Visualization of Global Sensitivity Analysis Results Based on a Combination of Linearly Dependent and Independent Directions

Misty D. Davies* and Karen Gundy-Burlet†
NASA Ames Research Center, Moffett Field, CA, 94035

A useful technique for the validation and verification of complex flight systems is Monte Carlo Filtering—a global sensitivity analysis that tries to find the inputs and ranges that are most likely to lead to a subset of the outputs. A thorough exploration of the parameter space for complex integrated systems may require thousands of experiments and hundreds of controlled and measured variables. Tools for analyzing this space often have limitations caused by the numerical problems associated with high-dimensionality and caused by the assumption of independence of all of the dimensions. To combat both of these limitations, we propose a technique that uses a combination of the original variables with the derived variables obtained during a principal component analysis.

Nomenclature

α	=	The angle between the plotting plane and the planes of the projected hyperrectangle

I. Introduction

Finding bugs in modern flight software is an arduous process. Interdependencies between the systems and subsystems of the flight vehicle and the environment are usually explored in the design phase using high-fidelity simulations. At this level, the interplay between the interfaces can be fully explored. Advances in computing power make it possible to try many thousands of combinations of input variables to exercise the behavior of the system. However, a thorough exploration of the entire flight envelope results in gigabytes of data that requires expert domain knowledge to interpret; and because of the high dimensionality of the data, it is difficult to pick out patterns with the human eye.

The Robust Software Engineering (RSE) group within the Intelligent Systems Division at NASA Ames Research Center has developed a multi-step Monte Carlo Filtering technique[1,2] called the Margins Analysis[3-5] that automates much of the process necessary to identify unusual behaviors within the system and to isolate the critical ranges for suspicious inputs. We develop input test vectors using an *n*-factor combinatorial process[9]—the assumption is that most bugs are triggered by a maximum combination of two or three input variables; by guaranteeing that each combination of *n* input variables (where *n* is 2 or 3) appears in the test suite at least once we get a coverage guarantee while limiting the number of total test vectors within the suite. The output

* Research Computer Engineer, Intelligent Systems Division, Mail Stop 269-3, AIAA Member.
† Research Scientist, Intelligent Systems Division, Mail Stop 269-3, AIAA Associate Fellow.

behavior is analyzed for non-compliance with requirements and for unexpected structure using a combination of unsupervised[6] and supervised[7,8] machine learning techniques. Failures or unexpected results are collected and, when enough examples of a particular undesired behavior have been gathered, the data is automatically examined to determine which inputs (and ranges) demonstrate the most sensitivity with respect to the undesired behavior. This information is exploited in order to cause the undesired behavior to occur more often in future runs, and the aggregated data over all of the iterations can be used to identify bugs within flight software or within the simulation, or to determine the margins to failure as the systems pass from nominal to off-nominal behavior.

In current practice at NASA, the analysis can involve several hundred parameters over tens of thousands of trials. The unsupervised learning technique, a clustering technique that utilizes an expectation-maximization algorithm, has numerical difficulties at this level of dimensionality; we use a principal component analysis to reduce the dataset. The supervised learning technique, a treatment learning algorithm developed at West Virginia University, does not suffer from the same numerical difficulties as the clustering technique and can handle large numbers of parameters effortlessly. However, the treatment learning algorithm assumes that all of the parameters are independent; this assumption, when false, causes parameter ranges to be much larger than necessary and can cause the treatment learner to miss important sensitivities. A treatment learning analysis performed solely in the principle components space benefits from the fact that each transformed variable is linearly independent, but suffers from the difficulty of understanding the results in the transformed space. Each new direction in the principal component analysis is a linear combination of the original variables, so the answer that the treatment returns is a hyperrectangle in the space of the original variables.

We utilize the ability of the treatment learner to handle many dimensions, and ask it to perform an analysis over the data simultaneously in the original space and in the rotated space of the principle component analysis. As a post-processing step, we reduce the hyperrectangles in the principle component space to two- and three-dimensional projections in the original variable space. We then combine all of the information gained from the non-rotated and rotated spaces into two- and three-dimensional plots in the original variable space, so that they can be understood and interpreted by domain experts.

II. Methodology

We will discuss at length how we perform the principal component analysis, how we reduce dimensionality for the clustering algorithm, what data is given to the treatment learner and what data the treatment learner returns, how we choose the number of plots for each treatment, how we determine the dimensionality and axes for each plot, and how we plot the boundaries for the treatments.

III. Results

In standard practice, we ask the treatment learner to return a maximum of 10 treatments. Each treatment can consist of up to 4 variables. We place no restriction on whether these variables are in the original space or the principle component space. Ranges in the principle component space are first transformed into hyperrectangles in the original space, and then projected into two- or three-dimensional plots depending on the value of α, the angle between the primary two-dimensional plotting plane and the parallel planes of the projected hyperrectangle. The primary two-dimensional plotting plane is defined by the maximum components of the unit vector in the principle component direction when written as a linear combination of the variables in the original directions.

Figure 1 shows an example treatment for a dataset obtained during a bicycle ride. There were 10 variables for this dataset, and each of the 4435 time steps in the series was treated as a different experiment. The power calculation from the measured variables in the dataset was particularly noisy. Each data point in the set was rated according to the noise in the power calculation for some small time period surrounding the data point; the colormap goes from blue to red with the red points in the plots corresponding to the noisiest data points. The treatment returned for Figure 1 had 3 components (chosen by the algorithm from 20 components—10 components in the original variable space and the 10 rotated directions): cadence, wind speed, and a principle component direction in 10-dimensional space where the two largest components when written in the original space were hill slope and wind speed. The two variables in the original space are plotted together as a two-dimensional plot on the left, and the red box shows that the noisiest data occurs at the highest wind speeds and cadence. The angle α for the

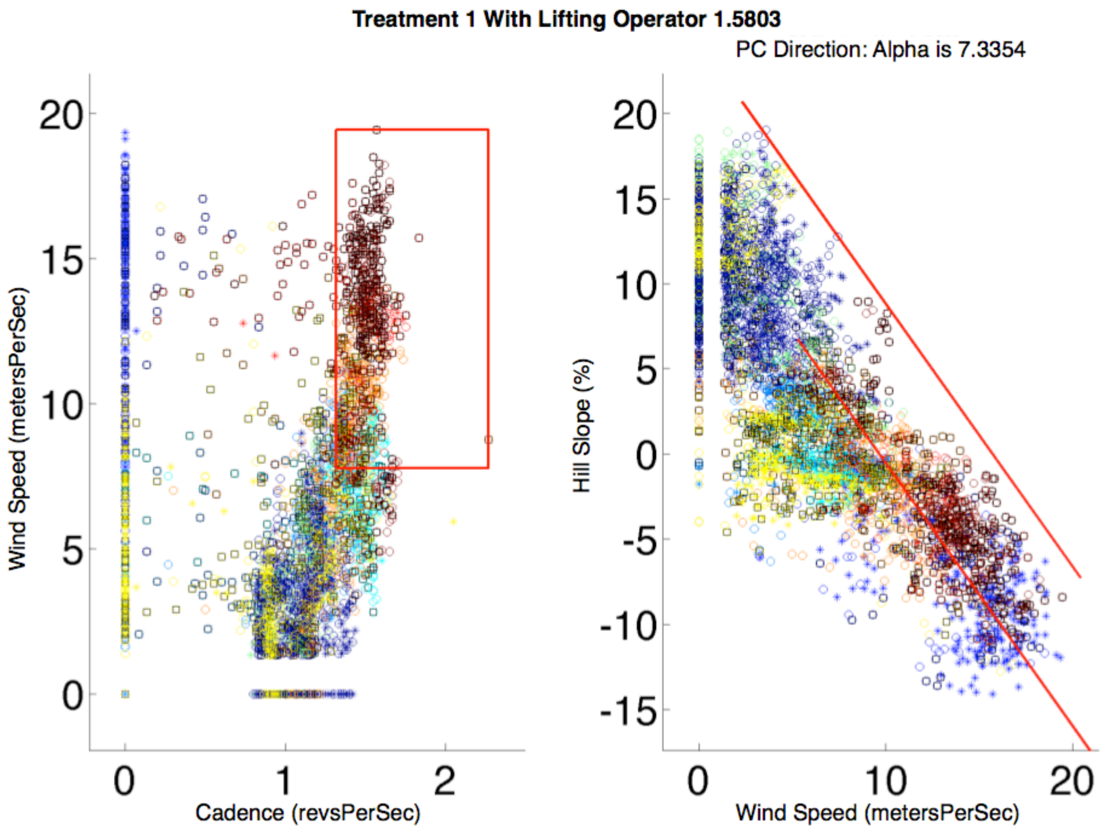

Figure 1. An example treatment for data obtained during the operation of a bicycle. *The data are color-coded from blue to red with red corresponding to the noisiest power data and blue corresponding to the least noisy. The red lines outline the regions identified by the treatment learner as the most likely to contain the undesirable noisy data. In this treatment, it is clear to see from the first plot that the undesired data (in this case, noisy power data, plotted in red and outlined with a black box) occurs most often at high wind speeds when the rider is continuing to pedal the bicycle. This second plot in the treatment is a projection of a principal component analysis direction result into the original space. This plot shows us that the hill slope and the wind speed have a strong linear correlation and that the least desirable data occurs for a linear combination of high wind speed and high hill slope. The two plots together constitute a visual representation of the highest scoring treatment.*

principal direction isolated in this treatment is less than 10 degrees, so the principal component information is plotted as a two-dimensional plot on the right. The red lines show the intersection of the almost vertical treatment hyperplanes with the plane defined by the hill slope and wind speed. What immediately becomes apparent from the plot on the right is that the hill slope and wind speed are negatively linearly correlated, so that the highest wind speeds occur at the steepest downhill slopes. Interpreting the two plots together leads to the conclusion that the calculated power measurement is most likely to be faulty when the cyclist is free pedaling on the downhill slopes.

IV. Conclusion

Acknowledgments

This research was conducted at NASA Ames Research Center. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government.

References

[1]Rose, K., Smith, E., Gardner, R., Brenkert, A. and Bartell, S. "Parameter Sensitivities, Monte Carlo Filtering, and Model Forecasting Under Uncertainty," *Journal of Forecasting*, Vol. 10, 1991, pp. 117-133.

[2]Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., and Tarantola, S., *Global Sensitivity Analysis: The Primer*, Wiley, Chichester, 2008, Chaps. 1, 5.

[3]Gundy-Burlet, K., Schumann, J., Barrett, T., and Menzies, T., "Parametric Analysis of ANTARES Re-entry Guidance Algorithms Using Advanced Test Generation and Data Analysis," *9th International Symposium on Artificial Intelligence, Robotics and Automation in Space*, 2007.

[4]Schumann, J., Gundy-Burlet, K., Pasareanu, C., Menzies, T., and Barrett, T. "Tool Support for Parametric Analysis of Large Software Systems", *Proceedings of Automated Software Engineering, 23rd IEEE/ACM International Conference,* 2008.

[5]Gundy-Burlet, K., Schumann, J., Barrett, T., and Menzies, T., "Parametric Analysis of a Hover Test Vehicle Using Advanced Test Generation and Data Analysis," *AIAA Aerospace*, 2009.

[6]Fischer, B., and Schumann, J. "Autobayes: A System for Generating Data Analysis Programs From Statistical Models," *Journal of Functional Programming*, Vol. 13, 2003, pp. 483-508.

[7]Hu, Y., "Treatment Learning: Implementation and Application," Masters Thesis, Department of Electrical Engineering, University of British Columbia, 2003.

[8]Hu, Y., and Menzies, T. "Data Mining for Very Busy People," *IEEE Computer*, Vol. 36, No. 11, 2003, pp. 22-29.

[9]Barrett, A., "A Combinatorial Test Suite Generator for Gray-Box Testing", *Third IEEE International Conference on Space Mission Challenges for Information Technology*, 2009.

Cite as:
Davies, M., and Gundy-Burlet, K., "Visualization of Global Sensitivity Analysis Results Based on a Combination of Linearly Dependent and Independent Directions." In *AIAA Infotech,* Atlanta, Ga., April 20-22, 2010.

Visualization of Global Sensitivity Analysis Results Based on a Combination of Linearly Dependent and Independent Directions

Misty D. Davies* and Karen Gundy-Burlet†
NASA Ames Research Center, Moffett Field, CA, 94035

A useful technique for the validation and verification of complex flight systems is Monte Carlo Filtering—a global sensitivity analysis that tries to find the inputs and ranges that are most likely to lead to a subset of the outputs. A thorough exploration of the parameter space for complex integrated systems may require thousands of experiments and hundreds of controlled and measured variables. Tools for analyzing this space often have limitations caused by the numerical problems associated with high-dimensionality and caused by the assumption of independence of all of the dimensions. To combat both of these limitations, we propose a technique that uses a combination of the original variables with the derived variables obtained during a principal component analysis.

I. Introduction

Finding bugs in modern flight software is an arduous process. Interdependencies between the systems and subsystems of the flight vehicle and the environment are usually explored in the design phase using high-fidelity simulations. At this level, the interplay between the interfaces can be fully explored. Advances in computing power make it possible to try many thousands of combinations of input variables to exercise the behavior of the system. However, a thorough exploration of the entire flight envelope results in gigabytes of data that requires expert domain knowledge to interpret; and because of the high dimensionality of the data, it is difficult to pick out patterns with the human eye.

The Robust Software Engineering (RSE) group within the Intelligent Systems Division at NASA Ames Research Center has developed a multi-step Monte Carlo Filtering technique[1,2] called the Margins Analysis[3-5] that automates much of the process necessary to identify unusual behaviors within the system and to isolate the critical ranges for suspicious inputs. We develop input test vectors using an *n*-factor combinatorial process[9]—the assumption is that most bugs are

* Research Computer Engineer, Intelligent Systems Division, Mail Stop 269-3, AIAA Member.
† Research Scientist, Intelligent Systems Division, Mail Stop 269-3, AIAA Associate Fellow.

triggered by a maximum combination of two or three input variables; by guaranteeing that each combination of n input variables (where n is 2 or 3) appears in the test suite at least once we get a coverage guarantee while limiting the number of total test vectors within the suite. The output behavior is analyzed for non-compliance with requirements and for unexpected structure using a combination of unsupervised[6] and supervised[7,8] machine learning techniques. Failures or unexpected results are collected and, when enough examples of a particular undesired behavior have been gathered, the data is automatically examined to determine which inputs (and ranges) demonstrate the most sensitivity with respect to the undesired behavior. This information is exploited in order to cause the undesired behavior to occur more often in future runs, and the aggregated data over all of the iterations can be used to identify bugs within flight software or within the simulation, or to determine the margins to failure as the systems pass from nominal to off-nominal behavior.

In current practice at NASA, the analysis can involve several hundred parameters over tens of thousands of trials. The unsupervised learning technique, a clustering technique that utilizes an expectation-maximization algorithm, has numerical difficulties at this level of dimensionality; we use a principal component analysis to reduce the dataset. The supervised learning technique, a treatment learning algorithm developed at West Virginia University, does not suffer from the same numerical difficulties as the clustering technique and can handle large numbers of parameters effortlessly. However, the treatment learning algorithm assumes that all of the parameters are independent; this assumption, when false, causes parameter ranges to be much larger than necessary and can cause the treatment learner to miss important sensitivities. A treatment learning analysis performed solely in the principle components space benefits from the fact that each transformed variable is linearly independent, but suffers from the difficulty of understanding the results in the transformed space. Each new direction in the principal component analysis is a linear combination of the original variables, so the answer that the treatment returns is a hyperrectangle in the space of the original variables.

We utilize the ability of the treatment learner to handle many dimensions, and ask it to perform an analysis over the data simultaneously in the original space and in the rotated space of the principle component analysis. As a post-processing step, we reduce the hyperrectangles in the principle component space to two- and three-dimensional projections in the original variable space. We then combine all of the information gained from the non-rotated and rotated spaces into two- and three-dimensional plots in the original variable space, so that they can be understood and interpreted by domain experts.

II. Methodology

A *Hamel basis*, or simply *basis*, is a set of vectors (usually orthonormal) that are used in linear combination to describe every other vector in a vector space. One familiar basis is the normal vectors along the x, y, and z axes of the Cartesian coordinate system. A principal components analysis (PCA) is a way of discovering the basis in which some data has the most meaning. Assume we have some two-dimensional data as in Fig. 1. This data was measured using the arbitrary x and y axes, but it is obvious by visual inspection that most of the data falls on a line. A PCA analysis will discover the orthogonal vectors ξ and η in which most of the variation is explained (all of the variation occurs in ξ and none in η). There are several techniques for performing a PCA analysis; we use the singular value decomposition (SVD)[10]. The SVD factors a data matrix $\mathbf{D}$ with m rows and n columns into three new matrices $\mathbf{U}$, $\mathbf{\Sigma}$, and $\mathbf{V}$

$$\mathbf{D} = \mathbf{U}\Sigma\mathbf{V}^{\mathbf{T}} \tag{1}$$

where **U** and **V** are orthogonal and **Σ** is diagonal. For the tests we perform, m is usually much larger than n. The matrix $\mathbf{U}^{\mathbf{T}}$ is the change of basis matrix from the basis k used to describe **D**,

(2)

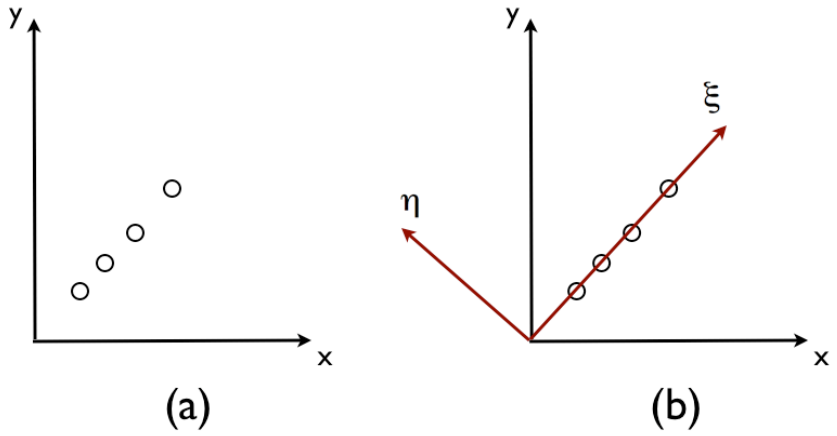

Figure 1. An illustration of basis transformation. *The circled data in plots (a) and (b) is highly linearly correlated. A PCA finds the new basis vectors, shown in red, that should be used in order to explain most of the variation in the data.*

the original data, to the basis κ used to describe **Δ**, the transformed data. The columns of **V** are the vectors of the new basis as described in the original basis. The transformed dataset **Δ** is found by

$$\Delta = \mathbf{D}\mathbf{V}^{\mathbf{T}}.$$

A PCA is usually done as a first step in reducing dimensionality for a dataset. The values on the diagonal of **Σ** are the eigenvalues σ of $\mathbf{D}^{\mathbf{T}}\mathbf{D}$, the covariance matrix of **D**. These eigenvalues are also known as singular values, and the *m* by *n* matrix **Σ** is ordered so that that largest singular value is at the upper left of the diagonal and the smallest is at the bottom right, or

$$\Sigma \equiv \begin{bmatrix} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_m \\ & 0 & \end{bmatrix} \tag{3}$$

where $\sigma_1 \geq \sigma_i \geq \sigma_m$. This means that the leftmost column of **V** is the basis vector that contains the most information about the data in **D** and the rightmost column of **V** is the basis vector that contains the least. A zero singular value signifies that there is no variation in the data when represented in the transformed basis κ.

The PCA is traditionally used to reduce dimensionality for a dataset. To decide how many eigenvectors in the new basis to keep we set a bottom threshold $0 \leq I_{\min} \leq 1$and define *I* as

$$I = \frac{\sum_{i=1}^{k} \sigma_i}{\sum_{j=1}^{m} \sigma_j}. \tag{4}$$

We keep the smallest number *k* columns of **V** such that *I* is larger than I_{min}. In practice for the large aerospace and space simulations run within the RSE group, choosing an I_{min} of 1 (essentially telling the algorithm to keep all principal directions that can explain any variation in the data), results in keeping about 50% of the principal directions. The reduced dataset $\mathbf{\Delta}_{\text{reduced}}$ is the first k columns of $\mathbf{\Delta}$. Most clustering algorithms, including the one used by the RSE group, are subject to numerical errors for data of high-dimensionality; giving the reduced dataset to the clustering algorithm increases performance and accuracy.

For our Monte Carlo Filtering analysis, we also perform supervised learning in the guise of *treatment learning*. Treatment learning is a fast sampling method that finds a minimum set of rules that predict a known outcome. In our case, we are usually looking for a set of inputs and ranges for a simulation that will cause some failure of a requirement. Treatment learning is not subject to the sorts of numerical errors that plague clustering algorithms, so there is no need to give the treatment learner a reduced dataset. However, treatment learning is subject to an error that many rule-learners make, and that is the assumption that all of the data parameters are independent. We use the PCA in a novel way to eliminate the independence assumption when data is highly linearly correlated. Figure 2 gives an illustrative hypothetical example. In Fig. 2, assuming the linear independence of the data forces the treatment learner to provide ranges much larger than necessary. If the ranges can be defined in the transformed basis, the graphical picture is easier to understand.

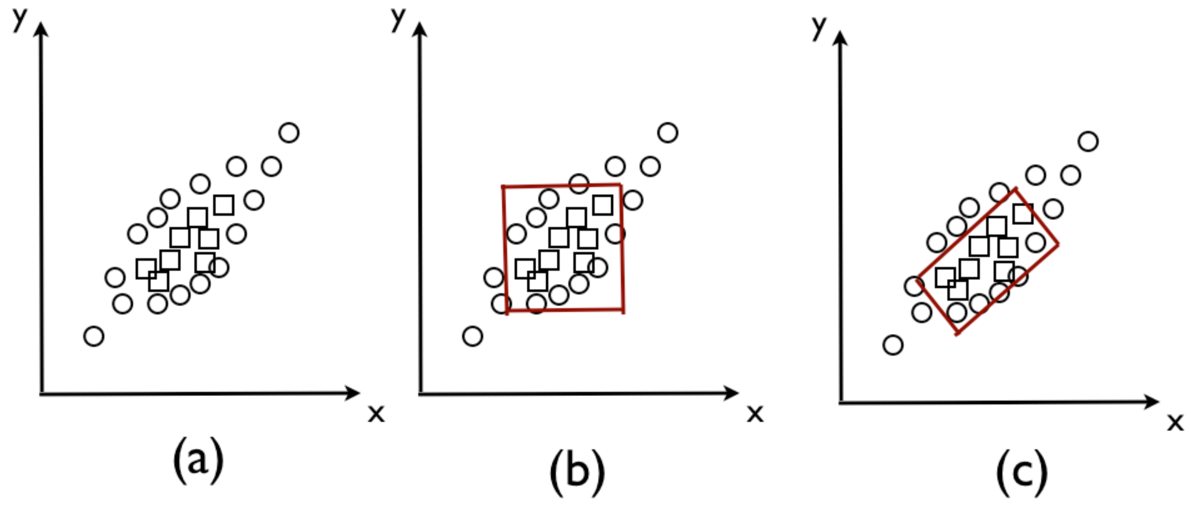

Figure 2. A simplified example of the difference between a treatment result in the independent and dependent directions. *The data in plots (a) through (c) is highly linearly correlated; the data marked by squares is the data of interest. The treatment learner is forced to find rules only on x and y for plot (b) because the data directions x and y were assumed to be independent. In plot (c), the treatment learner was allowed to use ranges in the transformed basis. This results in a treatment with a smaller area. If the data had been more highly linearly correlated, the volume of the box would have been even smaller.*

The treatment learner does not know which basis the data is defined in. As a result, the result from the treatment learner that the data of interest lies in the transformed basis direction between the values 0 and 1 may be difficult to interpret when the transformed basis direction is a linear combination of the original meaningful variables. In order to make sense of the treatment learner results when transformed data is used, we must transform the result back into the original basis and plot it graphically.

Assume that the treatment learner gives a result that transformed basis vector $\hat{t}_j$ should be restricted to values between *a* and *b*, and the original *n* variables form the basis $\hat{d}_i$ where *i* ranges between 1 and *n*. The first step is to use the j^{th} column of $\mathbf{V}$ to discover that $\hat{t}_j = c_1\hat{d}_1 + \cdots + c_n\hat{d}_n$.

In general, the human eye becomes confused when trying to interpret two-dimensional plots with more than two dimensions, so we will use the two original variables $\hat{d}_i$ with the largest values of c as the axes for the plot, c_{max} and c_{amax}. The range given by the treatment learner is two hyperplanes that intersect this two-dimensional plane in a line.

In order to interpret the plot, we also need to know how much information we've lost by projecting the data into two-dimensions. In order to do this, we project our hyperplane into a two-dimensional plane slicing through three-dimensional space. The three-dimensional space is itself a projection—two dimensions are formed by our two-dimensional plot and the third dimension contains all of the other original basis vectors. If our projected hyperplane is orthogonal to the two-dimensional plot, the graphical construction we've created is easily interpreted. Each transformed basis plot is labeled with the angle α, the angle between the orthonormal vector to our two-dimensional plot and the projected hyperplane. Figure 3 gives an example.

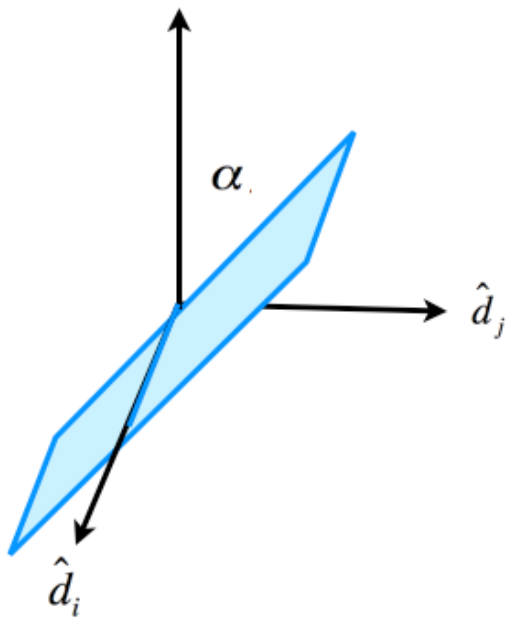

Figure 3. The projected hyperplane from the treatment results. *The basis vectors are variables from the original data—the variables labeled d are the original variables we are using for the two-dimensional plot. The vertical basis vector is the projected dimension of all of the other original variables. The treatment results form a hyperplane in the basis of the original data. This hyperplane is projected to a two-dimensional plane in the new three-dimensional space. The angle alpha is the angle between the vertical dimension and the projected hyperplane. If this angle is small, the two-dimension plot in d_i and d_j will be easy to interpret.*

III. Results

In standard practice, we ask the treatment learner to return a maximum of 10 treatments. Each treatment can consist of up to 4 variables along with the implicated ranges. We place no restriction on whether these variables are in the original space or the principle component space; both sets of data are handed simultaneously to the treatment learner. Ranges in the principle component space are first transformed into hyperrectangles in the original space, and then projected into two-dimensional plots. Each plot that uses the transformed basis reports the value of α, the angle between the normal vector to the primary two-dimensional plotting plane and the parallel planes of the projected hyperrectangle. The primary two-dimensional plotting plane is defined by the maximum components of the unit vector in the principle component direction when written as a linear combination of the variables in the original directions.

Figure 4 shows an example treatment for a dataset obtained during a bicycle ride. There were 10 variables for this dataset, and each of the 4435 time steps in the series was treated as a different experiment. The power calculation from the measured variables in the dataset was particularly noisy. Each data point in the set was rated according to the noise in the power calculation for some small time period surrounding the data point; the colormap goes from blue to red with the red points in the plots corresponding to the noisiest data points. The data for this figure contained information only in the primary basis.

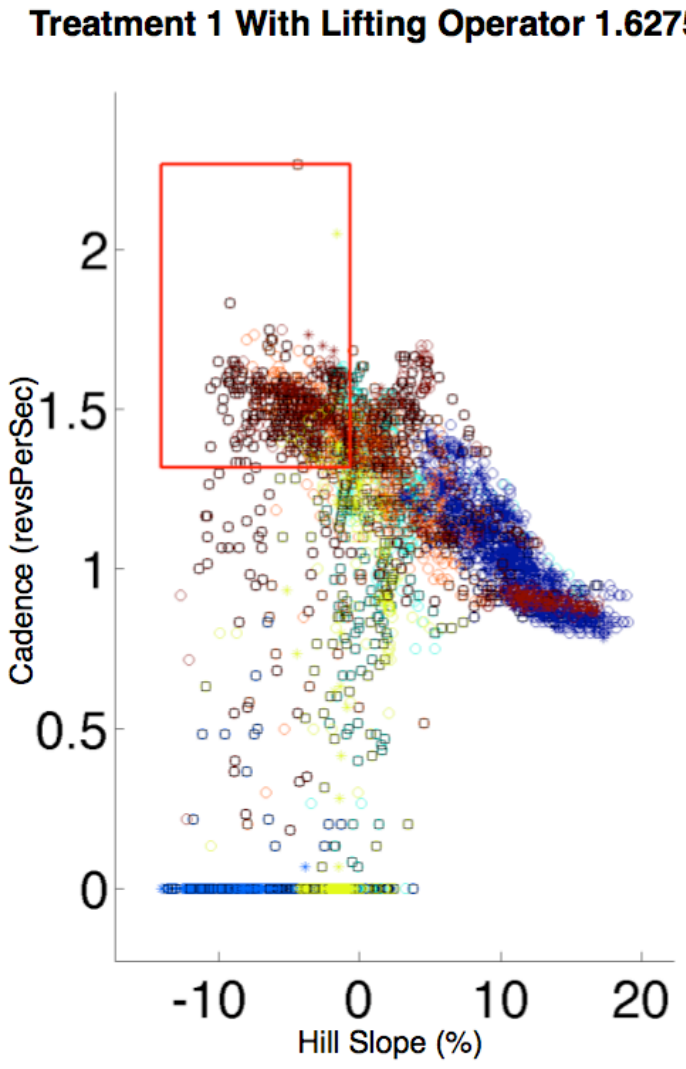

Figure 4. An example treatment using only the original variables. *The data are color-coded from blue to red with red corresponding to the noisiest power data and blue corresponding to the least noisy. The red lines outline the regions identified by the treatment learner as the most likely to contain the undesirable noisy data. In this treatment, it is clear to see that the undesired data (in this case, noisy power data, plotted in red) occurs most often at high wind speeds when the rider is continuing to pedal the bicycle.*

The treatment returned for Figure 5 had 3 components (chosen by the algorithm from 20 components—10 of these components were in the original variable space and the 10 were in the rotated directions). The treatment implicated cadence, wind speed, and a principle component direction in 10-dimensional space where the two largest components when written in the original space were hill slope and wind speed. The two variables in the original space are plotted together as a two-dimensional plot on the left, and the red box shows that the noisiest data occurs at the highest wind speeds and cadence. The angle α for the principal direction isolated in this treatment is less than 10 degrees, so the principal component information is plotted as a two-dimensional plot on the right and is easy to interpret. The red lines show the intersection of the almost vertical treatment hyperplanes with the plane defined by the hill slope and wind speed. What immediately becomes apparent from the plot on the right is that the hill slope and wind speed are negatively linearly correlated, so that the highest wind speeds occur at the steepest downhill slopes. Interpreting the two plots together leads to the conclusion that the calculated power measurement is most likely to be faulty when the cyclist is free pedaling on the downhill slopes.

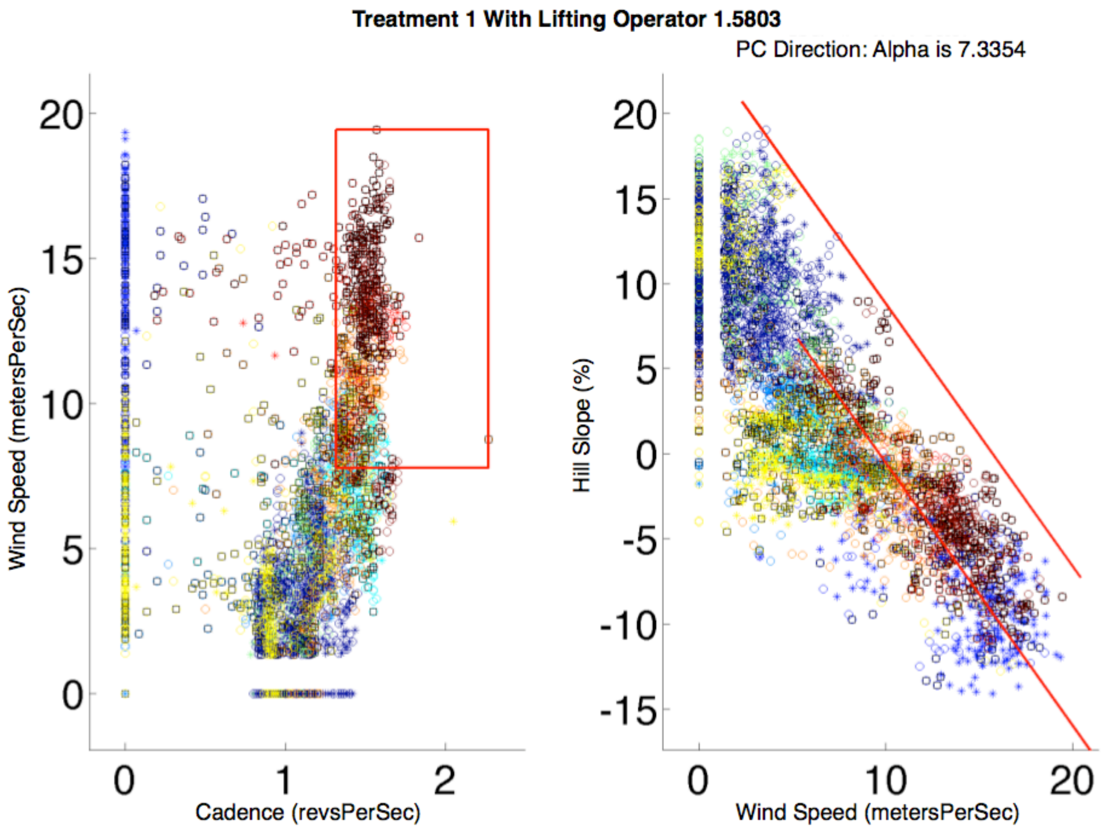

Figure 5. An example treatment for data obtained during the operation of a bicycle. *The data are color-coded from blue to red with red corresponding to the noisiest power data and blue corresponding to the least noisy. The red lines outline the regions identified by the treatment learner as the most likely to contain the undesirable noisy data. In this treatment, it is clear to see from the first plot that the undesired data (in this case, noisy power data, plotted in red) occurs most often at high wind speeds when the rider is continuing to pedal the bicycle. This second plot in the treatment is a projection of a principal component analysis direction result into the original space. This plot shows us that the hill slope and the wind speed have a strong linear correlation and that the least desirable data occurs for a linear combination of high wind speed and high hill slope. The two plots together constitute a visual representation of the highest scoring treatment.*

IV. Conclusion

These results demonstrate how the PCA can be used to overcome the limitations given by machine learning algorithms that assume the linear independence of data. In order to interpret the results, the transformed basis dimensions must be translated back to the original basis. Graphical representations of the rules given by these learners are then easy to interpret, and correlations between the original variables are easy to understand.

Acknowledgments

This research was conducted at NASA Ames Research Center. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government.

References

[1]Rose, K., Smith, E., Gardner, R., Brenkert, A. and Bartell, S. "Parameter Sensitivities, Monte Carlo Filtering, and Model Forecasting Under Uncertainty," *Journal of Forecasting*, Vol. 10, 1991, pp. 117-133.

[2]Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., and Tarantola, S., *Global Sensitivity Analysis: The Primer*, Wiley, Chichester, 2008, Chaps. 1, 5.

[3]Gundy-Burlet, K., Schumann, J., Barrett, T., and Menzies, T., "Parametric Analysis of ANTARES Re-entry Guidance Algorithms Using Advanced Test Generation and Data Analysis," *9th International Symposium on Artificial Intelligence, Robotics and Automation in Space*, 2007.

[4]Schumann, J., Gundy-Burlet, K., Pasareanu, C., Menzies, T., and Barrett, T. "Tool Support for Parametric Analysis of Large Software Systems", *Proceedings of Automated Software Engineering, 23rd IEEE/ACM International Conference,* 2008.

[5]Gundy-Burlet, K., Schumann, J., Barrett, T., and Menzies, T., "Parametric Analysis of a Hover Test Vehicle Using Advanced Test Generation and Data Analysis," *AIAA Aerospace*, 2009.

[6]Fischer, B., and Schumann, J. "Autobayes: A System for Generating Data Analysis Programs From Statistical Models," *Journal of Functional Programming*, Vol. 13, 2003, pp. 483-508.

[7]Hu, Y., "Treatment Learning: Implementation and Application," Masters Thesis, Department of Electrical Engineering, University of British Columbia, 2003.

[8]Hu, Y., and Menzies, T. "Data Mining for Very Busy People," *IEEE Computer*, Vol. 36, No. 11, 2003, pp. 22-29.

[9]Barrett, A., "A Combinatorial Test Suite Generator for Gray-Box Testing", *Third IEEE International Conference on Space Mission Challenges for Information Technology*, 2009.

[10]Strang, G., *Linear Algebra and Its Applications, 3rd Edition.* Harcourt Brace Jovanovich, San Diego, 1988. pp. 442-452.