Pattern Recognition for a Flight Dynamics Monte Carlo Simulation

Carolina Restrepo*
NASA Johnson Space Center, Houston, TX.
and John E. Hurtado†
Texas A&M University, College Station, TX.

The design, analysis, and verification and validation of a spacecraft relies heavily on Monte Carlo simulations. Modern computational techniques are able to generate large amounts of Monte Carlo data but flight dynamics engineers lack the time and resources to analyze it all. The growing amounts of data combined with the diminished available time of engineers motivates the need to automate the analysis process. Pattern recognition algorithms are an innovative way of analyzing flight dynamics data efficiently. They can search large data sets for specific patterns and highlight critical variables so analysts can focus their analysis efforts. This work combines a few tractable pattern recognition algorithms with basic flight dynamics concepts to build a practical analysis tool for Monte Carlo simulations. Current results show that this tool can quickly and automatically identify individual design parameters, and most importantly, specific combinations of parameters that should be avoided in order to prevent specific system failures. The current version uses a kernel density estimation algorithm and a sequential feature selection algorithm combined with a k-nearest neighbor classifier to find and rank important design parameters. This provides an increased level of confidence in the analysis and saves a significant amount of time.

I. Introduction

Spacecraft design is inherently difficult due to the nonlinearity of the systems involved as well as the expense of testing hardware in a realistic environment. The number and cost of flight tests can be reduced by performing extensive simulation and analysis work to understand vehicle operating limits and identify circumstances that lead to mission failure. A Monte Carlo simulation approach that varies a wide range of physical parameters is typically used to generate an umbrella of test scenarios. The results of these analyses bound the vehicle performance and eventually help certify a spacecraft for flight. NASA's Orion vehicle is a current example of the importance and benefits of the Monte Carlo design approach.[1]

As in any engineering problem, identifying variables that can drive the design is crucial. These variables need to be analyzed more thoroughly to ensure safety and reliability of the spacecraft. For a human-rated spacecraft, identifying the variables that could cause failures is particularly important. Given enough time, the Monte Carlo approach allows analysts to identify most of the individual design variables that influence certain system failures. This is a long and meticulous process because it involves the analysis of thousands of simulation runs. Engineers seek to pinpoint a few individual influential variables that directly affect a particular system requirement in order to address the necessary changes in the design. However, it is typically not the individual parameters that lead to critical failures such as missing a landing target, sub-optimal parachute deployment, or high g-forces on the crew. It is a series of complex variable interactions that cause these anomalies due to the high level of coupling throughout the flight of a spacecraft. Determining which variable combinations cause system failures is essential in the final design and testing phases, and they are

*Aerospace Engineer, Integrated GN&C Analysis Branch, AIAA student member, *carolina.i.restrepo@nasa.gov*
†Associate Professor, Aerospace Engineering Department, AIAA member, *jehurtado@tamu.edu*

extremely difficult to track down with a manual analysis of Monte Carlo data.

Currently, there is no general methodology that can be used to identify individual variables, or their critical interactions in a reliable and timely manner for a flight dynamics problem. There have been several methods developed to identify which variable uncertainties have a greater effect on the outcome of a simulation, but in most cases the algorithms are specific to a particular problem and require the analyst to write additional code, or to manipulate and re-run the Monte Carlo simulation. These are significant obstacles that must be addressed in order to automate the data analysis process. To overcome them, the authors have strived to solve this problem from *the perspective on a flight dynamics engineer who does not necessarily have access to the simulation, but is tasked with the analysis of a set of Monte Carlo data from a spacecraft designed by someone else.*

More specifically, the goal of this work is to develop a general methodology that can be used to analyze flight dynamics data for problems with a small number of design parameter but especially for problems with thousands of design parameters such as the Orion vehicle. In the past, the analysis of Monte Carlo data for problems with a relatively small number of design variables has been addressed in a number of ways, but the analysis of data for fully integrated spacecraft has mostly been performed manually on an individual basis by a great number of people working simultaneously. In fact, there are several recent publications that show how Monte Carlo data is used and analyzed for NASA's newest spacecraft.[2–4] The lack of a general methodology for the analysis of flight dynamics Monte Carlo data is evident.

On the other hand, aerospace problems with a smaller number of variables than a high-fidelity spacecraft simulation, have served as great test problems to develop a number of innovative analysis methods. Perhaps the most intuitive method to find individual influential variables is to perform a sensitivity analysis of all output parameters with respect to all input parameters, though this typically requires access to the model equations and to write additional pieces of code. This is an obstacle for an engineer that does not own the simulation.

Statistical methods such as Modern Design Of Experiments (MDOE)[5] and ANalysis Of VAriance (ANOVA)[6] have been used effectively to allocate how much of the output variance is due to the variance of the different inputs. Problems in the aerospace field have been addressed with this method[7,8] with the goal of understanding the sources of variance in experimental data. The goal of the Monte Carlo analysis problem is not to allocate the amount of variance among the different inputs, but to understand the interaction of the variance of each design parameter that was purposefully introduced by the analyst in the form of a simulation input file.

Another probability-based approach to the problem of characterizing input uncertainties that ultimately result in a system failure is to iteratively expand or reduce a region in the input space that contains a certain probability of failure. Reference[9] describes an algorithm that starts with a well-defined subset of the input uncertainty space and iteratively modifies it based on whether or not it encapsulates dispersed points that meet a certain performance criteria. A similar approach in the sense that new test input vectors are generated and analyzed iteratively to narrow down a critical input space is presented in.[10] Even though both of these papers demonstrate the ability to narrow down certain influential variables in their systems, the methods manipulate the Monte Carlo input deck and re-run the simulation. This is one of the obstacles the authors wish to overcome.

One last related approach is the use of the Markov Chain Monte Carlo algorithm as in reference.[11] The authors assume that input parameters have Gaussian distributions and use a Markov Chain to generate successive samples of inputs that would likely generate failures in the output space. The authors feel that making assumptions about the parameter input space could yield results that cannot be used in a certification task. Once again, using a non-deterministic technique to generate new samples and re-running the simulation would not be an option if the analyst does not have access to the simulation.

Another novel approach that has been used for identifying influential variables in a Monte Carlo data set is polynomial chaos. The method requires writing problem-specific code because the model equations must be reformulated. This is undesirable because the tool must be applicable to a wide range of problems. However, references[12, 13] use a modified non-intrusive version of the method that does not require modifying the system equations. They treat the simulation as a black box, and perform the analysis around it. This might seem a very practical approach, but if the analyst cannot fully understand and keep track of what the algorithm is doing, he or she cannot trust the method, especially when it comes to the certification of a human-rated vehicle. In fact, since one of the characteristics of this tool is to use tractable algorithms that a flight dynamics engineer can trust, the use of non-deterministic algorithms is not an option. This includes methods such as neural networks, or any other kind of method that assumes that the simulation is a black box.

The methods discussed above may be effective for small scale aerospace problems, but unfortunately cannot be generalized to complex flight dynamics problems. Currently, and as far as the authors are aware of, the only other attempt to develop a standard methodology for the analysis of flight dynamics Monte Carlo data is the work done by K. Gundy-Burlet et. al[14, 15] at the NASA Ames Research Center. The group has developed a method to analyze flight dynamics data which uses a combination of pattern recognition methods to identify qualitative trends between inputs and outputs. They create success maps of the most correlated variables. The work herein is similar in the sense that both seek to automate the complex data analysis task that NASA flight dynamics engineers face today through the use of pattern recognition, but different in the sense that it aims for a more deterministic answer, a *concrete ranking of influential parameters and most influential variable combinations that directly affect a specific system failure.*

This work provides a systematic way of listing the most influential variables for each particular failure metric sets the stage for a qualitative analysis of the physics of the problem, and reduces the number of variables requiring analysis from several hundreds or even thousands, to a short list of influential variables. This allows the analyst to reliably assume that the remaining parameters do not influence a particular failure metric and consequently do not need detailed analysis. This is valuable because it saves time and increases the level of confidence with which the design can be validated and certified. Time and confidence levels are important to reduce the design cycle costs which currently involve hundreds of engineers generating terabytes worth of Monte Carlo data and spending months analyzing said data.

A simple example, a satellite spinning about one of its axes, is used throughout this paper to explain the method and benefits of the analysis tool. The paper is organized as follows. Section II briefly introduces the simulation example. Section III discusses the challenges with current analysis methods and how this research contributes to the automation of this process. Section IV presents the use of pattern recognition methods in a flight dynamics problem. Section V, and section VI summarizes the contributions.

II. A Simple Flight Dynamics Problem

To illustrate the use of the Monte Carlo analysis tool, an example taken from reference[16] will be used throughout the rest of the paper. Consider a satellite represented by a rectangular box with a control moment gyro (CMG), that spins about one of its axes. The equations of motion are the following:

$$\dot{\omega}_1 = \frac{(I_2 - K_3)\omega_2\omega_3 + h\omega_3}{K_1} \tag{1}$$

$$\dot{\omega}_2 = \frac{(K_3 - K_1)\omega_1\omega_3 - u}{I_2} \tag{2}$$

$$\dot{\omega}_3 = \frac{(K_1 - I_2)\omega_1\omega_2 - h\omega_1}{K_3} \tag{3}$$

$$h = u \tag{4}$$

where $\omega_1, \omega_2, \omega_3$ are the angular velocities, I_1,I_2,I_3, and J_t are the inertias of the satellite and tangential inertia of the CMG respectively, h is the constant angular momentum, u is the CMG torque, and $K_1 = I_1 + J_t$, $K_3 = I_3 + J_t$.

The nominal motion of the satellite is simulated with given values of inertias and initial values of angular velocities. The motion is considered stable when there is one dominant angular velocity and two small angular velocities.[17] The motion is considered unstable when the three angular velocities are approximately the same magnitudes. A Monte Carlo simulation is performed by varying the box dimensions as well as the initial angular velocities to find out which design parameters are most influential in the spin stability of the satellite. Figure 1 shows the resulting Monte Carlo trajectories.

III. Analysis of a Flight Dynamics Monte Carlo Simulation

Designing a spacecraft is an iterative process that begins with the current vehicle design and configuration, current information about parameter uncertainties, and a set of nominal initial conditions for a nominal trajectory. The design is simulated and tested thousands of times through Monte Carlo simulations. The resulting set of simulations yield information about the probability of success of the spacecraft and its subsystems when operated under many different circumstances. The data is then analyzed by flight dynamics engineers who design guidance, navigation and control algorithms for the fully integrated vehicle. If the current design cannot perform adequately, the analysts then recommend changes for the next iteration.

The goal of a Monte Carlo simulation is to understand all critical design sensitivities that may prevent the design from meeting requirements. It is relatively easy to determine if a given simulation run in a set fails requirements. Typically, analysts look at a set of runs in which the failed ones have been somehow labeled as such. For example, Figure 1 is a time history of the satellite angular velocities where the red trajectories have failed the stability requirement and the blue trajectories successfully meet the stability criteria established in section II.

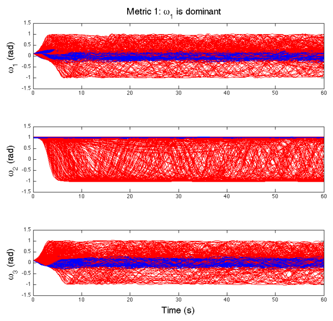


Figure 1: Satellite Trajectories

It is well known that a box that spins about its intermediate inertia is unstable. However, if this result was unknown, the first step in the analysis process would be to plot the different input and output parameters of the unstable simulation runs and try to track down the source of the problem. Flight dynamics engineers would look at a series of trajectory plots and scatter plots. They might see fit to plot combinations of two or three variables at a time to understand their relationship visually. Since there is no way of telling which of these variables should be plotted together, other than engineering intuition, this process can take several months for the thousands of variables in a spacecraft simulation. Realistically, it is not possible to plot and analyze every single combination of variables, so there is no guarantee that the analyst will be able to capture every problematic aspect of the design. In other words, the best way of obtaining confidence in the analysis is to be extremely familiar with the spacecraft design and the odds and ends of the simulation. This is one of the drawbacks of the way today's engineers analyze flight dynamics simulations. The tool developed here still requires a good understanding of the physics of a flight dynamics problem, but it does not require intimate knowledge of the parameter dispersions. A flight dynamics engineer will be able to use this analysis tool without having designed the specific spacecraft themselves, which will save a significant amount of work and time.

IV. Pattern Recognition and Flight Dynamics

Pattern recognition can be defined in several different ways. In general, it is the process of automatically finding common features and patterns in a data set with the purpose of describing the data, fitting a model, or classifying data points into classes that help us understand it better. References[18, 19] contain more precise definitions of pattern recognition and provide the mathematical background for the algorithms used in this problem.

There are two basic concepts, features and patterns, that are used in the field of pattern recognition. A feature is any characteristic that describes an object, and a pattern is a combination of specific features that can describe the object in a particular way. Features are informative when they provide information that differentiates an object from other objects. Features are not informative when the information they provide is not helpful in discriminating one object from another.

For a flight dynamics problem, the goal is to create a tool that can automatically rank individual variables (features) and combinations of variables (patterns) that are most relevant to the problem at hand. To create these rankings, there are two things that are taken into account: correlation between variables, and the separability of good data points from bad data points. More specifically, the goal is to find subspaces in the input and output parameter space that allow the successful cases be most separable from the failure cases.

The specific algorithms used in the analysis tool were selected based on the constraints listed below. The constraints are derived from the fact that the problem is addressed from the perspective on a flight dynamics engineer who does not have access to the simulation and does not own the design of the particular vehicle that must be analyzed.

1. Algorithm Constraints:
   (a) Algorithms are for *post-processing* data only and cannot rely on iteratively running several Monte Carlo sets
   (b) Algorithms must make *no assumptions about input probability density functions*
   (c) Algorithms must *compare all types of parameters at once* regardless of their units or relative magnitudes
   (d) Algorithms must filter out all obvious variable correlations that do not affect a particular failure. In order to reduce analysis time, the analyst must feel confident that detailed analysis of filtered variables is not necessary and focus solely on those influential parameters
2. Usability Constraints:
   (a) The tool must be general enough to address any GN&C issue that arises for any flight vehicle
   (b) The tool must be specific enough to capture subtleties buried in large data sets while ignoring obvious variable correlations that are not informative and do not affect system failures
   (c) Non-intrusive algorithms that do not require modification of existing code or writing new pieces of problem-specific code
   (d) Tractable methods that an aerospace engineer can *trust* and *understand* without being an expert in the fields of statistics or pattern recognition
   (e) Yield consistent results each time the algorithm is used on the same data sets (no heuristics)

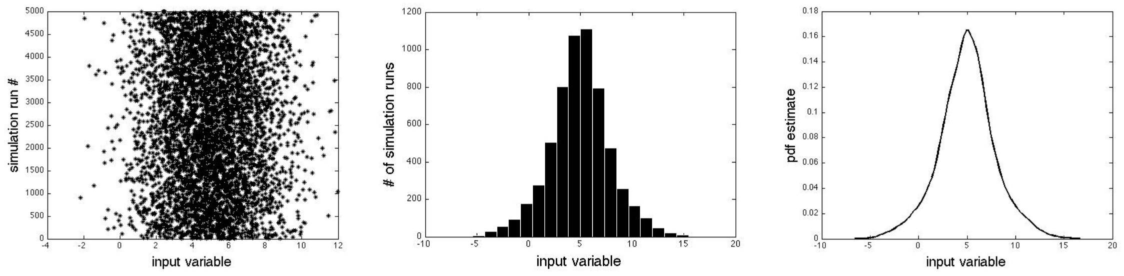
The current version of the analysis tool uses a non-parametric density estimation method, kernel density estimation, to identify individual influential variables and a k-nearest neighbors method to identify influential variable interactions that lead to system failures.

V. Analysis Tool

To further justify the rationale for the selected algorithms, consider again the satellite example, specifically the trajectory plot from Figure 1. It is necessary to find trends that are common among failed simulation runs but that are not common among successful runs in order to understand what causes the satellite to go unstable. The goal of the analysis tool is to automatically rank the design variables according to how useful they are in explaining the problem, given a metric for success. The best way to narrow down the variable space in search for the most influential variables is to highlight differences. And to highlight differences it is important to accurately describe the data, which implies that no assumptions should be made about the input distributions. Non-parametric density estimation methods, namely the kernel density estimation (KDE) method, and the k-Nearest Neighbors (k-NN) method are used here to describe and subsequently compare the data. Section A explains the basic theory of the kernel density estimation method, and how it is used to identify individual influential variables in a flight dynamics problem. Section B explains the k-NN method and how it is used to identify critical combinations of variables.

A. Identification of Individual Influential Variables

The KDE method[19] is similar to a smooth histogram. Figure 2a shows a scatter plot of a given input variable to a Monte Carlo simulation vs. the simulation run number, Figure 2b shows a histogram of the data, and Figure 2c shows a density estimate of the data using the KDE method. The KDE estimate is a normalized probability density function of the data, but the x-axis of 2c still preserves the original units of that input variable. This is a significant advantage over methods that require data normalization before any comparison between variables can be performed.



(a) Input Variable (b) Histogram (c) KDE

Figure 2: Example of Non-parametric Density Estimation

KDE is useful in understanding flight dynamics data when a density estimate is calculated separately for the failed simulation runs and the successful runs. Figure 4 demonstrates how the plotting both KDE curves can easily highlight which design variables are the best discriminators between successful and failed simulation runs. The variables that have one curve on top of the other are those that have little to no effect on the given success metric, whereas the variables that significantly different red and blue curves clearly show the trends that differentiate successful runs from failed ones.

The variables are then ranked according to how different the red and blue KDE curves are. Figure 4 shows them in the correct order, but a bar graph of the relative difference between the curves of each variable tells the user which variables are relatively more important to analyze. For example, variables 4, 5, 6, 8, and 10 are not individually affecting the stability metric, and variable 11, the width of the box, was the same for all simulation runs in the set no there is no difference between the curves and the bar height is zero.

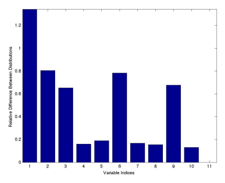
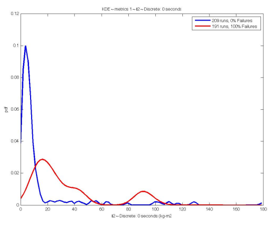


Figure 3

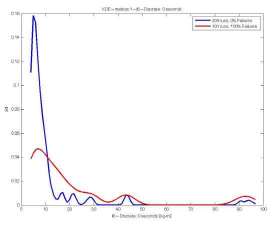
B. Identification of Influential Combinations of Variables

From Figure 4, it is clear that even though some of the top ranked variables are informative, none of them can completely separate the successful cases from the failed cases. For example, the second inertia curves show that the failed simulation runs tend to have values for I_2 between 0 and 60 and between 80 and 100. However, the successful runs also have a value between 0 and 20. So the region $I_2 = 0 - 20$ must be explored further because it is not possible to discriminate between success or failure. In other words, the single variable I_2 cannot provide a clear explanation, only a useful pointer for the user to include that variable in further analysis. As mentioned previously, even though it is important to identify the individual parameters that significantly influence system failures, these seldom cause failures by themselves, so their combinations and interactions are what matters. Typically, further analysis means combining or co-plotting the *interesting* individual variables with other variables to seek a more concrete answer. This section explains how this analysis tools helps with this process.

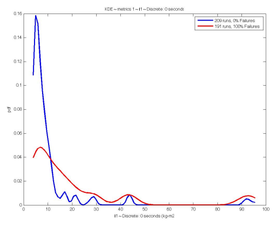
In section II it was established that what causes spin stability is to spin about the axis of the intermediate inertia. Since the success metric ensures that ω_2 is the dominant angular velocity, the satellite goes unstable when the value I_2 is between I_1 and I_3. Therefore, the subspace of variables in which all successful cases are completely separable from failed cases must involve a region encapsulated by I_1, I_2, and I_3.



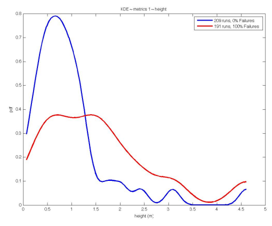
(a) I_2



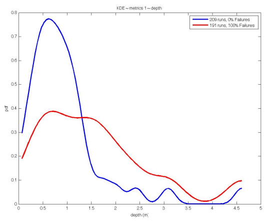
(b) I_3



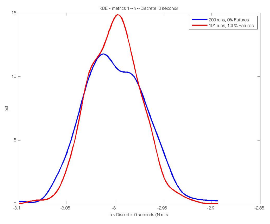
(c) I_1



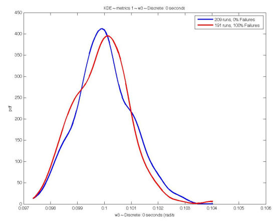
(d) satellite height



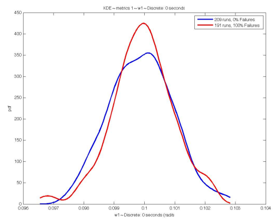
(e) satellite depth



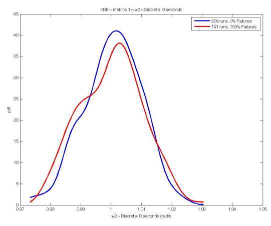
(f) CMG angular momentum



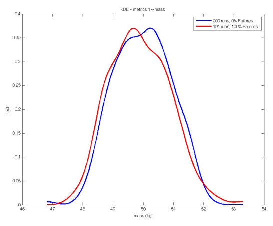
(g) $\omega_3(t = 0)$



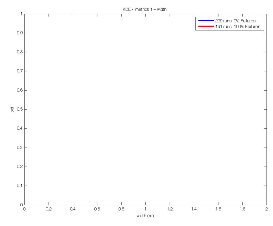
(h) $\omega_1(t = 0)$



(i) $\omega_2(t = 0)$



(j) satellite mass



(k) satellite width

Figure 4: KDE estimates for individual variables
Success metric: Stability

It is necessary to check those higher dimensional regions that might contain the answer to our question without forcing the flight dynamics engineer to select these regions manually. After some experimentation with a few example problems, including flight dynamics data from the Orion vehicle, the authors found that it is a reasonable assumption to consider only higher dimensional regions of up to four variables. Regions that contain more that four variables proved to be difficult to analyze.

Computing and analyzing an exhaustive list of four-variable combinations for a system with thousands of parameters is still too time consuming for an analyst. In order to alleviate this process, it was decided that all original variables will be combined into pairs, both as a difference and as a ratio, to form a new set of variables. Subsequently, these new variables are combined once again in the form of a two-dimensional region that is automatically searched for data separability and ranked accordingly through the use of the k-NN algorithm. The following steps summarize this process.

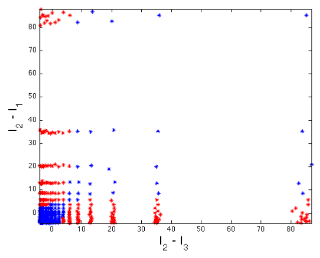
The total number of new variables (2D combinations) is calculated by

$$\#\ of\ new\ variables = \left(\frac{n!}{2!(n-2)!}\right) * (\#\ of\ arrangements) \tag{5}$$

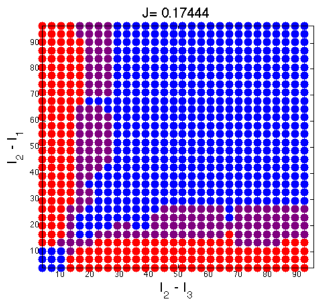
where n is the original number of dispersed variables, and the number of arrangements is two: for any pair of variables a, b, the possible (non-redundant) arrangements are $a - b$, and $\frac{a}{b}$. This is still a very large number of variables, but even though it is difficult to select which variables are the most important, it is safe to use a solid flight dynamics background to reduce the list by taking out those variables that are known to be independent of a specific failure.

Subsequently, the k-nearest neighbors algorithm[19] is used to determine the separability of the successful runs from the failed runs by creating a map of the 2D region and calculating the overlap between the red and the blue areas. These regions are then ranked using this overlap as a performance index: the least amount of overlap, the more informative the region is to the analyst. Figure 5a shows a scatter plot of two of the new variables generated from the original variables as described above. The successful region is completely separate from the failure region, and this is precisely the explanation for the satellite instability: I_2 must not be the intermediate inertia. Figure 5b shows a map of the region created with the k-NN algorithm. For visualization purposes, the overlap is colored purple, and the performance index is the purple percentage of the 2D region. For this example, the percentage is approximately 17%.

The analysis tool provides a ranking of the two-dimensional regions. The exact shape of the region doesn't matter as much as the fact that the three inertia values are the variables that interact together to cause a simulation run to fail. For a general flight vehicle, the nonlinear interactions that are to be avoided are typically certain dynamic or vibration modes that are catastrophic to the vehicle or the crew. These behaviors occur when certain parameters, namely aerodynamics and mass properties, interact in complex nonlinear ways. This method is capable of finding nonlinear regions as complex as the checker board shape of the satellite inertias plot. This would not be possible to find with a method that did not do the calculations based on local information of the data. These plots also confirm the reasons for not using any methods that are based on correlation coefficients or on assumptions of the shape of input parameter distributions. The authors are aware that such mapping could be achieved with artificial intelligence based methods, but again, the goal is for a flight dynamics engineer to be able to track and trust this analysis tool enough to aid in a certification task, without being an expert on statistics of pattern recognition.



(a)



(b)

Figure 5

VI. Summary of Contributions

The Monte Carlo data analysis tool developed has proved to be helpful in the analysis of general flight dynamics problems. The methods used are tractable algorithms that can be understood by an aerospace engineer, but above all, trusted. It has allowed analysts to save significant amounts of time by requiring a single Monte Carlo data set rather than multiple sets. The tool can be used to analyzed *any* flight dynamics problem because it does not require writing additional code or modifying any of the model equations or simulation input files. It can be used to compare simulation outputs without the need to normalize any of the data or losing the original units of any of the variables.

The general methodology developed here has proved to be useful in the analysis of problems as simple as the satellite spin stability example in this paper, and as complex as the Orion ascent abort guidance, navigation, and control algorithms. Additionally, the authors are currently in the process of programming the algorithms on a GPU and developing a graphical user interface for this tool.

Acknowledgments

The authors wish to acknowledge the support of the Johnson Space Center under project A3720. Any opinions, findings and conclusions or recommendations in this material are those of the authors and do not necessarily reflect the views of the National Aeronautics and Space Administration.

References

[1] "Crew Exploration Vehicle system requirements document," NASA CEV Document: CxP-72000, January 2007.

[2] Mark C. Jackson and Timothy Straube, "Orion flight performance design trades," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, Houston, TX, August 2010, NASA Johnson Space Center, number 2010-8443.

[3] John Davidson, Jennifer Madsen, Ryan Proud, Deborah Merrit, David Raney, Dean Sparks, Paul Kenyon, Richard Burt, and Mike McFarland, "Crew Exploration Vehicle ascent abort overview," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, August 2007, number 2007-6590 in AIAA.

[4] Dean Sparks and David Raney, "Crew Exploration Vehicle launch abort controller performance analysis," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, August 2007, number AIAA 2007-6595.

[5] Vijayan N. Nair, "Taguchi's parameter design: A panel discussion," *Technometrics*, vol. 34, no. 2, pp. 127–161, May 1992.

[6] Douglas C. Montgomery and George C. Runger, *Applied Statistics and Probability for Engineers*, John Wiley & Sons Inc., 3rd edition, 2002.

[7] Richard DeLoach, "Analysis of variance in the modern design of experiments," in *48th AIAA Aerospace Sciences Meeting*. AIAA, January 2010.

[8] Shinkyu Jeong, Kazuhisa Chiba, and Shigeru Obayashi, "Data mining for aerodynamic design space," *Journal of Aerospace Computing, Information, and Communication*, pp. 452–469, November 2005.

[9] Gianfranco Morani, Federico Corraro, and Antonio Vitale, "New algorithm for probabilistic robustness analysis in parameter space," *Journal of Aerospace Computing, Information, and Communication*, vol. 6, pp. 291–306, 2009.

[10] Toshikazu Motoda and Yoshikazu Miyazawa, "Identification of influential uncertainties in monte carlo analysis," *Journal of Spacecraft and Rockets*, vol. 39, no. 4, pp. 615–623, July-August 2002.

[11] Daniel P. Thunnissen, Siu Kui Au, and Edward R. Swenka, "Uncertainty quantification in conceptual design via an advanced monte carlo method," *Journal of Aerospace Computing, Information, and Communication*, vol. 4, July 2007.

[12] Serhat Hosder and Robert W. Walters, "Non-intrusive polynomial chaos methods for uncertainty quantification in fluid dynamics," in *48th AIAA Aerospace Sciences Meeting*, Orlando, Florida, January 2010, number 2010-129.

[13] Michael Balch, Serhat Hosder, and Robert W. Walters, "Modeling and propagation of physical parameter uncertainty in a Mars atmosphere model," in *46th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, January 2008, number AIAA 2008-450.

[14] Karen Gundy-Burlet, Johann Shumann, Tim Menzies, and Tony Barrett, "Parametric analysis of antares re-entry guidance algorithms using advanced test generation and data analysis," .

[15] Corina S. Pasareanu, Peter C. Mehlitz, David H. Bushnell, Karen Gundy-Burlet, Michael Lowry, Suzette Person, and Mark Pape, "Combining unit-level symbolic execution and system-level concrete execution for testing nasa software," in *Association for Computing Machinery ISSTA*, Seattle, Washington, July 2008.

[16] Peter C. Hughes, *Spacecraft Attitude Dynamics*, Wiley & Sons Inc., 1986.

[17] John. E. Hurtado, *A Kinematics and Kinetics Primer*, Lulu, August 2008.

[18] Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*, John Wiley & Sons Inc., 2nd edition, 2001.

[19] Christopher M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2007.