Anomaly Detection for Next-Generation Space Launch Ground Operations

Lilly Spirkovska, Ph.D.[1] and David L. Iverson[2]
NASA Ames Research Center, Moffett Field, CA, 94035-1000

David R. Hall[3]
Stinger Ghaffarian Technologies, Inc., Moffett Field, CA, 94035-1000

William M. Taylor[4] and Ann Patterson-Hine, Ph.D., P.E.[5]
NASA Ames Research Center, Moffett Field, CA, 94035-1000

Barbara L. Brown[6]
NASA Ames Research Center @ KSC, Kennedy Space Center, FL, 32899

and

Bob A. Ferrell[7] and Robert D. Waterman[8]
NASA Kennedy Space Center, Kennedy Space Center, FL, 32899

NASA is developing new capabilities that will enable future human exploration missions while reducing mission risk and cost. The Fault Detection, Isolation, and Recovery (FDIR) project aims to demonstrate the utility of integrated vehicle health management (IVHM) tools in the domain of ground support equipment (GSE) to be used for the next generation launch vehicles. In addition to demonstrating the utility of IVHM tools for GSE, FDIR aims to mature promising tools for use on future missions and document the level of effort – and hence cost – required to implement an application with each selected tool. One of the FDIR capabilities is anomaly detection, i.e., detecting off-nominal behavior. The tool we selected for this task uses a data-driven approach. Unlike rule-based and model-based systems that require manual extraction of system knowledge, data-driven systems take a radically different approach to reasoning. At the basic level, they start with data that represent nominal functioning of the system and automatically learn expected system behavior. The behavior is encoded in a knowledge base that represents "in-family" system operations. During real-time system monitoring or during post-flight analysis, incoming data is compared to that nominal system operating behavior knowledge base; a distance representing deviation from nominal is computed, providing a measure of how far "out of family" current behavior is. We describe the selected tool for FDIR anomaly detection – Inductive Monitoring System (IMS), how it fits into the FDIR architecture, the operations concept for the GSE anomaly monitoring, and some preliminary results of applying IMS to a Space Shuttle GSE anomaly.

---

[1] Computer Engineer, Intelligent Systems Division, M/S 269-3.
[2] Computer Engineer, Intelligent Systems Division, M/S 269-4.
[3] Senior Research Engineer, ISRDS, M/S 269-3.
[4] Computer Engineer, Intelligent Systems Division, M/S 269-2.
[5] Technical Area Lead, Intelligent Systems Division, M/S 269-1, AIAA Associate Fellow.
[6] ISHM-FDIR Task Lead, NASA-Ames Resident Office, MailCode ARC.
[7] Lead Electronics Engineer, Advanced Systems Branch, M/S NE-E9.
[8] Command & Control Architecture Lead, Architecture Integration & Management Office, M/S LX-S3.

I. Introduction

UNSURPRISINGLY, launch operations for manned spacecraft have evolved significantly from the first manned Mercury launches in the early 1960s. As spacecraft and launch vehicles became more complex, as operations personnel became more experienced with detecting and resolving pre-launch problems, as computer capabilities improved, and as automation technologies were developed, launch operations progressed toward the system we have in place today for Space Shuttle launch operations. Whereas the first mission controllers were inventing as they went along and gaining significant experiences with every mission, there is now a large body of knowledge that guides today's controllers. In preparation for the next generation of human exploration missions, we have been developing technologies to further improve launch operations. In particular, the goal of the Fault Detection, Isolation, and Recovery (FDIR) system is to provide tools – decision support systems – that will assist controllers with detecting faults and anomalies, diagnosing and isolating the cause of faults, and recommending procedures to recover from faults. FDIR is being developed as an integral element of the launch control system (LCS) to assist controllers in managing the health status of the ground support equipment (GSE) while the spacecraft and launch vehicles are undergoing checkout operations in the Vehicle Assembly Building (VAB) and on the launch pad at NASA Kennedy Space Center.

The FDIR system leverages two decades of technology development in the Integrated Vehicle Health Management (IVHM) area. Although the FDIR plug-and-play architecture can support a variety of tools, the initial implementation is incorporating two: Qualtech Systems Inc. TEAMS tool for fault isolation and NASA-developed Inductive Monitoring System (IMS) tool for anomaly detection. In this paper, we describe the anomaly detection portion of FDIR. Information about the FDIR architecture and the fault isolation portion of FDIR is described elsewhere.[1,2]

II. Anomaly Detection

FDIR provides for two methods of detecting anomalies. In the first method, anomalies are equated to faults – problems that can be anticipated and checked for explicitly. The typical method for this is limit checking – verifying whether a parameter value goes outside of a user-specified range of values. Multiple ranges can be defined; the user can specify a not-to-exceed value (high limit), a not-below value (low limit), or both. The user can also specify multiple high or low limits to define, for example, an advisory range, a caution range, and a warning range. The limit can be applied directly to the instantaneous value of the parameter (e.g., pressure), the change from the previous value (e.g., change in pressure), or the trend of the value over time (e.g., slope of change in pressure, i.e., pressure is slowly or rapidly decreasing). The limit(s) must be provided by the user based on expectations for system behavior that evolved from knowledge of the system and can vary during different modes of operation (i.e., the limits may differ for operations in the VAB vs. at the pad). This method is already in use for Shuttle and will be carried over for monitoring GSE for next generation vehicles. The discipline engineers will develop most of the limit checking detectors as part of their standard monitoring suite. FDIR will supplement it with additional detectors as needed for the fault isolation task.

To supplement limit checking, FDIR will provide an alternative method for anomaly detection. Rather than checking just a single parameter by itself, we will apply a data fusion method to monitor the relative relationship of values of a set of related parameters. By comparing the current relationship within a group of parameter values to the expected relationships within that group, the data fusion approach provides a more sophisticated monitoring process that can detect the presence of anomalies even when all parameter values remain within limits. Data fusion enables detection of anomalies at their earliest stages, before limits are crossed. Early detection of subtle system degradation can prompt operators to service the system or adjust system operations to maintain peak efficiency. Clearly this approach can be accomplished with standard limit checking for a small number of parameters whose relationships are known in advance. For example, the relationship between the temperature parameter and the corresponding pressure parameter can be pre-determined; that the expected relationship continues to hold during operations can be verified with a pre-programmed (simple) algorithm. Unfortunately, the complexity of the approach increases as the number of parameters increases and when the relationship between the parameters is not readily apparent.

To address these limitations, FDIR is providing a *data driven* data fusion method. Data-driven systems take a radically different approach to reasoning. Rule-based and model-based systems both require the manual extraction of system knowledge. That is, rule-based systems require manual coding of rules elicited from experts, while model-based systems require manual construction of a model of the system, a time-consuming and knowledge-intensive process. In contrast, data-driven systems automatically encode system knowledge. At the basic level, they start with

archives of data that represent the functioning of the system and learn, using various machine-learning techniques, the behavior of system operations.

Data-driven systems can use either supervised or unsupervised learning – the distinguishing characteristic is whether a "teacher" provides the "answer" for how the data should be classified. In unsupervised learning, the data-driven system is provided with only vectors of parameters. It is not told in advance whether a particular vector should be associated with a nominal operation or a particular type of failure mode. It must determine that classification on its own. In supervised learning, the data-driven system is provided with the association and only needs to learn how to distinguish between the various categories of data provided. A special case of supervised learning (arguably also called unsupervised learning) is so-called one-class learning. In this situation, the data-driven system is provided data for a single type (class) of behavior, typically nominal operation. The data-driven system, like in supervised learning, then only needs to learn the distinguishing characteristics of that data and, thus, learn to reject (or flag) data that do not resemble those characteristics. There are numerous examples of data-driven systems, including systems that use support vector machines[3] (SVM, e.g., NASA's Mariana), nearest neighbor (e.g., Orca[4], cooperatively developed by ISLE and NASA), neural networks[5] (numerous, including Gensym's Neuronline), clustering (e.g., NASA's IMS), hybrid (e.g., NASA/JPL's BEAM[6]), as well as conventional numerical statistics.

FDIR is utilizing the Inductive Monitoring System (IMS) developed at NASA Ames Research Center.[*] IMS has many desirable characteristics. The underlying algorithm of IMS is easy to understand, and therefore, more likely to be trusted by NASA mission controllers and operations personnel. It requires very little computational power. It is trained on only nominal data – almost a necessity for space systems that are so well maintained that failure data are very rare. And it is not limited to low-dimensional spaces and works as effectively with dozens of parameters as it does with a few.

IMS has successfully demonstrated real-time health monitoring for complex NASA systems in both space and aeronautics domains. In one of the most compelling demonstrations, IMS detected anomalies in archived ascent data of the impacted wing of Shuttle STS-107 Columbia, although the traditional monitoring techniques did not show any out of limit parameter values.[7] In another demonstration, when tested on archived data, IMS detected precursors of the International Space Station (ISS) Control Moment Gyroscope CMG1 failure nearly 15 hours in advance of the actual failure.[8] As a result of these and other demonstrations, IMS is now deployed as a certified full-time ISS flight control room tool to monitor a number of systems, including CMG health on the Attitude Determination and Control Officer (ADCO) console and external thermal control systems monitored on the ISS Thermal Operations and Resources Officer (THOR) console.

III.   Inductive Monitoring System (IMS)

IMS uses a data-driven machine learning technique that automatically extracts system parameter relationships and interactions from archived nominal system data producing a monitoring capability quickly for almost any system for which archived nominal data is available. IMS does not require knowledge engineers or modelers to capture precise details of system operation. It only requires archived nominal data. Lacking that, IMS can learn the relationship based on the parameter values from a high-fidelity simulator, with the accuracy of the simulator determining the accuracy of learned monitoring capability. When neither operational nor simulated data is available for a system, IMS can be trained on a very similar system. However, as with simulated data, the accuracy of the learned monitoring capability will depend on the similarity of the systems. We followed this approach and got reasonable results for anomaly detection in an FDIR prototype for the Ares I-X launch, discussed below.[9]

IMS has two phases: training and monitoring. Training is done off-line and monitoring can be done on-line (real-time monitoring) or off-line (post-flight analysis). Figure 1 shows a high-level architecture of IMS and how it connects to the FDIR architecture.

---

[*] IMS is available for technology transfer. Contact NASA Ames Research Center Technology Partnerships Division (technology.arc.nasa.gov; email: technology@mail.arc.nasa.gov; phone 650-604-1754).

American Institute of Aeronautics and Astronautics

The goal during the training phase is to learn how the system normally behaves. The input to the learning algorithm is a data set representing nominal system operation. The training flow consists of the following: From the full set of data sensors available, the user selects the subset of parameters that adequately encode the behavior of the system. These parameters, normalized and weighted to assign desired significance, are formed into a data vector. The user can then clean/validate the training set by eliminating outlier vectors. IMS is trained on most of the cleaned data set, with the exception of a portion left out to validate learning generalization. When satisfied with the performance (generalization and speed on the target hardware), IMS is trained on the full, cleaned data set. The output is a list of clusters that define nominal operations, where a cluster is defined as a

Figure 1. High-level architecture of IMS.

rectangular N-dimensional box (where N is the number of parameters used for training), with the cluster encoding the two extreme points of the box. (These "Nominal Ops Clusters" (shown in Fig. 1) are referred to as the knowledge base (KB) in the remainder of this paper.) We have found that this model representation is much easier for users to understand than the representation of other data-driven systems; for example, neural networks and similar techniques produce coefficients that are difficult to relate to system data. As mentioned previously, IMS is trained off-line but can be retrained if system characteristics, and thus expected behavior, change, or as additional data is collected. A supporting IMS knowledge base (KB) development environment facilitates KB development by FDIR customers who are not IMS tool experts. It allows the user to specify the data sets to be used for training, to customize the training parameters, to verify learning performance on test data sets, and to iterate until desired performance is achieved. Figure 2 shows an example of a simple two-parameter data set separated into clusters.

The goal of the monitoring step is to determine if the system is behaving differently than during operations when training data was collected. A data flow diagram of the run-time (or off-line, if so desired for post-flight analysis) process is shown in Fig. 3. The monitoring step extracts the relevant parameters from the incoming real-time system data, normalizes and weights them, as was done during training, and then compares the incoming vector to the clusters generated during training. It outputs similarity scores, the distance between the vector and the nearest cluster for the vector as a whole (a composite score) and a distance from the nearest cluster for each
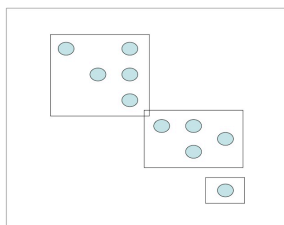

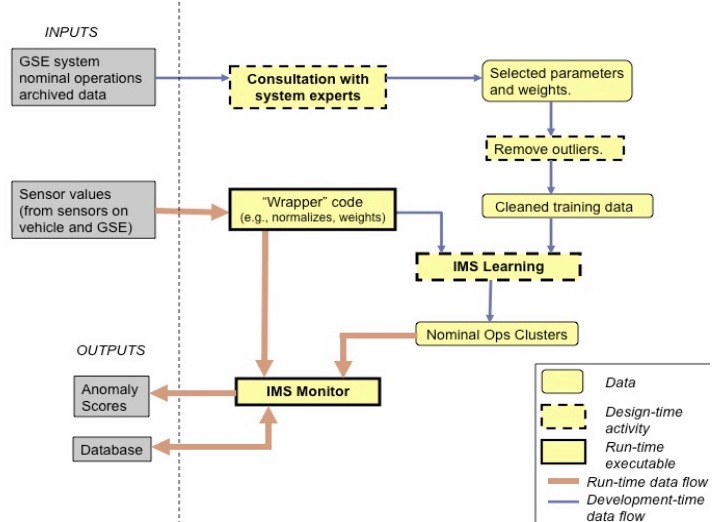**Figure 2. A simple example of a two-parameter data set separated into clusters.**
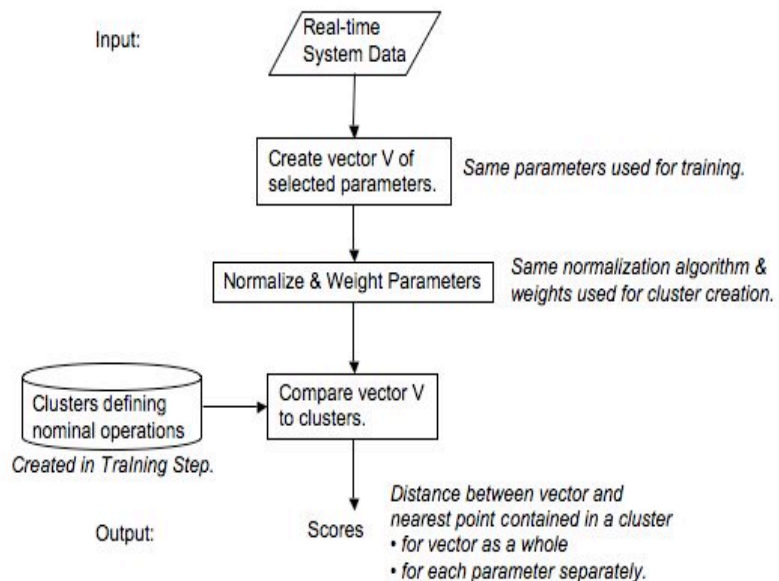

Figure 3. IMS monitoring flow to determine if the system is behaving differently than during training.

parameter separately. These distances represent the deviation of the current operation from nominal operations as defined by the training set – a measure of how "out of family" the behavior is. These scores can then be sent through an *Alert Logic Filter* to filter out data spikes and issue alerts only when a specified number of sequential incoming vectors are anomalous. Alternatively, the scores can be plotted allowing users to assess the severity of any system anomalies and watch for worsening trends over time. Moreover, the individual parameters' deviation scores can help users identify which subsystems are exhibiting off-nominal characteristics relative to expected system behavior under current operating conditions.

IV. Application Development Considerations

FDIR will supplement traditional single-parameter monitoring with an IMS-based anomaly detection tool only for those GSE subsystems for which it makes sense. FDIR technology maturation prototypes and ISS applications have led to a better understanding of the approach to take and the decisions that must be made during development of an IMS application. First, data must be available for training. If archived data from real-world system operations is not available, IMS can also be trained on high-fidelity simulator data. For optimal monitoring performance, the training data must accurately reflect nominal system operations. It should cover all expected situations: any situations that are not covered in the training set may be flagged as anomalies during monitoring.

A second consideration is choosing how to handle system modes. During nominal operations, many systems function in fairly distinct operating modes. In aircraft, for example, one can separate the operation, at a very high level, into engine start, taxi, takeoff, en route, approach and landing. During each of these phases or modes, we expect different system behavior. For example, for a few seconds after engine start, the oil pressure may be low. Although an acceptable condition during that phase, a low oil pressure is of high concern during the en route phase. When developing an IMS application, the applications engineer must decide whether nominal operations (and the associated training data) should be separated into different operating modes. If so desired, IMS can be separately trained on each mode to produce a mode-specific cluster knowledge base. During monitoring, the current operating mode would be determined first, and then the correct cluster knowledge base would be utilized to determine if the situation is nominal or anomalous. If separate mode training is not desired, IMS can be trained on the entire collection of archived data. However, the clusters produced may not be as focused as mode-specific clusters and could have higher tolerance for off-nominal behavior. Collaboration with a domain expert is desirable when making this decision.

Domain expertise is also very desirable for the third consideration – selecting parameters to monitor and specifying the influence (weight) to associate with each parameter. The goal is to select a minimal number of parameters that provide good monitoring capability. It is important to select parameters that are related to the behavior of the system, and not to just train on all parameters available in the data set. This prevents IMS from expending resources on modeling relationships with minimal monitoring value. Selecting appropriate parameters leads to improved generalization, that is, the algorithm will perform as well on previously unseen data as it does on the training set of data. Also, the number of parameters inversely affects the monitoring speed because during monitoring, each parameter in an incoming data vector must be compared to each parameter range for each cluster. (In practice, we have found that the speed consideration is of little consequence. IMS is very fast and can monitor faster than a human operator can read the resulting scores.)

The selected parameters can be continuous or discrete. The range of all parameters is scaled to cover similar ranges. This scaling process occurs automatically, without user involvement, and is based entirely on the training data. The goal is to eliminate any undue influence of any parameters, as explained by the following example. Assume one parameter, P1, has a range of 1-20. For P1, a change of 1 reflects a 5% change. Assume another parameter, P2, has a range of 1-100. For P2, a change of 1 reflects only a 1% change. Thus, a given change in P1 is likely to be more significant to the overall behavior of the system than an equal magnitude change for P2. To eliminate this disparity, all parameters can be scaled so the values for each parameter are expressed relative to their expected range. For example, normalized values for every parameter could range from 0-100%. Each parameter then has equal influence on the overall computation.

If a higher influence is desired for a parameter, the *weight* associated with the parameter can be increased. Similarly, if a smaller influence is desired, the weight associated with the parameter can be decreased. This weighting process typically requires user involvement – a system expert is best qualified to specify the contribution that each parameter should have. If a system expert is not available, the weights can be adjusted until adequate performance on a validation set is reached. Automated techniques to assist in parameter selection and weighting are under development.

Discrete parameters deserve special consideration. When a parameter with few possible values is scaled, the contribution of that parameter gets undue influence. For example, in the worst case, binary parameters are scaled such that one value maps to 0% and the other value maps to 100%. Thus, a flip of the value represents a 100% change. It is recommended that discrete parameters be avoided; if not possible, discrete parameters should be weighted to remove any undue influence. It has been found in practice that the weight associated with discrete parameters should be significantly decreased to accurately reflect their influence. Internally, IMS changes the 0-100% scaled values to 0-10000. It is recommended that the weight associated with binary discrete parameters is 1/10,000. To increase the influence, the weight can be increased, but should remain fairly small (e.g., 2/10,000).

A fourth consideration is customizing the algorithm. Three parameters are available to specify the cluster formation: maximum cluster extent (radius, $\varepsilon 0$), initial centroid tolerance ($\varepsilon 1$), and cluster growth tolerance ($\varepsilon 2$). The three parameters have a direct effect on the number of clusters generated. Limiting cluster radius (i.e., smaller $\varepsilon 0$) typically results in more clusters for a given training data set. Thus, monitoring sensitivity is improved, but because there are additional clusters for comparisons with incoming data, the monitoring speed may decrease. The other two parameters, $\varepsilon 1$ and $\varepsilon 2$ can be increased to account for larger sensor tolerance values and limited training data.

The fifth factor – data rate and target hardware – influences the customization parameter values. Clearly, the higher the incoming data rate, the faster the IMS algorithm must be. Similarly, the less capable the target hardware (slow processor or limited memory), the faster the IMS algorithm must be. It is important to consider these matters jointly with other decisions.

Finally, the applications engineer must consider the timing of various sensor inputs and how to deal with stale data. In some applications, any time a sensor value is received, the entire vector, with only that sensor value replaced, is sent through IMS for anomaly detection. In other situations, it may be more appropriate to delay processing so that stale data is avoided.

V. Ground Operations Approach

Ground Support Equipment (GSE) refers to non-flight systems, equipment, or devices necessary to support such operations as transporting, receiving, handling, assembly, inspection, test, checkout, servicing, launch, and recovery of space systems. A variety of GSE exist for the Space Shuttle at NASA Kennedy Space Center (KSC); we expect a similar variety of GSE to be needed to prepare the next generation vehicles for launch. Examples of ground systems include thermal control subsystems, environmental control subsystems, ground special power, hazardous-gas leak detection subsystems, liquid hydrogen, liquid oxygen, thrust vector control hydraulics, mobile launcher tower hydraulics, and the sound suppression subsystem. In general, FDIR will support spacecraft processing in the vehicle processing facilities, support vehicle stacking in the VAB, and support pre-launch operations at the launch pad. The data from each of these operations will be available through the Launch Control System (LCS) in the Launch Control Center (LCC) Firing Rooms (FR). The FDIR users are the ground systems and flight systems discipline engineers in the FRs.

Anomaly detection for ground operations can be configured to suit the needs of the system discipline engineers. FDIR could provide a single KB for all ground subsystems – not a likely scenario given the large number of parameters and the independence of some subsystems from the others. FDIR could provide one KB per subsystem – more viable but again not a likely scenario given the interdependence of some subsystems during some operating modes. Most likely, FDIR would provide multiple KBs per subsystem to reflect the need to separate behavior into operating modes and to combine parameters from multiple, closely related subsystems in one KB to benefit from, for example, the behavior of a flight subsystem being reflected in the behavior of the attached ground subsystem. The fully extendable FDIR architecture can support numerous processes and provides capabilities to start and stop processes as the user desires, gather and distribute the appropriate parameters to each process, and track operating mode changes. This enables multiple KBs to be run within one process (serially) or via separate processes (in parallel). LCS operators would arrange the process configuration through data files eliminating the need to modify any computer code. This approach not only provides flexibility but also facilitates certification of the application – the application code needs to be certified only once since it will not change for different configurations of processes or different KBs. Moreover, if KBs are updated after a mission, for example, to include nominal data from an expanded operating envelope of the monitored system(s), only the configuration data files would change. Again, there would be no need to modify – and thus, no need to recertify – any code.

Determining the desired process configuration and developing the appropriate set of KBs is done off-line. During launch preparations, subsystem sensor data would be made available to FDIR and hence to anomaly detection. Each second (or user-specified duration), the data would be sent through IMS for comparison with the

appropriate KB. The resulting deviations would be provided to the user, displayed as required (e.g., on a trend plot). The FDIR system would also track the current system operating mode and manage the appropriate set of KBs.

VI. Ground Operations Examples

We have developed two prototypes of the FDIR system. One prototype was used for pre-launch monitoring of the thrust vector control (TVC) subsystem of the Ares I-X vehicle and the corresponding hydraulic support system (HSS) of the Ares I-X GSE.[10] Because the Ares I-X TVC is very similar to the Shuttle Solid Rocket Booster (SRB) TVC, we trained IMS on the Shuttle data and then monitored Ares I-X. Although we found a few differences – flagged as anomalies by IMS – between the operation of the two vehicles, the method was quite effective. A comprehensive report on the approach and findings is in preparation.

The second prototype is being developed for the liquid hydrogen (LH2) GSE with the expectation that next generation launch vehicles will also have liquid-fueled engines. For anomaly detection capability demonstrations, we are training IMS on Shuttle data. One such demonstration involved the LH2 leak at the Ground Umbilical Carrier Plate (GUCP) – which is the interface of the External Tank (ET) and the hydrogen vent line – on Space Shuttle mission STS-119 in March 2009 and then again on STS-127 in July 2009. On both missions, during the final stages of loading LH2 ("LH2 tanking") into the ET, sensors measured a significant leak near the GUCP, leading to launch scrubs. Figure 4 shows the data for relevant parameters for LH2 tanking.

For the demonstration, we trained IMS on two configurations of the same data set from a single nominal tanking (from the final tanking prior to the launch of STS-119). In one configuration, we used all seven LH2 system parameters. In the other, we used just four parameters, eliminating a discrete valve-open indicator and two parameters indicating local H2 concentration (the leak detectors). The remaining four parameters included two pressure parameters and two temperature parameters. The test data set was from the off-nominal STS-119 tanking. The IMS composite deviation scores (i.e., the distance or deviation from a test set data vector to the nearest nominal cluster) for the seven-parameter configuration are shown in Fig. 5 and the composite deviation scores for the four-parameter configuration are shown in Fig. 6. Notice the similarity of the shape of the seven-parameter IMS composite deviation score to the tanking data shown in Fig. 4. The score is very closely tracking the leak detector measurement. Note that even after removing the leak detectors from the data set, IMS continues to indicate anomalous behavior, though a bit delayed and not quite as strong as the indications from the full vector. This demonstration is a foretaste of FDIR anomaly detection and highlights IMS's ability to determine off-nominal states even when direct measurements are not available.
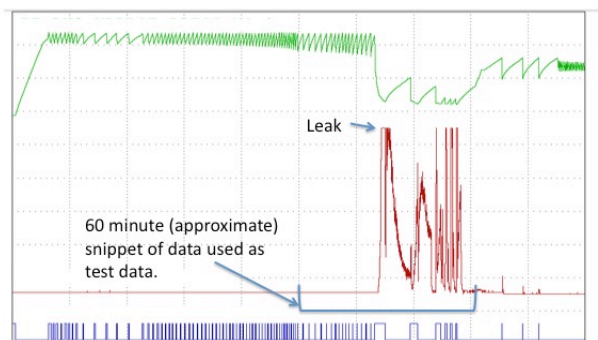

Figure 4. LH2 tanking data. The red plot shows measurements from a GUCP-local leak detection sensor. Approximately 6 hours of data is shown.
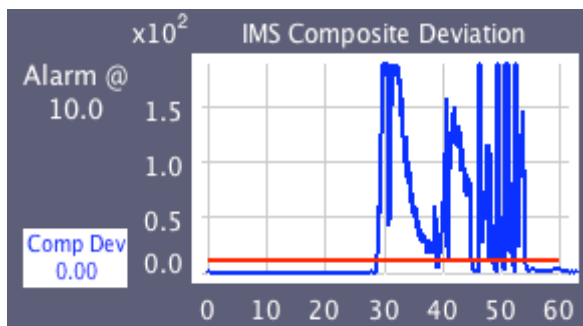

Figure 5. IMS composite deviation score for STS-119 off-nominal data set trained on seven parameters. One hour of data is shown.
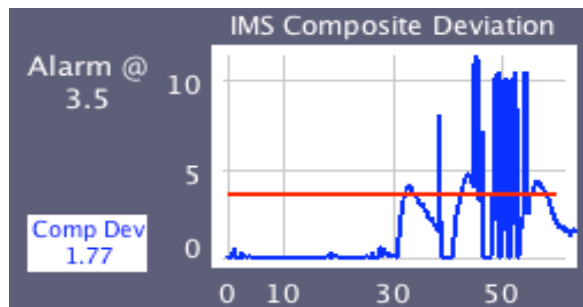

Figure 6. IMS composite deviation score for STS-119 off-nominal data set trained on four parameters. One hour of data is shown (same period as in Fig. 5).

As previously mentioned, STS-127 had a similar leak during tanking. We tested the same KB produced with STS-119 nominal data on STS-127 off-nominal data. The results resembled the STS-119 results shown here.

VII. Technology Development

In addition to demonstrating the current capabilities of IVHM tools, we are maturing the tools for ground operations applications. For anomaly detection, we are extending IMS to process data vectors even when some of the data parameters are not available. This allows FDIR to continue to monitor a system even with data dropouts. Also, the user may choose to bypass a sensor either to verify that other correlated sensors are providing similar information and thereby confirm that it is not likely just a sensor failure, or to remove the sensor from processing because it is known to be bad or even removed from the vehicle or ground system.

We are also performing experiments to determine the performance benefits of different approaches of KB development to provide guidelines and enable the customer to make informed decisions. Some of these experiments include understanding the effects on monitoring when some parameters are unavailable and the effects of training on subsets of parameters versus training on the full set and then marking some of them as unavailable. We are also developing methods to assist a KB developer in selecting a good set of parameters, i.e., selecting a group of parameters that are highly correlated and provide an effective monitoring capability. To this end, we will be comparing parameter selection based on computed correlation scores and parameter selection based on information available in the TEAMS model being built for fault isolation against parameters that are selected by a domain expert.

VIII. Conclusion

Although launch operations are continuously refined to accommodate lessons learned from launch experiences, the last time ground operations went through a major overhaul was for the Space Shuttle program in the late 1970s. With the imminent retirement of the Shuttle fleet, we have an opportunity for another overhaul to make the operation of next-generation launch vehicles more efficient. Determining the health state of complex space transportation and ground support systems using traditional parameter limit checking is becoming more difficult as the number of sensors and component interactions grows. Data-driven monitoring techniques such as IMS 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. FDIR's use of IMS to extend the anomaly detection capability is a big step toward improving monitoring of space systems prior to launch by providing alerts to system malfunctions or precursors of significant failures.

Appendix A
Acronym List

ADCO   Attitude Determination and Control Officer (ISS)
ARC   Ames Research Center (NASA)
CMG   Control Moment Gyroscope
ET   External Tank (Space Shuttle)
FDIR   Fault Detection, Isolation, and Recovery
FR   Firing Room (in LCC)
GSE   Ground Support Equipment
GUCP   Ground Umbilical Carrier Plate (interface between ET and hydrogen vent line)
H2   Hydrogen
IMS   Inductive Monitoring System (used for anomaly detection by FDIR)
ISLE   Institute for the Study of Learning and Expertise (www.isle.org)
ISS   International Space Station
IVHM   Integrated Vehicle Health Management
JPL   Jet Propulsion Lab (NASA)
KB   Knowledge Base
KSC   Kennedy Space Center (NASA)
LCC   Launch Control Center (at KSC)
LCS   Launch Control System
LH2   Liquid Hydrogen
SRB   Solid Rocket Booster
TEAMS   Testability Engineering and Maintenance System (used for fault isolation by FDIR)
THOR   Thermal Operations and Resources Officer (ISS)
TVC   Thrust Vector Control
VAB   Vehicle Assembly Building (at KSC)

Acknowledgments

References

[1]Ferrell, B., Lewis, M., Oostdyk, R., Goerz, J., Perotti, J., and Brown, B., "Fault Detection Isolation & Recovery (FDIR) Lessons Learned," *Proceedings of the AIAA Infotech@Aerospace Conference*, AIAA, Atlanta, GA, 2010. Submitted for publication.

[2]Oostdyk, R., Ferrell, B., Lewis, M., and Perotti, J., "Functional Fault Modeling of a Cryogenic System for Real-Time Fault Detection and Isolation," *Proceedings of the AIAA Infotech@Aerospace Conference*, AIAA, Atlanta, GA, 2010. Submitted for publication.

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

[4]Bay, S.D. and Schwabacher, M., "Mining Distance-Based Outliers in Near Linear Time with Randomization and a Simmple Pruning Rule," *International Conference on Knowledge Discovery and Data Mining*, 2003.

[5]Bishop, C. M., *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.

[6]Park, H., Mackey, R., James, M., Zak, M., Kynard, M., Sebghati, J., et al., "Analysis of Space Shuttle Main Engine Data Using Beacon-based Exception Analysis for Multi-Missions," *Aerospace Conference Proceedings*, Vol. 5, IEEE, Washington, DC, 2002, pp. 6-2835-6-2844.

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

[8]Iverson, D.L., Martin, R., Schwabacher, M., Spirkovska, L., Taylor, W., Mackey, R., and Castle, J.P., "General Purpose Data-Driven System Monitoring for Space Operations," *Proceedings of the AIAA Infotech@Aerospace Conference*, AIAA, Seattle, WA, 2009.

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

[10]Schwabacher, M., Martin, R., Waterman, R., Oostdyk, R., Ossenfort, J., and Matthews, B., "Ares I-X Ground Diagnostic Prototype," *Proceedings of the AIAA Infotech@Aerospace Conference*, AIAA, Atlanta, GA, 2010. Submitted for publication.