

Unsupervised Anomaly Detection and Diagnosis for Liquid Rocket Engine Propulsion

Rodney A. Martin
NASA Ames Research Center
Mail Stop 269-1
Moffett Field, CA 94035-1000, USA
(650) 604-1334
rmartin@email.arc.nasa.gov

Abstract— The results of a comprehensive array of unsupervised anomaly detection algorithms applied to Space Shuttle Main Engine (SSME) data are presented. Most of the algorithms are based upon variants of the well-known unconditional Gaussian mixture model (GMM). One goal of the paper is to demonstrate the maximum utility of these algorithms by the exhaustive development of a very simple GMM. Selected variants will provide us with the added benefit of diagnostic capability.

Another algorithm that shares a common technique for detection with the GMM is presented, but instead uses a different modeling paradigm. The model provides a more rich description of the dynamics of the data, however the data requirements are quite modest. We will show that this very simple and straightforward method finds an event that characterizes a departure from nominal operation. We show that further diagnostic investigation with the GMM-based method can be used as a means to gain insight into operational idiosyncrasies for this nominally categorized test. Therefore, by using both modeling paradigms we can corroborate planned operational commands or provide warnings for unexpected operational commands.

TABLE OF CONTENTS

1	INTRODUCTION	1
2	METHODOLOGY	4
3	RESULTS	8
4	CONCLUSION	13
5	ACKNOWLEDGEMENTS	14

1. INTRODUCTION

As part of either manned or unmanned spaceflight, the propulsion systems that support missions to lower earth orbit (LEO) and/or geosynchronous earth orbit (GEO) have varying measures for success or failure. A common evaluation criterion considers avoidance of loss of mission or loss of crew. These costs should be identified and quantified as early on as possible in the design process for development of robust failure detection systems. Such systems can be used to

alert the crew or ground operators of potential signatures and precursors to catastrophic failure on missions ranging from ground-based test fires to manned spaceflight. Characterization of these costs, as well as the sensitivity of the alarm system is critical to avoid spurious alerts or missing potential precursors altogether. In some cases the algorithms used to implement these advanced caution and warning systems can be used in real-time, or for forensic post-hoc analysis.

Pragmatically, it is common for such algorithms to consider costs that are assigned to very specific critical events. The question of how to optimize these costs by way of robust anomaly detection, diagnosis, and prognosis has great significance for integrated systems health management. When indications of system abnormalities are presented by fault detection algorithms, we want to be sure about the severity of the anomalies as well as their implications. As such, we thematically discuss statistical performance analysis and metrics for failure detection systems, although the results we present are preliminary in nature.

Well-known performance metrics can be derived from the confusion matrix and Receiver Operating Characteristic (ROC) analysis as detailed in [7]. They are clearly excellent candidates to aid in determining alarm system sensitivity, as well as follow-on assignment of costs as discussed above. However, the application of this method and resulting interpretations must be measured carefully. Barring the availability of consistent data with faults or failures that are comparable both in severity and functional impact, the overall results of such an analysis might falsely indicate poor performance. However, we still introduce the results by using the confusion matrix-based and related ROC metrics, forgoing the assignment of costs at this stage. The true value of the results can more clearly be demonstrated in the diagnostic utility and ability to detect faults by using data outside of their functional categorization with a visual representation of the anomalies detected.

Space Shuttle Main Engine (SSME) data has been investigated in several previous studies, using various methods from artificial intelligence such as the nearest neighbor-based and decision tree approaches. Two unsupervised anomaly detection algorithms implemented by Orca and GritBot demon-

U.S. Government work not protected by U.S. copyright.
IEEEAC paper # 1658

strate these methods, respectively. Detailed results of the application of these algorithms have been previously published by Schwabacher [24]. Furthermore, Tumer and Agogino [1] have presented an information theoretic entropy-based algorithm for anomaly detection in SSME data, and Fiorucci et al. [10] present a vibration-based analysis for a documented high pressure turbopump blade failure. The former are just a small sample of analyses performed on SSME data, and is by no means meant to serve as a comprehensive list. However, they serve as excellent benchmarks and address a technically challenging and very important applied research objective.

We use these research studies as motivation to pursue alternate means of the application of anomaly detection algorithms to SSME data. In doing so, we can provide further corroboration of existing failures as additional measures of detection and diagnostic ability. This adds to the growing repository of SSME-based studies, allowing for a more exhaustive investigation of previous failures and building a comprehensive array of tools. Therefore, we aim to provide an additional method from which to select in order to meet modest goals within integrated systems health management. These include the ability to corroborate potential failures, as well as to detect and diagnose previously unknown benign sensor failures, or even precursors to potential systemic failures.

In this paper we provide alternative techniques for anomaly detection in SSME data using two basic modeling paradigms. The algorithms to be presented are based upon standard parametric methods that can easily be discussed within the framework of Bayesian methods and probabilistic graphical modeling. The two modeling methods to be investigated are the unsupervised Gaussian mixture model, and a standard linear dynamic system. The latter of the two methods makes basic allusions to the algorithmic blending of control theory and machine learning. However, to the author’s knowledge, this is a novel comparison and corroboration of these basic techniques using SSME data.

Both modeling paradigms will use a very straightforward method for anomaly detection, based upon previous research by Pontoppidan and Larsen [22], where a maximum allowable threshold for the probability of false alarm can be selected. This is an objective common to many algorithms, such as the one presented by Bickford [3]. The overall goal of the paper is to provide a comprehensive assessment and comparison of the methods introduced, highlighting the diagnostic strengths and potential for extension to more theoretically rigorous methods that will enhance predictive capability.

Methods Investigated

The use of unsupervised Gaussian Mixture Models (GMM’s) represents the driver for most of the techniques to be investigated. In the probabilistic graphical model framework shown in Fig. 1, a sequence of outputs serve as N independent observations to train models for unsupervised, or unconditional GMM’s. The mixture of experts paradigm models both an

independent sequence of N outputs and N inputs. Shaded nodes represent observed data, and unshaded nodes represent hidden nodes which require inference. The unconditional unsupervised GMM on the left of Fig. 1, as well as the parameters listed as $\theta = (\alpha_1, \dots, \alpha_M, \mu_1, \dots, \mu_M, \Sigma_1, \dots, \Sigma_M)$ represent the model that will be used throughout this paper.

We take the time to point this out to make the distinction between using the terminology “unsupervised” as described above and the definition of “unsupervised learning,” where no labeled examples of failures and nominal trials are available to train a model. In our case, we do have labeled examples of failures and nominal trials, however, the labels are used only for validation, and not in the training of the model. As such, the methods we present fall not only within the “unsupervised” method from the input-output standpoint, but also for the purposes of machine learning and model development.

Because of the limited availability of examples of failures, we use nominal training data only, as in [2], [12], [24] without the use of techniques such as random stratification of the data, which is traditionally used to prevent any bias when comparing training results to validation results. Another modeling paradigm is also investigated, that can also be expressed within the probabilistic graphical model framework, as shown in Fig. 2. Here we can see that the model to be learned is a dynamic system, and observed data is also represented by shaded nodes. The hidden nodes still represent unobserved random variables which need to be inferred, however in this case they are continuous (Gaussian) random variables, not discrete (multinomial) random variables as in the case of the GMM.

Additionally, the hidden random variables are chained together in a Markovian fashion, which provides for a measure of dynamic memory, or temporal dependence such that the time slices are serially correlated. This linear dynamic system modeling representation can also be augmented to allow for input-output relationships to be enforced, as in the right hand side of Fig. 1. However, in this paper, we will consistently investigate unsupervised models. The motivation for using this paradigm as an alternate modeling construct and more detail on its mechanics will be presented in the subsequent section.

Data-Driven Modeling

The data to be used with the models presented previously can actually be categorized functionally, due to the nature of the systems and available sensors onboard the Space Shuttle Main Engine (SSME). Controller data contains important information relevant to the major systems of the engine, such as pressures, temperatures, control system parameters, valve positions, etc. Vibration data contains measurements recorded mainly from accelerometers that represent an array of sensors placed to monitor the structural integrity of the engine. All of the data obtained will allow for more effective health management systems to be developed. In this paper, both controller and vibration data will be studied independently, as well as

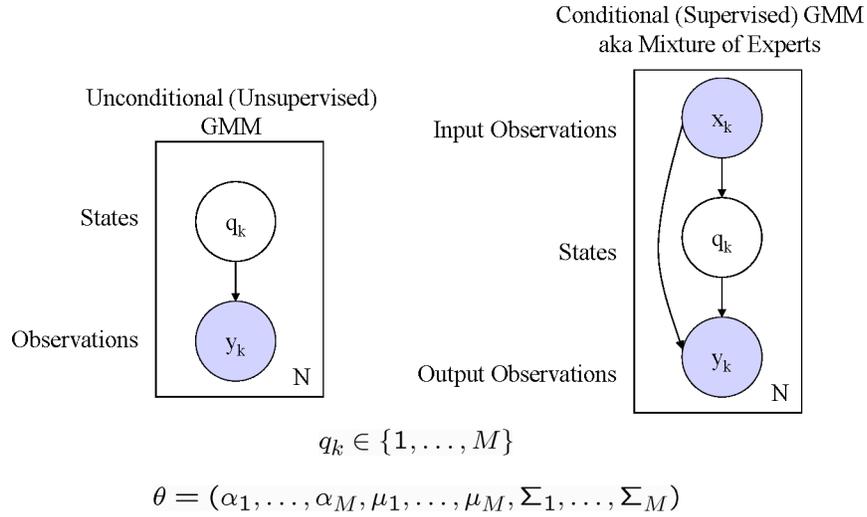


Figure 1. Gaussian Mixture Models

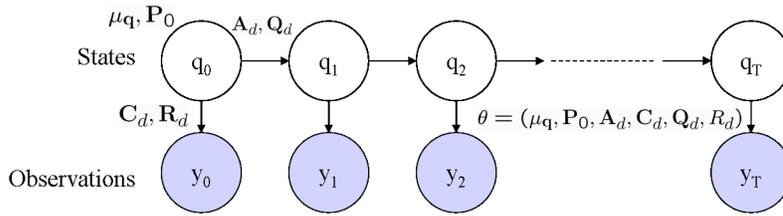


Figure 2. Linear Dynamic System

the combination of both controller and vibration data into a feature vector. There is also a third category of data, often stored separately in facility data files, that contain measurements from sensors placed on test stands only. This category will not be investigated in this paper.

Table 1 shows the data sources, and categorical determination of which tests are nominal or potentially anomalous. The breakdown of which tests are used for training the models and which are used for validation is presented. Controller data is available for all tests shown in Table 1, but vibration data is not necessarily available for all tests. As such, the training/validation breakdown serves the purpose of investigating controller data only.

In the column labeled “Data Sources,” either test stand data is used, or data from actual shuttle flights. For the latter, the “#” refers to the engine number for that particular flight, as the space shuttle is configured with 3 SSME’s. The test stand data naming convention can be parsed by reference to the actual physical test stand used (prefixed by A1 or A2), and concatenation of the test number. The first digit of the test number can loosely be used to determine if a different configuration is used. There are different configurations (Block I, Block IIA, Phase II) that span the flights shown in Table 1 as well. As such, we are using a hybrid mix of not only different data sources, but different SSME configurations and the flight/non-flight status as well.

This should contribute nicely towards the heterogeneity of the model, with more operating conditions available for training, and helping to prevent biased results, since no new configurations are part of the validation data set. However, this also has the disadvantage of fundamentally skewing the statistics of the model from the case in which the training and validation data sets are based only on a single configuration, as is often performed. In this case, the model would retain the homogeneity of the chosen configuration, and the results would be expected to exhibit a sharper discrimination due to more consistent statistics. However, there are significantly fewer within-configuration tests available to perform a ROC analysis that is statistically significant.

The tests that are labeled as potential anomalies in Table 1 are in most cases functionally correlated to a specific data type, i.e. they appear only in sensors of a particular functional category. This may also bias the results when using ROC analysis or confusion matrices. Given all of the biases and impacts on generating results whose metrics are based upon ROC analysis and confusion matrices, we will also demonstrate the diagnostic utility and ability to detect faults through visual interpretation as supporting evidence of performance.

When a subset of common sensors are selected for training and validation, the dimension of the feature vector is often reduced substantially. However, depending on the functional data category, there can still be as many as 130 parameters

Table 1. Training/Validation Breakdown for Controller Data

Data Sources	Training	Validation	
	Nominal	Nominal	Potential Anomalies
Flight Data	STS-77 (#1)	STS-103 (#2)	STS-77 (#2)
	STS-78 (#1)	STS-103 (#3)	STS-91 (#1)
	STS-78 (#2)	STS-106 (#1)	STS-93 (#1)
	STS-78 (#3)	STS-106 (#2)	STS-93 (#3)
Test Stand Data - Source #1	A10851	A10852	A10853
	A20726	A20750	A20619
	A20858		
Test Stand Data - Source #2	A20631	A20643	A10853
		A20823	A20619

to train on. The computational complexity of the algorithms will increase considerably as a result. As such, in order to decrease the computational burden, we can reduce the dimensionality of the data using various methods. There are many candidates, including Sammon nonlinear mapping [13]. However, the two that will be investigated in this paper are Principal Components Analysis (PCA), and taking the sum of z-scores over all sensors. These two methods are much less computationally intensive than using Sammon nonlinear mapping in which validation requires a redefinition of the map, although alternate methods exist to address this issue, such as the use of neural networks [4].

As a comprehensive exercise, the results of unreduced data will also be generated for comparison. In the case of GMM’s this means that the Gaussian distributions will be multivariate, with the cases of spherical, diagonal, and full covariance matrices investigated. In addition, an individual GMM can be trained for each sensor, which adds the dimension of diagnostic capability in addition to detection. However, this diagnostic capability is only available when the scoring metric used for classification is applied on an individual basis, and not on an aggregate basis. Finally, there will be no data reduction necessary for the models that use linear dynamic systems. This is due to the computational burden being reduced significantly as a result of the data requirements involving only the difference between two sensor values rather than all sensor values.

2. METHODOLOGY

In this section, we will describe the details of the two models investigated and used in conjunction with a common anomaly detection algorithm. The anomaly detection algorithm is implemented as an alarm system, which will also be described in detail.

Gaussian Mixture Models

The Gaussian mixture model has the advantage of standard parametric flexibility that is easily understood. Without loss of generality, Eqns. 1-2 represent the distribution of the feature vector, $\mathbf{y}_k \in \mathbb{R}^n$, at a single point in time, and repre-

senting the joint likelihood of the entire time series, up to N , respectively.

$$p(\mathbf{y}_k) = \sum_{j=1}^M \alpha_j \mathcal{N}(\mathbf{y}_k; \mu_j, \Sigma_j) \quad (1)$$

$$p(\mathbf{y}_0, \dots, \mathbf{y}_N) = \prod_{k=0}^N \sum_{j=1}^M \alpha_j \mathcal{N}(\mathbf{y}_k; \mu_j, \Sigma_j) \quad (2)$$

The parameters, as shown in Fig. 1, are trained using the standard EM (Expectation- Maximization) algorithm originally introduced by Dempster et. al. [5], and implemented with code freely available by Ian Nabney, detailed in [21]. Parametric initialization for the M mixture weights (α_j), means (μ_j), and covariance matrices (Σ_j) was performed using various implementations of the k-means algorithm, one of which was from [21]. Another initialization method can be used when $n = 1$, using a single feature or data reduction. The initialization is performed by binning the data as a univariate histogram. The dominant clusters are identified visually, and the statistics of the data contained in the bins corresponding to the dominant clusters are treated as the initial means and variances. The mixing proportions in this case are assigned equal probabilities.

Now this previously described method can only be used in the case when PCA or sum of z-scoring is used to dimensionally reduce the data, or $n = 1$ in any other case. Furthermore, in the case where an individual GMM is trained for each sensor, it is simplest to choose a default number of clusters based upon the dominant sensor rather than manually select the dominant bins for each sensor. For the SSME, the dominant sensor for controller data is the throttle, or power level. From this we can identify the maximum number of power levels visited throughout most of the excursions of the training data sets. This value can be used as a guide in setting the maximum number of clusters to be used for all sensors for which a GMM is trained individually.

In the case where there is non-convergence of the EM algo-

rithm or convergence to infeasible answers, this may indicate overfitting, and is a case in which choosing the maximum number of clusters based on the dominant sensor fails. In this case, we simply implement an automated procedure to decrease the number of clusters incrementally until the EM algorithm converges.

Selecting the optimal number of clusters is even more of a challenge when multi-dimensional Gaussian mixture components are used. In this case, with $\mathbf{y}_k \in \mathbb{R}^n$, and when $n \gg 1$, clustering can no longer be performed visually. Instead, clustering can be performed by using a prediction strength indicator. This prediction strength indicator introduced in [26] uses a supervised learning concept to determine the maximum number of clusters that can be accurately predicted by using a clustering technique such as k-means. The one disadvantage in practice is that the implementing code can be computationally burdensome when attempting to generate an ensemble estimate of the “breakpoint,” or minimum number of clusters yielding reasonable prediction strength. In the case of the multi-dimensional Gaussian, only a single model needs to be trained. Therefore, using the computationally intensive prediction strength algorithm is useful for initialization only. It is not used with the exhaustive GMM per-sensor approach, where applying the prediction strength method would be computationally prohibitive.

Some of the fundamental challenges and disadvantages of using GMM’s are very well elucidated in [14]. One of the challenges addresses our assumption that the data can be well described by a mixtures of Gaussian distribution with a select number of clusters. However, we do not truly have any intuitive cause for believing that this is innately how the physical mechanisms generating the SSME sensor data are manifested. In such cases, nonparametric density estimators may be used as a viable alternatives, and are suitable candidates for future research. However, the robustness of using GMMs will be elaborated on in detail in a subsequent section, specifically in reference to the quantitative and qualitative performance metrics introduced thus far.

Linear Dynamic Systems

The second modeling paradigm to be implemented in detail has a more thorough intuitive basis than the GMM. Essentially, the motivation behind training a linear dynamic system is threefold.

1. The training data is univariate, i.e. $n = 1$, and represents the difference between the commanded throttle and actual throttle. In control systems terminology, this is the control system error, $e(t)$, traditionally used as the input to a controller, as shown in Fig. 3.
2. To provide for a more rich description of the dynamics of the data in which the data requirements are quite modest.
3. To lay the groundwork for the application and advanced development of more sophisticated anomaly detection techniques requiring the use of Linear Dynamic Systems and

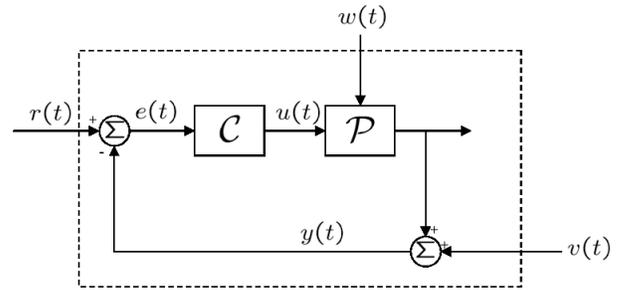


Figure 3. Closed-Loop Control System Block Diagram

Kalman Filtering in future research. These methods will provide an additional level of predictive capability.

There are two primary control systems that operate in support of the SSME. One is the throttle control system, which regulates the main combustion chamber pressure. This throttle, or power level, can also be determined by this pressure via a scaling factor. The other major control system that functionally supports the SSME is the mixture ratio control system. This system maintains the oxidizer/fuel mixture ratio at a desired level. We will only focus on the throttle control system error in this paper, due to the commanded throttle qualitatively being the apparent driver for so many other sensor readings.

In Fig. 3, the closed-loop control system representation illustrates that the actual throttle level, $y(t)$, is subtracted from the desired or commanded throttle level, $r(t)$, to obtain the control system error, $e(t) = r(t) - y(t)$. The block labeled C represents the controller, which we can nominally assume to be a very simple PI (proportional-integral) controller. The PI controller takes the control system error and computes the appropriate actuation to deliver to the plant, labeled as block P . The plant may be subject to input noise, $w(t)$, which is introduced directly into the state dynamics. Finally, as the feedback loop is closed, measurement noise, $v(t)$, may be additively introduced to the output of the plant to form $y(t)$, which represents the actual throttle level used by the control system.

The state dynamics of the open-loop plant, P , can be expressed by equations 3-4.

$$\dot{\mathbf{x}}(t) = \mathbf{A}_o \mathbf{x}(t) + \mathbf{B}_o \mathbf{u}(t) + \mathbf{B}_{wo} w(t) \quad (3)$$

$$y(t) = \mathbf{C}_o \mathbf{x}(t) + v(t) \quad (4)$$

where

$$w(t) \sim \mathcal{N}(0, Q_o)$$

$$v(t) \sim \mathcal{N}(0, R_o)$$

The state of the system is represented by $\mathbf{x}(t) \in \mathbb{R}^n$. In this case we choose $n = 2$, in order to strip the dynamics down to

the most minimal case in which the plant is a first order system, and the controller is a PI controller. All matrices in the equation above are also subscripted with “o” in order to disambiguate between the open-loop dynamics and the closed-loop state dynamics yet to be presented. Clearly this is an extravagant simplification, however in doing so we allow for the closed-loop control system dynamics to be represented in controllable canonical form. Furthermore, we are at the very minimum allowing for the introduction of *some* dynamics, as distinct from the GMM where no dynamics are modeled. Due to the unavailability of the actual plant dynamics and control system design parameters, we default to the simplest case in order to generate a workable model.

There are several transfer functions that can be formed from the closed-loop state dynamics. The one that we are most interested in from the machine learning standpoint is $TF_{w \rightarrow e}$, or the closed loop dynamics that represent the transfer function from input noise to error. Because the data available to us for training is the control system error, $e(t) = r(t) - y(t)$, we can reformulate the dynamics of the closed-loop feedback control system into a standard representation that can be treated as an unsupervised problem in machine learning (i.e., using output observations only). This is performed by loosely approximating the measured control system error using the transfer function $TF_{w \rightarrow e}$.

More formally, we know that the part of the block diagram shown within the dotted line in Fig. 3 represents the closed-loop dynamics. As such, we can express the control system error as a function of the variables that represent exogenous inputs to the closed-loop within the dotted line, as shown in Eqns. 5-6.

$$\begin{aligned} e(t) &= \| TF_{r \rightarrow e} \| r(t) + \| TF_{w \rightarrow e} \| w(t) \quad (5) \\ &+ \| TF_{v \rightarrow r} \| v(t) \\ &= e_r(t) + e_w(t) + e_v(t) \quad (6) \end{aligned}$$

Therefore, in a sense, everything within the dotted line can be reformulated to represent the closed loop dynamics, where the desired output is $e(t)$. Ultimately, we would like to be able to express these dynamics as shown in Eqns. 7-8 below.

$$\begin{aligned} \dot{\mathbf{q}}(t) &= \mathbf{A}\mathbf{q}(t) + \mathbf{B}_w w(t) \quad (7) \\ e_w(t) &= \mathbf{C}\mathbf{q}(t) + v(t) \quad (8) \end{aligned}$$

again, where

$$\begin{aligned} w(t) &\sim \mathcal{N}(0, Q) \\ v(t) &\sim \mathcal{N}(0, R) \end{aligned}$$

These equations fit the modeling paradigm represented by our machine learning problem represented in Fig. 2 perfectly

(after discretization). But we still need to determine how to enforce this paradigm, given our current objective which involves a slightly skewed output definition, by considering $e(t)$ in lieu of $y(t)$. We are interested in $e(t)$ as a whole, however only the first of the following exogenous inputs are readily available for measurement: $r(t)$, $w(t)$, and $v(t)$.

Therefore, the most accurate modeling representation of $e(t)$ requires supervision, where the input is the known reference signal $r(t)$. The remaining input and measurement noise inputs are never measured, and would need to be introduced into the standard formulation shown in Eqns. 7-8 if possible. Otherwise, constraints would be required to be imposed during the learning process to allow for $w(t)$ and $v(t)$ to remain uncorrelated. This would be necessary so that they can be introduced as a single augmented exogenous noise input via concatenation.

However, because the problem is being treated as unsupervised, we will need to make some approximations by removing both the closed-loop commanded input term $e_r(t)$, and the closed-loop measurement noise term, $e_v(t)$ from consideration. By ignoring these terms we lose some accuracy in approximating $e(t)$. Therefore, the final approximation to allow for a loose interpretation of the closed-loop transfer function of $TF_{w \rightarrow e}$ is $e(t) \approx e_w(t) = \| TF_{w \rightarrow e} \| w(t)$. Eqns. 7-8 clearly represent the state-space realization of the approximation for unsupervised learning. Note that after discretization of these equations, we can use the approximation $e_w(t)$ in place of $y(t)$ shown in Fig. 2.

The controllable canonical form shown in Eqns. 9-11 is used to allow for a mapping to intuitive canonical parameters: the natural frequency, ω_n , and the damping ratio, ζ . We can estimate the natural frequency by making an assumption of $e(t)$ to be represented by a zero-mean stationary Gaussian random process. In this case, we can use Rice’s formula for the level-upcrossing rate [15], [23], as shown in Eqn. 12, to compute the natural frequency, $\omega_n = \frac{\sigma_{\dot{e}}}{\sigma_e}$. This formula can be derived very easily [17], and is used in similar studies [8], [9], [18].

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -\omega_n^2 & -2\zeta\omega_n \end{bmatrix} \quad (9)$$

$$\mathbf{B}_w = \begin{bmatrix} 0 \\ \omega_n^2 \end{bmatrix} \quad (10)$$

$$\mathbf{C} = [1 \ 0] \quad (11)$$

$$\nu_e^+ = \frac{\sigma_{\dot{e}}}{2\pi\sigma_e} e^{-\frac{1}{2}\left(\frac{L-\mu_e}{\sigma_e}\right)^2} \quad (12)$$

By using $L = 0$ as a candidate level, all we need is to count the number of zero-upcrossings of the sample data, and compute the 2^{nd} -order statistics: μ_e , and σ_e in order to use Rice’s formula to find ω_n . In case $\mu_e = L = 0$, we simply need ν_e^+ , as $\omega_n = \frac{\sigma_{\dot{e}}}{\sigma_e} = 2\pi\nu_e^+$.

During the learning procedure for the linear dynamic system, the EM algorithm is used to find the parameters shown in Fig. 2. Details of this procedure are provided in Zoubin and Hinton [11] as well as Digalakis et. al. [6], and it is implemented using Murphy's BNT (Bayes' Net Toolbox) [20]. Initialization of the parameters shown as θ in Fig. 2 is also performed using some basic heuristics. By initializing $\zeta = 1$ and clamping ω_n during training, we can back out the learned value of the damping ratio ζ . Initial values for \mathbf{A} and \mathbf{B}_w can be derived as a function of ζ and ω_n , $\mathbf{C} = \begin{bmatrix} 1 & 0 \end{bmatrix}$ is fixed during learning, and R is initialized by making a guess at the SNR (signal to noise ratio), so that $R = \frac{\sigma_{e_w}^2}{SNR}$ ($\sigma_e^2 \approx \sigma_{e_w}^2$ can be computed directly from the data).

Using these assumptions, and by use of steady-state continuous-time Lyapunov equations for Eqns.7 and 8 (cf. \mathbf{P}_0 from Fig. 2), we can find an adequate initialization for \mathbf{Q} , as is performed in [17], [18]. We then discretize all parameters using the sampling interval T_s , and the procedure outlined in [18], allowing us to form Eqns. 13 - 14, which support the variables shown in Fig. 2, again using the approximation e_{w_k} in place of y_k .

$$\mathbf{q}_{k+1} = \mathbf{A}_d \mathbf{q}_k + \mathbf{w}_k \quad (13)$$

$$e_{w_k} = \mathbf{C}_d \mathbf{q}_k + v_k \quad (14)$$

where

$$\begin{aligned} \mathbf{w}_k &\sim \mathcal{N}(0, \mathbf{Q}_d) \\ v_k &\sim \mathcal{N}(0, R_d) \\ \mathbf{A}_d &= e^{\mathbf{A}T_s} \\ \mathbf{B}_d &= (e^{\mathbf{A}T_s} - \mathbf{I})\mathbf{A}^{-1}\mathbf{B} \\ \mathbf{C}_d &= \mathbf{C} \\ R_d &= \frac{R}{T_s} \\ \mathbf{Q}_d &= \int_0^{T_s} e^{\mathbf{A}\lambda} \mathbf{B}_w \mathbf{Q} \mathbf{B}_w^T e^{\mathbf{A}^T \lambda} d\lambda \end{aligned}$$

Throughout learning, we attempt to retain the controllable canonical structure in order to allow for determination of the learned value for ζ . This is easily performed by the allowance for enforcement of arbitrary constraints in Murphy's BNT [20], and slight modification of the appropriate open-source routines. Doing so introduces sub-optimality into the learning procedure, which means that the learning curve will not necessarily increase monotonically. However, a reasonable sub-optimal local minimum will be found that best represents the parameter space with enforcement of the controllable canonical form constraint.

Alarm Systems and Anomaly Detection

We now present the methods and motivations for designing the alarm systems, largely based upon a novelty detection

method introduced in Pontoppidan and Larsen [22] as well as Larsen et. al [16]. Here, the conditional likelihood or log-likelihood functions are evaluated with streaming observations against a threshold in order to meet the criterion set for the maximum allowable probability of false alarm. The design of this decision rule is facilitated by the formation of an empirical cumulative distribution of the likelihood or log-likelihood values, and setting an acceptance/rejection criterion for the null hypothesis of anomalous behavior. We can use any significance level (i.e. $p = 0.05$) as the maximum allowable probability of false alarm. Eqns. 16-17 are expressed without loss of generality, and the design criterion for the decision rule is expressed by the inequality shown in Eqn. 17. The likelihood or log-likelihood function values for the selected threshold, l , serves as a scoring metric that can be used for anomaly detection, or synonymously as the basis of the decision rule.

$$J(l) \triangleq P(\log(p(\mathbf{y}_k|\theta)) < l) \quad (15)$$

-or-

$$J(l) \triangleq P(p(\mathbf{y}_k|\theta) < l) \quad (16)$$

$$J(l) < p_{max} \quad (17)$$

Here we see that $J(l)$ represents the empirical cumulative distribution function of either the likelihood, or log-likelihood values. Eqns. 15-16 are presented in a generic enough fashion to allow for any representation of $p(\mathbf{y}_k|\theta)$, so that implicitly either modeling paradigm can be used. This anomaly detection-based alarm system can also easily be implemented in real-time. Eqn. 18 is the representation of $p(y_k|\theta)$ in the case of GMM's that are trained on reduced data, or individual sensors.

$$p(y_k|\theta) = \sum_{i=1}^M \alpha_i \mathcal{N}(y_k; \mu_i, \sigma_i^2) \quad (18)$$

where

$$\theta = (\alpha_1, \dots, \alpha_M, \mu_1, \dots, \mu_M, \sigma_1^2, \dots, \sigma_M^2)$$

As alluded to in previous section, when independent GMM's are trained on individual sensors, the anomaly detection algorithm is augmented with diagnostic capability. In fact, there are as many alarm systems as there are sensors, but an alarm in one sensor may or may not be indicative of an alarm for all sensors, which is how the diagnostic capability is demonstrated. Although this is an added benefit, any interactions between the sensors cannot be modeled using this exhaustive modeling method.

Alternatively, we can form an aggregate alarm system without having to reduce the data using methods described earlier.

This can be performed either for the case in which multiple GMMs are trained individually per sensor, or for the case of a single GMM for all sensors. In the latter case, the sensors can be modeled either as independent or as correlated observations, so that interactions between the sensors are modeled. Eqn. 19 represents $p(y_k^1, \dots, y_k^n | \theta)$ ¹ in the former case, and Eqn. 20 represents $p(\mathbf{y}_k | \theta)$ in the latter.

$$p(y_k^1, \dots, y_k^n | \theta) = \prod_{j=1}^n \sum_{i=1}^{M_j} \alpha_{ij} \mathcal{N}(y_k^j; \mu_{ij}, \sigma_{ij}^2) \quad (19)$$

where

$$\theta = (\alpha_{11}, \dots, \alpha_{M_n M_n}, \mu_{11}, \dots, \mu_{M_n M_n}, \sigma_{11}^2, \dots, \sigma_{M_n M_n}^2)$$

$$\begin{aligned} p(\mathbf{y}_k | \theta) &= \sum_i^M \alpha_i \mathcal{N}(\mathbf{y}_k; \mu_i, \Sigma_i) \quad (20) \\ &\stackrel{diag}{=} \sum_i^M \alpha_i \prod_{j=1}^n \mathcal{N}(y_k^j; \mu_i^j, (\sigma_i^j)^2) \\ &\stackrel{sph}{=} \sum_i^M \alpha_i \prod_{j=1}^n \mathcal{N}(y_k^j; \mu_i^j, \sigma_i^2) \end{aligned}$$

where

$$\theta = (\alpha_1, \dots, \alpha_M, \mu_1, \dots, \mu_M, \Sigma_1, \dots, \Sigma_M)$$

Eqn. 20 can be redefined as shown when the sensor observations are modeled as being correlated. In this case, $p(\mathbf{y}_k | \theta)$ has slightly different definitions when the covariance matrix, Σ , is spherical or diagonal instead of full. Furthermore, there is a subtle distinction to make between Eqns. 1 and 19. Note that Eqn. 1 represents the joint likelihood of the entire time series, up to $k = N$, for a multidimensional feature vector, $\mathbf{y}_k \in \mathbb{R}^n$. This form is often used to compute the expected complete log-likelihood function in deriving the equations used in the EM algorithm. Eqn. 19 represents the joint likelihood of n uncorrelated GMM's for n univariate sensor observations of y_k^i , $i \in \{1, \dots, n\}$ at a single point in time, k . In this case, if using the log-likelihood version of $J(l)$, then Eqn. 21 applies.

$$\log(p(y_k^1, \dots, y_k^n | \theta)) = \sum_{j=1}^n \log \sum_{i=1}^{M_j} \alpha_{ij} \mathcal{N}(y_k^j; \mu_{ij}, \sigma_{ij}^2) \quad (21)$$

¹Notationally, indexed superscripts refer to elements of a vector for all equations in the paper.

In case of Eqn. 20, for a multidimensional feature vector, $\mathbf{y}_k \in \mathbb{R}^n$, the log-likelihood version of $J(l)$ can be represented as Eqn. 22-23, for both a diagonal and a spherical covariance matrix, respectively.

$$\log(p(\mathbf{y}_k | \theta)) \stackrel{diag}{=} \log \sum_{i=1}^M \alpha_i \prod_{j=1}^n \mathcal{N}(y_k^j; \mu_i^j, (\sigma_i^j)^2) \quad (22)$$

$$\log(p(\mathbf{y}_k | \theta)) \stackrel{sph}{=} \log \sum_{i=1}^M \alpha_i \prod_{j=1}^n \mathcal{N}(y_k^j; \mu_i^j, \sigma_i^2) \quad (23)$$

The representation of $p(\mathbf{y}_k | \theta)$ can also be expressed as a function of the parameters of the linear dynamic system, as shown in Eqn. 24

$$p(y_k | \theta) = \mathcal{N}(y_k; 0, \mathbf{C}_d \mathbf{P}_{ss} \mathbf{C}_d^T + R_d) \quad (24)$$

where²

$$\mathbf{P}_0 = \mathbf{P}_{ss} = \mathbf{A}_d \mathbf{P}_{ss} \mathbf{A}_d^T + \mathbf{Q}_d$$

Recall that y_k in Eqn. 24 is actually represented by e_{w_k} , and that the continuous time equivalent is $e(t) \approx e_w(t) = \|TF_{w \rightarrow e}\| w(t)$. Although this an approximation, of the three additive terms that form $e(t)$, the transfer function $\|TF_{w \rightarrow e}\|$ represents the quantity that we want to penalize the most from an anomaly detection perspective. Because this transfer function most closely characterizes the magnitude of input noise disturbances, we can assume that any outliers are truly significant.

The control system was most likely designed with both reference command following and disturbance rejection in mind. However, when large disturbances influence the plant, \mathcal{P} , to the extent that the control system cannot reject them expeditiously, this may be indicative of a significant event which is cause for diagnostic investigation. The reference command following transients and measurement noise are almost always representative of nominal plant behavior during throttle changes and standard sensor noise, respectively. Therefore, it is reasonable to assume that the terms reflecting these characteristics are negligible.

3. RESULTS

Testing Conditions

The sample data used for both training and validation shown in Table 1 were often recorded at different sampling rates. Because of memory limitations, the data was downsampled

²Steady-State (Algebraic) Discrete-Time Lyapunov Equation

using the largest sampling interval prior to loading into memory for algorithmic processing. The characterization of this preprocessing step as pure downsampling is not entirely accurate, due to the fact that the raw data was averaged as well. Some files could not be downsampled prior to loading into memory due to incompatible file formats. These files were decimated after being loaded into memory using the same sampling interval, and by using a low pass Chebyshev filter.

Each of the failures listed in Table 1 has a unique description and functional categorization. Often, the failure will present most clearly in a single sensor, depending on its severity. Table 2 lists the type of failure, approximate severity, and functional categorization.

The differences in failure types and severities have clear implications for the aggregation of the results using ROC analysis. However, we can still speak to the ability of each algorithm under consideration to maximize true positives based upon the severity of the failure. Table 3 lists all of the possible scenarios to be investigated. It should be noted that sensor data which is functionally categorized as vibration data, or “both” controller and vibration data use a slightly different training/validation breakdown than is shown in Table 1.

In Table 1, eight nominally classified training datasets are listed along with the same number of nominal validation datasets and anomalous validation sets. Of the latter, two datasets are redundantly listed because their origination differs. Of the eight nominally classified training datasets, tests A20858 and A20631 do not contain any vibration data or vibration sensors that are used in the other tests. The same is true for tests A20643 and A20823 of the eight nominally classified validation datasets, and tests A10853 and A20619 corresponding to data source #2 only, for the eight anomalous validation sets. Therefore, they are eliminated for experiments shown in Table 3 to have a functional categorization of vibration data, or “both” controller and vibration data.

Other important conditions for both training and validation data are the consideration of the operating conditions of the system. It may very well be that the available data contains statistical outliers that are completely nominal from an operational standpoint. The main example of this is during startup, shutdown, or any major throttling transient that the engine experiences. As such, we will train and validate models only during periods of nominally steady state operation. For the purposes of this paper, this is to include any time after the startup transient, before shutdown, and allowing for a 1 sec settling time after major throttling changes.

When training a linear dynamic system model with vibration in lieu of controller data, the modeling paradigm makes a subtle shift. The control system motivation presented in Sec. 2 no longer applies, and as such training of the model using the EM algorithm no longer requires the clamping and constraining of specific parameters. The only modeling consideration

Table 3. Cases for Investigation

Label	Model	Data Reduction	Functional Categorization
A	GMM	PCA	Controller
B	GMM	Z-Scoring	Controller
C	GMM	PCA	Vibration
D	GMM	Z-Scoring	Vibration
E	GMM	PCA	Both
F	GMM	Z-Scoring	Both
G	GMM	None (spherical)	Controller
H	GMM	None (diagonal)	Controller
I	GMM	None (full)	Controller
J	GMM	None (aggregate)	Controller
K	GMM	None (diagnostic)	Controller
L	GMM	None (spherical)	Vibration
M	GMM	None (diagonal)	Vibration
N	GMM	None (full)	Vibration
O	GMM	None (aggregate)	Vibration
P	GMM	None (diagnostic)	Vibration
Q	GMM	None (spherical)	Both
R	GMM	None (diagonal)	Both
S	GMM	None (full)	Both
T	GMM	None (aggregate)	Both
U	GMM	None (diagnostic)	Both
V	LDS ^a	None (CSE) ^b	Controller
W	LDS	Z-scoring	Vibration
X	LDS	Z-scoring	Both

^aLinear Dynamic System

^bControl System Error

is for the data reduction to be performed using the sum of z-score method. This is necessary in order to allow for enforcement of the zero-mean Gaussian random process assumption.

Although presented with a very straightforward design criterion in Sec. 2, $J(l) < p_{max}$, there is still flexibility in choosing a particular type of threshold. Recall that p_{max} is the significance level (i.e. $p = 0.05$), or the maximum allowable probability of false alarm. As presented previously, anomalies are classified by the corresponding decision rule $\log(p(\mathbf{y}_k|\theta)) < l$. However, recall that either $J(l) = P(\log(p(\mathbf{y}_k|\theta)) < l)$ or $J(l) = P(p(\mathbf{y}_k|\theta) < l)$ can be used as the basis of designing the decision rule. This is often useful when using the former log-likelihood version of $J(l)$. In this case there is more allowance for infinitesimal values of p_{max} to be equivalently selected, which otherwise would not be accessible using the latter likelihood version. As such, we will use this version for all results to be presented. The fact that straightforward evaluation of $p(\mathbf{y}_k|\theta)$ is the limiting computational bottleneck of this algorithm facilitates its ease of translation to real-time implementation.

Table 2. Characterization of Failures

Failure Data	Failure Type	Functional Categorization	Severity
STS-77 (#2)	Anomalous Spike in Sensor Reading	Controller	Mild
STS-91 (#1)	Sensor Failure	Controller	Mild
STS-93 (#1)	Controller Failure	Controller	Moderate
STS-93 (#3)	Fuel Leak and Controller Failure	Controller	Moderate to Severe
A20619	Knife Edge Seal Crack	Vibration	Moderate to Severe
A10853	Turbine Blade Failure	Vibration	Severe

ROC Analysis

The ROC analysis can be performed using either relaxed or strict conditions for forming the confusion matrix. The strict condition dictates that an alarm sound prior to the known time of the anomaly, where the relaxed condition allows for the alarm to sound at any time for a test that was classified as anomalous. For cases K, P, and U, where the diagnostic GMM per-sensor approach is used, we will define the strict condition as at least one sensor alarming prior to the known time of the anomaly. For the case of the relaxed condition, at least one sensor should alarm at any time during the test. The metrics to be used as a basis for assessing the performance will be the ROC curve statistics and the percentage accuracy as a function of the appropriate threshold (see [7] for precise definition of % accuracy as a function of confusion matrix elements). The quantitative summary for all cases meeting the following criteria is provided in Table 4.

1. The subject case yields a feasible model.
2. The maximum accuracy is better than 50%.
3. At least one point on the ROC curve lies above the “random guessing” line.
4. The case is subject to using the relaxed criterion.

This table shows the maximum accuracy, and corresponding log-likelihood value, as well as the best ROC curve statistics (true positive and false positive rates). The “best” ROC curve statistics do not necessarily correspond to the same log-likelihood threshold that yields the maximum accuracy. Rather, the resulting true and false positive rates were often identical over entire ranges of thresholds due to the lack of sufficient data. As such, very few unique points were found on the ROC curve, making it easy to select the “best” points (with more emphasis on minimizing false positives). This is a subjective matter due to the fact that some alarm system designers may balance true and false positives quite differently, speaking directly to the issue of the assignment of costs. Therefore, when there was not a clear ‘best’ point, more than one was provided in Table 4.

The experiments which used data reduction techniques such as PCA and sum of z-scoring did not yield any better or worse results than for the other cases shown in Table 4. The same can be said for the assorted variety of unreduced data experiments. Allowing for correlated features by lifting the restriction on the covariance matrix from spherical or diagonal to

Table 4. ROC Analysis

Label	Max Accuracy	Threshold ^a	Best ROC Stats ^b
A	62.5%	-15	(25%/0%)
B	62.5%	[-100,-12]	(25%/0%)
C	83.3%	[-4,-3]	(83.3%/16.6%)
D	62.5%	[-140,-12]∪-9	(33.3%/0%) (66.6%/33.3%)
E	58.3%	[-417,-80]∪ [-19,-11]	(16.6%/0%)
F	66.6%	[-100,-20]∪ [-11,-8]	(33.3%/0%)
G	68.8%	-417	(37.5%/0%)
H	68.8%	[-700,-400]	(37.5%/0%)
J	56.3%	-417	(50%/37.5%)
K	62.5%	[-700,-400]	(25%/0%)
L	66.6%	[-280,-140]	(33.3%/0%)
M	83.3%	-139	(66.6%/0%)
N	83.3%	-139	(66.6%/0%)
O	66.6%	[-278,-140]	(33.3%/0%)
P	66.6%	-140	(33.3%/0%)
U	75%	[-700,-400]	(50%/0%) (66.6%/16.6%)
V	75%	-15	(62.5%/12.5%)
W	83.3%	-15	(66.6%/0%)

^aCorresponding Log-Likelihood based threshold range

^bTrue/False Positive Rates

fully correlated would intuitively seem to provide some additional information. However, the parameter space is still limited only to the mixtures, not the sensors. Inadequate parametric flexibility may be the reason for the lack of a boost in performance in using multivariate Gaussian mixture models with a full feature space, even when using full covariance matrices. When using vibration data, however, the results are consistently better (Cases C, M, and N).

Full parametric flexibility can be allowed by training a GMM per sensor, and implementing an alarm system that operates in an aggregate sense (cf. Eqns. 19 and 21). When doing so, however, the diagnostic power of the algorithm is removed, and the aggregation is implicitly an attempt at post-analysis data reduction. Therefore, failures that present only in a par-

ticular sensor may be overlooked by this method of implementing an alarm system. As a result, GMM-based experiments in which each individual sensor has its own alarm system may perform better. However, without more examples per experiment, it is difficult to ascertain this with any level of confidence.

Comparing the LDS (Linear Dynamic System) based experiments V and W may be dubious as well, because one case uses a larger dataset, which also has no exposure to vibration data. Specifically, for Case V, which is trained on larger dataset of controller data only, the statistics are still favorable, even though Case W has a higher maximum accuracy. This is largely due to the fact that failures presenting in vibration data can quite advantageously be detected with models that are based on controller data only. At the same time, the model in Case V is still able to detect moderate to severe controller-based failures, yielding a 75% max accuracy, and a 62.5%/12.5% split for the true/false positive rates. Because of the larger dataset, and even more importantly the lack of sufficient data to claim statistical significance, these statistics are not necessarily any better or worse than that of Case W.

For Cases C, M, N, and W, only the subset of tests specified previously that contain vibration data or vibration sensors used in all other tests were used for validation. As such, the correctly classified anomalies are on average the more severe examples, as shown in Table 2, with the exception of the Knife Edge Seal Crack in Test A20619, which is rarely detected using low frequency data. Furthermore, the statistics are naturally higher due to the fact that vibration data was used to train these models, in which all vibration-based failures were detected in addition to the more severe controller-based failures. This naturally yields more favorable statistics, as there is a bias towards vibration failures for a reduced dataset.

On average, the anomalies incorrectly classified as nominal are the tests labeled in Table 2 to have a “mild” severity. False positives also present in the ROC analysis, sometimes due to statistical modeling noise, and other times having a very clear operational explanation. Perhaps if we used either a more accurate method of selecting the number of clusters per sensor in the model for the GMM method, i.e. the BIC (Bayes Information Criterion) as is used in [22], or used a finer threshold grid for the ROC analysis for all methods, the accuracy may have been increased.

Another drawback of the GMM method is that the final validation results are often hyper-representative of the raw training data used to build the models. When controller data is used for training, we know that almost all sensors are driven by the commanded throttle. The qualitative signature of the commanded throttle is present even when the data is reduced. As such, if commanded throttle levels had not been visited previously in training data sets, validation data sets containing unseen throttle excursions will be incorrectly classified as

anomalous. The same is true for other sensors that may or may not be directly correlated the commanded throttle, such as valve positions. Such sensors may have had planned excursions throughout the course of a trial run or flight, but experienced a different one during validation.

A common cause for this phenomenon is due a change in the configuration of the SSME. Although this bears out as a false positive with respect to the ROC analysis, in an operational sense, we can look at this as being useful from the standpoint of providing warnings for unexpected operational commands. Furthermore, this phenomenon does not exist for the LDS method, where the training data is transformed by using control system error. Here, the influence of throttle is only apparent during major transients, and not throughout the course of the trial run or flight. Because models were trained using data only during periods of nominally steady state operation, there can be no bias due to unseen throttle levels using this method.

As alluded to a number of times, the statistical significance of results shown in Table 4 is dubious for two reasons. One reason is the absence of consistent data with faults or failures that are comparable both in severity and functional impact. The second reason has to do with the lack of sufficient data to support a statistically significant analysis. The overall results of the analysis might therefore falsely indicate poor performance, or even superior performance. As this is a preliminary assessment, the interpretation of these results must be measured carefully barring the availability of more examples of failures to work with. However, one advantage that still bears out in the ROC analysis is that failures presenting in vibration data can be found with models based on controller data only, primarily by using the LDS-based algorithmic technique.

Diagnostic Findings

There is a distinction to be made between the cases which are used purely for detection, and those that enable diagnostic reporting capability. Cases K, P, and U use a diagnostic GMM per-sensor approach, in which independent GMM’s are trained on individual sensors. Again, for these cases there are as many alarm systems as there are sensors, however, an alarm in one sensor may or may not be indicative of an alarm for all sensors. This logic allows for determination of which sensor is causing the problem, and its approximate severity. The severity can be inferred by corroboration with an array of other sensors that may be indicative of a more systemic problem rather than a more benign single sensor failure. This corroborative functionality and qualitative determination of severity is how the diagnostic capability is demonstrated.

The examples which most clearly demonstrate the utility of the diagnostic benefit are for one of the failures shown in Table 2, and for another nominally categorized test. For test A10853, in which a high pressure turbopump blade failure was found in vibration data [10], the LDS model was used to detect this anomaly at the same time as was indicated in [10].

Validation Data with Log-Likelihood-based Alarm System

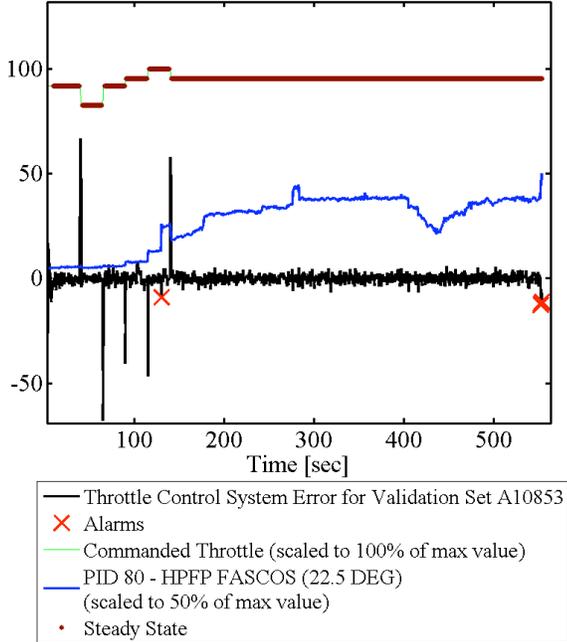


Figure 4. A10853 Control System Error

An illustration of this anomaly found in the control system error is provided in Fig. 4, along with the commanded throttle, steady-state periods, and offending vibration sensor (PID 80) superimposed.

Commanded throttle and steady-state periods are superimposed to allow for the demonstration of how large changes in the control system error are the result of nominal throttling transients. However, there should not be any large excursions in the control system error during steady-state periods. The first indication of an anomaly is at 130 sec, however, there are also some alarms at the very end of the trial run. These alarms are the result of a negative log-likelihood value (cf. log of Eqn. 24) exceeding a chosen threshold. ROC curves can be used to select this optimal design threshold for implementation. The procedure is to select an optimal design point from the portion of the ROC curve that corresponds to all training files being classified as nominal. This heuristic provides us with a guideline for ensuring that an optimal design point is not selected purely on the basis of the accuracy of the validation data, but of the training data as well.

We can use the GMM-based diagnostic models to gain further insight into the potential reasons for these anomalies found by deviations from control system error. As shown in Figs. 5 and 6, both alarm times are corroborated by other sensors. In the case of the first indication of an anomaly at 130 sec in Fig. 5, a vibration sensor is determined to be the basis of alarm. The algorithm-specific scoring metric (negative log-likelihood value), is illustrated in lieu of the actual sensor value, along with the chosen threshold. Presentation of the anomaly in this sensor can also be corroborated by the work

First detection for PID # 80 - HPFP FASCOS (22.5 DEG) is at 130.642 sec

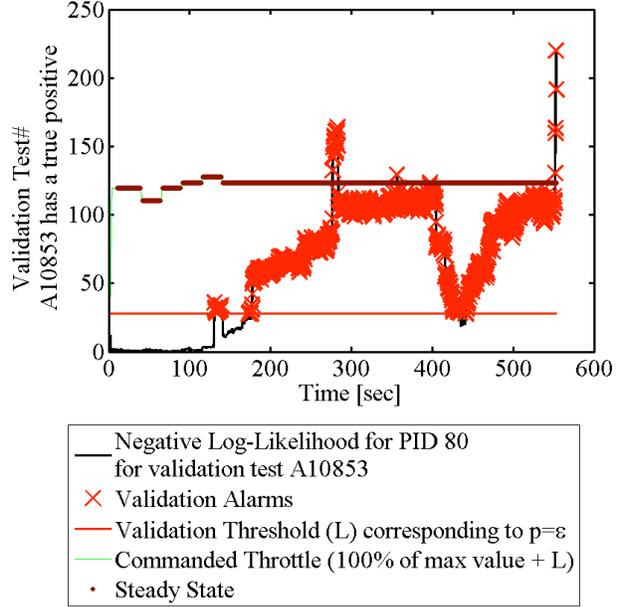


Figure 5. A10853 Vibration Sensor Diagnosis

of Fiorucci et. al. [10].

Additionally, the alarms at the very end of the trial run are found to be diagnostically linked to a controller sensor, shown in Fig. 6. This is a temperature sensor that clearly rises rapidly at the very end of the test, providing us with further indication of an anomaly most likely caused by the event detected previously as a precursor to this rise. It should be clear from Figs. 5 and 6 that in general these alarms correspond to the same alarms in Fig. 4 for which the LDS model was used. We should point out that different thresholds were used for both the GMM and LDS models, since the sensitivity of the threshold for each model will vary accordingly. As a result, the alarms for both methods will not coincide exactly. It is nonetheless important to perform an exhaustive search in order to determine the optimal threshold for each sensor and each model.

In validation test A10852, which was nominally categorized, an anomaly was detected using the LDS model in the range of 210 to 240 sec. An illustration of this anomaly is provided in Fig. 7. There appears to be no reason for the large change in control system error, as the anomalies occur during a steady-state period. However, after performing a diagnostic investigation using the GMM per-sensor approach, we find that the cause of the anomaly is due to a change in mixture ratio during this time period, as was found in [1]. This is illustrated in Fig. 8.

In hindsight, we found that this change of mixture ratio was a planned event. However, there is still utility in being able to characterize departures from nominal operation, whether

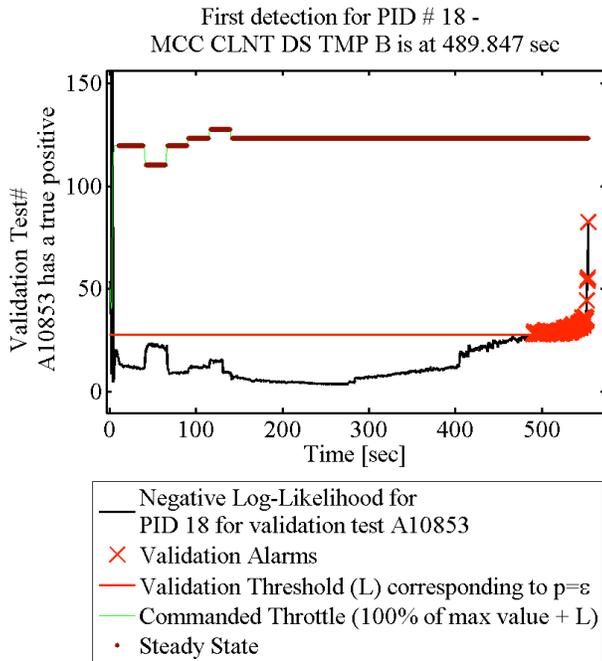


Figure 6. A10853 Controller Sensor Diagnosis

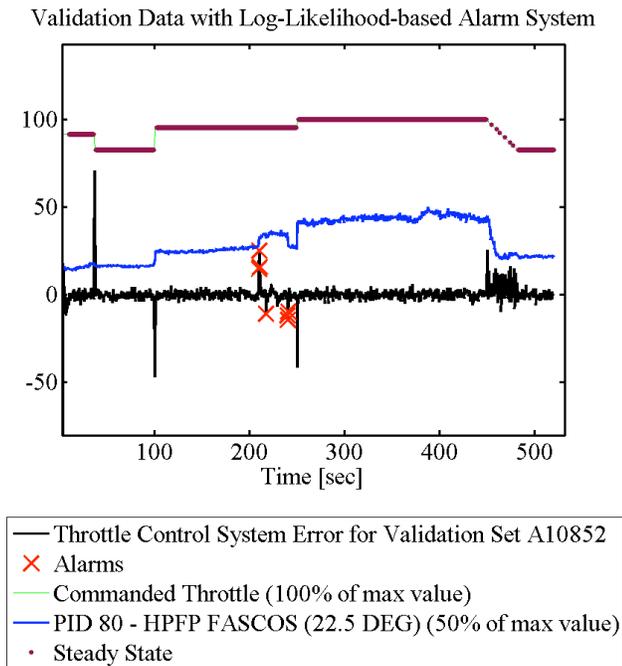


Figure 7. A10852 Control System Error

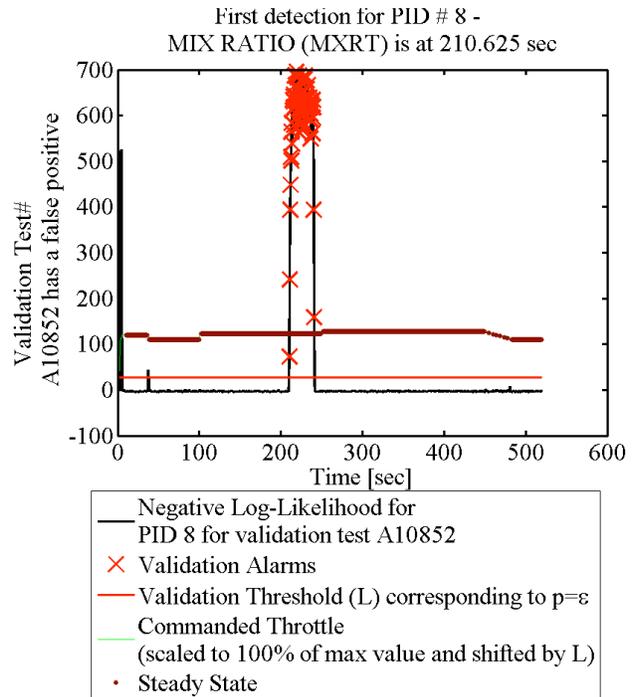


Figure 8. A10852 Mixture Ratio Diagnosis

they are planned or unplanned. This allows us to gain insight into operational idiosyncrasies, when planned reference changes are incorporated into the training data. Therefore, by using both modeling paradigms we can corroborate planned operational commands or provide warnings for unexpected operational commands.

The diagnostic utility of the GMM per sensor method does not end with the corroboration of anomalies found by different means. For tests that were either anomalously or nominally categorized, it is very possible to diagnose sensor failures of a less severe nature that weren't previously detected. An example is test A20619, in which case there was a temperature sensor failure that can be corroborated by other algorithms such as the one implemented by Orca [2], [24].

4. CONCLUSION

The main contribution of this paper has been to present the results of a very simple approach in data mining, using various methods of implementation. The two main modeling paradigms used were the unsupervised Gaussian mixture model and a simple canonical 2^{nd} order linear dynamic system. In the former approach, all available data was used, whereas in the latter approach, only the control system error is used. The following list highlights the most important points covered in this paper.

- Many more examples of failures are needed to allow for a ROC analysis that is statistically significant.
- The validation results of the GMM method are heavily influenced by the training data. The same is not true for the LDS method, where the training data is transformed by using

control system error.

- The GMM per sensor modeling paradigm provides a useful diagnostic reporting capability.
- With the two modeling paradigms used in tandem, they serve to corroborate planned operational commands or provide warnings for unexpected operational commands.
- Failures presenting in vibration data can be found with models based on controller data only, primarily by using the LDS-based algorithmic technique.

In future research studies, we plan to investigate alternate and more sophisticated modeling paradigms such as mixture of experts, supervised LDS, and switching Kalman filters [19]. In addition, one of the motivations for introducing the LDS paradigm was to lay the groundwork for the application and advanced development of more sophisticated anomaly detection techniques requiring the use of Linear Dynamic Systems and Kalman Filtering. Specifically, the design of optimal alarm systems [18], [25] can incorporate thresholds in a more theoretically rigorous fashion, while providing an additional level of predictive capability.

5. ACKNOWLEDGEMENTS

The author thanks Dr. Mark Schwabacher and Dr. Nikunj Oza of NASA Ames Research Center for useful feedback and insightful comments, as well as John Butas and Anthony Kelly of NASA Marshall Space Flight Center for providing data, and astute reviews of the results. The author would also like to thank Matt Davidson, Al Daumann, and John Stephens of Pratt & Whitney Rocketdyne for providing data, as well as Randall Bickford and Edwina Liu of Expert Microsystems for providing data and accompanying detailed descriptions. The research was supported by the Liquid Propulsion portion of NASA's Integrated System Health Management project within the Exploration Technology Development Program funded by the Exploration Systems Mission Directorate.

REFERENCES

- [1] Adrian Agogino and Kagan Tumer. Entropy based anomaly detection applied to space shuttle main engines. In *Proceedings of the IEEE Aerospace Conference*, Big Sky, MO, March 2006.
- [2] Stephen D. Bay and Mark Schwabacher. Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In *KDD '03: Proceedings of The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 29–38, New York, NY, 2003. ACM Press.
- [3] Randall Bickford. MSET Signal Validation System Final Report. Technical report, NASA Contract NAS8-98027, August 2000.
- [4] Dick de Ridder and Robert P.W. Duin. Sammon's mapping using neural networks: A comparison. *Pattern Recognition Letters*, 18(11–13):1307–1316, November 1997.
- [5] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.
- [6] Vassilios V. Digalakis, Jan Robin Rohlicek, and Mari Ostendorf. ML estimation of a stochastic linear system with the EM algorithm and its application to speech recognition. *IEEE Transactions on Speech and Audio Processing*, 1(4):431–442, 1993.
- [7] Tom Fawcett. ROC graphs: Notes and practical considerations for data mining researchers. Technical Report HPL-2003-4, Hewlett Packard Laboratories, 2003.
- [8] Clifford C. Federspiel, Rodney A. Martin, and Hannah Yan. Thermal comfort models and “call-out” (complaint) frequencies. Technical report, University of California, Berkeley, Center for the Built Environment, 2003.
- [9] Clifford C. Federspiel, Rodney A. Martin, and Hannah Yan. Re-calibration of the complaint prediction model. *International Journal of HVAC&R Research*, 10(2), April 2004.
- [10] Tony R. Fiorucci, David R. Lakin II, and Tracy D. Reynolds. Advanced engine health management applications of the SSME real-time vibration monitoring system. In *Proceedings of the 36th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, Huntsville, AL, July 2000.
- [11] Zoubin Ghahramani and Geoffrey E. Hinton. Parameter estimation for linear dynamical systems. Technical Report CRG-TR-96-2, Department of Computer Science, University of Toronto, 1996.
- [12] David L. Iverson. Inductive system health monitoring. In *Proceedings of The 2004 International Conference on Artificial Intelligence (IC-AI04)*, Las Vegas, Nevada, June 2004. CSREA Press.
- [13] John W. Sammon, Jr. A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, C-18(5):401–409, May 1969.
- [14] Michael I. Jordan. An introduction to probabilistic graphical models. Manuscript used for Class Notes of CS281A at UC Berkeley, Fall 2002.
- [15] Benjamin Kedem. *Time Series Analysis by Higher Order Crossings*. IEEE Press, 1994.
- [16] Jan Larsen, Lars Kai Hansen, Anna Szymkowiak, Torben Christiansen, and Thomas Kolenda. Webmining: Learning from the world wide web. *Computational Statistics and Data Analysis*, 38:517–532, 2002.
- [17] Rodney A. Martin. Optimized response to thermal sensation complaints in buildings. Master's thesis, University of California, Berkeley, December 2000.
- [18] Rodney A. Martin. *Optimal Prediction, Alarm, and Control in Buildings Using Thermal Sensation Com-*

- plaints*. PhD thesis, University of California, Berkeley, 2004.
- [19] Kevin Murphy. Switching Kalman Filters. Technical report, Department of Computer Science, University of California, Berkeley, 1998.
- [20] Kevin P. Murphy. The Bayes' Net Toolbox for MATLAB. *Computing Science and Statistics*, 33, 2001.
- [21] Ian Nabney. *NETLAB: Algorithms for Pattern Recognition*. Springer, January 2001.
- [22] Niels Henrik Pontoppidan and Jan Larsen. Unsupervised condition change detection in large diesel engines. In *Proceedings of the 13th IEEE Workshop on Neural Networks for Signal Processing*, pages 565–574. IEEE Press, September 2003.
- [23] S. O. Rice. Mathematical analysis of random noise. *Bell System Technology Journal*, 24:46–156, 1945.
- [24] Mark Schwabacher. Machine learning for rocket propulsion health monitoring. In *Proceedings of the SAE World Aerospace Congress*, volume 114-1, Dallas, Texas, 2005. Society of Automotive Engineers.
- [25] Anders Svensson. *Event Prediction and Bootstrap in Time Series*. PhD thesis, Lund Institute of Technology, September 1998.
- [26] Robert Tibshirani, Guenther Walther, David Botstein, and Patrick Brown. Cluster validation by prediction strength. Technical report, Department of Biostatistics, Stanford University, 2001.



Dr. Rodney Martin is a researcher at NASA Ames Research Center, Moffett Field, CA. His research interests include the intersection of control theory, machine learning and mathematical statistics. Dr. Martin earned an M.S. degree in mechanical engineering from UC Berkeley in 2000, and a Ph.D. in Mechanical Engineering from UC Berkeley in 2004.