SpaceOps Conferences
5-9 May 2014, Pasadena, CA
SpaceOps 2014 Conference

Spacecraft State-of-health (SOH) Analysis via Data Mining

Steve Lindsay[1] and Diane Myung-kyung Woodbridge[2]
Sandia National Laboratories, Albuquerque, NM, 87185

Spacecraft state-of-health (SOH) analysis typically consists of limit-checking to compare incoming measurand values against their predetermined limits. While useful, this approach requires significant engineering insight along with the ability to evolve limit values over time as components degrade and their operating environment changes. In addition, it fails to take into account the effects of measurand combinations, as multiple values together could signify an imminent problem. A more powerful approach is to apply data mining techniques to uncover hidden trends and patterns as well as interactions among groups of measurands. In an internal research and development effort, software engineers at Sandia National Laboratories explored ways to mine SOH data from a remote sensing spacecraft. Because our spacecraft uses variable sample rates and packetized telemetry to transmit values for 30,000 measurands across 700 unique packet IDs, our data is characterized by a wide disparity of time and value pairs. We discuss how we summarized and aligned this data to be efficiently applied to data mining algorithms. We apply supervised learning including decision tree and principal component analysis and unsupervised learning including k-means and orthogonal partitioning clustering and one-class support vector machine to four different spacecraft SOH scenarios after the data preprocessing step. Our experiment results show that data mining is a very good low-cost and high-payoff approach to SOH analysis and provides an excellent way to exploit vast quantities of time-series data among groups of measurands in different scenarios. Our scenarios show that the supervised cases were particularly useful in identifying key contributors to anomalous events, and the unsupervised cases were well-suited for automated analysis of the system as a whole. The developed underlying models can be updated over time to accurately represent a changing operating environment and ultimately to extend the mission lifetime of our valuable space assets.

I. Introduction

TYPICAL spacecraft state-of-health (SOH) analysis requires experts to check incoming measurand values against their predetermined thresholds. The threshold limits are determined by expert knowledge on a component itself and relationships with other components and outside influences including operating environment changes. All of these relationships evolve over time. However, as the aforementioned approach is not data-driven, it is possible to fail on managing and evolving threshold limits properly, resulting in undesirable biased outcomes. For instance, a spacecraft with four batteries could have two that operate well within their normal ranges, while the remaining two approach their nominal limits. While these occurrences may not be flagged during an automated limit check, it could be indicative of a malfunctioning controller that may soon lead to an overcharge condition and ultimately to battery failure and reduced time on orbit.

A more powerful solution is applying data mining techniques to SOH data for finding underlying correlations, trends and patterns among groups of measurands. Since a high volume of SOH data overwhelms the mining process, data preprocessing is a required first step. Additionally, as components can have different sensing frequencies, sensing time, and units of measures, data transformation is needed to change the data into a format which is easily fed to the data mining procedure. For example, the spacecraft data used in this study has 30,000 measurands across 700 different packet IDs with a wide disparity of time and value pairs and requires data

[1] Data Management Capability Lead, Decision Support Systems, MS 1202
[2] Software Engineer, Data Analysis and Exploitation, MS 1326

processing including data integration, discretization, summarization, and transformation. In this paper, we discuss how we preprocessed this data to allow it to be fully leveraged by the data mining step.

In data mining, depending on whether the class label of training data is provided, learning can be classified as either supervised or unsupervised. Supervised learning has labeled training data, while unsupervised learning does not have class labels on training data and sometimes does not include the number of classes. In this study, we cover four different scenarios we explored for finding data correlations and trends using supervised and unsupervised learning. We applied supervised learning to two scenarios and unsupervised learning to the other two scenarios. For each scenario, we define problems and objectives and then discuss the applied data mining techniques and their results.

The first scenario is the fault diagnosis and prognosis of spacecraft subsystems, which detects contributors and precursors of a system failure and predicts the amount of time left before failure. Similar to other studies which monitor vibration of the system for condition monitoring and damage detection[1], this scenario focused on features relevant to vibration. A vibration-sensitive hardware component in the scenario was experiencing a failure event at intermittent and unpredictable intervals, and we were interested in algorithms for predicting a transition from a nominal status to a status which is an hour before failure and ultimately identifying the contributing measurands. We partitioned the data into four categories – nominal, pre-event, in-event, and post-event – and used decision tree and attribute importance algorithms in an attempt to characterize SOH values that can predict a transition from nominal to pre-event.

Our second scenario was a supervised analysis of our ground-based data processing pipeline, which occasionally experiences unusually long delays in creating our downstream data products. We assume that there are hardware components that cause failures and try to diagnose the problem with data-driven methods. We discretized the execution time into five categories and used the decision tree algorithm to pinpoint the hardware contributing most to the slowdowns.

The third scenario involved an unsupervised analysis of the thermal subsystem. Among the 54 measurands monitored in the thermal subsystem, we were most interested in grouping measurands with common features and reducing cardinalities. We used two separate algorithms, K-means clustering[2] and orthogonal partitioning clustering[3] to create unique data clusters and compared the characteristics of each one. We were then able to correlate the clusters with low cardinalities to their corresponding trigger events.

In the fourth scenario, we used an unsupervised one-class support vector machine (OC-SVM) with a Gaussian kernel to perform anomaly identification across a set of measurands deemed most critical in determining overall health and status. OC-SVM builds an initial model representing nominal operations and compares new data against the model, assigning a probability of inclusion (nominal) or exclusion (anomalous) for each time slice.[4] We assumed data that does not correspond with the model built from training data is anomalous. This method is particularly well-suited to spacecraft SOH analysis, as one can periodically regenerate the model to account for component degradation as well as the changing solar seasons. The method is completely data driven as opposed to other studies which require prior knowledge.[5, 6]

The remainder of this paper is organized as follows. In Section II, we describe the spacecraft SOH data repository. Section III explains the data preprocessing step for our spacecraft SOH data including data discretization, summarization, and transformation. Sections IV and V present experimental results for the four different scenarios described above.

II. Data Repository

Our data mining endeavor builds upon a previous effort whose primary goal was to transition SOH data from flat files into a relational database.[7] We implemented a commercial off-the-shelf (COTS)-based solution using an Oracle® 11g relational database management system (RDBMS)[9] to store data in time-based partitions for rapid query response and to facilitate the archival and removal of older data. We also incorporated compression to minimize the amount of disk space required for storage on the file system.

Our compression scheme uses a change-based approach to avoid storing every single sample downlinked from the spacecraft. Our implementation was for a Department of Energy (DOE) remote sensing payload that contains over 30,000 measurands and produces 150M values daily. Through compression, however, we store only 30M rows daily while also providing a way to reconstruct the data as sampled on the spacecraft – all within a reasonable response time.

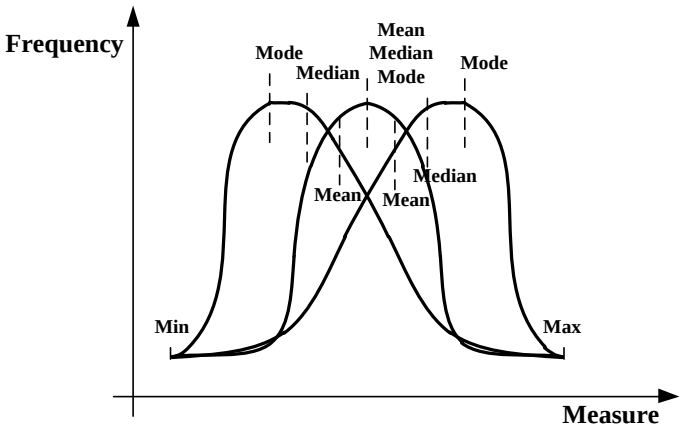


Figure 1. Central tendency measures used in the study.

The abilities to reconstruct the data and present it quickly to the client were key to moving forward with the predictive analytics discussed in this paper. Without an accurate reconstruction, we would have been unable to produce precise statistics across various time intervals. In addition, without the benefit of responsive queries, our data presentation layer would have required too much execution time. Fortunately we achieved both of these goals in the previous study, resulting in a solid foundation for our journey into data mining.

Finally, we were fortunate in that our database also contains a limited number of measurands that represent performance of the ground system hardware, including values for central processing unit (CPU) and memory usage as well as system and network response times gathered via ping. We will demonstrate later in this paper how this feature created unique opportunities to fuse ground and flight data prior to mining for hidden trends and patterns.

The reference[7] provides more details regarding our SOH database implementation. However, for the purposes of this paper the reader can assume our starting point was a vast repository of time-series data sampled at various rates and stored in both raw and engineering unit (EU) converted form, along with structured query language (SQL) constructs for quickly retrieving that data in minimal time.

III. Data Preparation

In this study, SOH subject matter experts (SMEs) define a group of relevant measurands for each scenario, as described in Sections IV and V in detail. However, as measurands in a domain have different sampling frequencies and different monitoring times, it is difficult to align measurements and find data correlations and trends. Therefore, a data preprocessing step is required for mining the SOH data.

In the data preprocessing step, it is necessary to discretize and summarize data with different sampling frequencies into descriptive data having the same sampling frequencies while preserving the original data as much as possible.[8] Descriptive data summarization techniques are used for identifying data properties and characteristics per time unit (u) within given time interval, $[\,t_{start}, t_{end}\,]$. Continuous time-series data are transformed into discretized and descriptive format for each time interval. In this study, mean, median and mode are used for measuring central tendencies of the summarized data. Mean is a numerical central value of a set of data. Median is the middle value of the ordered measures x_i and mode is the value occurring most in a set. With aforementioned central tendency measures, it is possible to find whether the data distribution is skewed or symmetrical. For example, if mean, median and mode values are the same for a curve with one mode value, the measures are symmetrically distributed. Otherwise, the data distribution is skewed and its skewedness can be estimated, using the mean, median and mode values (Figure 1).

In addition, the standard deviation, minimum and maximum operators are measures of data dispersion. The minimum and maximum are the smallest and largest values within a set, and the standard deviation (σ) shows how much dispersion from the mean (μ) exists in a set.

Data summarization results in $\left\lceil \frac{t_{end}-t_{start}+1}{u} \right\rceil$ rows per measurand. However, as each measurand's data are not aligned with other measurands in the same scenario with the same timestamp, it is difficult to calculate correlations between different measurands. In order to integrate measurands' data with the same timestamp, we used the pivot

function in Oracle®. Pivot provides summarization functionality by transposing data columns into the header row and aggregating data. In our study, the pivot function rearranges a table with descriptive values per measurand into a table with the entire measurands' descriptive values sharing the same Unix timestamp (UTM) (Figure 2).

The developed data preprocessing procedure provides options to choose a scenario, time ranges of interest (t_{start}, t_{end}), time intervals (u) for data summarization, and types of descriptive data. In order to keep concept hierarchy and monitor repeated patterns, the timestamp is also transformed to time of day and month of year. Transforming timestamps into different granularity helps monitor cyclic changes in data such as environment changes due to Earth's rotation. This data preparation step reduces the data complexities by reducing volume and aligning timestamps, and thus enhances the quality of data mining.

IV. Supervised Scenarios

For all four scenarios, we used Oracle® Data Miner (ODM) as our data mining tool and leveraged its graphical user interface (GUI) embedded in Oracle® SQL Developer.[9, 10, 11]

A. Vibration-sensitive Hardware Component

Our first supervised data mining scenario involved a vibration-sensitive payload hardware component. As with any supervised activity we were trying to predict a particular value. In this case, the hardware component in question was taking itself offline infrequently but at seemingly random times and with no apparent cause – an occurrence affectionately known as a hardware trip. After consulting with the SME, we felt it was worth exploring whether data mining could help identify the cause.

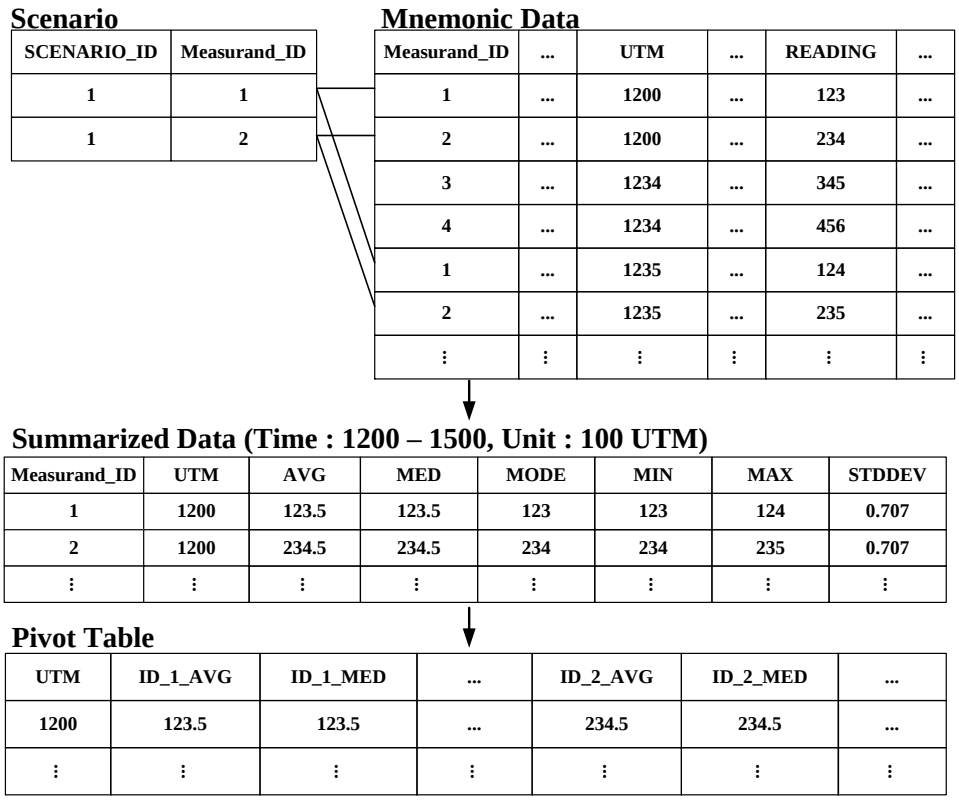


Figure 2. Example of data preprocessing.

Table 1. Distribution of *eventState* Values for Hardware Trips.

Value of *eventState*	Number of rows	Percent of all rows
Normal operations	86653	99.9%
Pre-event (60 minutes before In-event)	18	0.02%
In-event	5	0.01%
Post-event	13	0.02%

1. Approach

Our approach consisted of first identifying the relevant measurands, which the SME helped us pinpoint. He chose 54 values including voltages, temperatures, temperature rates, and measurements of specific vibration harmonics within the hardware assembly itself. The SME recommended not mining on all six aggregations, but rather a subset: minimum, maximum, mean, and standard deviation. In his opinion, mode and median would provide no added value and the mean would be enough for representing the central tendency.

After selecting the candidate measurands and aggregations, we decided upon the definition of our target variable and named it *eventState*. Our plan was to give it a range of four possible values: normal operations, pre-event, in-event, and post-event settling. For the pre-event duration, our SME recommended a constant value of one hour prior to a given event. The in-event and post-event durations were event-specific and ranged from ten to thirty minutes.

2. Algorithm and Challenges

When characterizing the pre-event state, we faced three significant challenges. The first was that the SME's chosen measurands were sampled on the spacecraft only at one-minute intervals. Consequently, if the underlying cause presented itself in mere seconds before a given event, it would likely be undetectable by the data mining software.

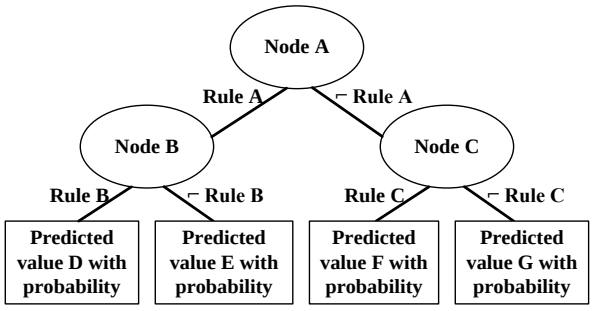
The second challenge was that very few hardware failures had occurred up to this point, causing a highly skewed distribution of *eventState* values as shown in Table 1. Consequently, we knew we would have to account for this skewed distribution by adjusting the weights in the implementation algorithm.

The final challenge was the most daunting of all yet common to any supervised data mining scenario: the root cause might not manifest itself at all in the underlying data. The SME was inclined to believe this might be the case; however, it was worth pursuing to determine whether data mining could provide any additional insight into the intermittent hardware failures.

Due to the demanding nature of this problem, we chose to corroborate the results of two separate algorithms: decision tree and attribute importance (Figure 3). The decision tree algorithm determines each measurand's threshold values that define a given state, listed in decreasing order of significance.[12] The attribute importance algorithm also produces a decreasing order of significance but it omits threshold value determination. This algorithm is based on principle components analysis (PCA).[13]

3. Execution and Results

We ran the data preparation step over ten-minute intervals and produced 86,689 rows of mineable data. We were



a. Decision Tree Output

Input

Predictor A	Predictor B	Predictor C	Target
A	B	C	T
A'	B'	C'	T'
A''	B''	C''	T''

Output

Variable Name	Importance
Predictor A	Importance A
Predictor B	Importance B
Predictor C	Importance C

b. Attribute Importance Algorithm Input, Output

Figure 3. Example Inputs and Outputs used in Supervised Scenarios.

surprised to see that this step took more than six days to complete, but because it was our first execution attempt we knew we would have additional opportunities for optimization (discussed in Section B.3.). We then used standard SQL to populate the *eventState* values.

Unfortunately, the resulted state definition from the decision tree for pre-event encompassed more than the single hour prior to each in-event. In other words, ODM was unable to exclusively characterize pre-event to the extent that all other states were omitted. In fact, the pre-event state definition from the applied supervised learning was valid for 694 rows of mineable data: the 18 occurrences of actual pre-event in Table 1 plus 676 occurrences of normal operations. Restated, this means that over 97% of the time, the pre-event characterization represented a time interval that did not correspond to an actual pre-event situation. However, the applied method still shows the high true positive rate (t_p) and the low false negative rate (f_n) for pre-event, which yields high recall.

$$Recall = \frac{t_p}{t_p + f_n} \tag{1}$$

Data mining results illuminated the key contributors for defining the pre-event state with both algorithms. We rank-ordered the corresponding measurands and presented the two lists to the SME. He quickly noticed that each list showed the lower-magnitude vibration harmonics contributed more to the state transition than did the higher-magnitude vibration harmonics. As a result, he was able to conclude that the failure cause was most likely internal to the component rather than originating from an external force. While the definitive cause remains unknown, this piece of information has helped him narrow the search space for his analysis.

B. Data Product Save Times

Our next supervised scenario was unique in that it dealt almost exclusively with SOH data from our satellite ground system. After the ground system creates a data product from payload data, it saves the corresponding metadata in the database. The delay from the last frame of data collected on the payload to successful database insertion on the ground is typically less than five seconds. However, our operators noticed the ground system occasionally requires much longer to save a data product – sometimes on the order of tens of minutes. Our goal in this scenario was to determine the cause of these periodic delay spikes.

1. Approach

As mentioned in Section. II, we were fortunate to have a limited number of measurands that represented point-in-time performance of the ground system hardware sampled on regular intervals. We met with the SMEs to determine the relevant hardware components, which in this case consisted of nine servers and workstations comprising a significant portion of the data processing pipeline. For each machine, we decided to mine CPU usage, random access memory (RAM) utilization, and ping delays across the four aggregations of minimum, maximum, mean, and standard deviation. We chose to ignore other ground system metrics that we felt were irrelevant, including swap utilization and disk space usage for both the temporary and permanent file systems.

Because the data product save times were stored in a separate product database, we first needed to capture and fuse this data with the SOH database to present a unified view to the data mining tool. We chose to calculate data product counts and maximum save times (in seconds) over the aggregation window. For the target variable we leveraged the same *eventState* value but we defined it in a new manner. We discretized the maximum save times into 5 linear bins with integer numbers between -1 and +3. A value of -1 represented a non-applicable or non-

Table 2. Distribution of *eventState* values for product save times.

Value of *eventState*	Number of rows	Percent of all rows
-1 = N/A	7289	10%
0 = best	51865	73%
1 = fair	6393	9%
2 = poor	1809	3%
3 = worst	3684	5%

```
Populate lookahead hash with time = -1 for all measurands
Foreach measurand loop
       If (lookahead time) > (current time) + (time interval) then
              Use current value
       Else
              Query database for (value, time) of next change
              Update lookahead time with queried time
       End if
End loop
```

Figure 4. Populate lookahead hash with time = -1 for all measurands.

computable value, while 0 through 3 represented increasing maximum times: 0 being the best and 3 being the worst. Table 2 shows the distribution of these values, which fortunately was much more favorable than the distribution of the hardware trips discussed in Section A.

2. Algorithm and Challenges

We chose to again leverage the decision tree algorithm (Figure 3). This algorithm gave us good results in the prior supervised scenario and we felt it was also a good candidate for this one.

Our first challenge was that the ground system measurands were sampled relatively infrequently: on ten-minute intervals. While providing us less data than we would have wished, this did have the benefit of driving our aggregation sampling window which we set at ten minutes for consistency. In addition, the software collecting the ground system measurands was less stable than other ground system components – likely due to its relatively low level of mission criticality – resulting in frequent data dropouts. Such dropouts and overall lacks of "clean data" are typical of data mining in general however, so we embraced these as natural and expected challenges that would provide good tests of the data mining tools.

3. Execution and Results

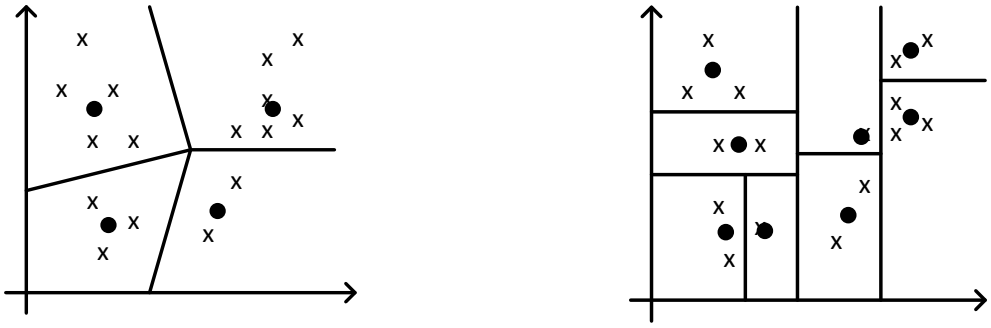
We used this scenario as our first opportunity to improve the data preparation performance, which we did by implementing a lookahead function. This function improves performance by querying the database only when needed rather than for every time step. The corresponding algorithm is summarized in Figure 4. Upon startup and whenever we generated 5% of the overall scenario data, we re-initialized the entire lookahead table and tracked the number of times each measurand successfully leveraged its lookahead function for a duration of 100 operations. If that number was less than a threshold of 20%, we assumed the overhead was too expensive and avoided further lookaheads for the given measurand until the next evaluation period. In this manner, we were able to guard against periods of high activity where a value was rapidly changing, while still realizing a performance boost when that same measurand was relatively stable. Using a lookahead function, the data preparation step for this scenario produced 71,040 rows of mineable data and took only three days to complete. We were pleased this resulted in a 50% performance boost.

ODM was quick to determine that the server CPU on which the database resides was the overwhelming contributor to the high product save time delays. We expected this result, as that same server also hosts the application software that drives the database insertions and has a highly taxed CPU.

We then removed all of the metrics – CPU, ping, and memory – for the database server in an attempt to pinpoint the primary external factor related to the high save times. Interestingly, ODM determined that memory on one particular metadata server was the primary contributor. Upon closer inspection of the native SOH data we did indeed notice this server's memory was highly utilized at key intervals. We forwarded this information to our system administration team so they could schedule a memory upgrade at our next available maintenance opportunity, along with a CPU upgrade of the database server.

V. Unsupervised Scenarios

Our next objective was to investigate the merits of unsupervised SOH data mining using ODM. Rather than predictive in nature, unsupervised techniques attempt to identify unique ways to describe the underlying data.



a. K-Mean (K = 4) and observation values and mean (black dot) of each cluster.

b. O-Cluster and observation values and centroids (black dot) of each cluster.

Figure 5. Example Outputs used in Unsupervised Clustering Scenarios.

A. Thermal Analysis

Our first unsupervised scenario delved into the thermal subsystem. While this subsystem was relatively stable, the SME was constrained to the typical process of strip-chart limit checking and trending common to temperature monitoring. Our goal in this scenario was to determine if he could benefit from viewing his data from an alternative perspective.

1. Approach

Our SME chose 54 measurands that primarily represented various temperature measurements on the payload. Because of their inherent stability, we chose to incorporate all six aggregations into our data preparation step, knowing that we could later filter out some of these values (such as mode and median) if we decided they were unimportant.

2. Algorithm and Challenges

We chose to use clustering as a means of dividing the thermal data into separate groups with common features. The two algorithms available to ODM are K-Means and Orthogonal Partitioning Clustering (O-Cluster) (Figure 5).

K-Means is a distance-based clustering algorithm that partitions the data into a specified number of clusters.[2] The underlying distance function drives the determination on whether cases are similar and should therefore belong to the same cluster.

O-Cluster is an Oracle-proprietary clustering algorithm that creates a hierarchical grid-based clustering model.[3] Operating recursively, the algorithm produces a hierarchical structure that represents clusters of dense areas in the attribute space.

Our primary challenge in this scenario was dealing with nebulous payoff. While the supervised scenarios attempted to identify an underlying cause for a given effect, the thermal clustering scenario simply dealt with data groupings. As a result, we found it difficult to determine when our work within this particular scenario was complete. In addition, the clustering algorithms themselves had numerous modifiable parameters, any of which could be changed and re-executed to produce a new set of results for analysis. Because the SME had limited availability, we felt we could modify only a small subset of these parameters rather than step through multiple iterations.

3. Execution and Results

We configured both algorithms to generate 12 clusters. For O-Cluster, the cardinality of the resulting clusters ranged from 600 to 16,000. The K-Means algorithm, however, produced five clusters of approximately 14,000 members and seven clusters with only one member. These seven singletons mapped to five unique time intervals of between ten and twenty minutes in duration.

We presented both sets of results to the SME. He found the O-Cluster results to be of limited utility and was unable to derive any useful information from their individual characteristics. He found the K-Means results much more interesting and useful. For each of the five unique time intervals, he successfully identified the cause

regarding why these periods were cluster outliers: three of the cases corresponded to flight software uploads, and the remaining two corresponded to hardware resets. These results represented high potential utility in identifying future anomalous operations.

B. Anomaly Identification

Our final unsupervised scenario was our most ambitious and comprehensive to this point. Our objective was to determine if we could proactively warn of current or pending payload anomalies for the payload as a whole.

1. Approach

Our SME helped us determine which measurands would be the most useful in making this overall assessment. Fortunately, our SOH data already contained a group of measurands flagged as critical to payload operation. This group consisted of 53 measurands that represented key temperatures, voltages, currents, and status flags. Like the thermal analysis scenarios, we chose to incorporate all six aggregations to present the most complete picture possible with this reduced and critical subset of measurands. We settled on a six-minute aggregation time interval.

2. Algorithm and Challenges

We chose the one class support vector machine (OC-SVM) algorithm to support our objective. As mentioned earlier (Section I), this algorithm builds a long-term model of nominal operations against which recent data can be judged to match or mismatch with a given probability, resulting in an assessment of the recent data being nominal or anomalous respectively (Figure 6).[4]

The primary challenge was finding a handful of measurands that truly captures the overall state of a complex payload with over 30,000 measurands. Of course, toward this end we trusted that our SME had given careful consideration to the measurands he selected for us to model. In addition, certain time intervals may exist which should be omitted from the nominal model build due to their off-nominal or even anomalous characteristics. These too should be carefully considered and the corresponding analysis may take a large amount of time – especially if the determinations are made on a per-measurand basis, which is a certain prerequisite for producing the most accurate results.

An additional challenge exists with respect to verifying the detection of an anomaly: if no such phenomenon exists, simulation is the only alternative. This opens up further questions regarding the duration, intensity, and complexity of the simulated anomaly. At one extreme, an analyst can inject a data dropout for a period of several minutes. At the other more subtle extreme, the analyst can cause multiple measurands to approach (but not exceed) their nominal or critical limits. Many possibilities exist and the corresponding testing can represent a significant time commitment for both the analyst and the SME.

3. Execution and Results

We built a model of our 53 critical measurands for the 24-month period ending at midnight on 31 July 2013. For simplicity, we chose to consider all data nominal rather than filter out specific time periods where data could have been considered off-nominal.

We then treated the following week (ending 7 August 2013) as our new data. We chose a time period of one week since that seemed to be the most appropriate level of granularity for operational reporting. That is, in a production environment we envisioned the software generating the prior week of data in the early hours each Monday, applying it to the previously built model, and emailing the results to the relevant SMEs for evaluation and further investigation where required.

Next we chose a single measurand – a system controller in this case – and experimented with data dropouts

DTM	PREDICTION	PROBABILITY
28-OCT-2010 19:06:31.000000000	0	.82451853983496037
28-OCT-2010 19:06:31.000000000	1	.17548146016503965
29-OCT-2010 19:06:31.000000000	0	.71361628409556332
29-OCT-2010 19:06:31.000000000	1	.28638371590443673

Figure 6. Anomaly Identification Example Output.
(A prediction value of zero is considered anomalous. A prediction value of one is considered normal.)

varying from between four and six minutes. Our goal in this endeavor was to determine a reasonable probability threshold that would accurately report an anomalous situation. We ran multiple scenarios and determined that for probability values between 0.639 and 0.675, the software successfully detected the dropouts. Restated, this means the OC-SVM would accurately report an anomaly for the system controller measurand if the likelihood is 0.639 or greater that the new data mismatched the previously built model.

This iterative approach to pinpointing a reasonable probability value is a necessary step in determining the starting point for our deployed system. The ultimate objective is to find a number that detects anomalies when present while simultaneously raising no false alarms. A thorough analysis requires cycling through multiple types of measurands and several kinds of anomalies in addition to simple data dropouts. Once the analyst determines this initial value, the next step is to deploy and monitor the results and adjust it accordingly. Toward this end, a conservative initial value that perhaps allows a small number of false alarms is better than the opposite extreme, which could result in undetected anomalies. Unfortunately time and resource constraints prevented us from deploying our software to the live system. However, we remain poised and ready to do so should those constraints become lessened at some point in the future.

VI. Conclusion

We feel that data mining has excellent potential as a tool for spacecraft SOH analysis. Our first two scenarios demonstrate that supervised data mining can find cause and effect relationships between measurands and is highly useful in a forensic capacity. If a particular cause manifests itself within SOH and the corresponding effect can be quantified, a decision tree or PCA analysis will likely reveal the results. Of course, the quality of the output depends in large part on solid data preparation, which is precisely why we spent a significant amount of time ensuring our discretization and summarization step was accurate and optimized for efficient execution. Without the latter, usability suffers and analysts will be less inclined to exercise the software.

Unsupervised data mining, and in particular the one class support vector machine algorithm, appears to be very useful for anomaly identification and automated notification. Once the analyst identifies a subset of measurands that adequately represent overall system state and builds a corresponding model, the system can prepare new data and apply it to the model on any desired period - all in a completely automated capacity. Because the process is data driven rather than analysis driven, it requires minimal long-term involvement from the hardware engineers. Instead, these same engineers can leverage their time in the most effective way possible: on demand and only when required by extenuating circumstances.

Data mining also represents an excellent way to leverage a large data repository. As this repository continues to grow, so does the potential for useful knowledge extraction. Conveniently, the longer a spacecraft flies, the more data mining is useful to ensure survivability as components degrade and suffer the effects of the harsh space environment. We feel that data mining is a vital tool for any program office wishing to increase the duration and return on investment of a multi-million dollar mission. In addition, due to the domain-agnostic nature of time-series data mining, the time and resources spent to develop a suite of tools can be leveraged again and again for future missions of any type.

One area that remains to be examined is real-time anomaly resolution. We feel this also has strong potential, especially in the context of unsupervised execution via the one-class SVM we discussed in our second unsupervised scenario. Provided the underlying model remains reasonably up-to-date, an analyst can quickly apply a new set of data containing the anomalous interval to determine suspicious time periods. The first or second periods flagged in the corresponding report may very well represent the beginning of the anomaly and provide valuable focus in a time-senstive troubleshooting operation.

Additionally, applying dynamic window sizes on data mining may enhance the quality of anomaly detection and prevention in our future works.[14] As our current study applies a static window size determined by SMEs, data are summarized within uniform intervals and the method doesn't reflect relationships between the window size and system evolution. Developing algorithms for managing the window size dynamically and evaluating the prediction rates may offer effective anomaly prognostics and prevention in the SOH domain.

Acknowledgments

We would like to thank Clark Poore for initiating the original project of transitioning SOH data from flat files into a relational database. Without his vision and perseverance, our data mining effort would never have been possible. We wish to thank Todd Jenkins, Mark Strong, and their system administration team for delivering and maintaining a bullet-proof database platform that continues to operate with no unscheduled downtime. We would also like to thank our managers John Feddema and Dan Schmitt for encouraging us to pursue this endeavor and for setting aside the necessary funding to make it happen.

References

[1] Hyers, R. W., McGowan, J. G., Sullivan, K. L., Manwell, J. F., and Syrett, B. C., *Condition monitoring and prognosis of utility scale wind turbines*, Energy Materials: Materials Science and Engineering for Energy Systems, 1(3), 2006, 187-203.

[2] Hartigan, J. A., and Wong, M. A., *Algorithm AS 136: A k-means clustering algorithm*, Journal of the Royal Statistical Society. Series C (Applied Statistics), 28(1), 1979, pp.100-108.

[3] Milenova, B. L., and Campos, M. M., *O-cluster: Scalable clustering of large high dimensional data sets*, IEEE International Conference on Data Mining, IEEE, 2002, pp. 290-297.

[4] Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., and Williamson, R. C., *Estimating the support of a high-dimensional distribution*, Neural computation, *13*(7), 2001, pp.1443-1471.

[5] Yairi, T., Nakatsugawa, M., Hori, K., Nakasuka, S., Machida, K., and Ishihama, N., *Adaptive limit checking for spacecraft telemetry data using regression tree learning*, Systems, IEEE International Conference on Man and Cybernetics, Vol. 6, IEEE, , 2004, pp. 5130-5135

[6] He, F., and Shi, W., *WPT-SVMs based approach for fault detection of valves in reciprocating pumps*, IEEE American Control Conference, Vol. 6, IEEE, 2002, pp. 4566-4570.

[7] Lindsay, S., and Poore, C., *An optimized database for spacecraft state-of-health analysis*. IEEE Aerospace Conference, IEEE. 2012, pp. 1-8.

[8] Han, J., Kamber, M., and Pei, J., *Data mining: concepts and technique*, Morgan kaufmann, 2006.

[9] Oracle®, "Oracle® Database", URL: http://www.oracle.com/us/products/database, [September 2013].

[10] Oracle®, "Oracle® SQL Developer", URL: http://www.oracle.com/technetwork/developer-tools/sql-developer/overview/index.html, [September 2013].

[11] Oracle®, "Oracle Data Mining", URL: http://www.oracle.com/technetwork/database/options/advanced-analytics/odm/index.html, [September 2013].

[12] Safavian, S. R., and Landgrebe, D., *A survey of decision tree classifier methodology*, Transactions on Systems, Man and Cybernetics, 21(3), 1991, IEEE, pp.660-674.

[13] Shlens, J., *A tutorial on principal component analysis*, Systems Neurobiology Laboratory, University of California at San Diego, 2005.

[14] Suh, M. K., Woodbridge, J., Moin, T., Lan, M., Alshurafa, N., Samy, L., Mortazavi, B., Ghasemzadeh, H., Bui, A., Ahmadi, S., and Sarrafzadeh, M., *Dynamic Task Optimization in Remote Diabetes Monitoring Systems*, Healthcare Informatics, International Conference on Imaging and Systems Biology (HISB), IEEE, 2012, pp. 3-11.