JOURNAL OF AEROSPACE COMPUTING, INFORMATION, AND COMMUNICATION
Volume 9, No. 2, October 2012

General Purpose Data-Driven System Monitoring for Space Operations

David L. Iverson*, Rodney Martin†, Mark Schwabacher‡, Lilly Spirkovska§, and William Taylor¶
NASA Ames Research Center, Moffett Field, California 94035

Ryan Mackey#
California Institute of Technology, Jet Propulsion Laboratory, Pasadena, California 91109

J. Patrick Castle** and Vijayakumar Baskaran††
Stinger Ghaffarian Technologies, NASA Ames Research Center, Moffett Field, California 94035

DOI: 10.2514/1.54964

Modern space propulsion and exploration system designs are becoming increasingly sophisticated and complex. Determining the health state of these systems using traditional methods is becoming more difficult as the number of sensors and component interactions grows. Data-driven monitoring techniques have been developed to address these issues by analyzing system operations data to automatically characterize normal system behavior. The Inductive Monitoring System is a data-driven system health monitoring software tool that has been successfully applied to several aerospace applications. Inductive Monitoring System uses a data mining technique called clustering to analyze archived system data and characterize normal interactions between parameters. This characterization, or model, of nominal operation is stored in a knowledge base that can be used for real-time system monitoring or for analysis of archived events. Ongoing and developing Inductive Monitoring System space operations applications include International Space Station flight control, spacecraft vehicle system health management, launch vehicle ground operations, and fleet supportability. As a common thread of discussion this paper will employ the evolution of the Inductive Monitoring System data-driven technique as related to several Integrated Systems Health Management elements. Thematically, the projects listed will be used as case studies. The maturation of Inductive Monitoring System via projects where it has been deployed or is currently being integrated to aid in fault detection will be described. The paper will also explain how Inductive Monitoring System can be used to complement a suite of other Integrated System Health Management tools, providing initial fault detection support for diagnosis and recovery.

Received 16 May 2011; accepted for publication 20 February 2012.

* Computer Engineer, Intelligent Systems Division, Mail Stop 269-4. David.L.Iverson@nasa.gov.
† Computer Engineer, Intelligent Systems Division, Mail Stop 269-1.
‡ Computer Scientist, Intelligent Systems Division, Mail Stop 269-3.
§ Computer Engineer, Intelligent Systems Division, Mail Stop 269-3.
¶ Computer Engineer, Intelligent Systems Division, Mail Stop 269-2, posthumous.
# Senior Researcher, 4800 Oak Grove Drive, Mail Stop 126-147.
** Computer Scientist, Intelligent Systems Division, Mail Stop 269-3.
†† Computer Scientist, Intelligent Systems Division, Mail Stop 269-3.

I. Introduction

As modern space propulsion and exploration systems improve in capability and efficiency, their designs are becoming increasingly sophisticated and complex. Determining the health state of these systems using traditional parameter limit checking, functional model-based, or rule-based methods is becoming more difficult as the number of sensors and component interactions grows. Data-driven monitoring techniques have been developed to address these issues by analyzing system operations data to automatically characterize normal system behavior. System health can be monitored by comparing real-time operating data with these nominal characterizations, providing detection of anomalous data signatures indicative of system faults or failures.

Data-driven techniques, which automatically generate system models from data, have a number of advantages over other methods for monitoring complex space vehicles. Unlike manually constructed functional model-based systems, the developer does not need to understand or encode the internal operation of the system. The information required to monitor the system is automatically derived from archived data collected during system operation. Unlike rule-based systems, data-driven systems do not require system analysts to define nominal relationships among sensors. Analysts can and often do determine these relationships for systems with few sensors; it is more difficult to analytically determine the nominal relationship among a large number of sensors. Data-driven techniques are not limited to low-dimensional spaces and can work as effectively with dozens of parameters as they do with a few. Knowledge bases formed by data-driven techniques are also easy to update. As the operating envelope of the monitored system is expanded, data-driven techniques can be quickly retrained to incorporate the new behavior into the knowledge base. The expertise and time-consuming process of updating a hand-crafted functional model or rule base to maintain consistency with the new operation is not required. Furthermore, as the training data set is updated with additional operations data, the resulting nominal behavior models tend toward improved system characterization.

Several data driven software tools have been successfully applied to aerospace operations for both real-time system monitoring and archived data analysis [1]. One such tool, Inductive Monitoring System (IMS) [2], uses a data mining technique called clustering to analyze archived system data and characterize nominal interactions between selected parameters. This characterization, or model, of normal operation is stored in a knowledge base that can be used for real-time system monitoring or analysis of archived events. In monitoring, system data is periodically compared with the nominal IMS model to produce a measure of how well current system behavior matches normal behavior defined by the training data used to construct the knowledge base. The degree of deviation from expected nominal system behavior is summarized with a single scalar "distance from nominal" value for each tested data sample. Significant deviations from the nominal system model can be indicative of potential system malfunctions or precursors of significant failures. Inductive Monitoring System also provides information pertaining to the relative amount of contribution from each monitored parameter to any detected deviations. This can be helpful in isolating the cause of an anomaly.

This paper discusses the use of the IMS data-driven technique as related to several Integrated System Health Management (ISHM) projects. The scope of IMS-based monitoring applications continues to expand with current development activities. Successful IMS deployment in the International Space Station (ISS) flight control room has led to generalization and applications in other ISS flight control disciplines. It has also generated interest in data-driven monitoring capability for next-generation ground operations and spaceborne systems. Several projects have been undertaken to evaluate and mature the IMS technology and complementary tools for use in future spaceflight programs. These include a vehicle system management experiment for the Air Force TacSat-3 satellite, ground systems monitoring for NASAs Ares I-X launch vehicle, monitoring of Space Shuttle cryogenic fuel loading, and fleet supportability applications. This paper describes the maturation of IMS via these projects and also explains how IMS can be used to complement a suite of other ISHM tools, providing initial fault detection support for diagnosis and recovery.

II. Data Mining for Space Operations

Many space operations organizations maintain extensive archives of operational data from both spacecraft and ground support systems. Data mining methods can be used to analyze the data found in these archives and extract information about typical parameter behavior and parameter interactions. In particular, data driven anomaly detection (AD) techniques can process the data to find unusual events, or outliers, in data for a given subsystem. These AD

Pressure A	Valve 1 Position	Pressure B	Valve 2 Position	Pressure C	Temperature 1	Temperature 2
2857.2	86.4%	1218.4	96.2%	1104.1	49.8	37.6

Fig. 1 Sample data vector.

techniques can also automatically analyze archived nominal system data to characterize normal system performance. Comparing incoming real-time data to a nominal model produced from historic data can inform the user when current system behavior differs from previous system performance.

A. Distance-Based AD

One powerful feature of many data driven AD techniques is the ability to simultaneously analyze multiple parameters. This feature allows them to discover and model interactions between related parameters that might be difficult to notice when monitoring the parameters individually. A basic data structure used for distance-based analysis is a vector of parameter values (Fig. 1). Vectors containing N values are treated as points in an N-dimensional vector space. An appropriate distance metric is used to calculate the distance between these points. The familiar Euclidean distance metric has proved effective in many applications, though other metrics may also be useful. The underlying premise of distance-based AD is that anomalous data points will fall a significant distance away from typical, nominal data points.

For system health monitoring applications, vector parameters are typically instantiated with concurrent sensor values collected from the data stream. Additional computed (derived) values or parameter values from previous data samples can be included in the vector as well. For instance, increased system insight can often be obtained by incorporating values in the vector such as the rate of change of a pressure, the difference between two related temperature sensors, or the difference between commanded and actual values for a set point or actuator position. Also, augmenting the vector with values collected during previous time slices can implicitly capture short-term system operation patterns and trends. System experts can often suggest useful telemetry and derived parameters to use in the health monitoring vectors.

Before processing, vector values are typically normalized by applying z-score normalization or a similar method to each of the parameters. Additionally, in many cases, it may be advantageous to increase or decrease the significance (weight) given to certain vector parameters. For instance, if maintaining a specific operating pressure is critical to a system, the weight of that pressure value could be increased so a small deviation in the pressure would manifest as a larger change in the associated vector parameter, increasing monitoring sensitivity to variations in that parameter. Conversely, if the monitored system is not particularly sensitive to a certain parameter, such as ambient temperature, the weight of the ambient temperature value could be decreased to reduce the chance of unnecessary alarms when that parameter value changes by an insignificant amount.

Each monitored system will present unique characteristics. Techniques have been developed to assist with parameter selection and weighting based on automated data analysis. However, normalization and parameter weighting schemes may sometimes benefit from manual tuning to suit the situation or meet particular monitoring goals.

B. Inductive Monitoring System

The IMS is a distance-based AD tool that uses a data-driven technique called clustering to extract models of normal system operation from archived data [2]. Inductive Monitoring System works with vectors of data values as described in the previous section. During the learning process, IMS analyzes data collected during periods of normal system operation to build a system model. It characterizes how the parameters relate to one another during normal operation by finding areas in the vector space where nominal data tends to fall. These areas are called nominal operating regions and correspond to clusters of nearby, similar points found by the IMS clustering algorithm. Inductive Monitoring System represents these nominal operating regions as hyper-boxes in the vector space, providing a minimum and maximum value limit for each parameter of a vector contained in a particular hyper-box. These hyper-box cluster specifications are stored in a knowledge base that IMS uses for real-time telemetry monitoring or archived data analysis. Figure 2 shows an overview of the IMS method.

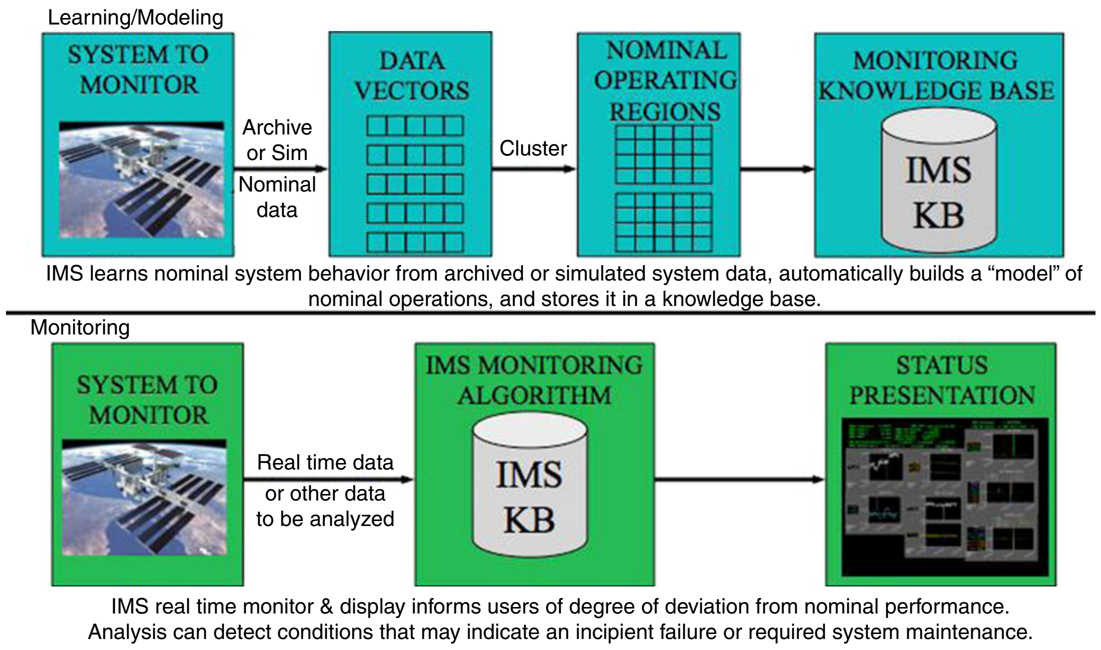

Learning/Modeling
SYSTEM TO MONITOR
Archive or Sim
Nominal data
DATA VECTORS
Cluster
NOMINAL OPERATING REGIONS
MONITORING KNOWLEDGE BASE
IMS KB
IMS learns nominal system behavior from archived or simulated system data, automatically builds a "model" of nominal operations, and stores it in a knowledge base.
Monitoring
SYSTEM TO MONITOR
Real time data or other data to be analyzed
IMS MONITORING ALGORITHM
IMS KB
STATUS PRESENTATION
IMS real time monitor & display informs users of degree of deviation from nominal performance. Analysis can detect conditions that may indicate an incipient failure or required system maintenance.

Fig. 2 Inductive Monitoring System overview.

1. *IMS Learning Process*

In general, the number and extent of nominal operating regions created during the IMS learning process is determined by three learning parameters: the "maximum cluster radius" is used to adjust the size and number of clusters derived from a fixed number of training data points, the "initial cluster size" is used to adjust the tolerance of newly created nominal operating regions, and the "cluster growth percent" is used to adjust the percent increase in size of a nominal operating region when incorporating new training data vectors. More specifically, the learning algorithm builds a knowledge base of clusters from successively processed vectors of training data. As such, the clustering approach is incremental in nature, which distinguishes it from well-known methods such as k-means clustering where the resulting clusters are independent of the ordering of the vectors. With the processing of each new training data vector, the distance from this new vector to the centroid of the nearest cluster in the knowledge base is computed. If this distance is below a prespecified value—the "maximum cluster radius"—the new vector is summarily incorporated into that cluster. The upper or lower limits for each affected dimension of the cluster are expanded, respectively, according to the "cluster growth percent" parameter to reflect the inclusion of the new vector. This incremental, inductive process gives IMS an advantage over other clustering methods such as k-means, since it tends to group temporally related points during the learning process. The grouping of temporally related points may also aid in discovering distinct system operations, making IMS more amenable to the specific goal of monitoring time series data for system operations.

The "cluster growth percent" parameter is used to adjust the learning rate. It establishes a fixed "growth" percentage difference for expansion of each dimension when updating previously formed clusters. This "cluster growth percent" learning parameter is therefore clearly proportional to the learning rate, due to the increased number of training data points that will be assigned to each new cluster per iteration for higher values of the "cluster growth percent" parameter. Naturally, the number of clusters in the knowledge base for a given training data set will increase as the "maximum cluster radius" and "cluster growth percent" values are decreased. Therefore, an inverse relationship between the maximum cluster radius and the number of clusters in the knowledge base exists. This dependence can be exploited to regulate the final size of the knowledge base in order to accommodate resource limitations in the computers running IMS.

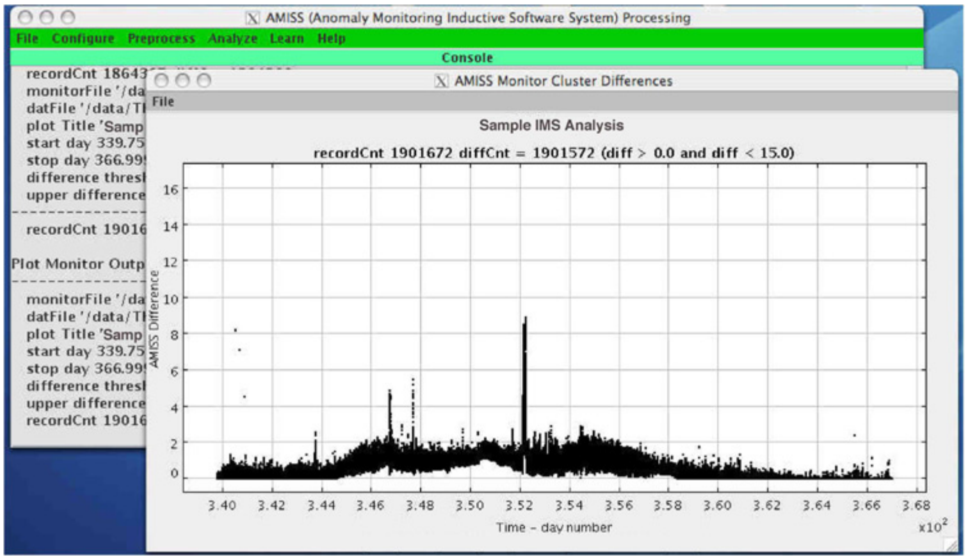
If the distance between a newly processed vector and the centroid of the nearest cluster in the knowledge base is above the prespecified "maximum cluster radius" value, a new cluster is created. The formation of a new cluster is accomplished by creating a hyper-box whose dimensions are based upon forming a window around each element of the new training data vector. The window is defined by introducing the "initial cluster size" parameter which is used to adjust the learning tolerance. This "initial cluster size" learning parameter represents a fixed percentage of the value for each dimension of the new training vector. As such, it relates directly to the size of newly established clusters, otherwise known as the "learning tolerance". The "initial cluster size" and "cluster growth percent" learning parameters also act as buffers which enable a provisional allowance for manufacturing sensor tolerances and for sensors that may have suffered from deterioration due to wear. Furthermore, these learning parameters provide increased coverage to compensate for training data that may not fully characterize the nominal performance envelope. A more algorithmic discussion of the knowledge base generation process has been previously published [3] and is motivated by comparisons to k-means and density-based clustering techniques.

2. *IMS Monitoring Process*

During the monitoring operation, IMS reads and normalizes real-time or archived data values, marshals the data into the predefined vector structure, and searches the knowledge base of nominal operating regions to see how well the new data vector (i.e., the query vector) fits the nominal system characterization. After each search, IMS returns the distance from the query vector to the nearest nominal operating region, called the composite distance. Query vectors comprised of data that match the normal training data well will fall within a nominal operating region hyper-box and have a composite distance of zero. If one or more of the data parameters is slightly outside of expected values defined by the boundaries of the nominal hyper-box, a small nonzero result is returned. As incoming data deviates further from the normal system data, indicating a possible malfunction, IMS will return a higher composite distance value to alert users to the anomaly. Inductive Monitoring System also calculates the contribution of each individual parameter to the composite deviation, reporting the distance between the selected nominal hyper-box boundary and the new vector parameter in each dimension; this information can help identify and isolate the cause of the anomaly.

3. *IMS Development Environment*

An IMS development environment facilitates production of IMS monitoring applications. It centers on a graphical user interface (GUI) that consolidates common IMS monitoring application development tasks (Fig. 3) including


AMISS (Anomaly Monitoring Inductive Software System) Processing
File
Configure
Preprocess
Analyze
Learn
Help
Console
AMISS Monitor Cluster Differences
File
Sample IMS Analysis
recordCnt 1901672 diffCnt = 1901572 (diff > 0.0 and diff < 15.0)
AMISS Difference
Time – day number
x10^2

```
recordCnt 18643
monitorFile '/da
datFile '/data/Tl
plot Title 'Samp
start day 339.75
stop day 366.99
difference thresl
upper difference
---------------
recordCnt 19016

Plot Monitor Outp
---------------
monitorFile '/da
datFile '/data/Tl
plot Title 'Samp
start day 339.75
stop day 366.99
difference thresl
upper difference
recordCnt 19016
```

Fig. 3 Inductive Monitoring System development environment GUI showing composite results review and data editing tool.

selecting useful data parameters and parameter weights, extracting and validating data from archive files, detecting and removing spurious and off-nominal data points from the training data set, and building IMS knowledge bases. This development environment is intended to enable IMS end users, such as flight controllers or mission engineers, to conveniently develop their own IMS monitoring applications without consulting data mining specialists.

III. IMS Maturation via Space Operations Applications

The IMS software tool has been deployed in NASA mission control to support real-time telemetry monitoring and has generated interest in data-driven monitoring capability for other NASA programs. Several diverse applications and deployments have enabled evaluation and maturation of the IMS technology and complementary tools for use in future spaceflight programs. These include a vehicle system management experiment for the Air Force TacSat-3 satellite, prelaunch ground diagnostics for NASAs Ares I-X development flight test, real-time monitoring of Space Shuttle cryogenic fuel loading, and monitoring of an analog planetary habitat. Additionally, IMS is under evaluation for spaceflight fleet supportability tasks. The maturation opportunities afforded by each application are listed at the start of its subsection.

A. ISS Mission Control

Opportunities: real-time mission control operations, tool generalization

Thus far, IMS has been deployed in NASAs ISS flight control room in support of two flight control disciplines: Attitude Control and Thermal Operations (THOR). The ISS Control Moment Gyroscope (CMG) attitude control system consists of four large gyroscopes, each mounted in a gimbal system that can rotate the CMG about the two axes perpendicular to the gyroscope spin axis (Fig. 4). The CMGs operate as nonpropulsive attitude control devices that exchange momentum with the ISS through induced gyroscopic torques.

As they have aged, some of the CMGs have degraded enough to malfunction and require replacement. Given their history, the ISS Attitude Determination and Control Officer (ADCO) flight controllers are interested in detecting early symptoms of degradation in the CMGs. A deployment of data-driven system health monitoring applications in the ISS flight control room is assisting with that task.

Working with ADCO flight controllers, nine telemetered and four derived CMG parameters were selected for real-time monitoring. These parameters include CMG vibration, bearing temperatures, rotation speed, gimbal rates, electric current draw, and ISS rotation rates, along with derived parameters for temperature and electrical current rates of change over a time window. Seven to 10 months of archived data were analyzed for each of the four CMGs. The data were sampled at a 1 Hz rate, formed into vectors of 13 values, and four IMS monitoring knowledge bases were constructed from the collected data. Each CMG was analyzed individually to capture its unique characteristics.

The IMS monitoring application was integrated with the NASA Mission Control data server software to access real-time telemetry in the ISS flight control room. Four IMS processes, one per CMG, run on the ADCO flight control

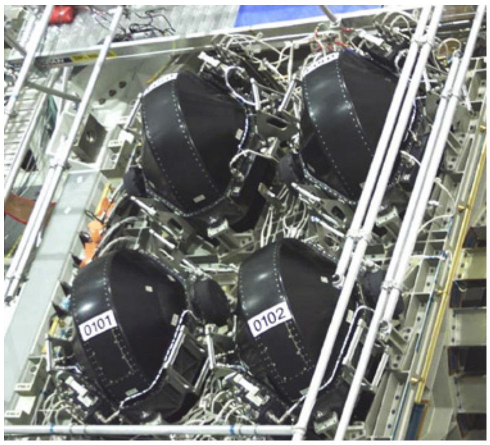


Fig. 4 International Space Station CMGs.

console to provide continuous real-time monitoring. Once per second, each process compares incoming telemetry data with the appropriate CMG knowledge base and returns the amount of overall deviation, if any, from the nominal training data. It also returns the contribution of each individual parameter to any deviation to aid in isolating the source of the deviation. These IMS results are published back to the Mission Control data server for access and monitoring by other Mission Control software applications. Inductive Monitoring System composite distances are plotted on ADCO console displays and automated alerts issued if significant deviations occur.

Successful deployment and certification of the IMS CMG monitoring system led to further development of real-time data-driven monitoring for ISS subsystems. The IMS CMG application was generalized to accept an arbitrary number of user-selected input parameters and to run on any controller console in the ISS flight control room. The resulting tool, called AMISS for Anomaly Monitoring Inductive Software System, has been applied to additional ADCO subsystems and the THOR domain, monitoring subsystems of the ISS External Thermal Control System (ETCS).

The ETCS is used to dissipate heat onboard ISS. Excess thermal energy from inside the ISS is transferred to liquid ammonia cooling loops in the ETCS. The heated ammonia is then circulated to radiators (RAD) and cooled as thermal energy is released into space (Fig. 5). External Thermal Control Systems are separated into two independent loops with three major subsystems each: the Pump Module (PM), the Ammonia and Nitrogen Tank Assemblies (ATA/NTA), and the RAD. The PM circulates coolant through the ETCS, the ATA/NTA stores reserve ammonia coolant and maintains pressure within the ETCS systems, and the RAD system controls the flow of coolant to each of three thermal RAD [4].

Anomaly Monitoring Inductive Software System knowledge bases were constructed from a year of archived ETCS operations data, one knowledge base for each major ETCS subsystem, and two knowledge bases covering all pertinent parameters in each ETCS loop. The subsystem modules monitor for anomalies that occur within each subsystem whereas the full loop modules also watch for anomalies that are only apparent in subsystem interactions. The AMISS monitoring application and ETCS knowledge bases were certified for operations in June 2009 and have supported flight control activity continuously since installation. The ETCS knowledge bases have been regularly updated with new telemetry data sets to adapt to the occasional thermal system reconfigurations performed to accommodate new ISS modules and additional cooling requirements for experiment facilities and increased crew size.

B. TacSat-3

Opportunities: multi-tool integration, flight hardware deployment

The TacSat-3 Vehicle System Management (TVSM) project was an experiment to implement fault detection and AD algorithms and diagnosis tools integrated with actual flight software. This was done to evaluate ISHM algorithms, including IMS, for suitability and ease of integration and to provide a realistic estimate of performance in future space missions. In 2007, NASA teamed with the Air Force Research Laboratory (AFRL), Alliant Techsystems Inc. (ATK), and Interface and Control Systems Inc. (ICS) to integrate these algorithms with AFRLs TacSat-3 flight software,

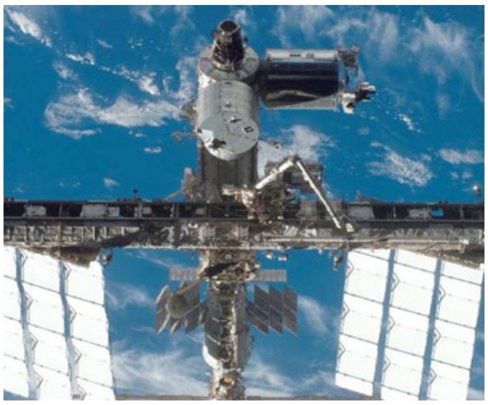


Fig. 5 International Space Station external view showing ETCS RAD at lower left and lower right.

using a flight-like avionics testbed at NASA Ames Research Center, ground control software from ICS, and test data from the TacSat-3 vehicle.

The resulting TVSM software package demonstrated [5] the ability to monitor spacecraft subsystems including guidance, navigation, and control and power. Its capabilities included data analysis in real-time to detect faults and unusual conditions, fault diagnosis, the ability to combine reports from dissimilar ISHM algorithms, and recommendation of a positive recovery decision to the flight software executive. The TVSM experiment successfully demonstrated the feasibility of IMS in a spaceflight context with respect to both functional and performance considerations. In TVSMs nominal configuration as would be expected on-board TacSat-3, the IMS reasoner examined 10 major spacecraft components using a model containing over 1000 data clusters, producing diagnostic conclusions at a rate of one per five seconds. The CPU load for IMS in this configuration was measured at under 1% (AITech S950 processor board). Alternately, the TVSM was stress-tested with simulated data rates of over 100 times those found on the actual TacSat-3 spacecraft (maximum rate tested approx. 27 Hz before overloading the CPU), demonstrating error-free operation of all TVSM algorithms including IMS.

Following this experiment, NASA has modified the TVSM software to investigate use of IMS and other algorithms in safety-critical implementations, including ARINC-653 compliant software, in anticipation of application to future crewed space exploration and commercial aviation. The TVSM is also under consideration for flight demonstration, either involving a small spacecraft or an experiment on the ISS.

C. Ares I-X

Opportunities: ground system operations, modeling new systems with minimal operations data, receiver operating characteristic curves for optimization

NASAs Constellation program (canceled in 2010) intended to develop vehicles to replace the Space Shuttle after its retirement in 2011. The first planned vehicle consisted of the Ares I crew launch vehicle and the Orion crew exploration vehicle. Following these craft, the heavy-lift Ares V cargo launch vehicle and the Altair lunar lander were to be built to provide beyond low Earth orbit capability. Ares I-X, launched in October 2009 from NASA Kennedy Space Center (KSC), was an uncrewed test flight of the Ares I crew launch vehicle. The Ares I-X test vehicle was powered by a single, four-segment reusable solid rocket booster (SRB), like those used on the Space Shuttle, modified to include a fifth inactive segment to simulate the Ares I five-segment booster. The Ares I-X vehicle also included mock-ups of the upper stage, the Orion crew module, and the launch abort system to simulate the integrated spacecraft.

The Ares I-X Ground Diagnostic Prototype (GDP) [6,7] was developed to detect and diagnose faults in the Ares I-X first-stage thrust-vector control (TVC) and associated hydraulic support system (HSS) ground equipment while the vehicle was in the Vehicle Assembly Building (VAB) and while it was on the launch pad. Similarly to the TVSM project, GDP combined IMS with a rule-based tool used to determine the system mode and a functional model-based tool used to help operators diagnose and isolate the cause of detected system anomalies. The three tools were interfaced with live data from the Ares I-X vehicle and from the ground hydraulics; the resulting GDP outputs were displayed on screens in the Ares I-X control center.

Separate IMS knowledge bases were constructed for the VAB and for the launch pad from data collected in each respective location. Because it was a new vehicle, there was no historic Ares I-X sensor data available early in the project. However, the first stage of Ares I-X was derived from the Space Shuttle SRB and the Ares I-X first-stage TVC was very similar to the Shuttle SRB TVC. Moreover, the Space Shuttle HSS was used with Ares I-X. Therefore, IMS was trained and tested on historical Shuttle SRB and HSS data with the expectation that it would be similar. Postflight analysis revealed that these assumptions held up modestly well. “Anomalies” identified by IMS were not failures in the modeled TVC or HSS systems but rather reflected differences between Shuttle operations and Ares I-X operations. There were no failures in the modeled systems during Ares I-X operations.

The IMS “distance from nominal” scores mentioned earlier—a composite score for the set of parameters as a whole and a separate score for each individual parameter—were displayed in the Ares I-X control center. Additionally, to facilitate observation of out of bounds conditions, a visual alarm annunciated when scores exceeded a specified threshold.

There are a number of methods to determine the alert threshold. Theoretically, if there were a comprehensive training data set available, IMS could learn the high and low thresholds of each individual parameter under every

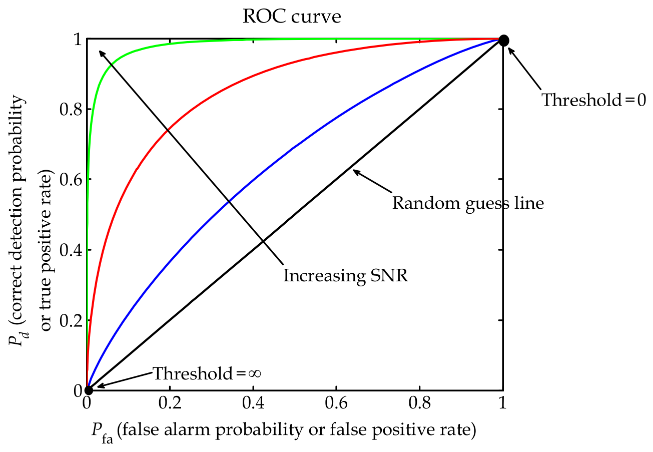


Fig. 6 Sample ROC curve.

operating condition, so any deviation from the IMS model would warrant an alert. However, as a practical measure, the simplest and most often used method in previous deployments has been for the user to specify a threshold value based heuristically upon the statistics of available validation data. For example, a three-sigma standard deviation value of the composite score applied to validation data may serve as an alert level, or higher multiples of this value, depending on the skewness and kurtosis of the underlying distribution of the composite score. However, due to the fact that the distribution will invariably change as a function of the data provided, the alert thresholds may vary drastically from one data set to another. For GDP, we employed an alternative method which has gained traction and is quickly becoming the standard for assessing performance of classification algorithms within the machine learning community and more recently the aerospace ISHM domain. This method involves the use of receiver operating characteristic (ROC) curve analysis, and the area under the ROC curve (AUC).

The ROC curve essentially plots the true positive rate against the false alarm rate for all possible threshold values, as shown in Fig. 6. It therefore can be used as a design tool in order to select an alert threshold according to preestablished requirements for minimum missed detection and/or false alarm rates. However, in order to compute true positive and false alarm rates, it is necessary to obtain a "ground truth" representation for each monitored example. In this case, an "example" can represent an individual validation flight, or a single point in time. Furthermore, in order to compute true positive and false alarm rates with a reasonable level of accuracy, there is a need to obtain a statistically significant number of labeled examples, both nominal and anomalous. However, the availability of nominally classified examples often surpasses the availability of anomalously classified examples, with the latter often considered to be a rare commodity. In fact, the Shuttle SRB TVC and associated HSS have had very few failures. We thus had available to us an abundance of nominal data for training, but very little anomalous data for testing before launch. It was necessary to simulate faults in part to make up for the deficit of available anomalously classified examples. These simulations involve the injection of specific system faults or degraded modes of operation into historical Shuttle data. (More detail on these simulations can be found in Martin et al. [8]) As such, in order to artificially boost the number of anomalously classified examples, classified time points are used as examples in lieu of individual validation flights for construction of the ROC curve, even with the availability of simulated fault data.

The selection of an alert threshold by using the ROC curve is therefore in part based upon an implicit requirement for the availability of a statistically significant number of anomalous examples. As such, this requirement may be recognized as a distinct disadvantage, since it does not exist for the heuristic method described earlier. However, one advantage of using ROC curve analysis is in its inherent robustness against the use of skewed distributions. The distribution here, however, is not the distribution of the IMS score mentioned previously, but the distribution representing the population of nominally and anomalously categorized examples. Furthermore, unlike other alert threshold selection techniques, use of the ROC curve may also be used for optimized selection of IMS learning parameters (i.e., the number of clusters in a knowledge base). This can be performed by using the AUC, which represents overall classification discriminability. Therefore, by maximizing the AUC with respect to the IMS parameters that control the number of clusters, we can ensure that the knowledge base used by IMS in order to perform threshold or alert selection has the best AD capability possible.

As seen in Fig. 6, the AUC has a classic increase in relation to the signal-to-noise ratio (SNR). This relationship is well established, and is derived from the origins of the ROC curve for use in radar applications. Unlike the SNR, it is often the case that IMS tuning parameters do not have a similar straightforward relationship to the AUC. However, this relationship is implicit in Fig. 6 due to the fact that optimization of algorithmic design parameters is performed by using the AUC. In a sense, the ROC curve is “tuned” by attempting to maximize the AUC by choosing the appropriate value of IMS tuning parameter(s) (e.g., the number of clusters) to allow for maximum predictive capability. This can be thought of as choosing the ROC curve with the highest SNR based upon such parameters. In this case theoretically the “signal” can loosely be assumed to characterize anomalous behavior, while the “noise” can be assumed to characterize nominal behavior. More detail on the optimization of IMS parameters in the context of a similar simulation-based study is provided in a study by Martin [9].

D. Launch System Ground Support Equipment

Opportunities: monitoring system executive control, missing parameter adaptation, level of effort assessment

The IMS tool is under consideration for continued use at NASA KSC to monitor vehicle and ground systems for future spaceflight programs, such as the 21st Century Launch Complex now under development. To demonstrate additional applications in this domain, prototype IMS-based systems have been developed and deployed to monitor the liquid hydrogen (LH2) ground support equipment (GSE) used to fuel the Space Shuttle. Equipment used to fuel future liquid powered launch vehicles is likely to be similar to the Shuttle LH2 GSE, though the operating characteristics will differ (for example, tank capacity and fill rates will differ). Still, much of the development can be done in advance based on the Shuttle LH2 GSE, including selecting and weighting parameters as well as integrating with the data and application architecture. To this end, a prototype IMS-based LH2 GSE real-time monitoring system was deployed in the KSC Launch Control Center (LCC) to monitor the LH2 fuel loading process for the STS-134 Shuttle launch.

The Shuttle LH2 ground system consists of a large storage sphere located near the launch pad which contains LH2 that is transferred to the Shuttle external tank via a series of fuel lines, valves, and various control and safety systems (Fig. 7). The system is instrumented with sensors to measure pressure, temperature, flow rate, fuel level, atmospheric gaseous hydrogen concentration, and other relevant parameters. Inductive Monitoring System parameter vectors were specified by Shuttle LH2 engineers for several LH2 GSE subsystems. These subsystems included LH2 system pressures and temperatures, anti-ice equipment, orbiter to external tank connections, and the ground umbilical carrier plate (GUCP) that connects the external fuel lines to the Shuttle external tank. The GUCP system was of particular interest due to previous occasions when hydrogen leaks developed in the GUCP connection. The LH2 GSE vectors contained from 6 to 22 parameters, depending on the subsystem. Two IMS knowledge bases were constructed for the GUCP system. One GUCP knowledge base used 22 parameters that provide general coverage of the system, and

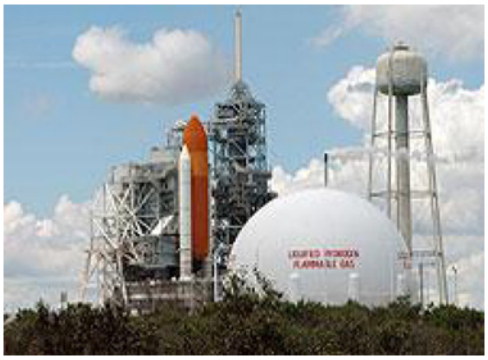


Fig. 7 Space Shuttle LH2 fueling system.

the other used 9 parameters that provide a more focused view of values that were more likely to be influenced by a leak in the system.

Data sets collected during 30 previous Shuttle LH2 fuel loads were retrieved from the archives to use for training and testing IMS. Some of these data sets contained anomalies, such as GUCP leaks that occurred during STS-119 and STS-133 fuel loading. These anomalous data sets were used to test, tune, and evaluate IMS performance before the system was deployed in the LCC. Each data set was further divided into up to eight distinct fuel load phases. A separate IMS knowledge base was constructed for every phase for each subsystem using all available nominal data from each phase. This resulted in a total of 40 knowledge bases. Anomaly detection executive software was developed to connect to the data acquisition system, determine the current operating phase, activate the phase-appropriate knowledge bases, marshal the parameter values to send to IMS, and relay IMS results to graphical display software. Inductive Monitoring System composite deviation values and the amount of individual parameter contributions to any deviations were available for display.

After integration and installation in the LCC, the system was tested with a playback data stream from the final (nominal) STS-133 fuel load. Displayed results matched IMS output obtained from off-line analysis of STS-133 archive data, and phase transitions, with associated IMS knowledge base switches, proceeded smoothly. The monitoring system was online and running with live telemetry data for the duration of the first STS-134 fuel load. There were no significant anomalies in the monitored LH2 system during that load, and observed IMS results accurately reflected the nominal operations. This first STS-134 launch attempt was scrubbed due to a Shuttle hydraulic line heater malfunction. The scrub occurred after the fuel loading was complete and the process was well established in the replenish phase. Because the scrub, only seven of the eight loading phases were monitored with live STS-134 data. The final phase, terminal count, occurs in the final minutes before launch and did not occur on this attempt. Since the data vectors, parameter weights, and other IMS tuning parameters have been established and tested on this first STS-134 launch attempt, updating the knowledge bases to include new nominal data for use during future launch attempts is simply a matter of extracting the fresh data into a file for each of the predefined vectors and phases and then rerunning the IMS training routines.

We were fortunate to have extensive data archives from prior Shuttle launches to use for IMS training on this project. This may not be the case with newer vehicles that do not have extensive previous operations history. In these situations, initial IMS training data sets may come from high fidelity simulations, or from data collected from similar systems on other vehicles, as was the case for Ares I-X. In the case of LH2 and other subsystems, test data can also be used to build initial knowledge bases. For instance, before the first flight a vehicle will typically be repeatedly fueled and de-fueled so operators can learn the actual operational characteristics of the interacting hardware. Inductive Monitoring System can be trained on these fueling tests. Thus, even before the first launch of a new vehicle, high fidelity training data should be available to enable production of a prototype IMS monitoring application for LH2 GSE or other systems as early as the first launch. It is expected that the operational data from that launch and perhaps the subsequent three to four following launches will provide enough data to capture expected nominal operations well, especially with systems such as the LH2 GSE, which have historically been stable, that is, they have not had many changes after initial development.

In addition to proving feasibility, one goal of the LH2 GSE prototyping effort was to perform a trade study to determine what types of systems lend themselves well to a data-driven approach. Another goal was to document the effort (time and cost) required for building an IMS application. In this case, initial LH2 IMS knowledge bases were available within a month of acquiring the archived data. Because future vehicles and much of the supporting ground equipment are still being designed and developed, archived data is not currently available. Working with existing equipment to determine the characteristics of suitable systems and the effort required to build monitoring applications will allow effective planning for future deployments of IMS and similar data-driven systems. Lessons learned from IMS application to Ares I-X and Shuttle systems will also provide valuable insight for determining effective uses of data-driven techniques in the future programs.

1. *Missing Parameters*

Sometimes during the course of operations, one or more monitored parameters will become unavailable or invalid for various reasons such as restricted data bandwidth, parameter substitution in the data stream, or partial loss of signal. We experienced such partial data losses during real-time monitoring of Ares I-X and STS-134 fuel loading.

This situation presents a challenge in distance-based AD, since the system models are based on full vectors of parameter values. To provide continuous monitoring even when parameter values are unavailable, the algorithm must adapt to the reduced input data set and continue to provide useful results. There are several techniques to handle missing parameter values, some of which are described subsequently. A comparison of these techniques was performed before the STS-134 monitoring demonstration, and the most effective method was implemented for this system. In practice, the implemented technique handled the missing parameters with favorable results.

Four approaches were considered for handling missing parameters: imputation, marginalization, relearning, and cluster projection. Imputation, in which the missing parameter value is filled in with the most recent valid value, a default value, or an average or projected value from previous data samples, is a popular approach. While there is some merit in filling the missing parameter with a “reasonable” value, it can lead to problems in certain situations. For example, if the most recent value is reused, it might not be consistent with the current state of the system. Even worse, an average value might not even be a valid value that the system can generate.

A second popular approach is marginalization wherein the entire vector of data associated with the current time record is discarded. The problem with this approach is that potentially useful information in the remaining valid data is being thrown out and the system is left unmonitored for the duration of the data drop out. Because the drawbacks of imputation and marginalization, we investigated two alternative approaches: relearning and cluster projection.

In anticipation of data drop out, it is possible to build, or relearn, knowledge bases in advance using partial parameter sets. During the monitoring phase, when it is discovered that certain parameters are missing, it would then be possible to load into memory the corresponding knowledge base prebuilt from the reduced parameter set. Such an approach should at least in theory handle missing parameters but the time taken to generate all possible knowledge bases can be prohibitive, especially if the system being monitored has a few dozen variables and multiple missing parameters are considered.

Instead of building a knowledge base for each reduced set of parameters, one can exploit the underlying structure of clusters in IMS in order to handle missing parameters. The clusters in IMS are axis-aligned hyper-boxes and therefore projecting them to a lower dimension is straightforward and still maintains cluster integrity. Projection of clusters in a knowledge base is achieved by simply discarding the lower and upper bounds of the corresponding parameters from the clusters, a step that can be done with a negligible computational expense during the monitoring phase. In general, a knowledge base originally consisting of n parameters can be projected down to a reduced knowledge base with $n - m$ parameters when m parameters become unavailable.

The idea behind cluster projection when two and three parameters are involved is illustrated in Fig. 8. The figure on the left shows a two-dimensional knowledge base with four clusters whereas the one on the right depicts a three-dimensional knowledge base with three clusters. In both cases, the effect of projection when the parameter p_2 drops out is shown. In the two-dimensional case, the clusters project down to intervals whereas in the three-dimensional case, the boxes project down to rectangles. It is evident from the illustration that the separation between clusters in IMS, or any clustering algorithm for that matter, may decrease as we project down from a higher dimension to a

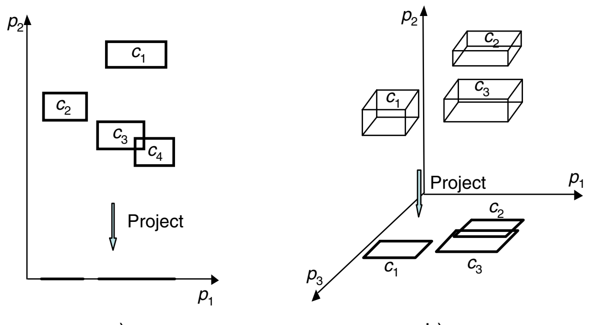


Fig. 8 Cluster projection of IMS clusters due to missing parameters: a) projecting clusters in the two-parameter case when p_2 becomes unavailable reduces rectangular regions into intervals along the axis of p_1, b) similarly, in the three-parameter case, rectangular boxes project down to rectangles in the $p_1 - p_3$ plane.

lower dimension, possibly diminishing its ability to detect anomalies. In other words, eliminating the information provided by a (now missing) parameter could decrease the IMS monitoring sensitivity, as would be expected from any monitoring technique. Not having information about a parameter precludes an assessment of whether that parameter is within an acceptable range. But the use of this projection technique still allows the system to produce useful results when parameters go missing. The possibility of false negative results may increase, due to the unavailability of information, but it has been shown that the technique should not result in increased false positive results. The projection technique was tested on several diverse data sets and systems with favorable results, comparable to the much more costly relearning technique. Thus, it was selected for implementation in this deployment of IMS for LH2 GSE.

E. Spacecraft Fleet Supportability

Opportunities: data-driven supportability analysis

The data-driven approach can be applied to fleet supportability tasks. Fleet supportability encompasses the comprehensive performance monitoring and analysis conducted to assure that a vehicle fleet retains its intended design performance, reliability, and safety throughout the program lifecycle. It includes all nonreal-time analyses of recorded data, including data generated during manufacturing, transport, mission preparation and prelaunch operations, in-flight, and postmission. It establishes a system that can determine the health condition of reusable components, the expected need for imminent replacement, the need for redesign of reusable or expendable components, or the need for procedure/process redesign.

Fleet supportability is typically associated with large fleets of similar equipment, for example, a fleet of F/A-18 aircraft. The operating characteristics of the fleet are used to develop a failure distribution. This profile can then be used to predict the failure of an individual component on similar instances of the fleet type. Thus, from F/A-18 operational profiling it can be inferred that a fuel pump may fail in as few as 500 h or as many as 2000 h but the majority of pumps fail at 1000 h (rates are notional, for explanation purposes only). This failure rate curve can be used to extend (or shorten) replacement periods to maintain a desired in-service failure rate. The replacement threshold can be set very low to account for early failures. This can result in increased costs due to much unnecessary replacement or maintenance. Alternatively, the replacement threshold can be set high to mitigate such costs. This can result in some components failing in service—not an acceptable solution for critical components on spacecraft.

Another factor against relying on the life usage model described earlier for spacecraft applications is that it will be difficult to collect enough performance data for statistical significance and accurate inference. Reusable launch vehicle components, such as SRBs, will be in service for only a few minutes per launch and expendable components (e.g., upper stage and upper stage engine; new components for each launch) will be in service for less than 10 min. Additionally, the fleet size of space launch programs will be much smaller than the typical aircraft fleet, with a handful of launches per year.

With the limits imposed by number and duration of operational cycles, alternative approaches for fleet supportability are under evaluation. Spacecraft programs will not have the benefit of evaluating the performance of components in thousands of instances. Instead, a data-driven approach like IMS can be applied to look for subtle differences in performance that may indicate impending failure.

Performance-degrading conditions can occur throughout a component's lifetime, from design through launch and reentry to refurbishment. Numerous prelaunch tests verify compliance with expected performance. If a subsystem fails a test, a diagnostic system can isolate the fault to a line replaceable unit or perhaps even to a component. An analysis is then performed to determine why that component failed—whether it was due to changes in manufacturing, materials, or quality control, environment during transport, or effects from previous launches, reentry, recovery, and refurbishment. The results of that analysis can then inform required maintenance planning, guide part restocking decisions, or identify needed manufacturing improvements.

Subtle differences in performance may not be noticed if the prelaunch tests pass. Current monitoring techniques—primarily visual comparison of graphs—focus on identifying differences in a single parameter. Because IMS analyzes the interaction between all the parameters simultaneously, it can complement these tests by detecting that the performance on a test is still within limits but is different than on previous tests either on this system or on previous systems of the same type. This may lead to early identification of under-performing components before in-flight failures on subsequent flights. It may also identify the need for revised limits for prelaunch tests. Further, it could

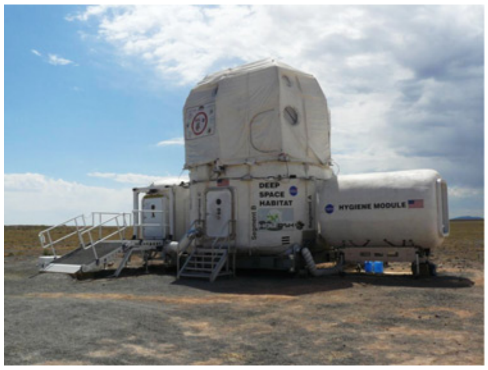


Fig. 9 Habitat Demonstration Unit—DSH.

assist with both the postfailure analysis as well as degraded-performance analyses by identifying conditions that degrade performance or reduce a component's lifetime. Alternatively, if components are performing well even in the presence of unusual conditions, it may indicate that environmental constraints could be relaxed. Finally, IMS output may be useful as an input to prognostics algorithms if a correlation with future performance can be determined.

F. Habitat Demonstration Unit: Deep Space Habitat

Opportunities: operational checkout, incremental in situ training

To explore concepts, mature technology, and establish procedures, NASA often uses analog missions—Earth-based missions that simulate a distant destination. One such analog is the Habitat Demonstration Unit (HDU), a facility that could represent lunar operations, missions to near-Earth objects (e.g., an asteroid), or a habitat on Mars (Fig. 9).

For a portion of each year, the HDU is fielded in the Arizona high desert with a given set of test objectives for a so-called Design Reference Mission (DRM). In 2011, the HDU was representative of a Deep Space Habitat (DSH) and the HDU-DSH DRM centered on concepts for integrated crew-ground mission operations for remote missions that will incur communication delay due to speed-of-light constraints. In some situations, because of significant communication delay with ground controllers, spaceborne crews must have the ability to independently identify anomalies, determine the cause of faults, and recover system functionality. Tools to help the crew attain this autonomy will need to be easy to use, require little crew interaction, and provide information that is easy to interpret with minimal crew training. Toward this end, IMS was demonstrated during the 2011 HDU-DSH deployment to evaluate and mature concepts for a next-generation system health management tool under design to assist both crew and ground-based flight controllers.

In the initial project concept, we anticipated using IMS to help detect if the HDU-DSH suffered damage during transport, whether it was reassembled properly at the test site, and if there was any performance impact from the new operating environment. With this aim, IMS was trained on nominal operating data collected from the HDU-DSH before its departure from the NASA Johnson Space Center (JSC) en route to Arizona. Unfortunately, changes to the data system configuration between the JSC installation and the Arizona-deployed HDU-DSH did not allow a direct comparison between the fielded facility and the baseline data collected at JSC and, thus, IMS was not demonstrated for operational checkout.

Nevertheless, IMS proved effective starting from scratch in the field deployment. Inductive Monitoring System was trained on nominal HDU-DSH operating data collected during in-the-field dry-runs of integrated mission operations tests. Even with the small set of available training data, IMS was able to detect a data network system failure within the first day of operation. This failure went unnoticed by operators performing traditional telemetry monitoring until they were informed of the IMS result. Periodic updates of the HDU-DSH knowledge bases in the field using newly collected operations data further refined the models and helped to evaluate and mature techniques that allow nonexperts to effectively train and update data-driven monitoring systems, like IMS, in situ. This capability will allow “space-based” crews to use just-in-time system training to keep their health monitoring systems up to date and well correlated with the current operating environment.

IV. Related Work

Inductive Monitoring System is an example of a one-class data-driven AD method, meaning that the algorithm relies on training data from only one class—nominal data. One-class data-driven AD methods are appropriate when little or no failure data are available for training, as is often the case in space applications. However, when large amounts of failure data are available (such as data from a high-fidelity simulator or system stress testing), better accuracy can often be obtained by using multiclass data-driven methods, better known as supervised classification algorithms.

Training data is not always available; for example, for a new system without adequate nominal operations history. A useful alternative to the data-driven approach in these situations is the functional model-based approach. Methods utilizing this approach require a domain expert to encode knowledge about the operation of the system into a model. In this section, we briefly describe each approach and provide some of the benefits, drawbacks, and application of example algorithms.

It is often beneficial to apply multiple algorithms to achieve ISHM goals. We conclude this section with a discussion of the synergy possible through such a multipronged approach.

1. One-class Data-driven Methods

IMS, Orca, one-class Support Vector Machines (SVMs), GritBot, and Dynamical Invariant Anomaly Detector (DIAD) are all members of the one-class data-driven methods category. For AD applications, these algorithms require only nominal data, though some can also use failure data when it is available.

Orca [10] is similar to IMS in that it is distance based, but it does not use clustering. It uses the average distance to a point’s nearest neighbors as an anomaly score. One advantage that Orca has over IMS is that it can be used to find the outliers within a heterogeneous data set with respect to that data set. Because IMS assumes that all training data is nominal data, it builds clusters large enough to contain all of the training data points. Thus, if IMS analysis is run on the data set that is used to train it, it will return zero as the anomaly score for every data point, including possible outliers. We have therefore sometimes used Orca to remove outliers from a data set before using the data set to train IMS. Inductive Monitoring System extensions that approximate nearest neighbors analysis are under development. These extensions will allow IMS to perform analyses similar to, but somewhat less precise than Orca within the existing IMS framework, including application to outlier removal and real-time monitoring.

One-class SVMs [11] seek to describe the range of normal training data in such a way as to enable it to distinguish normal data from abnormal data in the future. The name “one-class SVM” is due to the possibility that only one class of data (normal data) may be available during training (if abnormal training data is available, it can be used). One-class SVMs first map the training data from the original data space into a much higher-dimensional feature space and then find a linear model, such as a hyper-plane, in that feature space that allows normal data to be segregated to one side of the linear structure (and to be separated from abnormal training data if available). The idea is that linear models in a higher-dimensional space correspond to more complicated nonlinear models in the original data space. The use of linear models allows SVMs to retain the benefit that the algorithm finds the globally optimal solution given the training set, whereas still effectively using nonlinear models. For each test point, the one-class SVM returns a measure of how strongly normal or anomalous the data point is. Since one-class SVMs are not distance based, the anomalies that they detect are often quite different from the anomalies detected by Euclidean distance-based methods such as IMS and Orca.

GritBot is a commercial product from RuleQuest Research [12]. Rather than just looking for points that are anomalous with respect to the entire data set (like IMS), GritBot searches for subsets of the data set in which an anomaly is apparent. For each anomalous point, it reports a description of the relevant subset of the data set, based on values of discrete variables or ranges of continuous variables, in which the target variable usually has a particular value (if it is discrete) or range of values (if it is continuous). The point is considered to be an anomaly because the target variable at that point is significantly different from the value of the target variable at the vast majority of the other points in the subset.

Park et al. [13] applied the BEAM (Beacon-based Exception Analysis for Multi-Missions) system to AD in Space Shuttle Main Engine data. BEAM has nine components that use nine different approaches to AD. However, in Park et al. [13] only one of the nine components of BEAM was used: the DIAD. Dynamical Invariant Anomaly Detector is an unsupervised AD algorithm, like IMS. Dynamical Invariant Anomaly Detector differs from IMS in that DIAD only considers one variable at a time, whereas IMS considers all of the variables together and looks for anomalies in the relationships among the variables, in addition to anomalies in individual variables.

2. *Multiclass Data-driven Methods*

Multiclass data-driven methods are more familiarly known as supervised classification algorithms. These methods learn a model that distinguishes between two or more classes of data. In the ISHM domain, the classes typically include nominal and one or more failure modes. When adequate data sets covering the distinct classes are available, supervised classification algorithms cannot only detect anomalies, but also determine the failure mode. Numerous supervised classification algorithms are available, including decision trees, artificial neural networks, and SVMs; they are well documented in the machine learning literature. Thus, we present only one example application.

Schwabacher et al. [14] used a supervised learning approach to automatically detect and diagnose failures in the J-2X rocket engine. They used a high-fidelity simulator to simulate a large number of faults, and then used those simulated faults to train a C4.5 [15] decision tree to detect the faults and to identify the fault mode. They demonstrated that the resulting decision tree had a very low false alarm rate and low missed detection and misdiagnosis rates.

3. *Functional Model-Based Methods*

Besides the data-driven approach, where the system model is automatically derived from data, another major technique for automating AD is the functional model-based approach. This approach manually encodes human knowledge of the structure and behavior of the target system into a model, which is then used to automatically detect faults. Examples of systems that use the functional model-based approach include Livingstone [16–18], HyDE [19], Titan [20], TEAMS-RT [21], RODON [22], SHINE [23], and MEXEC [24].

Functional model-based approaches have some benefits over data-driven approaches. First, clearly, a lack of training data impedes data-driven application development. Although some aspects can be started in advance of data availability, such as selecting parameters to include in a vector or assigning weights, knowledge bases cannot be trained and verified until adequate high-fidelity data becomes available. In contrast, for a new system whose operation is well understood, a functional model-based approach could be operational for the first turn of the key or push of the start button. Second, the development of a fault detection or fault diagnosis model could exploit simulation or engineering models built for system development purposes, though these types of models would typically require translation into a representation suitable for use by a model-based reasoning algorithm. Similarly, models being built for ISHM purposes could be used earlier in the development process to help with system design, e.g., to determine the best location for sensors to facilitate system testability and diagnostics. In addition to being useful for AD, a powerful feature of functional model-based approaches is that they can provide diagnostic capabilities with better isolation than many data-driven approaches, i.e., more specific failure cause analysis. And because they encode explicit knowledge of the system, functional model-based systems can provide more detailed explanation of results.

Functional model-based approaches also have drawbacks. Building the models can be very knowledge intensive. The developer must know or have access to knowledge about how the system operates, how its components are integrated, and how the function (or malfunction) of each part propagates to the remainder of the system. (Some functional model-based approaches model only the nominal operation and do not require modeling of possible failure modes.) It is often difficult for one person to understand in detail every piece of a highly complex system, such as a spacecraft, leading to the necessity of coalescing knowledge from multiple experts and multiple, often inconsistent, documents, and then encoding that knowledge into a consistent model that accurately captures operations. For fault diagnosis purposes, this often labor intensive process leads to a trade-off between detailed modeling of every part of a complex system or consulting these same system experts only if and when a fault occurs. In some cases, this may not be feasible. For instance, deep space missions require crew autonomy due to extended communication delays, necessitating automation independent of Earth-bound experts to achieve that capability. Also, when models are extended or updated to reflect additional knowledge or system changes, it is necessary to carefully validate that model updates do not introduce unintended side effects. Lastly, for complex systems the computing resources required to execute these models for real-time health monitoring may exceed available resources (e.g., on-board spacecraft, satellites, or aircraft).

4. *Synergy of Methods*

As alluded to in some of the aforementioned example applications, it is often beneficial to use multiple diverse system health monitoring tools for a particular application. Since each tool provides a different perspective on the system data analysis, they frequently provide complementary results. For instance, both IMS and BEAM were simultaneously deployed onboard an F/A-18 jet aircraft to monitor engine health [25]. Both tools use data-driven techniques, but each exhibited individual strengths toward the monitoring task. In many cases, the tools corroborated each other's results. In other cases, an anomaly would manifest more strongly in BEAM results than IMS results, and vice-versa, depending on the nature of the anomaly. Similar results were obtained when several different data-driven AD tools, including IMS, Orca, one-class SVMs, and GritBot, were applied to data from the Space Shuttle Main Engine [26,27]. Moreover, some tools are more suited to different aspects of the problem. For instance, one-class data-driven tools excel at detecting unanticipated anomalies, those faults that are not identified in typical failure modes analyses, and thus may not be included in functional model-based tool application. Rather than viewing various system health monitoring techniques as competitors, it is beneficial to consider them as potential partners that can provide overall synergistic improvement in any particular system monitoring task.

V. Summary and Future Work

Through practical application, it has been demonstrated that data-driven system health monitoring can be useful in a variety of space mission operations settings. Furthermore, data-driven methods can complement other data analysis and diagnosis tools, together providing ISHM and fault recovery recommendations. A number of NASA projects incorporating data-driven AD have been deployed or are under development. These projects provide proof-of-concept demonstration by combining data-driven techniques with functional model-based and rule-based software tools for fault management. They also help refine recommendations on which types of systems would most benefit from data-driven and integrated data-driven, rule-based, and functional model-based approaches for fault detection and health management. This information will help guide NASA as it builds operations and support systems for future spacecraft and space launch systems. Finally, these data-driven concepts also show promise for fleet supportability applications and will be evaluated for that task as the support plans for future space operations grow and mature.

For future work, near term plans call for continued development of tools that allow operations personnel to build and maintain their own data-driven monitoring applications without direct support from data mining specialists. This includes refinement and maturation of utilities that help select and appropriately weight useful system monitoring parameters. These will be combined with updated tools that detect and remove spurious data from archived data sets used to build monitoring applications. Initial versions of these tools have already been incorporated into the graphical integrated development environment, allowing users to conveniently proceed from raw archived data to a deployable monitoring knowledge base. In continuing development, these techniques are becoming more tightly integrated and automated toward the goal of "one-click" system model production with minimal user interaction required.

NASA mission controllers have also suggested building an on-line learning capability, where IMS automatically updates the system model based on incoming real-time telemetry data. This is a feasible project with some additional research and development. Incoming real-time data vectors could be vetted based on their IMS deviation scores, and those that fell within an acceptable threshold of previous nominal data could be incorporated into the existing knowledge base. This approach could reduce the required amount of initial training data download and analysis, and automatically incorporate nominal events that were not included in the off-line IMS training data. However, such an on-line learning capability should be used with caution in some situations since a system drifting very slowly toward an off-nominal condition might subtly update the IMS model to accept anomalous behavior as normal. Running the on-line learning system in parallel with a static baseline system or discontinuing on-line learning after a given time period may be a wise course of action.

Longer-term activities will address capabilities required for wider application of data-driven and complementary technologies to future spaceflight programs. One important task to enable the use of data-driven approaches with newly developed hardware systems will be determining how to obtain adequate system characterizations with limited operations data. This will likely involve augmentation with simulation data or synthetic data extrapolated and generalized from available operations data. A related endeavor will address generalizing and normalizing training data collected during operations performed under varying conditions, such as launches in differing weather conditions (e.g., hot, humid summer launches vs cooler winter or night-time launches), and determining how much influence the disparate conditions have on the monitored system characteristics.

The study of combining data-driven technology with other ISHM techniques is still in its infancy and holds promise of further utility, including more sophisticated diagnostic tests and prognostic capabilities. There are also many applications of data-driven approaches yet to be explored for fleet supportability tasks. For instance, multivariate data-driven approaches are naturally complementary to traditional fleet supportability statistical analysis, and may be more effective when limited instances of supported equipment are available. It can also be a valuable technique for sorting through massive amounts of data to focus attention on key pieces of information, both from extended operations and from test and assembly of a new vehicle or system. Additionally, using these techniques to pick out the most divergent data points in operations data can either be used to drive risk-based testing and verification, or in a more traditional sense to simply hand the most interesting periods to design analysts. It may also be possible to classify current excursions from data-derived models to analogous historical operating periods by the degree and pattern of divergence. This could be used to drive procedural and recovery decisions.

There are many interesting paths toward improving data-driven and hybrid ISHM techniques to provide solutions that enable future space operations to low-Earth orbit, near-Earth objects, or deep space as well as addressing Earth-based applications. While it may not provide a comprehensive, highly structured development and test regimen, we have found the opportunities provided by the application-driven development approach to be economical and effective for evaluating, determining, and maturing necessary capabilities to evolve toward that goal.

References

[1] Iverson, D. L., "Data Mining Applications for Space Mission Operations System Health Monitoring," *Proceedings of the SpaceOps 2008 Conference,* ESA, EUMETSAT, AIAA, Heidelberg, Germany, May 2008.

[2] Iverson, D. L., "Inductive System Health Monitoring," *Proceedings of the 2004 International Conference on Artificial Intelligence (IC-AI04)*, CSREA, Las Vegas, NV, June 2004.

[3] Iverson, D. L., "Inductive System Health Monitoring With Statistical Metrics," *Proceedings of the 4th JANNAF Modeling & Simulation Subcommittee (MSS) Meeting,* JANNAF, Charleston, SC, June 2005.

[4] Bolt, K., Harrison, S., and Juillerat, R., "International Space Station Thermal Control System Training Manual," NASA Document ISS TCS TM 21109, Jan. 2004.

[5] Mackey, R., Castle, J. P., and Sweet, A., "Getting Diagnostic Reasoning off the Ground: Maturing Technology with TacSat-3," *IEEE Intelligent Systems,* Vol. 25, No. 5, Sept./Oct. 2010, pp. 27–35.

[6] Schwabacher, M., and Waterman, R., "Pre-Launch Diagnostics for Launch Vehicles," *Proceedings of the IEEE Aerospace Conference*, IEEE, Big Sky, MT, March 2008.

[7] Schwabacher, M., Martin, R. A., Waterman, R., Oostdyk, R., Ossenfort, J., and Matthews, B., "Ares I-X Ground Diagnostic Prototype," *AIAA Infotech@Aerospace Conference*, AIAA, Atlanta, GA, March 2008.

[8] Martin, R. A., Schwabacher, M., and Matthews, B., "Data-Driven Anomaly Detection Performance for the Ares I-X Ground Diagnostic Prototype," *Proceedings of the International Conference on Prognostics and Health Management*, CALCE (Center for Advanced Life Cycle Engineering), Portland, OR, Oct. 2010.

[9] Martin R. A., "Evaluation of Anomaly Detection Capability for Ground-Based Pre-Launch Shuttle Operations," edited by T. T. Arif, *Aerospace Technologies Advancements,* INTECH, Jan. 2010, ISBN: 978-953-7619-96-1.

[10] Bay, S. D., and Schwabacher, M., "Mining Distance-Based Outliers in Near Linear Time with Randomization and a Simple Pruning Rule,"*Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,* Association for Computing Machinery, New York, 2003.

[11] Tax, D. M. J., and Duin, R. P. W., "Support Vector Domain Description," *Pattern Recognition Letters*, Vol. 20, No. 1113, 1999, pp. 1191–1199.

[12] GritBot. RuleQuest Research Web site. http://www.rulequest.com [cited 9 May 2011].

[13] Park, H., Mackey, R., James, M., Zak, M., Kynard, M., Sebghati, J., and Greene, W., "Analysis of Space Shuttle Main Engine Data Using Beacon-based Exception Analysis for Multi-Missions," *Proceedings of the Aerospace Conference Proceedings*, Vol. 6, IEEE, Washington, DC, 2002, pp 6-2835–6-2844.

[14] Schwabacher, M., Aguilar, R., and Figueroa, F., "Using Decision Trees to Detect and Isolate Simulated Leaks in the J-2X Rocket Engine," *Proceedings of the IEEE Aerospace Conference*, IEEE, Big Sky, MT, March 2009.

[15] Quinlan, J. R., *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, CA, 1993.

[16] Kurien, J., and Nayak, P. P., "Back to the Future for Consistency-Based Trajectory Tracking,"*Proceedings of the National Conference on Artificial Intelligence*, American Association for Artificial Intelligence, Menlo Park, CA, 2000.

[17] Williams, B. C., and Nayak, P. P., "A Model-Based Approach to Reactive Self-configuring Systems," *Proceedings of the National Conference on Artificial Intelligence*, American Association for Artificial Intelligence, Menlo Park, CA, 1996.

[18] Narasimhan, S., Dearden, R., and Benazera, E., "Combining Particle Filters and Consistency-Based Approaches for Monitoring and Diagnosis of Stochastic Hybrid Systems," *Proceedings of the 15th International Workshop on Principles of Diagnosis (DX04)*, Carcassonne, France, June 2004.

[19] Narasimhan, S. and Brownston, L., "HyDE—A General Framework for Stochastic and Hybrid Model-based Diagnosis," *Proceedings of the 18th International Workshop on Principles of Diagnosis (DX07)*, Nashville, TN, May 2007, pp. 162–169.

[20] Williams, B. C., Ingham, M., Chung, S., Elliott, P., and Hofbaur, M., "Model-Based Programming of Fault-Aware Systems," *AI Magazine*, Vol. 24, No. 4, Fall 2003, pp. 61–75.

[21] TEAMS-RT Web page. http://www.teamqsi.com/RT.html [cited 9 May 2011].

[22] RODON Web page. http://www.sorman.com/Site/System/Rodon.aspx [cited 9 May 2011].

[23] Colgren, R., Abbott, R., Schaefer, P., Park, H., Mackey, R., James, M., Zak, M., Fisher, F., Chien, S., Johnson, T., and Bush, S., "Technologies for Reliable Autonomous Control (TRAC) of UAVs," *Proceedings of the 19th Digital Avionics Systems Conference*, IEEE, Philadelphia, PA, Oct. 2000.

[24] Barrett, A., "Model Compilation for Real-Time Planning and Diagnosis with Feedback," *Proceedings of the International Joint Conference on Artificial Intelligence*, IJCAI, Edinburgh, Scotland, 2005.

[25] Mackey, R., Iverson, D., Pisanich, G., Toberman, M., Hicks, K., "Integrated System Health Management (ISHM) Technology Demonstration Project Final Report", NASA Technical Memo TM-2006-213482, Feb. 2006.

[26] Schwabacher, M., Oza, N., and Matthews, B., "Unsupervised Anomaly Detection for Liquid-Fueled Rocket Propulsion Health Monitoring," *Proceedings of the AIAA Infotech@Aerospace Conference*, AIAA, Reston, VA, 2007.

[27] Martin, R. A., Schwabacher, M., Oza, N., and Srivastava, A., "Comparison of Unsupervised Anomaly Detection Methods for Systems Health Management Using Space Shuttle Main Engine Data," *Proceedings of the JANNAF Propulsion Meeting*, JANNAF, Denver, CO, 2007.

Ella Atkins
Associate Editor