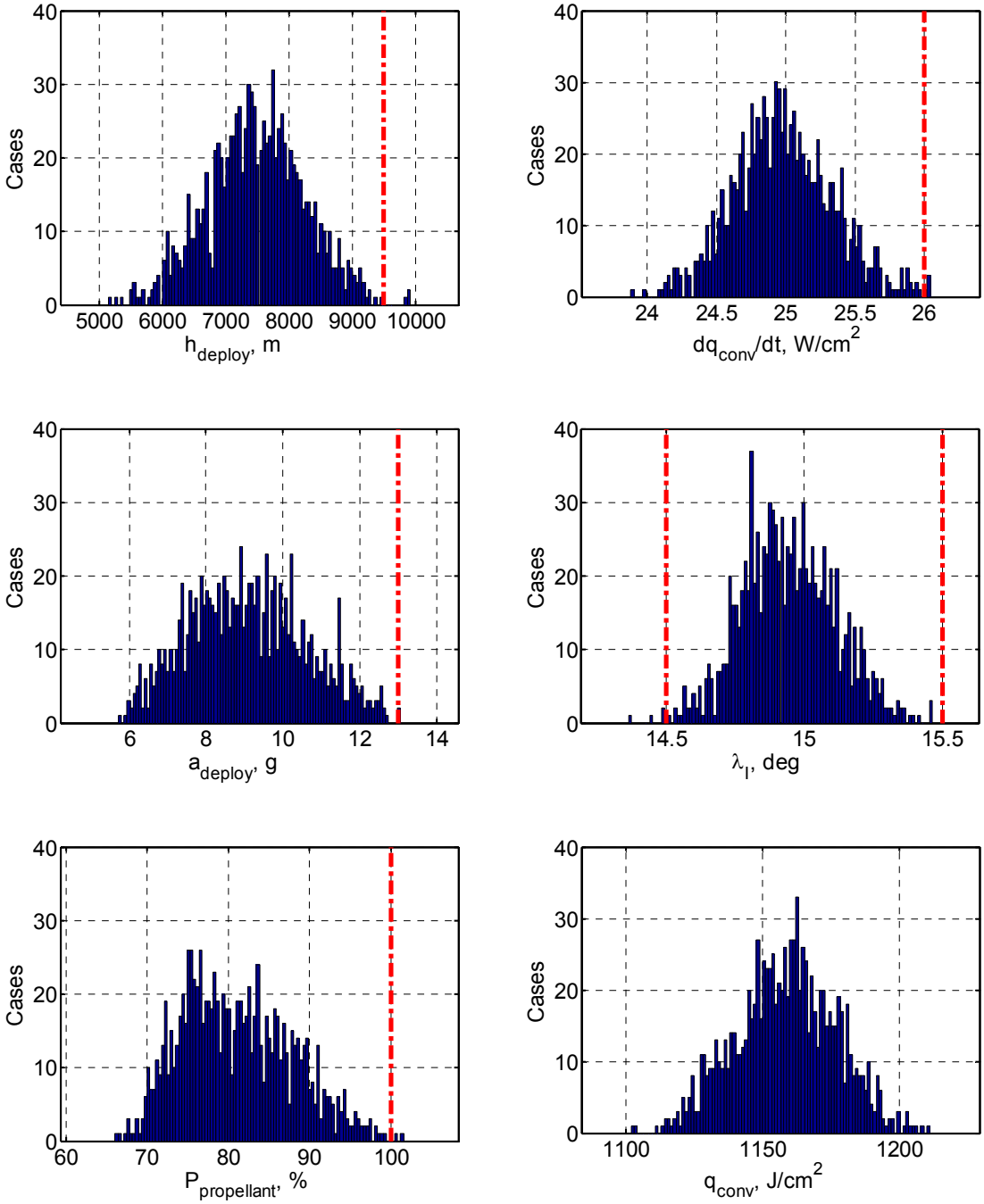


Figure 3.5: Histograms of selected outputs with failure limits

After reviewing the output statistics from the simulation, six outputs were selected for tracking by establishing failure constraints as shown in Figure 3.5. They were arbitrarily

chosen to give an overall low probability of failure, around 1.37% (i.e., a constraint violation of any one tracked output constitutes failure). The failure constraint limits are:

- Peak heating rate greater than 26 W/cm^2
- Altitude at parachute deployment greater than 9.5 km
- Maximum acceleration at parachute deployment greater than 13 g
- Final longitude less than 14.5°
- Final longitude greater than 15.5°
- Percentage fuel consumed greater than 100%

A simplified EDL simulation was developed to test the methods. Engineers studying EDL trajectories would typically run a multi degree of freedom simulation, such as POST. Ultimately these methods need to be evaluated with such a program. The simulation developed has a reduced number of perturbed inputs easing interpretation of the results.

4 Failure Domain Bounding

This method was published in 2008 by engineers at NASA LaRC and a National Institute of Aerospace (NIA) staff scientist for estimating very small failure probabilities [4]. For a model with many uncertain inputs, MCS can be used to estimate the failure probability. However, if the probability of failure is very small, MCS can be computationally expensive. Instead of sampling the entire input space, FDB gives an upper bound on the failure region, thus only cases within this boundary need to be sampled when approximating the probability of failure. This upper bound can be considered a conservative boundary since it over-estimates the failure region. Sample sets generated outside of this upper bound, in the complement safe region, can be assumed to have only successful output cases (i.e. non-failures). It would be costly to characterize the failure domain boundaries in the n-dimensional design space.

4.1 Method Description

To find the upper bound on the failure region, all the uncertain input variables are mapped into standard normal distributions with a mean of zero and standard deviation of one. A transformation of this type can be accomplished if the cumulative distribution function is known, even for correlated variables. The origin is now considered the nominal design. An optimization routine is used to find the Critical Point (CP) of failure, which represents the point of the failure domain that is closest to the origin or nominal design. The distance from the origin to this point can be interpreted as the radius of a hypersphere, which represents a conservative boundary of the failure region. The hypersphere radius R_{CP} is calculated from a vector with components for each input as shown in Equation (4.1) with three inputs.

$$R_{CP} = \sqrt{x_1^2 + x_2^2 + x_3^2} \tag{4.1}$$

Figure 4.1 illustrates this idea with a two dimensional example. The hatched area is the failure region. The design space within the hypersphere can be considered a safe region, while everything beyond is a risky region that could have a failure. After the optimizer

locates the CP and the radius of the hypersphere, the failure probability can be estimated by sampling outside the safe region as represented by the dots.

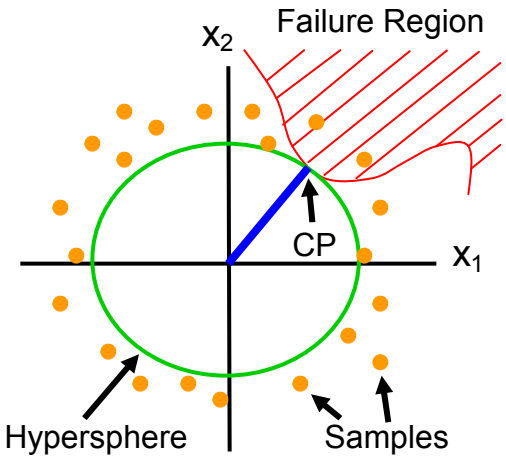


Figure 4.1: Example of safe region defined by hypersphere radius

The optimizer chosen was a gradient based algorithm. Therefore each step requires $(n+1)$ function evaluations to pick a search direction, evaluating the current position and the change caused by changing each input variable. Each failure definition is represented by an inequality constraint function in the optimization program. If there are failure bounds on an output, such that it is only successful within those bounds, these bounds are spilt into two separate inequality constraints. The output hypersphere radius can be mapped into the original input design space to identify the safe region where no failures are assumed to occur.

When evaluating the failure probability, no cases within the hypersphere need to be considered. A MCS analysis can now be performed on the parameters in the space beyond this radius by rejecting points generated within the radius [4] or by conditional sampling, so as to never generate points within the radius [8]. This method may greatly reduce the number of model evaluations needed to estimate the probability of failure. The flow chart in Figure 4.2 aids in conceptualizing the method.

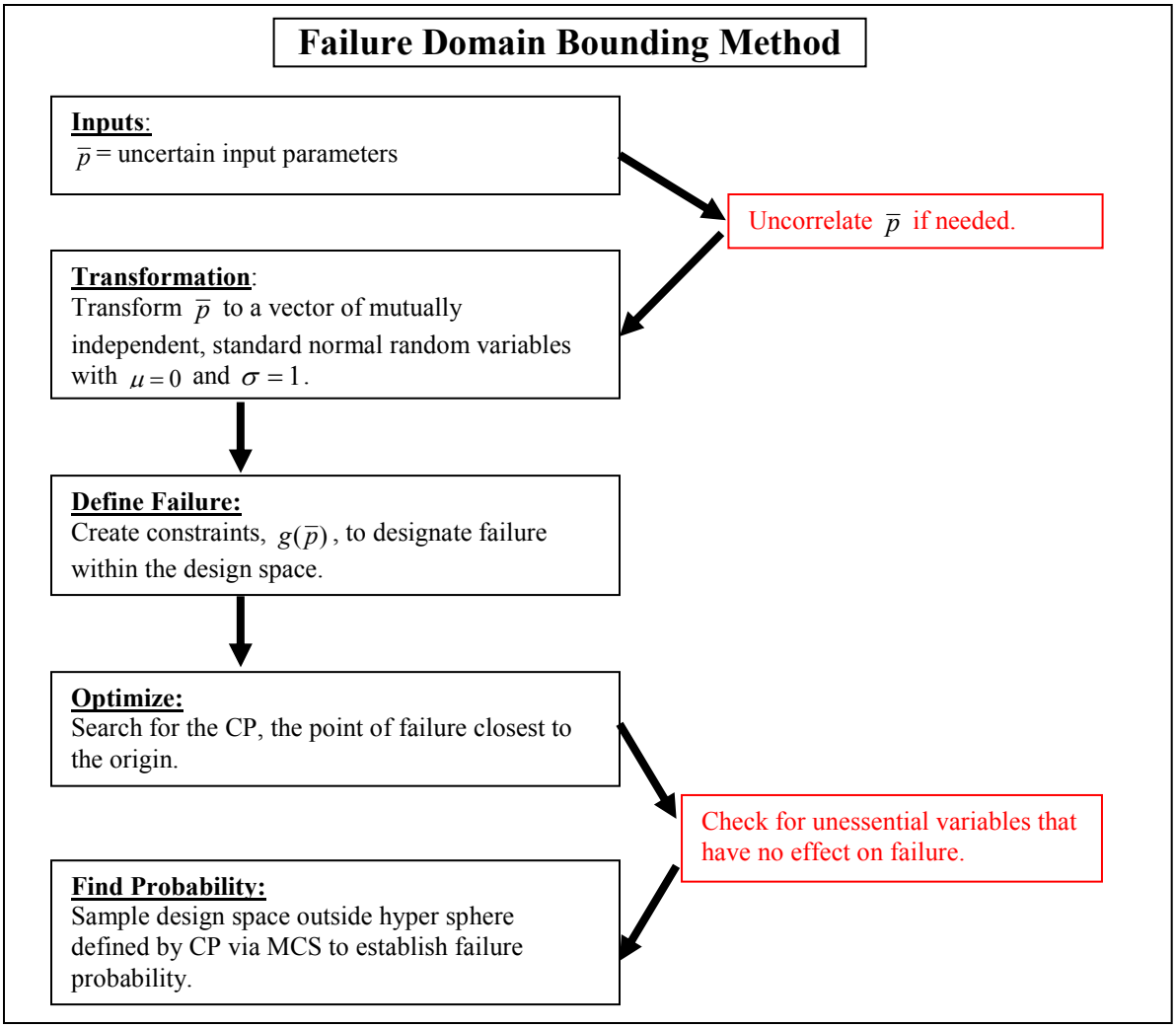


Figure 4.2: Failure domain bounding flow chart

4.1.1 Advantages

The main advantage of this method is the possible reduction in model evaluations required to obtain the failure probability. By assuming cases within the hypersphere are successful the number of evaluations should be significantly reduced. Ignoring samples generated within the hypersphere saves runtime for more samples to be evaluated within the upper bounds on the failure region. Thus a better approximation of the failure probability can be obtained with fewer computations. Random samples are generated like a typical MCS, but any sample within the hypersphere can be assumed successful (i.e. not run through the

simulation) when computing the failure probability. There is also the benefit of generating more failed cases to study the statistics on the inputs causing failure.

4.1.2 Disadvantages

An optimizer is utilized to find the minimum distance from the origin to the failure domain. Use of an optimizer introduces several potential complications. For this problem a gradient based optimizer was utilized. For nonlinear models with several extrema the optimizer may fail to find the global minimum. For situations where the model is highly nonlinear with multiple extrema a gradient method could still be used if multiple starting points were tested to gain confidence. A potentially better optimization method to use for a highly nonlinear model is a genetic algorithm or similar algorithm that tests multiple starting points.

4.2 Application to Projectile Trajectory

Application of FDB for a failure region resulting in approximately 1.7% failures was studied. Constraints defining failure on the outputs were: X_f greater 47,000 m, γ_f less than -14°, and V_f greater than 1250 m/s. First a sense of the actual probability of failure is needed. One million MCS cases were generated and evaluated, obtaining a failure probability of approximately 1.68%. Next for comparison purposes a theoretical upper limit on the number of model runs is set to 2000 cases.

The optimization routine must be applied to each constraint separately so that each will have a hypersphere radius $\|\bar{u}\|$ associated with it. The projectile trajectory has three failure constraints, thus three separate vectors are needed. Equation (4.2) shows the components of the hypersphere radius vector. Table 4.1 presents the vector components found for each failure constraint when FDB is applied to the projectile trajectory. The optimizer required 176 model evaluations to find the hypersphere radius vector components.

$$\bar{u} = \begin{bmatrix} u_g \\ u_y \\ u_v \\ u_\gamma \end{bmatrix} \tag{4.2}$$

Table 4.1: Hypersphere radius found for each constraint in the projectile trajectory

	$X_f > 47{,}000$ m	$\gamma_f < -14°$	$V_f > 1250$ m/s
u_g	-0.3946σ	0.0129σ	0.0019σ
u_y	0.0022σ	0.0052σ	0.0008σ
u_v	0.9467σ	-0.0325σ	2.4900σ
u_γ	2.2028σ	2.7415σ	0.0000σ
$\|\bar{u}\|$	2.4299σ	2.7418σ	2.4900σ

The maximum position constraint provides the smallest radius of 2.4299σ. The smallest safe region is the complement to the largest risky region, or the upper bound on failure. Now random inputs can be generated via MCS in the standard normal space with all cases having a radius less than 2.4299σ assumed successful. Only cases outside this radius need to be run through the model or simulation. Individual components of the $\bar{u}$ vector for each constraint also give an indication of the level of significance each input has on that failure. Initial velocity has the greatest effect on final velocity as seen in the right column of Table 4.1. The middle column shows that the initial flight path angle has the greatest effect on the final flight path angle. As expected, the final position constraint has three variables affecting failure.

Evaluating the cannon problem with 2000 input cases using typcial MCS, provided 35 failed cases, giving a failure probability of 1.75%. Since 176 function evaluations were required to find the minimum radius by the optimizer, there are 1824 evaluations left when there is a 2000 case maximum enforced. Random inputs were generated one at a time until 1824 cases were generated in the risky region. Each input set in the safe region was stored, but not evaluated. A total of 8526 input variable sets were generated before the maximum of 1824 function evaluations was reached, giving 6702 safe cases. Of the 1824 risky cases, 141

were failures giving a failure probability of 1.65%. This failure probability is within 0.03% of the probability found by running one million cases (1.68%), but only requiring 2000 evaluations. Running 2000 cases using typical MCS gave a failure probability within 0.07% for comparison.

A visual comparison of the typical convergence of MCS vs. FDB for 2000 runs is shown in Figure 4.3. This figure only shows the cases evaluated on the x-axis, thus ignoring the cases in the safe region for FDB. The FDB method converges on the failure probability with fewer evaluations than MCS. The samples generated in the safe region were evaluated to check if there were any failures, and none were present.

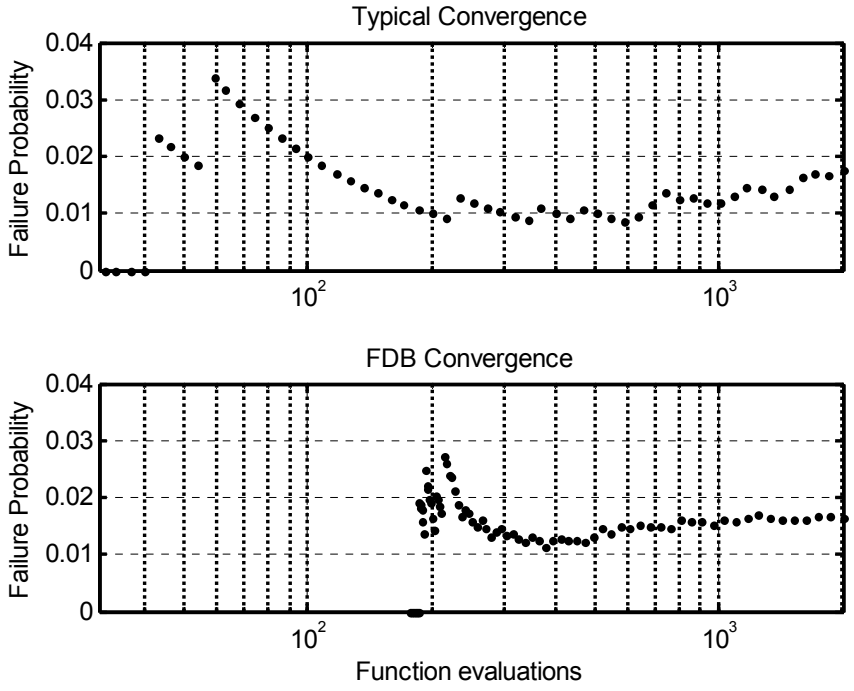


Figure 4.3: Convergence comparison between MCS and FDB with 2000 run limit

By identifying the risky region, the percentage of failed cases produced increases from 1.75% to 7.0% of the cases run. Four times as many failed cases can be produced for future study for the same number of function evaluations. Acquiring a large number of failed cases is useful. The inputs from the failed cases can be plotted as histograms to study which inputs are causing failure. Figure 4.4 shows the histograms of inputs causing a γ_f less than

-14°. In the top left plot, 7 of the 47 cases with $\gamma_f < -14^\circ$ had a g between 9 and 9.1 m/s^2. It is easily seen from the bottom right plot that γ_0 is the main influence on failure. The gap in the plot shows that no failures occurred when γ_0 had values between roughly -13° and $+13^\circ$. Figure 4.5 shows failures from the projectile traveling greater than 47,000 m, indicating trends in three of the input variables. Acceleration of gravity has a slight trend toward the lower bound, velocity is distributed around the 1σ value of 1100 m/s, and γ_0 is distributed around the 2.5σ value of 12°. Finally in Figure 4.6 the input sets causing the velocity constraint violations are shown. Here the main contributor is V_0 which causes failure when it is above ~1250 m/s.

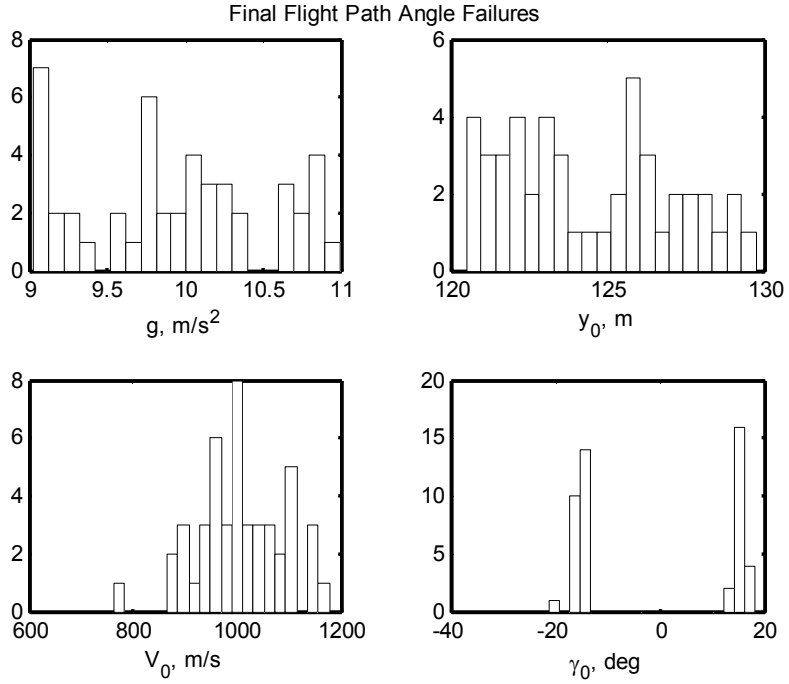


Figure 4.4: Input histograms of cases causing γ_f failure

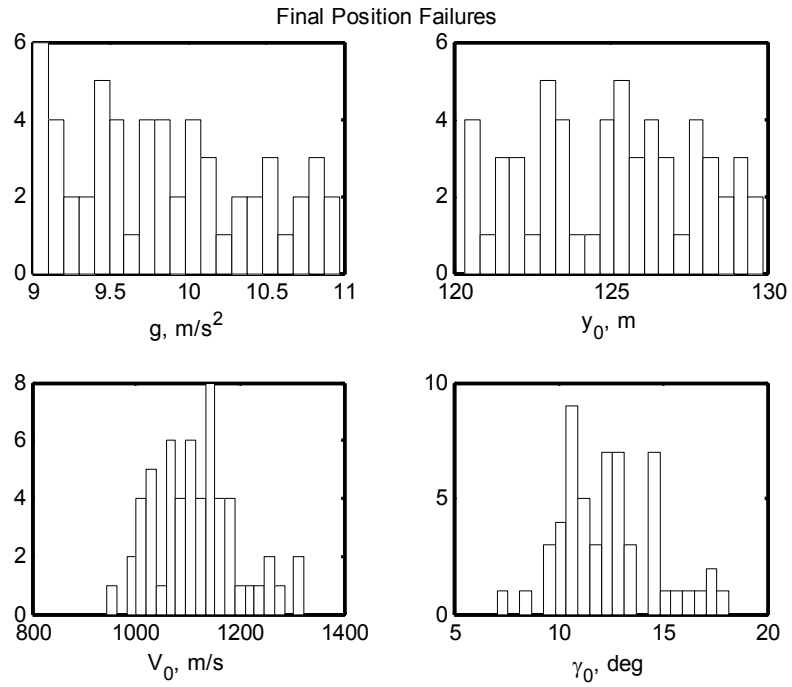


Figure 4.5: Input histogram of cases causing X_f failure

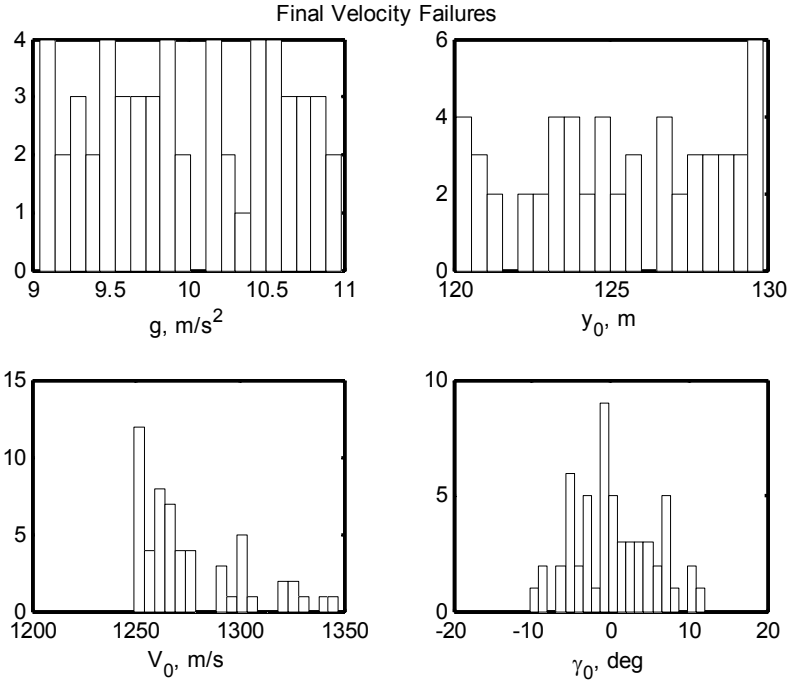


Figure 4.6: Input histograms of cases causing V_f failure

The size of the failure region has an effect on the benefits from this method. A region with approximately 1.7% failure probability has been studied. If the region is reduced to only constitute about 0.4% of the design space the benefits are amplified. The failure criteria were modified to be: X_f greater 58,000 m, γ_f less than -16°, and V_f greater than 1300 m/s.

Previously the hypersphere radius was 2.4299σ, now with these new constraints it has increased to 2.9177σ. The optimizer required 175 runs to find this radius. Applying the FDB method identified a safe region containing 21,937 cases that were ignored. The risky region gave 107 failures out of the 1825 cases run. This gives a failure probability of 0.45%. This number of failed cases is a 10 fold increase over running the typical MCS analysis, whereas it was 4 fold previously. Post analysis of these cases showed no failures occurred in the safe region. A convergence study is shown in Figure 4.7 where MCS requires more than 10,000 cases to converge, well over the 2,000 case limit that was set.

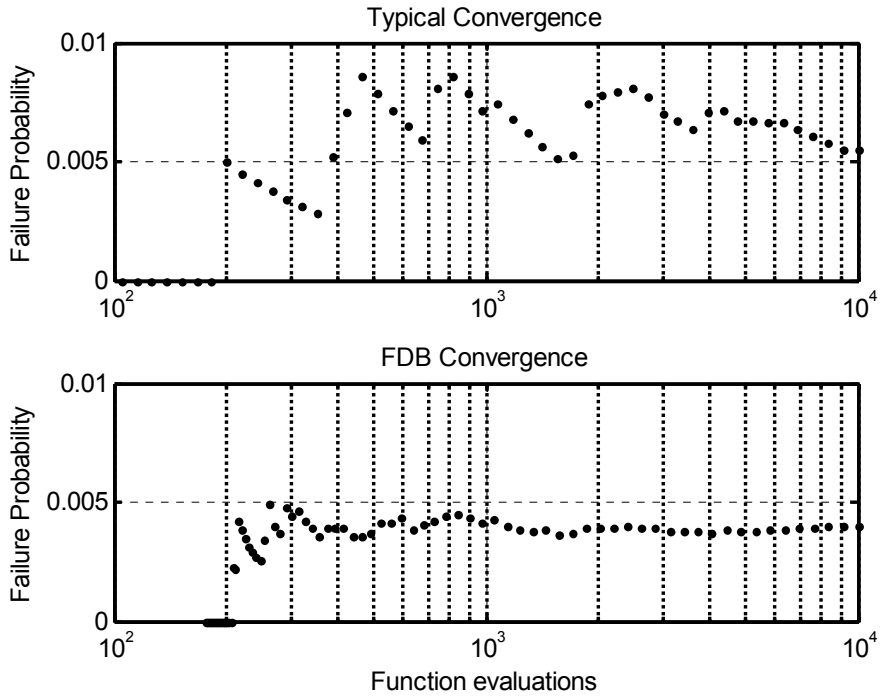


Figure 4.7: Convergence comparison between MCS and FDB for ~0.4% failures

4.2.1 Conclusions

Assuming that there is a failure region and that an optimum hypersphere radius is obtainable, the FDB method can yield insights into the failure characteristics. For the projectile trajectory it was shown that more failed cases can be generated for further study. As shown above in Figure 4.4-Figure 4.6, these additional failed cases provide more informative histograms for studying the cause of failure. Also the output $\overline{u}$ vectors for each failure constraint give an indication of the inputs contributing to each failure.

If there are limitations on the runs allowable, due to time or computing constraints, this method gives a better estimate of the failure probability then performing a traditional MCS analysis when applied to the projectile trajectory. When the accurate estimate of the failure probability is desired, the FDB will converge on a solution with less runs then the typical MCS analysis. Finally the smaller the failure region is, the greater the benefits of the FDB become.

4.3 Application to EDL Simulation

To recap from section 3.5, a failure region was defined to have a probability of 1.37% for the EDL simulation example. This includes failures due to max heat rate greater than 26 W/cm^2, altitude at parachute deployment greater than 9.5 km, max acceleration at parachute deployment greater than 13 g, a final longitude less than 14.5° or greater than 15.5°, and greater than 100% propellant used.

Application of this method returned a hypersphere radius of 2.5441σ. Table 4.2 shows the radius for each constraint since they are optimized separately. Also shown is the number of model evaluations the optimizer required. The high number of evaluations is caused by the large number of inputs and the use of a gradient based optimizer. Each time the optimizer takes a step toward the optimum, it must run the model 18 times, once at the current point, and 17 times to perturb each input. The final result is the minimum value of the hypersphere radius from each constraint.

Table 4.2: Hypersphere radius found for each constraint in the EDL simulation

Constraint	Radius	Model Evaluations
$(dq_{conv}/dt)_{\max} > 26 W/cm^2$	2.5729σ	1705
$h_{deploy} > 9.5 km$	2.7949σ	859
$a_{deploy} > 13 g's$	2.5441σ	221
$\lambda_f < 14.5^\circ$	2.6970σ	348
$\lambda_f > 15.5^\circ$	3.2909σ	909
$P_{propellant} > 100\%$	2.6307σ	1712
Final	2.5441σ	5754

The radius presented in Table 4.2 is the Euclidian norm of a vector pointing to the CP. Plotting this vector as a bar graph gives Figure 4.8 for the output altitude of parachute deployment. This plot indicates which variables are affecting failure. The inputs defining the upper atmosphere, and the drag force acting on the vehicle are the most influential as expected. Those variables that have no effect on parachute deployment failure are zero as expected, such as the I_{SP}. A study of the physics in the EDL simulation indicates that the sensitivity to ρ_0 and C_{dae} should be equivalent. The results do not make this obvious however. The reason lies in the variance assigned to the inputs. Referring to Table 3.2, ρ_0 has a 3σ value that is three times greater than that on C_{dae}, thus the sensitivity to ρ_0 is three times greater since these sensitivities are based on the statistics.

The next step is to start generating input cases and identifying those cases that can be assumed successful. Generating 30,000 random input cases gave 309 cases in the safe region, identified with the hypersphere radius, which is only ~1%. The remaining 29,691 risky cases had 416 failures. Notice that it takes more evaluations to identify the safe region (5754), than there are cases in the safe region (309). There are no benefits to this method when the safe region is so small.

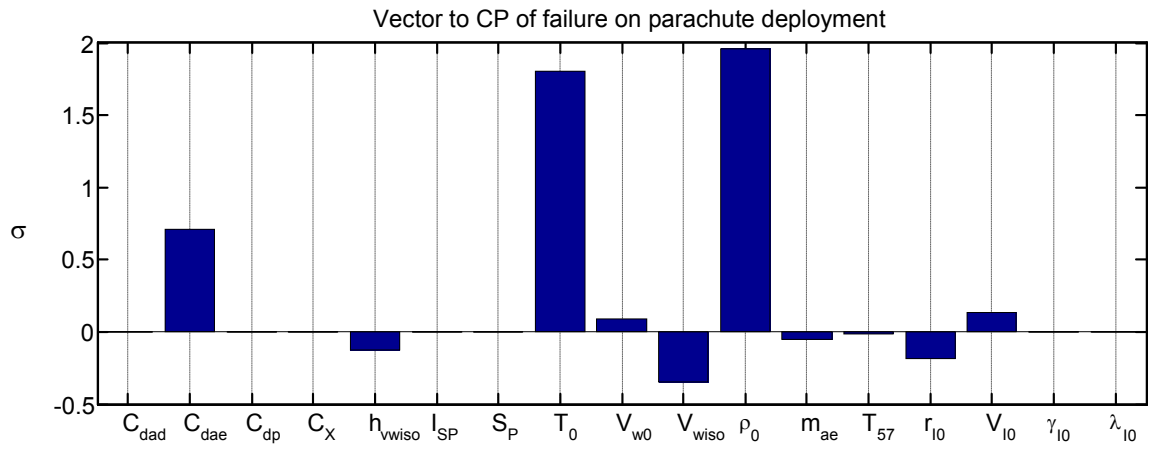


Figure 4.8: Vector components to the CP of failure on parachute deployment altitude

4.3.1 Complications

After investigating why the safe region was so small two complications were identified. First was the assumption that a hypersphere, which encompasses the safe region, would contain a significant portion of the design space. The second complication was with the minimization of the hypersphere over all dimensions (or inputs).

Investigating the first complication shows that the hypersphere does not contain a significant portion of the input design space due to how a large number of input dimensions spread the design space. This spreading undermines the usefulness of minimizing a hypersphere around the nominal design. Since the inputs have standard normal distributions, a radius around the origin should encompass a large portion of the sample set. For a standard normal distribution of a single variable, a hypersphere radius of 2σ would encompass ~95% of the design space. Adding a second independent dimension has a spreading effect on the design space. There is a greater chance of being far from the nominal point. Since this is a 2D situation, the hypersphere would be a circle. A hypersphere radius of 2σ now encompasses ~86% of the design space. Adding a third dimension further reduces the probability of being within 2σ to 74%. This trend continues as more dimensions are considered.

The second complication involves minimization of the hypersphere radius vector over all inputs. The vector component for any input with minimal influence on the failure region will have a distance of zero. Figure 4.9 illustrates this scenario where the input represented

by x_2 has no influence on failure. The vector to the CP in this scenario is 2σ in the x_1, and 0σ in x_2. This second input can have any value without affecting failure. Thus including the component for x_2 in the hypersphere radius vector unnecessarily reduces the safe region.

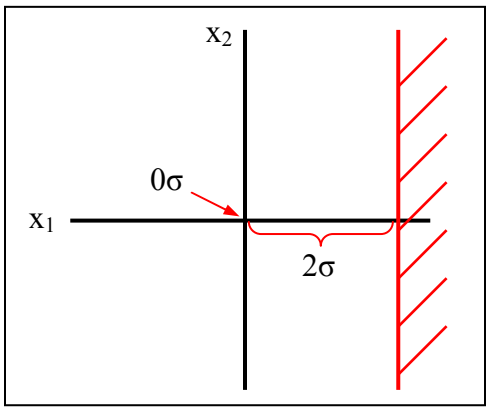


Figure 4.9: Hypersphere radius with a non-influential variable

To get around this complication, components of the output hypersphere radius vector with values less than some tolerance can be ignored in defining the safe region. The main justification for this action is the insensitivity as shown in Figure 4.9. It is desirable to ignore insensitive inputs to reduce the dimensionality of the problem. When identifying whether a case is safe or not, only the inputs affecting failure are considered. For example, if there are three inputs, and the hypersphere radius is $\bar{u} = [3 \quad 0 \quad 4]$, then the second input could be ignored. Thus for each case generated, only inputs x_1 and x_3 are checked to see if the case lies within a radius of $\sqrt{3^2 + 4^2} = 5$. The remaining variable x_2 is not taken into account, it could have any value, including 5, without causing that case to be designated risky. However, now each constraint must be considered in defining the safe region. An input that does not influence one failure constraint may be important to another. If that variable were ignored there would be failed cases in the safe region due to the second constraint. In essence this new approach identifies separate safe regions, or individual hyperspheres, for each constraint, where each hypersphere has minimal dimensions. Reducing the dimensionality also counteracts the spreading effect.

The old methodology would have had a single 17 dimensional hypersphere radius that should in theory touch the CP and contain most of the design space. It does not contain a significant portion of the space, however, due to the spreading effect of increasing the dimensions. For the EDL simulation the new methodology proposed has six hyperspheres, of only 4-6 dimensions each. The tolerance used to ignore a component was set to 0.2σ. Input dimensions less than this threshold can be thought of as having infinite bounds to define the safe region for that particular dimension. These six regions are merged into one, so that only points common to all six hyperspheres are considered safe. The effect is to increase the safe region from containing 1% to over 42% of the sample set. No failed cases were identified in the safe region at this tolerance. Figure 4.10 shows the convergence of the estimated failure probability by performing typical MCS versus using FDB. The FDB dots on the plot start at 5754 model runs, since those were required by the optimizer to find the CP. Both methods have similar convergence. The benefits would increase if the dimensionality of the hyperspheres could be reduced further, or if there were less failure constraints.

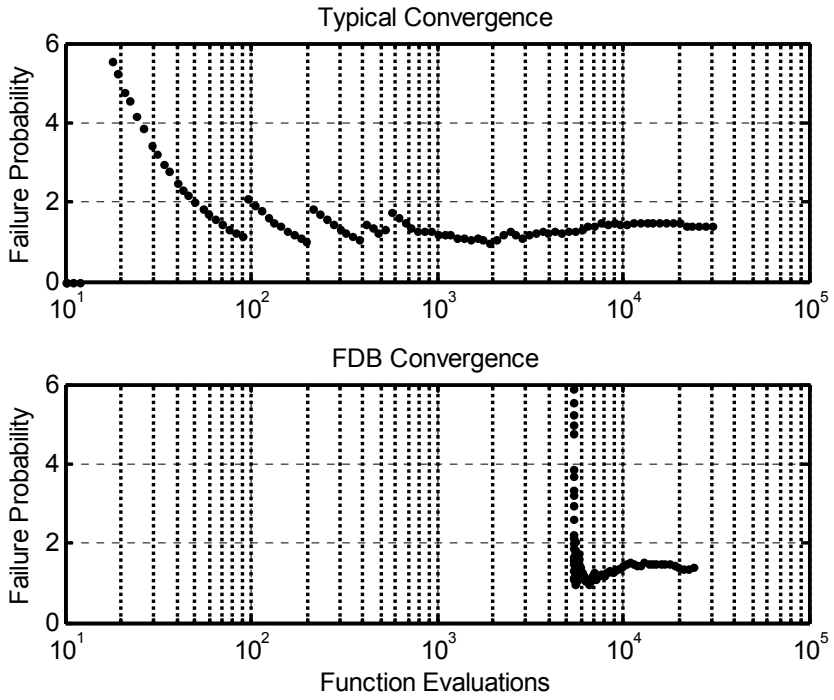


Figure 4.10: Convergence of EDL simulation failure probability for MCS vs. FDB

5 Global Variance-Based Sensitivities

This method was created in 1990 by Dr. I.M. Sobol, a Russian mathematician. It provides a way of representing a global type of sensitivity based on the variance of the output [15]. Advances to the original method reduce the required computations and are incorporated in this work [13]. Global sensitivity measures the effect input variance, for a specific input or combination of inputs, has on the output variance.

5.1 Method Description

Appendix B provides an in depth derivation of this method, here is a brief explanation. The output is approximated as a series of summands of different dimensions:

$$f(\bar{x}_1,...,\bar{x}_n) = f_0 + \sum_{i=1}^{n} f_i(\bar{x}_i) + \sum \sum_{1\le i<j\le n`} f_{ij}(\bar{x}_1,\bar{x}_j) + ... + f_{1,2,...,n} \tag{5.1}$$

Sobol calls this a Fourier-Haar series expansion. This approach is similar to performing a Fourier series approximation; however the individual functions do not need to calculated. The function approximations are only needed to show that the total variance can be decomposed into individual components. These functions are orthogonal, with the special property that the area under each function sums to zero. Each function in this series is associated with a single input, or a combination of inputs as represented by the number of subscripts.

Computing the expected value or mean in equation (5.2) requires an integration of these functions over each input, giving zero for each individual function and leaving only the constant mean value f_0. This is occurs since the area under each function sums to zero. That is the functions have a value of -1 over a portion of the domain and a value of +1 over the same size portion with a value of zero everywhere else. The portion of the domain with a non-zero value is different depending on the subscript in equation (5.1). For example f_i is -1 on [0 ½] and +1 on [½ 1], but f_{ij} may be -1 on [0 ¼] and +1 on [¼ ½], zero everywhere else.

The functions are only defined on the unit hypercube [0,1] and each integration below is from 0 to 1.

$$E\left[f\left(\bar{x}\right)\right] = \int f\left(\bar{x}\right)d\bar{x} = f_0 \tag{5.2}$$

To find the variance the expected value operator must be evaluated twice as seen in equation (5.3). The expected value of the function squared is no longer canceled giving equation (5.4). The variance due to a specific input is given in equation (5.5). This allows the comparison of the variance due to any input with the total variance on the output.

$$V\left[f\left(\bar{x}\right)\right] = E\left[f\left(\bar{x}\right)^2\right] - E\left[f\left(\bar{x}\right)\right]^2 \tag{5.3}$$

$$V = \int f^2\left(\bar{x}\right)d\bar{x} - f_0^2 \tag{5.4}$$

$$V_i = \int f_i^2\left(\bar{x}_i\right)d\bar{x}_i \tag{5.5}$$

$$S_i = \frac{V_i}{V} \tag{5.6}$$

The sensitivity can be found for each input separately or in any combination with other input variables. Sensitivities due to individual variables are considered main or 1st order effects. Coupling or 2nd order effects comprise all possible combinations of two variables. Combinations of variables can be up to order n. All the sensitivities sum to a value of 1. Stated another way, the variance due to each input and combination of inputs must add up to the total variance. Another parameter of interest is the total effect sensitivity. Summing sensitivities, such as the main effect and all higher order terms involving the variable of interest, gives the total effect. These ideas are portrayed in equations (5.7) and (5.8) for a 3 input model, where the number of subscripts represents the order (1st, 2nd…) and the superscript T is for total effect.

$$S = 1 = S_1 + S_2 + S_3 + S_{12} + S_{23} + S_{13} + S_{123} \tag{5.7}$$

$$S_1^T = S_1 + S_{12} + S_{13} + S_{123} \tag{5.8}$$

Conceptually Figure 5.1 shows what type of information Sobol's method provides. The left plot shows the output of a function with two inputs sampled 20 times. When the 1st input set is resampled, while leaving the 2nd set the same, the center plot is generated. Hence each output would change since one of the inputs is different. Next the right plot is produced by resampling the 2nd variable and using the original for the 1st. Again the outputs would change. The original outputs were left in the plot for reference. By taking this new variance and comparing it to the original, an idea of the influence that variable has is obtained. In Figure 5.1 it is obvious that the second variable has a larger effect on the output variance. Hence the second variable would have a larger sensitivity.

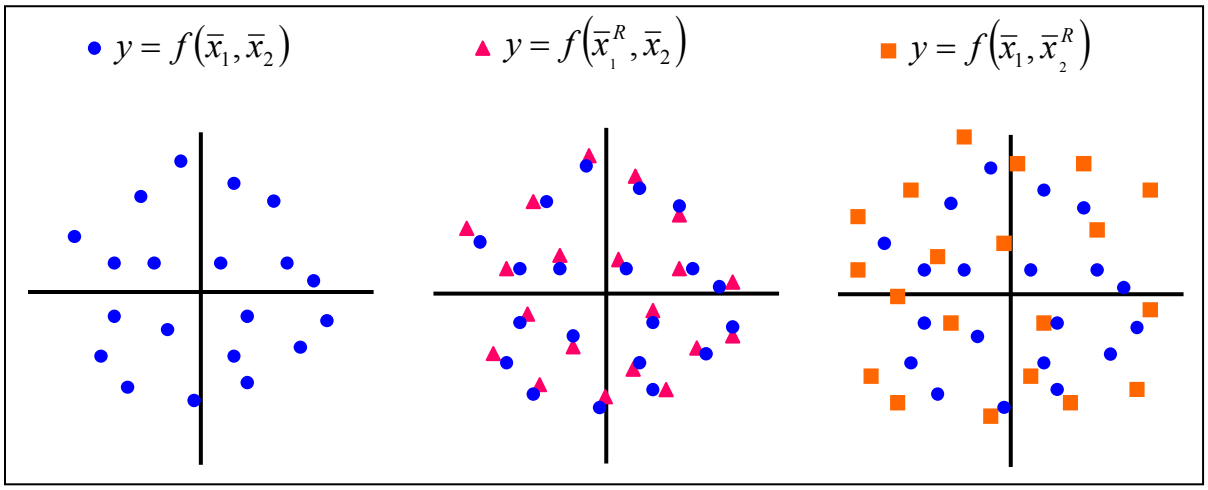


Figure 5.1: Visualization of Sobol's method, resampled sets have superscript R

To isolate the variance of a specific input i all the inputs are varied via MCS to approximate the total variance V then the complimentary variables $j \neq i$ are resampled to approximate the variance caused by i. The required integrals for calculating the variance are approximated by MCS. The equations in approximate form are:

$$f_0 \approx \frac{1}{N}\sum_{i=1}^{N} f(\bar{x}) \tag{5.9}$$

$$V + f_0^2 \approx \frac{1}{N}\sum_{i=1}^{N} f^2(\bar{x}) \tag{5.10}$$

$$V_i + f_0^2 \approx \frac{1}{N}\sum_{i=1}^{N} f(\bar{x}) f(\bar{x}_i, \bar{x}_j^R) \tag{5.11}$$

Here $\bar{x}$ is a matrix of the n inputs with N samples and $\bar{x}_i$ is the 1 by N set of samples of the input for which the sensitivity is desired. Finally $\bar{x}_j^R$ is the set of $(n-1)$ by N samples that are resampled. Choice of sampling method can be important. Low discrepancy sequences, or quasi-random number generators, get better results then pseudo-random number generators by picking numbers in a manner that attempts to guarantee uniformity across the design space [10]. Two such quasi-random number generators available to use are Halton sequence leaped [10] and Sobol' sequence generators [18].

These global sensitivities give a measure of the sensitivity of the output statistics to the input statistics. The most influential variable can be identified by comparing the individual effects of each variable on the output variance. Such knowledge can help engineers decide where to focus their efforts in reducing the uncertainty on the output to obtain the most benefit. Considering combinations of variables, as opposed to individual variables, gives an indication of the coupling between inputs.

5.1.1 Advantages

This method is ideal for simulations since it is model independent. There are no assumptions of the type of model or functions used. Just as the most influential variable can be found, unessential variables can also be identified. These variables can be removed from the analysis by being set to their nominal value with minimal effect on the output uncertainty.

5.1.2 Disadvantages

A large sample set is required to obtain reliable statistics. Each input requires N runs times the number of higher order terms involving that input. Thus for each individual or coupled sensitivity desired, the model must be evaluated with a separate sample set. Often surrogate models are developed based on a design of experiments, and then the surrogate is

used to obtain the sensitivities. Errors would be introduced by using a surrogate, however computational expense is reduced.

There are some requirements on the input parameters. For example, the inputs must be uncorrelated and transformed to uniform distributions on the unit hypercube. In addition, there is no way to evaluate the accuracy of the results, although there are methods to obtain confidence intervals [1].

5.2 Application to Projectile Trajectory

Final position from the projectile trajectory is studied to explain the meaning of variance-based sensitivity. Instead of constantly resampling, two separate samples are generated to be used throughout the process. If the number of variables is n, and the number of samples is N, then both sets will be independently generated to be N by n. For the projectile trajectory n is four. Variables from the first set will have the superscript A; while those from the second set will have superscript B. A numerical integration scheme is implemented to calculate the variances. The variance calculated from the first sample set is used as the basis of the sensitivities. The expected value is computed to give the mean value f_0. Numerically this is just the average of the output:

$$f_0 = \frac{1}{N}\sum_{1}^{N} X_f\left(g^A, y_0^A, V_0^A, \gamma_o^A\right) \tag{5.12}$$

$$V = \frac{1}{N}\sum_{1}^{N}\left[X_f\left(g^A, y_0^A, V_0^A, \gamma_o^A\right)^2\right] - f_0^2 \tag{5.13}$$

In each future calculation of a variance, the mean f_0 shown above with sample set A is used. To find the sensitivity of final position to each input or coupling of inputs, the model is rerun with the variable/s of interest taken from sample set A while the other variables are from set B. For the sensitivity to y_0 the variance is calculated as:

$$V_{y_0} = \frac{1}{N}\sum_{1}^{N}\left[X_f\left(g^A, y_0^A, V_0^A, \gamma_o^A\right) * X_f\left(g^B, y_0^A, V_0^B, \gamma_o^B\right)\right] - f_0^2 \tag{5.14}$$

$$S_{y_0} = \frac{V_{y_0}}{V} \tag{5.15}$$

Figure 5.2 shows how the variance on the output changes as each input is resampled. In these plots the green dots represent the variance between the two sample sets, i.e. they are plots of the model run with set A on the x-axis versus the model run with set B on the y-axis. The blue dots are model runs of set A versus runs with combinations of the two sets which are labeled.

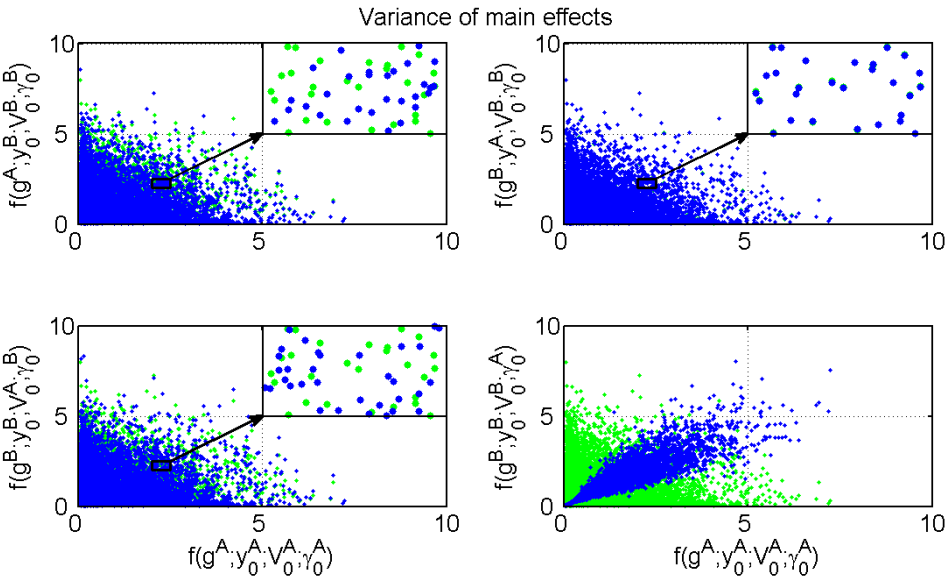


Figure 5.2: Effect of resampling each input variable on final position

From the top right plot the effects of y_0 are negligible as can be seen. In this simulation y_0 has nearly no effect on the variance, hence it would have a very small sensitivity. The blue dots lie directly on top of the green dots. Thus changing the input y_0 does not affect the output. At the other extreme is the bottom right plot showing the main effect of γ_0. Here the blue dots show final position calculated with the same sample set of γ_0 on both axes, with different sample sets for the other inputs. The plot appears to be approaching a straight line, as would occur if a variable was plotted versus itself. Final

position is so dependent on γ_0 that it plots as if position was plotted against itself. In these plots the labels have been simplified to ignore the order of magnitude which was 10^4. The two left plots show the effects of g and V_0 where it is difficult to judge which is more influential between the two.

Another way to portray the change in variance is with residuals shown in Figure 5.3. They show the same information as above, in a different form. Starting with the top left and going clockwise the plots show:

$$X_f\left(g^A, y_0^B, V_0^B, \gamma_o^B\right) - X_f\left(g^B, y_0^B, V_0^B, \gamma_o^B\right) \tag{5.16}$$

$$X_f\left(g^B, y_0^A, V_0^B, \gamma_o^B\right) - X_f\left(g^B, y_0^B, V_0^B, \gamma_o^B\right) \tag{5.17}$$

$$X_f\left(g^B, y_0^B, V_0^A, \gamma_o^B\right) - X_f\left(g^B, y_0^B, V_0^B, \gamma_o^B\right) \tag{5.18}$$

$$X_f\left(g^B, y_0^B, V_0^B, \gamma_o^A\right) - X_f\left(g^B, y_0^B, V_0^B, \gamma_o^B\right) \tag{5.19}$$

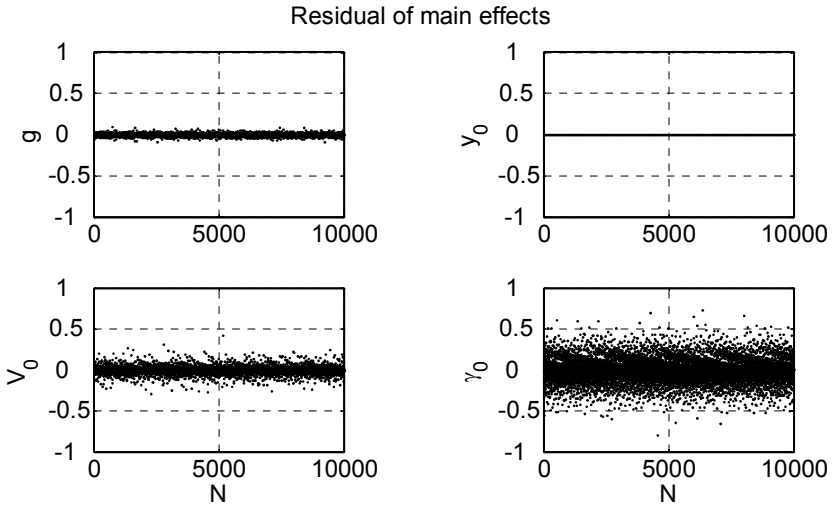


Figure 5.3: Residual variance of resampling each input variable on final position

They illustrate how swapping the sample set of a particular input changes the output. Viewing the data in this form shows that between g and V_0, velocity has a greater influence on the output variance, which was indistinguishable in Figure 5.2. In the bottom two plots

there is some structure visible; this is a consequence of using a quasi-random number generator.

The above figures serve as an aid in visualizing what a global variance-based sensitivity provides. In calculation, the sensitivities are obtained by taking a ratio of the variances as described earlier.

Accurate estimates of the sensitivities require enough samples be run through the simulation for the statistics to be well represented. Convergence is studied in the next plot. Figure 5.4 shows convergence of the main effects to their sensitivities. The solution has converged with just over 1,000 samples. The sensitivities of g and y_0 are about zero, γ_0 is around 0.93, and V_0 has slight value of 0.02. Based on the main effects γ_0 is the most influential input affecting final position.

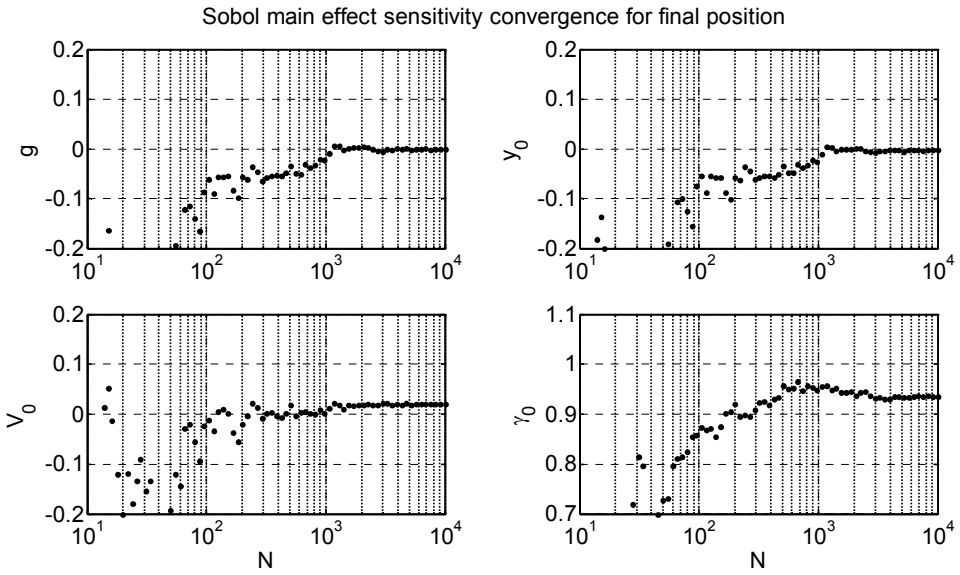


Figure 5.4: Convergence analysis for sensitivities to main effects

The total function or model evaluations required to obtain sensitivities is dependent on how many terms are desired. For the projectile trajectory, if just the main effects are required then there are four terms sought. In the following T is the number of terms sought.

$$\text{main effects cost} = N * (T + 1) \tag{5.20}$$

Based on the convergence study N could be set to 1,000 samples giving a total computational cost of 5,000 runs. The total effect terms, which will hint at the couplings present, can be obtained with a small increase in computations:

$$\text{main} + \text{total effects cost} = N * (T + 2) \tag{5.21}$$

The additional 1,000 runs give an indication of which couplings should be investigated, if necessary, to understand the system. If the main, total, and 2nd order couplings are desired, the above equation would be used with 10 terms, 4 main and 6 coupling pairs, requiring 12,000 runs. If run times become prohibitive a design of experiments approach could be implemented to create a surrogate model. Since all the sensitivities must add up to one, the main and total effects can be used as a guide to search until the sum of the sensitivities found approaches one. A precursor analysis could be performed by running a low number of cases which uniformly span the design space. Graphing these responses would give an idea of which terms to focus on initially.

Values for the sensitivities of position are shown in Table 5.1. Generally since the sensitivity is a ratio of the variances they should never be negative. Negative values can exist since different sample sets are used while the same mean from set A is always subtracted. Hence the mean could be slightly different, however the means should approach the same value as the size of the sample set is increased.

Table 5.1: Final position sensitivities by Sobol's method for projectile trajectory

Total Effects	
g^T	0.0049
y_0^T	-0.0001
V_0^T	0.0599
γ_0^T	0.9799

Main Effects	
g	-0.0008
y_0	-0.0028
V_0	0.0207
γ_0	0.9344

2nd Order Effects	
$g - y_0$	0.0028
$g - V_0$	0.0028
$g - \gamma_0$	0.0063
$y_0 - V_0$	0.0028
$y_0 - \gamma_0$	0.0028
$V_0 - \gamma_0$	0.0395

3rd Order Effects	
$g - y_0 - V_0$	-0.0028
$g - y_0 - \gamma_0$	-0.0028
$g - V_0 - \gamma_0$	-0.0029
$y_0 - V_0 - \gamma_0$	-0.0028

5.2.1 Conclusions

Final position main effect sensitivities show that the dominant effect is from γ_0 as expected, with the next largest contributor being the coupling between γ_0 and V_0. With the current input distributions, position is the only output with significant nonlinear effects. Final flight path angle and velocity have only one major contributor to their variance, being γ_0 and V_0 respectively. This method worked well on the projectile trajectory giving the insights desired at a reasonable computational cost.

5.3 Application to EDL Simulation

Due to the large number of inputs going into the simulation, which increases the samples needed for convergence, only main effects and total effects were sought. To give an idea of the sensitivities obtained, two of the outputs will be investigated. Figure 5.5 shows the main effect or 1st order sensitivities of the parachute deployment altitude.

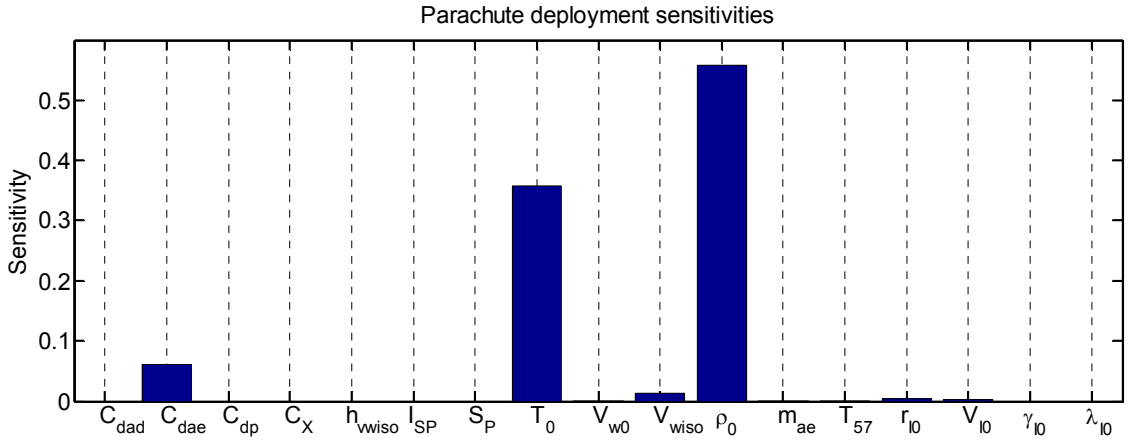


Figure 5.5: Global sensitivities on parachute deployment altitude

The uncertainty in the parachute deployment altitude comes mainly from the surface density and temperature which determine the atmospheric model. These terms affect the density in the upper atmosphere, which causes drag on the vehicle. The entry configuration drag coefficient is the other primary contributor to the drag acting on the vehicle. The values

make physical sense and match between the two analyses presented thus far. In this case however, the sensitivity to ρ_0 is roughly nine times higher than that on C_{dae}, due to this method giving sensitivities based on the variance ($\nu = \sigma^2$).

The second output investigated is the final percentage of propellant consumed. Figure 5.6 indicates that the parachute drag coefficient is the main contributor to the output uncertainty. Three other terms that have an effect are the engine specific impulse, surface density, and reference area of the parachute. Specific impulse is the only variable directly included in the calculation of propellant consumed. The parachute will affect the vehicle altitude at which the thrusters are switched on.

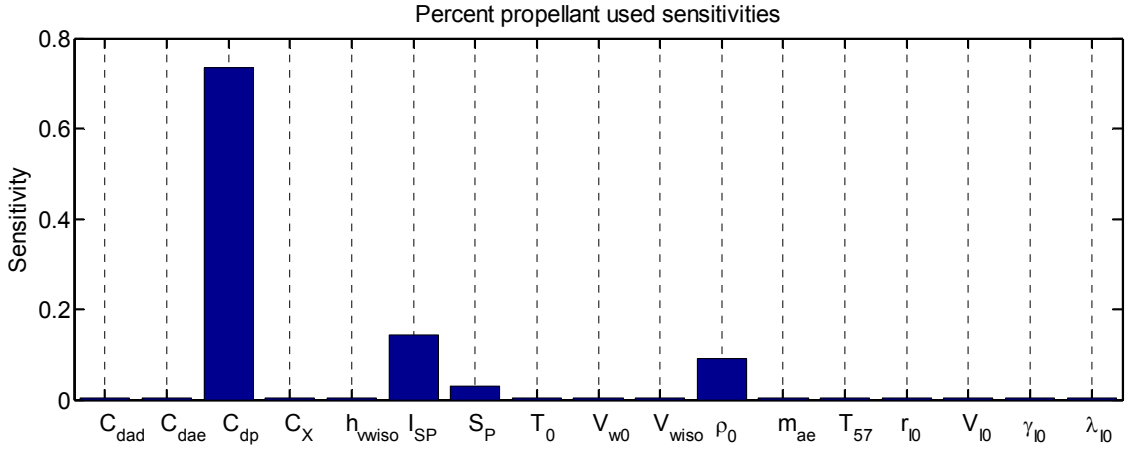


Figure 5.6: Global sensitivities on percent propellant used

A large number of runs were required to obtain these sensitivities. Figure 5.7 shows an example of the convergence for the sensitivity of parachute deployment altitude to surface temperature. It shows that around 6,000 runs are required for convergence. That is 6,000 runs for just this one input sensitivity. Referring to equation (5.20), for 17 inputs it takes 108,000 runs to obtain the main effect sensitivities.

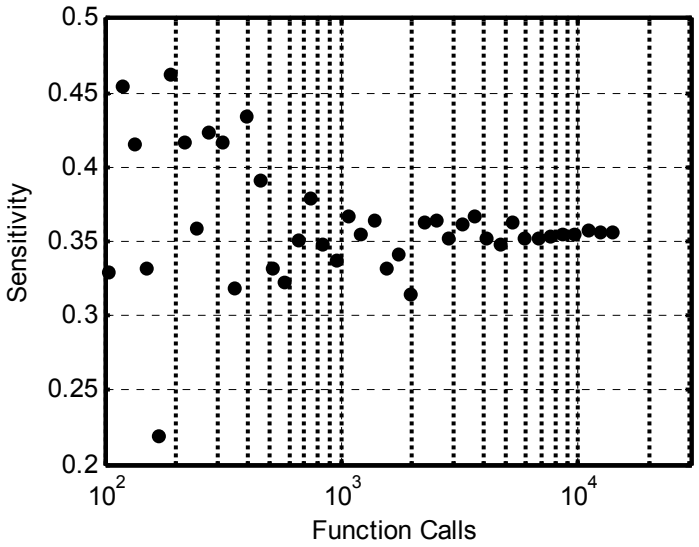


Figure 5.7: Parachute deployment altitude sensitivity convergence to surface temperature

An additional 6,000 runs will provide the total effects terms, which should indicate which variables have higher order couplings. An investigation of these terms does not indicate any significant couplings for this output.

5.3.1 Complications

Sobol's method had difficulty approximating the sensitivities to outputs with small output variances. Only 19 of the 44 tracked output variables converge on a solution. Of those, the outputs of interest are the total heat load, altitude at parachute deployment, longitude, maximum force and acceleration at parachute deployment, and the propellant consumed.

If convergence to an accurate sensitivity isn't required, but instead the relative sensitivities are acceptable, a smaller sample set could be run. After N = 2,000 runs (for each sensitivity) the top three contributors remained constant compared to each other for those 19 outputs. For example the top three contributors to the maximum acceleration at parachute deployment were C_{D_P}, C_X, and S_P in that order. Increasing N does not change that order. The actual sensitivity values haven't converged in 2,000 runs, but the relative values compared to each other have.

5.3.2 Conclusions

Knowledge of which variables directly influence the output uncertainty, along with which interactions between variables exist, is very useful. Testing on the EDL simulation proved questionable however. If the variance on the output is low the method has convergence issues. Applying this method to a POST simulation may provide valuable insights, but at a high computational cost.

6 Probabilistic Sensitivity Analysis

This method was published in 2008 by NASA LaRC engineers and a NIA staff scientist for approximating probabilistic sensitivities [5]. The results of PSA give the approximation of the sensitivities of output statistics to input statistics. These probabilistic sensitivities are found by taking the partial derivative of the mean or variance of the output with respect to the mean or variance of a specific input variable: $\partial\mu_{out}/\partial\mu_{in}$, $\partial\mu_{out}/\partial\nu_{in}$, $\partial\nu_{out}/\partial\mu_{in}$, or $\partial\nu_{out}/\partial\nu_{in}$.

6.1 Method Description

Calculation of the mean or variance can be found with the expected value operator in equation (6.1), where $y(\ )$ is an output function and $\overline{X}$ is the matrix of input parameters with known PDFs represented jointly as $p(\ )$. The input matrix $\overline{X}$ can have n dimensions with N samples. Two methods are proposed by Crespo [5] to obtain the sensitivities, the first of which is finite differencing of the expected output in equation (6.2), where θ_i is either the mean, μ_{x_i}, or variance, ν_{x_i}, of the input vector $\bar{x}_i$ for input variable x_i. Since $\overline{X}$ represents a sample set it has a mean and variance associated with each input. The variable of interest $\bar{x}_i$ can be transformed to reflect a perturbation of Δ to the μ_{x_i} or ν_{x_i}, which is incorporated into $\overline{X}_\Delta$ with all other inputs unchanged.

$$E\left[y\left(\overline{X}\right)\right] = \int y\left(\overline{X}\right)p\left(\overline{X}\right)d\overline{X} \tag{6.1}$$

$$\frac{\partial E\left[y\left(\overline{X}\right)\right]}{\partial\theta_i} \approx \frac{1}{\Delta}\left(E\left[y\left(\overline{X}_\Delta\right)\right] - E\left[y\left(\overline{X}\right)\right]\right) \tag{6.2}$$

The second method applies Leibniz's rule to find the derivative of the integral in the expected value operator in equation (6.1). Calculating the sensitivity requires moving the derivative inside the integral simplifying to equation (6.3). An in-depth derivation of equation (6.3) is found in Appendix C. In this equation $p_i(\ )$ is the PDF for $\bar{x}_i$. The

multidimensional integral in the expected value operator in equation (6.1) is integrated over the bounds of each input. Equation (6.3) includes the bounds of $\bar{x}_i$ as a and b. Sample set $\overline{X}_a$ has the input vector $\bar{x}_i$ replaced with the boundary value a for all samples, while leaving all other variables unchanged. Similarly $\overline{X}_b$ has b replacing the input vector $\bar{x}_i$.

$$\frac{\partial E\left[y\left(\overline{X}\right)\right]}{\partial \theta_i} = E\left[\frac{y\left(\overline{X}\right)}{p_i\left(\bar{x}_i\right)}\frac{\partial p_i\left(\bar{x}_i\right)}{\partial \theta_i}\right] + \frac{\partial b}{\partial \theta_i}p_i(b)E\left[y\left(\overline{X}_b\right)\right] - \frac{\partial a}{\partial \theta_i}p_i(a)E\left[y\left(\overline{X}_a\right)\right] \tag{6.3}$$

It is assumed that the partial derivatives of $p_i(\ )$ can be found analytically. Appendix D lists the partial derivatives for the PDFs used in this analysis. At first glance there is no apparent benefit over finite differencing in terms of computational expense: $y\left(\overline{X}\right)$ must be evaluated, then for each partial $y\left(\overline{X}_b\right)$ and $y\left(\overline{X}_a\right)$ must also be evaluated via a MCS method. However for PDFs with bounds at infinity, the second and third terms in equation (6.3) become zero (e.g. for a normal distribution $p(\infty) = 0$) reducing the number of cases simulated. If a set of cases are simulated to obtain the mean and variance of the outputs, e.g. $E\left[y\left(\overline{X}\right)\right]$, the partial derivatives of inputs with normal distributions can be obtained at no additional computational cost (i.e., no additional samples), since $y\left(\overline{X}_b\right)$ and $y\left(\overline{X}_a\right)$ need not be simulated. Similarly, equations can be defined for the sensitivity of the variance $V\left[y\left(\overline{X}\right)\right]$.

$$\frac{\partial V\left[y\left(\overline{X}\right)\right]}{\partial \theta_i} = E\left[\frac{\left(y\left(\overline{X}\right) - E\left[y\left(\overline{X}\right)\right]\right)^2}{p_i\left(\bar{x}_i\right)}\frac{\partial p_i\left(\bar{x}_i\right)}{\partial \theta_i}\right] + \frac{\partial b}{\partial \theta_i}p_i(b)V\left[y\left(\overline{X}_b\right)\right] - \frac{\partial a}{\partial \theta_i}p_i(a)V\left[y\left(\overline{X}_a\right)\right] \tag{6.4}$$

To provide a fair comparison between the derivatives, the outputs are given as percent derivatives. Calculation of percent derivatives requires defining step sizes. The step Δ is used as a percentage of the statistic to be perturbed. In this work Δ is set to 1% of the input variable mean if the partial derivative is with respect to the mean (similarly for the variance). If the perturbed mean is zero, then Δ is taken as 1% of the range. Where the range is twice the standard deviation. For finite differencing the step size is directly used in the equation. When using Leibniz's rule the step size is used to scale the results of the method.

Sample sets, for the calculation of the sensitivities, can be generated by several methods such as: Sobol sequence generator [18], pseudo-random number generator, or Halton sequence generator [10].

6.1.1 Advantages

Obtaining the sensitivity of an output to each input mean allows for the identification of the largest contributing input variable. Calculating the sensitivity to the variance on the inputs would show where the most benefits could be gained by performing further experimentation (such as analysis or hardware changes, etc) to reduce uncertainty on an input. Application of Leibniz's rule is promising because it requires the evaluation of only one sample set of N samples to obtain many sensitivities when the bounds of the input PDFs are infinite such as in a normal distribution.

6.1.2 Disadvantages

There are some limitations to this method. The analytic partial derivative of the PDF is required for the Leibniz approach. While no additional sets of runs are required to obtain sensitivities for normal distributions beyond a first set, there is an additional computational cost for PDFs having bounds that are functions of θ, such as uniform distributions. For the worst case scenario, the computational cost of using Leibniz's rule is equivalent to that required for finite differencing, since the computational cost of using Leibniz's rule is dependent on the types of input variable PDFs present. Therefore the entire analysis can be performed via the Leibniz's rule with the benefit of fewer simulations if there are input variables with normal distributions.

6.2 Application to Projectile Trajectory

The results of the application of Leibniz's rule to the projectile trajectory are shown in Table 6.1. Each value represents the percent change of the output from a 1% change in the indicated input. To produce these values the sample size N was set to be 100,000. The first

value is $\partial\mu_{X_f}/\partial\mu_g$, showing that a 1% change to the mean of the gravitational acceleration (0.1 m/s^2) causes a -0.83 % change to the mean of the final position. These sensitivities were also found by finite differencing, matching to within 2 significant figures.

Table 6.1: Sensitivities to projectile trajectory outputs via Leibniz's rule

	X_f	γ_f	V_f
$\partial\mu/\partial\mu_g$	-0.828	0.180	0.001
$\partial\mu/\partial\mu_{y_0}$	0.178	0.180	0.001
$\partial\mu/\partial\mu_{v_0}$	1.630	-0.369	0.997
$\partial\mu/\partial\mu_{\gamma_0}$	1.910	0.000	0.000
$\partial\mu/\partial\nu_g$	0.003	-0.000	-0.000
$\partial\mu/\partial\nu_{y_0}$	-0.000	-0.000	-0.000
$\partial\mu/\partial\nu_{v_0}$	0.007	0.005	-0.000
$\partial\mu/\partial\nu_{\gamma_0}$	0.314	0.319	-0.000
$\partial\nu/\partial\mu_g$	-2.061	-0.244	-0.003
$\partial\nu/\partial\mu_{y_0}$	-0.044	-0.243	-0.003
$\partial\nu/\partial\mu_{v_0}$	3.925	0.484	0.002
$\partial\nu/\partial\mu_{\gamma_0}$	2.177	0.002	0.001
$\partial\nu/\partial\nu_g$	0.009	0.000	-0.000
$\partial\nu/\partial\nu_{y_0}$	0.000	0.000	-0.000
$\partial\nu/\partial\nu_{v_0}$	0.081	0.004	0.999
$\partial\nu/\partial\nu_{\gamma_0}$	1.007	1.238	0.000

The sensitivities found match the physics of the model. Both the final velocity and flight path angle are dependent on the coupled term gy_0 in equations (2.4) and (2.5). The inputs g and y_0 only appear together within the equations. Therefore any sensitivity to gravitational acceleration or initial height on these outputs should have the same value as seen in the table. Final velocity is dependent mostly on the initial velocity, with almost a one

to one basis, $\partial\mu_{V_f}/\partial\mu_{v_0}$ = 0.997 and $\partial\nu_{V_f}/\partial\nu_{v_0}$ = 0.999. Final position is the most non-linear since it has several terms with significant sensitivities.

Inspecting the last four terms in the first column for final position, those showing the sensitivity of the output variance with respect to the input variance, shows the same trend as found with Sobol's method in Table 5.1. The output variance on final position is mostly affected by the initial flight path angle, followed by initial velocity, with a slight influence by gravitational acceleration.

It was mentioned that sensitivities to inputs with normal distributions can be obtained without simulating additional sample sets. In equation form, the computational cost to obtain sensitivities to the statistics is:

$$\text{Leibniz Cost} = N*(2*U+1) \tag{6.5}$$

Here U is the number of inputs with a uniform distribution, since the inputs had either normal or uniform distributions. The computational cost for the projectile trajectory to obtain Table 6.1 was 500,000 runs or $5N$ since two inputs had uniform distributions. Using finite differencing would have required $9N$. Obviously the benefits are increased by having more distributions with infinite bounds. The computational cost for finite differencing is dependent on the total number of inputs n. Each input requires the simulation of two additional sample sets to obtain all four possible sensitivities: $\partial\mu_{out}/\partial\mu_{in}$, $\partial\mu_{out}/\partial\nu_{in}$, $\partial\nu_{out}/\partial\mu_{in}$, and $\partial\nu_{out}/\partial\nu_{in}$.

$$\text{Finite Difference Cost} = N*(2*n+1) \tag{6.6}$$

For Table 6.1 N was set to 100,000 to obtain accurate results. Similar results could have been obtained if the sample set was reduced to 1,000 samples, as indicated by the convergence data in Figure 6.1 for the final position variance sensitivity to initial flight path angle variance. This plot shows the worst case convergence, since output variance requires two approximations via MCS numerical integration: $V\left[y(\overline{X})\right] = E\left[y(\overline{X})^2\right] - E\left[y(\overline{X})\right]^2$.

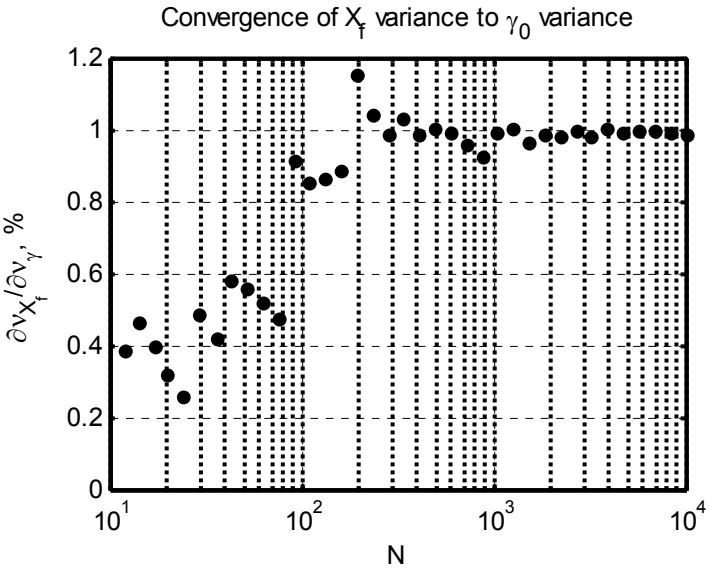


Figure 6.1: Typical convergence of PSA sensitivities via Leibniz's rule

6.2.1 Complications

Percent derivatives were employed to overcome scaling and dimensionality issues. Without doing so the sensitivities provided would not be a fair comparison. There is also the concern of how to deal with inputs that have zero mean when choosing the percent change. In this analysis they are arbitrarily set to a percentage of the range.

6.2.2 Conclusions

Application of Leibniz's rule provides sensitivities similar to Sobol's method at about the same computational cost for the projectile trajectory. Inputs with PDFs having infinite bounds allow the approximation of sensitivities without additional simulation of sample sets for each input. Since the computational cost to calculate sensitivities to inputs with uniform distributions is equivalent to finite differencing, and fewer evaluations are needed for inputs with normal distributions, Leibniz's rule can be used to calculate all sensitivities.

6.3 Application to EDL Simulation

A large amount of information is provided by PSA. For each output there are four partial derivatives per input. Thus for the EDL simulation there are 68 partial derivatives for each output (4*17 inputs). The parachute deployment altitude output is analyzed here. Four bar charts are presented, one for each set of partial derivatives.

First the sensitivities to the parachute deployment altitude mean with respect to the input means are shown in Figure 6.2. The initial states, were omitted from the following plots because they were correlated. The sensitivities are scaled to show how a 1% change in the mean surface temperature causes a 2.63% change in the mean parachute deployment altitude for example. This plot suggests that the mean surface temperature, surface density, and initial mass have the greatest influence on the mean parachute deployment altitude. These results make physical sense since the first two terms dictate the atmospheric density, while the last term will control how fast the vehicle is traveling.

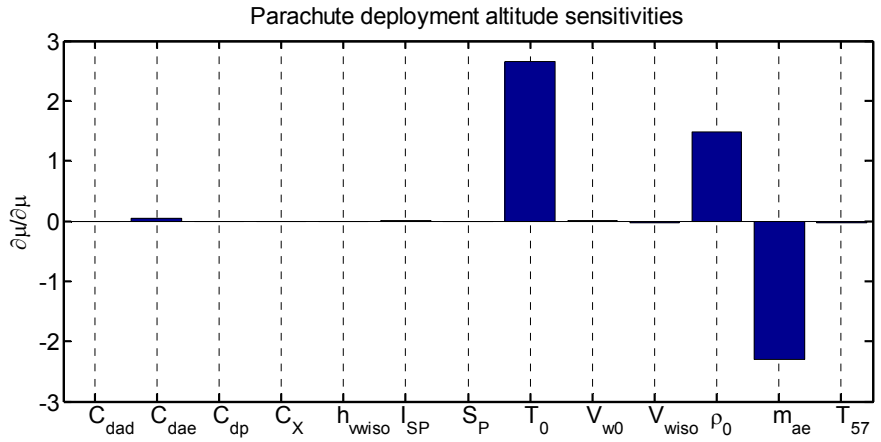


Figure 6.2: Sensitivity of parachute deployment altitude mean to each input mean

Next the sensitivity of the output mean with respect to the input variance can be investigated. Figure 6.3 shows that a change to the variance of any input has little effect on the output mean. The largest term only causes a 0.002% change to the output mean. While the larger terms in this plot are those that define the upper atmospheric conditions, this plot

appears to be influenced by numerical noise due to running a limited sample set (e.g., not running infinite simulations).

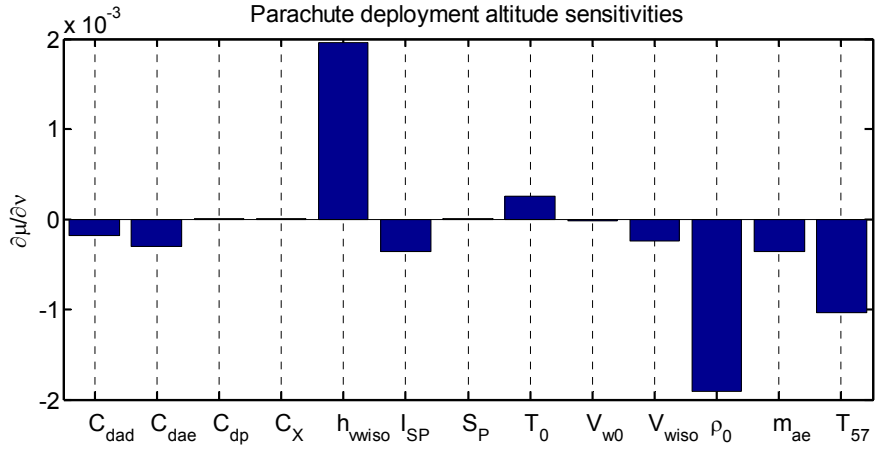


Figure 6.3: Sensitivity of parachute deployment altitude mean to each input variance

The sensitivity of the output variance due to the input mean is shown in Figure 6.4. Here there is only one term with a major influence, being the initial mass with a small effect from the surface temperature. Increasing the mass of the vehicle should reduce the effect of the aerodynamic forces acting upon it.

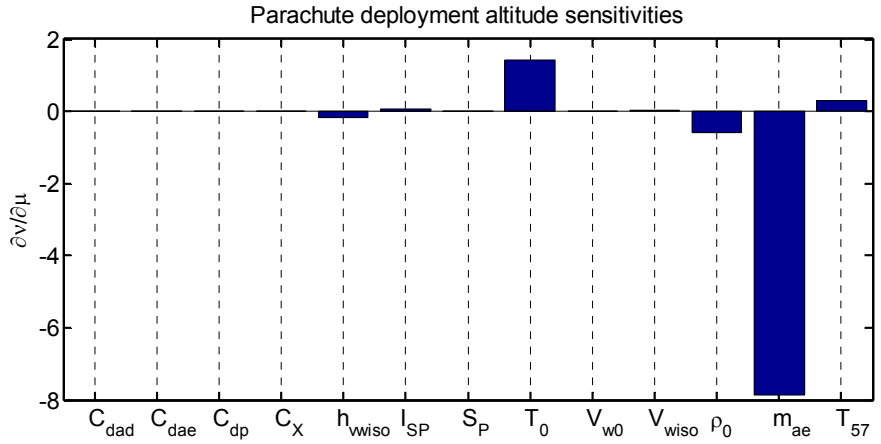


Figure 6.4: Sensitivity of parachute deployment altitude variance to each input mean

Finally the output variance sensitivity to input variance is shown in Figure 6.5. Again the atmospheric variables, surface temperature and density, are the main contributors. In

addition there is some influence from the coefficient of drag for the aeroshell. These results again match with those found by the previous two methods explored in this report. The sensitivity in this chart is based on the variance, thus if there are different variances assigned to the inputs C_{dae} and ρ_0, this variation will appear in the sensitivities. Again the sensitivity to ρ_0 is roughly nine times greater than that of C_{dae} since the variance is equal to σ^2 and there was a factor of three difference between the assigned 3σ values on those inputs.

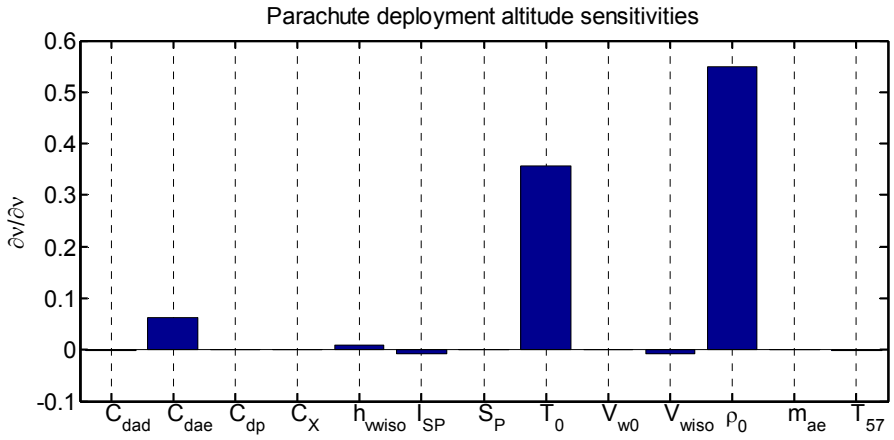


Figure 6.5: Sensitivity of parachute deployment altitude variance to each input variance

Additional insights into the EDL simulation can be gained by investigating another output variable. The following figures show the sensitivities of the final longitude at touchdown statistics with respect to each of the inputs. The final longitude, λ_f, is a measure of how far the vehicle traveled before touching down. Figure 6.6 shows the sensitivities of the mean λ_f with respect to the mean of each input. The two main contributors are the mass of the vehicle and the surface temperature. This information shows that increasing the mass will generally increase the distance traveled by the vehicle. Increasing the mass would decrease the deceleration of the vehicle caused by drag during entry, thus the horizontal component of the velocity decreases at a slower rate increasing the distance traveled.

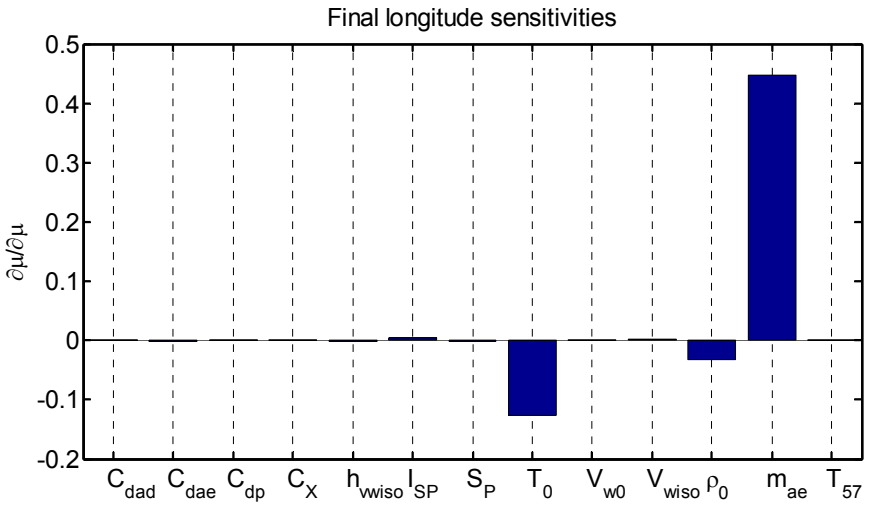


Figure 6.6: Sensitivity of final longitude mean to each input mean

Not shown are the sensitivities of the mean λ_f with respect to the variance on each input because the scale of the sensitivities is 10^{-4} indicating noise rather than true sensitivities. Next Figure 6.7 shows the sensitivities of the variance of λ_f with respect to the mean of each input. The main contributor is again the mean mass of the vehicle. An increase to the mean mass by 1% causes a decrease in the λ_f variance by about 7%.

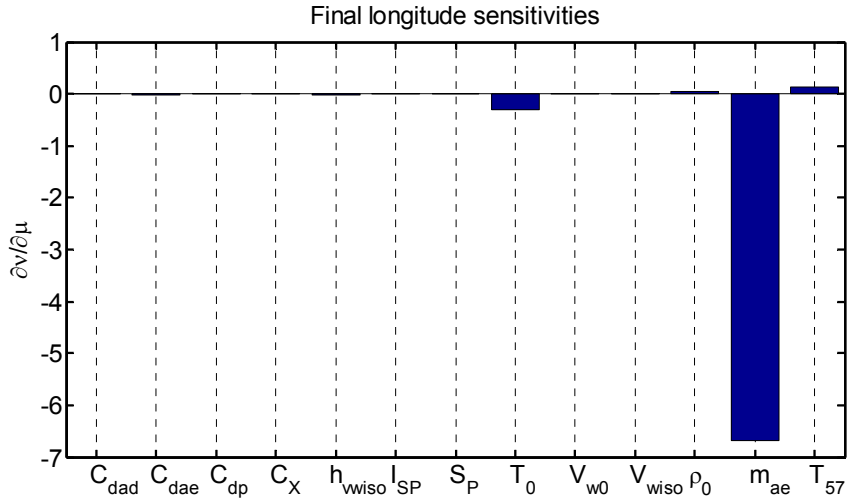


Figure 6.7: Sensitivity of final longitude variance to each input mean

The final chart to be investigated is Figure 6.8 showing the sensitivities of the λ_f variance with respect to the variance of each input. The top two contributors are the surface temperature and surface density which dictate the atmospheric density. Increasing the variance on these two inputs will increase the variance on the density which affects entry and descent, the two segments of EDL where the vehicle has the greatest horizontal movement. To reduce the variance on λ_f an engineer would gain the most benefit by reducing the uncertainties in T_0 and ρ_0. The ratio between the sensitivities to ρ_0 and C_{dae} is about nine in Figure 6.8, again due to the increased uncertainty associated with ρ_0.

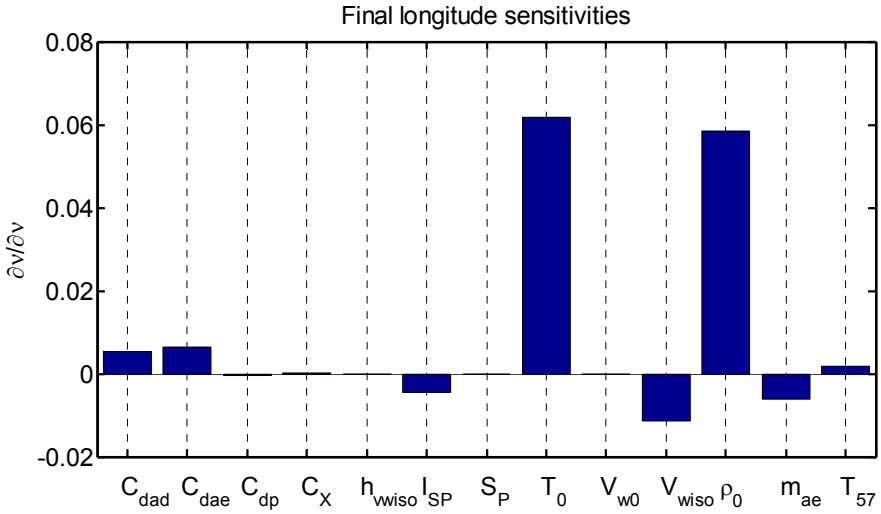


Figure 6.8: Sensitivity of final longitude variance to each input variance

The sensitivities shown above were obtained using a sample size N of 15,000 to give accurate results. Figure 6.9 shows that the method converges within 5,000 samples. Using equation (6.5) with $U = 6$, since there are six inputs with uniform distributions, gives 65,000 total function calls. Calculating these sensitivities using finite differencing would have required 175,000 function calls to obtain the same results.

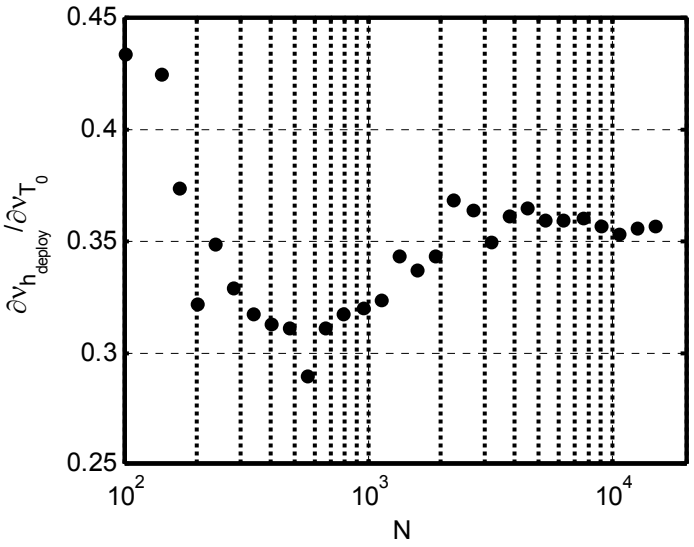


Figure 6.9: Convergence of parachute deployment altitude to surface temperature

6.3.1 Complications

The previous sections ignored the initial states. The partial derivatives required by Leibniz's rule are difficult to calculate for correlated variables. Since the four ignored inputs are correlated, the joint PDF associated with them cannot be separated when calculating the analytic partial derivatives as can be done with independent inputs.

6.3.2 Conclusions

This method reduces computational cost by providing sensitivities to any inputs having a normal distribution without simulating additional sample sets. The results found match with what is calculated by finite differencing. Given a situation where partial derivatives of the statistics to an input set having several normally distributed inputs are desired, application of Leibniz rule has clear advantages. It provides the same approximations as finite differencing at a reduced computational cost.

7 Conclusions

This study analyzed the merits of three methods that provide insights into the statistics of EDL simulations. The failure domain bounding method promised to provide a more accurate approximation of the failure probability with less runs than a typical MCS analysis. Global variance-based sensitivities and probabilistic sensitivity analysis are methods that provide the sensitivity of the output statistics to the input statistics. These three methods were applied to a 2D EDL simulation to assess their potential.

FDB was able to successfully identify a safe region. When applied to the EDL simulation an optimizer required 5,754 runs to find a hypersphere radius defining the safe region. For the variables that dictate failure, the radius vector also gives a sense of the sensitivities to the inputs. The safe region was found to encompass an insignificant portion of the design space due to a spreading effect of increasing the input dimensions. A modification was implemented to partially fix the problem. Instead of identifying one tiny safe region, it was modified to find several regions, than combine them giving a larger safe region. With the given failure constraints convergence to the probability of failure was equivalent to MCS. FDB gives better results when there are fewer constraints, smaller failure probability, or fewer input variables.

Sobol's method successfully gave the sensitivities to each input variable. Accurate results required 108,000 runs for main effect sensitivities. Inspection of the total effects terms did not indicate many significant couplings. It was found that the method did not work well if the output variance was small. This method works better with large output variances and less input variables. With longer run times a surrogate model could be used to alleviate the computational cost.

The PSA method also successfully gave the sensitivities to the input statistics. While it does not provide a direct sense of couplings that may be present, it does provide sensitivities to both the mean and variance. It required 65,000 runs for accurate results, providing similar information to Sobol's method at a fraction of the computations. PSA

provides the same information as finite differencing, again with fewer computations. The method requires the analytic partial derivatives of the PDFs with respect to the mean and variance, which was not an issue for the situation studied, but could be for other PDFs. Application of Leibniz's rule to variables that are correlated requires further study. Sensitivities of the failure probability or to correlated inputs can still be obtained via finite differencing. The benefits of PSA are augmented with an increase in the number of input variables defined with normal distributions, or other distributions with infinite bounds.

References

[1] Archer G., Saltelli A., Sobol' I.M., "Sensitivity measures, ANOVA like techniques and the use of bootstrap", *Journal of Statistical Computation and Simulation*. V. 58 (1997) 99-120.

[2] Blake, W.W., "Experimental Aerodynamic Characteristics of the Viking Entry Vehicle Over the Mach Range 1.5 - 10.0", NASA CR-159225, 1971.

[3] Braun, R.D., Manning, R.M., "Mars Exploration Entry, Descent and Landing Challenges", *2006 IEEE Aerospace Conference*, IEEEAC paper #0076, Big Sky, Montana, March 2006.

[4] Crespo L.G., Giesy D.P., Kenny S.P.. "Reliability-based Analysis and Design via Failure Domain Bounding", *Structural Safety*, 31, July 2009, 306-315.

[5] Crespo L.G., Kenny S.P., Giesy D.P.. "Sampling-based Strategies for the Estimation of Probabilistic Sensitivities", *AIAA Conference in Non-deterministic Approaches*, Palm Springs, California, 4-7 May 2009. AIAA-2009-2283.

[6] Desai, P., Schoenenberger, M, and Cheatwood, F. M., "Mars Exploration Rover Six-Degree-of-Freedom Entry Trajectory Analysis" *Journal of Spacecraft and Rocket*s, Vol. 43, No. 5, Sept-Oct 2006.

[7] Flaherty, T. "Viking Aerodynamics Data Book, VER-10", Martin Marietta Corp., Denver Division, TR-3709014, Revision D, January 1973.

[8] Giesy D.P., Crespo L.G., Kenny S.P.. "Approximation of Failure Probability Using Conditional Sampling", *12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Victoria, Canada, 10-12 September 2008. AIAA 2008-5946.

[9] Kaplan, D., Environment of Mars, 1988, NASA TM-100470, 1988.

[10] Kocis, L., and Whiten W. J., "Computational Investigations of Low-Discrepancy Sequences", *ACM Transactions on Mathematical Software*, Vol. 23, No. 2, June 1997, 266-294.

[11] Powell, R.W., et. al., "Program to Optimize Simulated Trajectories: Volume II, Utilization Manual", Version 1.1.1.G, NASA Langley Research Center; Lockheed Martin Corporation, May 2000.

[12] Regan, F. J. and Anandakrishnan, S. M. *Dynamics of Atmospheric Reentry*. Washington DC: American Institute of Aeronautics and Astronautics, Inc., 1993.

[13] Saltelli, A., "Making Best Use of Model Evaluations to Compute Sensitivity Indices", *Computer Physics Communications*, Vol. 145, pp. 280-297, 2002.

[14] Seiff, A. "Post-Viking Models for the Structure of the Summer Atmosphere of Mars". *Adv. Space Res*. Vol. 2, pp. 3-17, 1982.

[15] Sobol' I.M.. "Sensitivity estimates for nonlinear mathematical models", *Matematicheskoe Modelirovanie*, V. 2, N 1, 1990, 112-118 (in Russian), English translation in : *Mathematical Modeling and Computational Experiment*, V. 1, N 4, 1993, 407-414.

[16] Tolson, R. H. *Space Flight Guidance and Navigation*, MAE 783. Spring 2009.

[17] Weisstein, Eric W. "Haar Function." From MathWorld--A Wolfram Web Resource. http://mathworld.wolfram.com/HaarFunction.html, Accessed October 2008

[18] http://www.broda.co.uk, Accessed September 2008

Appendices

Appendix A: Generating Correlated Entry States

An entry trajectory analysis first requires a set of initial conditions comprising position and velocity vectors. These states contain six variables as vector components in an inertial Cartesian system.

$$\{R\} = \begin{Bmatrix} x \\ y \\ z \end{Bmatrix} \quad \{V\} = \begin{Bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{Bmatrix} \tag{A.1}$$

Combining these terms into one column vector gives the state vector.

$$\{S\} = \begin{Bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{Bmatrix} \tag{A.2}$$

To perform a MCS analysis numerous perturbations of the initial conditions must be chosen randomly. Most of the variables involved in this report are perturbed independently; however the states are not independent having a covariance matrix $\boldsymbol{\Gamma}$.

$$\boldsymbol{\Gamma} = \begin{bmatrix} \sigma_x^2 & \rho_{xy}\sigma_x\sigma_y & \rho_{xz}\sigma_x\sigma_z & \rho_{x\dot{x}}\sigma_x\sigma_{\dot{x}} & \rho_{x\dot{y}}\sigma_x\sigma_{\dot{y}} & \rho_{x\dot{z}}\sigma_x\sigma_{\dot{z}} \\ \rho_{xy}\sigma_x\sigma_y & \sigma_y^2 & \rho_{yz}\sigma_y\sigma_z & \rho_{y\dot{x}}\sigma_y\sigma_{\dot{x}} & \rho_{y\dot{y}}\sigma_y\sigma_{\dot{y}} & \rho_{y\dot{z}}\sigma_y\sigma_{\dot{z}} \\ \rho_{xz}\sigma_x\sigma_z & \rho_{yz}\sigma_y\sigma_z & \sigma_z^2 & \rho_{z\dot{x}}\sigma_z\sigma_{\dot{x}} & \rho_{z\dot{y}}\sigma_z\sigma_{\dot{y}} & \rho_{z\dot{z}}\sigma_z\sigma_{\dot{z}} \\ \rho_{x\dot{x}}\sigma_x\sigma_{\dot{x}} & \rho_{y\dot{x}}\sigma_y\sigma_{\dot{x}} & \rho_{z\dot{x}}\sigma_z\sigma_{\dot{x}} & \sigma_{\dot{x}}^2 & \rho_{\dot{x}\dot{y}}\sigma_{\dot{x}}\sigma_{\dot{y}} & \rho_{\dot{x}\dot{z}}\sigma_{\dot{x}}\sigma_{\dot{z}} \\ \rho_{x\dot{y}}\sigma_x\sigma_{\dot{y}} & \rho_{y\dot{y}}\sigma_y\sigma_{\dot{y}} & \rho_{z\dot{y}}\sigma_z\sigma_{\dot{y}} & \rho_{\dot{x}\dot{y}}\sigma_{\dot{x}}\sigma_{\dot{y}} & \sigma_{\dot{y}}^2 & \rho_{\dot{y}\dot{z}}\sigma_{\dot{y}}\sigma_{\dot{z}} \\ \rho_{x\dot{z}}\sigma_x\sigma_{\dot{z}} & \rho_{y\dot{z}}\sigma_y\sigma_{\dot{z}} & \rho_{z\dot{z}}\sigma_z\sigma_{\dot{z}} & \rho_{\dot{x}\dot{z}}\sigma_{\dot{x}}\sigma_{\dot{z}} & \rho_{\dot{y}\dot{z}}\sigma_{\dot{y}}\sigma_{\dot{z}} & \sigma_{\dot{z}}^2 \end{bmatrix} \tag{A.3}$$

Off-diagonal terms have a linear correlation coefficient ρ. It is desirable to randomly choose each perturbation independently, which requires a transformation. Before getting into the transformation the dimensions must be reduced since in this analysis a 2D simulator is

being utilized requiring a simplification of the six initial states to the variables r_I, λ_I, v_I, and γ_I.

$$r_I = \sqrt{x^2 + y^2 + z^2} \tag{A.4}$$

$$\lambda_I = \cos^{-1}\left(\frac{\{R_n\}^T \cdot \{R_p\}}{\|R_n\| \cdot \|R_p\|} \right) \tag{A.5}$$

$$v_I = \sqrt{\dot{x}^2 + \dot{y}^2 + \dot{z}^2} \tag{A.6}$$

$$\gamma_I = \frac{\pi}{2} - \cos^{-1}\left(\frac{\{V_p\}^T \cdot \{R_p\}}{\|V_p\| \cdot \|R_p\|} \right) \tag{A.7}$$

The subscript n represents the nominal state, while the subscript p is for vectors projected into the 2D plane defined by the nominal position and velocity vectors. In Equation (A.5) the sign is dictated by whether the projected position is down track (positive) or not.

A sensitivity matrix $\mathbf{A}$ was developed to transform the 6x6 $\mathbf{\Gamma}$ matrix to a 4x4 matrix allowing the generation of four independent variables instead of six. The superscripts are added to the covariance matrices to represent the size of the matrix.

$$\mathbf{A} = \begin{bmatrix} \frac{\partial r}{\partial x} & \frac{\partial r}{\partial y} & \frac{\partial r}{\partial z} & \frac{\partial r}{\partial \dot{x}} & \frac{\partial r}{\partial \dot{y}} & \frac{\partial r}{\partial \dot{z}} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} & \frac{\partial v}{\partial z} & \frac{\partial v}{\partial \dot{x}} & \frac{\partial v}{\partial \dot{y}} & \frac{\partial v}{\partial \dot{z}} \\ \frac{\partial \gamma}{\partial x} & \frac{\partial \gamma}{\partial y} & \frac{\partial \gamma}{\partial z} & \frac{\partial \gamma}{\partial \dot{x}} & \frac{\partial \gamma}{\partial \dot{y}} & \frac{\partial \gamma}{\partial \dot{z}} \\ \frac{\partial \lambda}{\partial x} & \frac{\partial \lambda}{\partial y} & \frac{\partial \lambda}{\partial z} & \frac{\partial \lambda}{\partial \dot{x}} & \frac{\partial \lambda}{\partial \dot{y}} & \frac{\partial \lambda}{\partial \dot{z}} \end{bmatrix} \tag{A.8}$$

$$\mathbf{\Gamma}^4 = \mathbf{A}\mathbf{\Gamma}^6\mathbf{A}^T \tag{A.9}$$

The transformation is accomplished by calculating a matrix φ whose columns are the eigenvectors of $\mathbf{\Gamma}^4$, which can be used to diagonalize $\mathbf{\Gamma}^4$.

$$\mathbf{\Gamma}_{\mathbf{d}} = \varphi^T \mathbf{\Gamma}^4 \varphi \tag{A.10}$$

Because $\mathbf{\Gamma}^4$ is positive definite symmetric, $\mathbf{\Gamma}_{\mathbf{d}}$ exists and has the eigenvalues of $\mathbf{\Gamma}^4$ on the diagonal with off-diagonal elements equal to zero. Regan describes this process thoroughly in the chapter on error analysis [12]. The orthogonal rotation matrix φ defines the linear combination of perturbations that provide linearly independent random values [16]. The square root of $\mathbf{\Gamma}_{\mathbf{d}}$ gives the independent standard deviations of the states as a diagonal 4x4 matrix $\boldsymbol{\sigma}$. Now a set of random values can be chosen to perturb the four simplified states independently as a 4x1 array $\{P_{rand}\}$. With this information the perturbations to the initial states are:

$$\{\Delta S\} = \varphi^4 \boldsymbol{\sigma} \{P_{rand}\} \tag{A.11}$$

$$\{S\} = \{S_n\} + \{\Delta S\} \tag{A.12}$$

Appendix B: Derivation of Sobol Global Sensitivities

Description of Haar Function

Haar functions are described on Wolfram MathWorld [17] as:

$$\psi(x) \equiv \begin{cases} 1 & 0 \le x < 1/2 \\ -1 & 1/2 < x \le 1 \\ 0 & otherwise \end{cases} \tag{B.1}$$

$$\psi_{jk}(x) \equiv \psi(2^j x - k) \tag{B.2}$$

for a nonnegative integer j and $0 \le k \le 2^j - 1$. A function can be written as a series expansion:

$$f(x) = c_0 + \sum_{j=0}^{\infty} \sum_{k=0}^{2^j-1} c_{jk} \psi_{jk}(x) \tag{B.3}$$

The functions ψ_{jk} and ψ are orthogonal on [0,1] such that:

$$\int_0^1 \psi(x) \psi_{jk}(x) dx = 0 \tag{B.4}$$

$$\int_0^1 \psi_{jk}(x) \psi_{lm}(x) dx = 0 \tag{B.5}$$

for $(j,k) \neq (0,0)$ in (B.4) and $(j,k) \neq (l,m)$ in (B.5). Inputs are defined in a unit n-dimensional cube:

$$x = x_1, \ldots x_n$$

$$K^n = \{x \mid 0 \le x_i \le 1; i = 1, \ldots, n\} \tag{B.6}$$

The function $f(x)$, equation (B.3), is a model that can be calculated at any given point *x*. All integrations to follow are from 0 to 1 for each variable.

Fourier-Haar Series Expansion

A function can be represented by an expansion into summands of different dimensions based on the Haar function replacing ψ with $f(\ )$:

$$f(x_1,...,x_n) = f_0 + \sum_{i=1}^{n} f_i(x_i) + \sum \sum_{1 \le i < j \le n`} f_{ij}(x_1, x_j) + ... + f_{1,2,...,n} \tag{B.7}$$

Sobol calls this a Fourier-Haar series expansion [15]. This approach is similar to performing a Fourier series approximation; however the individual functions do not need to calculated. The function approximations are only needed to show that the total variance can be decomposed into individual components.

Orthogonality

Integrating (B.7) with respect to the input variables from 0 to 1 givies zero for each individual function and leaving only the constant mean value f_0. This is occurs because the area under each function sums to zero. That is the functions have a value of -1 over a portion of the domain and a value of +1 over the same size portion with a value of zero everywhere else as seen in (B.1). The portion of the domain with a non-zero value is different depending on the subscript in equation (5.1). For example f_i is -1 on [0 ½] and +1 on [½ 1], but f_{ij} may be -1 on [0 ¼] and +1 on [¼ ½], zero everywhere else. The functions are only defined on the unit hypercube [0,1]. Assuming that the integrals of $f_{i1,...is}$ with respect to their 'own' variables are zero:

$$\int_0^1 f_{i1,...is}(x_{i1},...,x_{is}) dx_{i_k} = 0 \quad ; \quad 1 \le k \le s \tag{B.8}$$

Thus all the summands are orthogonal:

$$\int_{K^n} f_{i1,...,is} f_{j1,...,jl} dx = 0 \quad \text{if} \quad (i_1,...,i_s) \ne (j_1,...,j_s) \tag{B.9}$$

Since at least one index is not repeated. From the above relations it can be shown that:

$$f_0 = \int_{K^n} f(x)dx \tag{B.10}$$

Variance Calculation

Any function can be represented by a unique Fourier-Haar series with input defined on K^n and a constant mean value of f_0. The variance can be computed:

$$V(X) = \int (x-\mu)^2 p(x)dx \tag{B.11}$$

where $p(x)$ is the probability density function. Equation (B.11) can be restated as:

$$V(X) = E(X^2) - [E(X)]^2 \tag{B.12}$$

where $E(\ )$ is the expected value operator. Looking at the first term of (B.12) in terms of the Fourier-Haar representation:

$$E(X^2) = \int x^2 p(x)dx \tag{B.13}$$

$$E(f(x)^2) = \int_a^b [f(x)]^2 \frac{1}{b-a} dx = \int [f(x)]^2 dx \tag{B.14}$$

For the second term in (B.12):

$$E(X) = \int xp(x)dx \tag{B.15}$$

$$E(f(x)) = \int_a^b f(x)\frac{1}{b-a} dx = \int f(x)dx = f_0 \tag{B.16}$$

where $a = 0$ and $b = 1$. Thus the variance becomes:

$$V(f(x)) = \int f^2(x)dx - f_0^2 \tag{B.17}$$

For any individual or combination of variables, similar expressions can be developed:

$$E(f_i^2) = \int f_i^2 dx_i \tag{B.18}$$

$$E(f_i) = \int f_i dx_i = 0 \tag{B.19}$$

by (B.8), or integration with respect to its 'own' variable.

$$V(f_i(x_i)) = \int f_i^2(x_i) dx_i \tag{B.20}$$

Thus all sets of variances for each variable can be computed. By squaring and integrating the function representation (B.7):

$$V = \sum V_i + \sum\sum V_{ij} + ... + V_{ij...n} \tag{B.21}$$

Sensitivity

Now the global sensitivity can be estimated by the ratio of any individual or combination of variances to the total variance.

$$S = \frac{V_i}{V} \tag{B.22}$$

$$\sum S_i + \sum\sum S_{ij} + ... + S_{ij...n} = 1 \tag{B.23}$$

Algorithm

For calculation of sensitivities to functions, such as a black box calculation, Sobol divides the input set into two groups $x = \{y, z\}$; where $y = \{x_i, ..., x_s\}$ and $z = \{x_{s+1}, ..., x_n\}$. Now the expansion can be shown with subscript 1 for *y* and 2 for *z* as:

$$f(x) = f_0 + f_1(y) + f_2(z) + f_{12}(y, z) \tag{B.24}$$

The variances are:

$$V_1 = \int f_1^2(y) dy \tag{B.25}$$

$$V_2 = \int f_2^2(z) dz \tag{B.26}$$

$$V = \int f^2(x)dx - f_0^2 \tag{B.27}$$

But to calculate the variance in a computer environment, the f_1 and f_2 terms must be removed. In doing so a new set of independent random variables $t = \{u, v\}$ is introduced. This set is similar to the original set *x* in dimensions.

$$V_1 = \int f_1^2(y)dy = \int \left(\int f(x)dz - f_0 \right)^2 dy \tag{B.28}$$

$$= \int \left(\int f(x)dz \right)^2 dy - 2f_0 \int\int f(x)dzdy + \int f_0^2 dy = \int \left(\int f(x)dz \right)^2 dy - 2f_0^2 + f_0^2 \tag{B.29}$$

Now where the function is squared the second independent set of variables is introduced. Since the variance with respect to *y* is desired, the subset *y* is kept as input with both function calls. However the subset *z* is replaced with the subset *v* in the second function call.

$$V_1 = \int f(y,z)f(y,v)dzdvdy - f_0^2 \tag{B.30}$$

Similarly for the variance of subset *z*.

$$V_2 = \int f(y,z)f(u,z)dydudz - f_0^2 \tag{B.31}$$

At this point all the necessary integrals can be calculated using MCS methods as $N \to \infty$:

$$f_0 = \int f(x)dx \approx \frac{b-a}{N} \sum_{i=1}^{N} f(x_i) \tag{B.32}$$

again $(b-a) = (1-0) = 1$. For each of the required integrals:

$$f_0 \approx \frac{1}{N} \sum_{i=1}^{N} f(y_i, z_i) \tag{B.33}$$

$$V + f_0^2 \approx \frac{1}{N} \sum_{i=1}^{N} f^2(y_i, z_i) \tag{B.34}$$

$$V_1 + f_0^2 \approx \frac{1}{N} \sum_{i=1}^{N} f(y_i, z_i) f(y_i, v_i) \tag{B.35}$$

$$V_2 + f_0^2 \approx \frac{1}{N}\sum_{i=1}^{N} f(y_i, z_i) f(u_i, z_i) \tag{B.36}$$

The left hand side of the above equations can then be used to calculate sensitivities. There are advances in the literature that decrease the function evaluations, decrease errors, and increase convergence. Pseudo-random Monte Carlo methods work well, however low discrepancy sequences, which pick numbers in a way that attempt to guarantee uniformity across the design space, are better. If the sensitivity to a single variable is desired, then subset *y* is set just to that variable while *z* is all other variables in *x*.

$$y = \{x_i\} \qquad z = \{x \neq x_i\} \qquad \Rightarrow S_i \tag{B.37}$$

Similarly for coupled terms:

$$y = \{x_i, x_j\} \qquad z = \{x \neq x_i, x_j\} \qquad \Rightarrow S_{ij} \tag{B.38}$$

Appendix C: Derivation of Probabilistic Sensitivities via Leibniz's Rule

Application of Leibniz's Rule to the Expected Value:

Application of Leibniz's rule is discussed in [5], however the derivation of the equations are left to the reader. The final equations are developed here for completeness:

$$\frac{\partial}{\partial z}\int_{a(z)}^{b(z)} f(x,z)dx = \int_{a(z)}^{b(z)} \frac{\partial f(x,z)}{\partial z}dx + f(b(z),z)\frac{\partial b}{\partial z} - f(a(z),z)\frac{\partial a}{\partial z} \tag{C.1}$$

Consider a two input system:

$$E\left[y(\bar{x}_1,\bar{x}_2),\theta_1,\theta_2\right] = \int_{a(\theta_1)}^{b(\theta_1)}\int_{c(\theta_2)}^{d(\theta_2)} y(\bar{x}_1,\bar{x}_2)p(\bar{x}_1,\bar{x}_2,\theta_1,\theta_2)d\bar{x}_2 d\bar{x}_1 \tag{C.2}$$

where $p(\ )$ is the joint probability density function. To be concise: $\bar{X} = [\bar{x}_1,\bar{x}_2]$; $\theta = [\theta_1,\theta_2]$. Also the bounds and PDF will no longer include the independent variable θ, e.g. $a(\theta_1)$ will just be a & $p_1(\bar{x}_1,\theta_1)$ will be $p_1(\bar{x}_1)$. If the inputs are independent then the joint PDF can be separated giving:

$$E\left[y(\bar{X}),\theta\right] = \int_a^b\int_c^d y(\bar{X})p_1(\bar{x}_1)p_2(\bar{x}_2)d\bar{x}_2 d\bar{x}_1 \tag{C.3}$$

The parameters, θ_i, are for the PDF, such as the μ (mean) and v (variance) for normal distributions. Below θ_1 represents some parameter of interest that is in p_1. Also E is shorthand for $E\left[y(\bar{X}),\theta\right]$ unless otherwise specified:

$$\frac{\partial E}{\partial \theta_1} = \frac{\partial}{\partial \theta_1}\int_a^b\int_c^d y(\bar{X})p_1(\bar{x}_1)p_2(\bar{x}_2)d\bar{x}_2 d\bar{x}_1 \tag{C.4}$$

Application of Leibniz's rule gives:

$$\frac{\partial E}{\partial \theta_1} = \int_a^b \frac{\partial}{\partial \theta_1}\left(\int_c^d y(\bar{X})p_1(\bar{x}_1)p_2(\bar{x}_2)d\bar{x}_2\right)d\bar{x}_1 + \ldots \tag{C.5}$$

$$\frac{\partial b}{\partial \theta_1}\int_c^d y(b,\bar{x}_2)p_1(b)p_2(\bar{x}_2)d\bar{x}_2 - \frac{\partial a}{\partial \theta_1}\int_c^d y(a,\bar{x}_2)p_1(a)p_2(\bar{x}_2)d\bar{x}_2 \tag{C.6}$$

Focusing on equation (C.5) Leibniz's rule can be invoked again, however this time the limits of integration are not functions of θ_1, so the additional terms disappear $\left(\frac{\partial c}{\partial \theta_1} = 0\right)$. Rearranging to simplify notation:

$$\int_a^b \int_c^d \frac{y(\bar{X})}{p_1(\bar{x}_1)} \frac{\partial p_1(\bar{x}_1)}{\partial \theta_1} p_1(\bar{x}_1) p_2(\bar{x}_2) d\bar{x}_2 d\bar{x}_1 \tag{C.7}$$

$$= E\left[\frac{y(\bar{X})}{p_1(\bar{x}_1)} \frac{\partial p_1(\bar{x}_1)}{\partial \theta_1}\right] \tag{C.8}$$

Now looking at the remaining terms in equation (C.6):

$$\frac{\partial b}{\partial \theta_1} \int_c^d y(b, \bar{x}_2) p_1(b) p_2(\bar{x}_2) d\bar{x}_2 = \frac{\partial b}{\partial \theta_1} p_1(b) E[y(b, \bar{x}_2)] \tag{C.9}$$

$$\frac{\partial a}{\partial \theta_1} \int_c^d y(a, \bar{x}_2) p_1(a) p_2(\bar{x}_2) d\bar{x}_2 = \frac{\partial a}{\partial \theta_1} p_1(a) E[y(a, \bar{x}_2)] \tag{C.10}$$

Combining equations (C.8), (C.9), and (C.10) gives the final formula:

$$\frac{\partial E}{\partial \theta_1} = E\left[\frac{y(\bar{X})}{p_1(\bar{x}_1)} \frac{\partial p_1(\bar{x}_1)}{\partial \theta_1}\right] + \frac{\partial b}{\partial \theta_1} p_1(b) E[y(b, \bar{x}_2)] - \frac{\partial a}{\partial \theta_1} p_1(a) E[y(a, \bar{x}_2)] \tag{C.11}$$

Application of Leibniz's Rule to the Variance:

$$V[y(\bar{X}), \theta] = E\left[(y(\bar{X}) - E)^2\right] \tag{C.12}$$

$$V[y(\bar{X}), \theta] = E\left[y(\bar{X})^2 - 2y(\bar{X})E + E^2\right] \tag{C.13}$$

Note that the joint PDF is used assuming independent inputs ($p(\bar{X}) = p_1(\bar{x}_1) p_2(\bar{x}_2)$). Also V will be shorthand for $V[y(\bar{X}), \theta]$ unless otherwise specified.

$$\frac{\partial V}{\partial \theta_1} = \frac{\partial}{\partial \theta_1} \int_a^b \int_c^d (y(\bar{X}) - E)^2 p(\bar{X}) d\bar{x}_2 d\bar{x}_1 \tag{C.14}$$

Application of Leibniz's rule gives:

$$= \int_a^b \frac{\partial}{\partial \theta_1} \left(\int_c^d \left(y(\bar{X}) - E \right)^2 p(\bar{X}) d\bar{x}_2 \right) d\bar{x}_1 + \ldots \tag{C.15}$$

$$\frac{\partial b}{\partial \theta_1} p_1(b) \int_c^d \left(y(b, \bar{x}_2) - E[y(b, \bar{x}_2), \theta] \right)^2 p_2(\bar{x}_2) d\bar{x}_2 - \ldots \tag{C.16}$$

$$\frac{\partial a}{\partial \theta_1} p_1(a) \int_c^d \left(y(a, \bar{x}_2) - E[y(a, \bar{x}_2), \theta] \right)^2 p_2(\bar{x}_2) d\bar{x}_2 \tag{C.17}$$

The last two terms ((C.16) and (C.17)) can be expressed as:

$$\frac{\partial b}{\partial \theta_1} p_1(b) V[y(b, \bar{x}_2)] - \frac{\partial a}{\partial \theta_1} p_1(a) V[y(a, \bar{x}_2)] \tag{C.18}$$

Returning to the first term (C.15), Leibniz's rule can be applied again, however the bounds c and d are not functions of θ_1 so the partial derivatives will be zero. Ignoring the additional terms and applying the chain rule to (C.15):

$$\int_a^b \int_c^d \frac{\partial}{\partial \theta_1} \left(\left(y(\bar{X}) - E \right)^2 \right) p(\bar{X}) + \left(y(\bar{X}) - E \right)^2 \frac{\partial}{\partial \theta_1} \left(p(\bar{X}) \right) d\bar{x}_2 d\bar{x}_1 \tag{C.19}$$

Expanding and solving each component of the first derivative in the equation (C.19):

$$\frac{\partial}{\partial \theta_1} \left(y(\bar{X})^2 - 2y(\bar{X})E + E^2 \right) = 0 - 2y(\bar{X}) \frac{\partial E}{\partial \theta_1} + 2E \frac{\partial E}{\partial \theta_1} \tag{C.20}$$

Returning (C.20) into the integration:

$$\int_a^b \int_c^d \left(-2y(\bar{X}) \frac{\partial E}{\partial \theta_1} + 2E \frac{\partial E}{\partial \theta_1} \right) p(\bar{X}) d\bar{x}_2 d\bar{x}_1 \tag{C.21}$$

Evaluating the first term:

$$-2 \frac{\partial E}{\partial \theta_1} \int_a^b \int_c^d y(\bar{X}) p(\bar{X}) d\bar{x}_2 d\bar{x}_1 = -2 \frac{\partial E}{\partial \theta_1} E \tag{C.22}$$

Evaluating the second term:

$$2E\frac{\partial E}{\partial \theta_1}\int_a^b\int_c^d p(\bar{X})d\bar{x}_2 d\bar{x}_1 = 2\frac{\partial E}{\partial \theta_1}E(1) \tag{C.23}$$

Therefore:

$$\int_a^b\int_c^d \frac{\partial}{\partial \theta_1}\left(\left(y(\bar{X})-E\right)^2\right)p(\bar{X})d\bar{x}_2 d\bar{x}_1 = 0 - 2\frac{\partial E}{\partial \theta_1}E + 2E\frac{\partial E}{\partial \theta_1}(1) = 0 \tag{C.24}$$

Leaving only the second term in equation (C.19) which can be expressed as:

$$\int_a^b\int_c^d \left(y(\bar{X})-E\right)^2\frac{\partial}{\partial \theta_1}\left(p(\bar{X})\right)d\bar{x}_2 d\bar{x}_1 \tag{C.25}$$

$$= \int_a^b\int_c^d \frac{\left(y(\bar{X})-E\right)^2}{p_1(\bar{x}_1)}\frac{\partial p_1(x_1)}{\partial \theta_1}p_1(\bar{x}_1)p_2(\bar{x}_2)d\bar{x}_2 d\bar{x}_1 \tag{C.26}$$

$$= E\left[\frac{\left(y(\bar{X})-E\right)^2}{p_1(\bar{x}_1)}\frac{\partial p_1(x_1)}{\partial \theta_1}\right] \tag{C.27}$$

Combining what was developed in (C.18) and (C.27) gives:

$$\frac{\partial V}{\partial \theta_1} = E\left[\frac{\left(y(\bar{X})-E\right)^2}{p_1(\bar{x}_1)}\frac{\partial p_1(x_1)}{\partial \theta_1}\right] + \frac{\partial b}{\partial \theta_1}p_1(b)V\left[y(b,\bar{x}_2)\right] - \frac{\partial a}{\partial \theta_1}p_1(a)V\left[y(a,\bar{x}_2)\right] \tag{C.28}$$

Appendix D: Equations for Relevant Probability Density Functions

Normal probability density function:

$$p(\bar{x},\mu,\sigma)=\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(\bar{x}-\mu)^2}{2\sigma^2}\right) \tag{D.1}$$

$$\frac{\partial p}{\partial \mu}=\frac{1}{\sqrt{2\pi}}\frac{(\bar{x}-\mu)}{\sigma^3}\exp\left(-\frac{(\bar{x}-\mu)^2}{2\sigma^2}\right) \tag{D.2}$$

$$\frac{\partial p}{\partial \nu}=\frac{1}{\sigma^3 2\sqrt{2\pi}}\left(\frac{(\bar{x}-\mu)^2}{\sigma^2}-1\right)\exp\left(-\frac{(\bar{x}-\mu)^2}{2\sigma^2}\right) \tag{D.3}$$

Uniform probability density function:

$$p(a,b)=\frac{1}{b-a} \tag{D.4}$$

$$\mu=\frac{a+b}{2} \qquad \sigma=\frac{b-a}{\sqrt{12}} \tag{D.5}$$

$$a=\mu-\sqrt{3}\sigma \qquad b=\mu+\sqrt{3}\sigma \tag{D.6}$$

$$p(\mu,\sigma)=\frac{1}{2\sqrt{3}\sigma} \tag{D.7}$$

$$\frac{\partial p}{\partial \mu}=0 \tag{D.8}$$

$$\frac{\partial p}{\partial \nu}=\frac{-1}{4\sqrt{3}\sigma^3} \tag{D.9}$$