Prognostics for Advanced Compressor Health Monitoring

Michael Krok* and Kai Goebel+
GE Global Research

ABSTRACT

Axial flow compressors are subjected to demands for ever-increasing levels of pressure ratio at a compression efficiency that augments the overall cycle efficiency. However, unstable flow may develop in the compressor, which can lead to a stall or surge and subsequently to gas turbine failure resulting in significant downtime and cost to repair. To protect against these potential aerodynamic instabilities, compressors are typically operated with a stall margin. This means operating the compressor at less than peak pressure rise which results in a reduction in operating efficiency and performance. Therefore, it is desirable to have a reliable method to determine the state of a compressor by detecting the onset of a damaging event prior to its occurrence. In this paper, we propose a health monitoring scheme that gathers and combines the results of different diagnostic tools to maximize the advantages of each one while at the same time minimizing their disadvantages. This fusion scheme produces results that are better than the best result by any one tool used. In part this is achieved because redundant information is available that when combined correctly improves the estimate of the better tool and compensates for the shortcomings of the less capable tool. We discuss the usage of diagnostic information fusion for a compressor event coupled with proactive control techniques to support improved compressor performance while at the same time avoid the increased damage risk due to stall margin reduction. Discretized time to failure windows provide event prediction in a prognostic sense.

Keywords: Sensor Validation; Prognostics; Diagnostics; Information Fusion; Data Fusion; Simulation; Surge; Stall;

1. INTRODUCTION

Recent focus in gas turbine research has emphasized improving power generation performance and affordability of the installed base. Field reports from gas turbine operation and maintenance personnel suggest that the lack of a reliable and accurate diagnostic/prognostic system not only decreases affordability but also increases turnaround time when maintenance is warranted due to substantial damage caused by infrequent, hard to detect subsystem events. This means more protection margin, increased diagnostic time, increased spare parts inventory, increased component test time, and reduced affordability. Gas turbine customers are therefore demanding improved diagnostics and prognostics in order to improve engine affordability while at the same time obtaining more power output performance from their existing machines. Therefore, power generation systems require onboard intelligence in a Prognostics Health Monitoring (PHM) system to support extraction of real-time information from the key subsystems of the gas turbine to protect the system from catastrophic events as ever increasing levels of performance (both power and efficiency) are demanded.

In this paper, we address the detection of stall and surge precursors as a pre-requisite to protecting a compressor of a gas turbine from catastrophic fluid instability events. Stall and surge induce stresses, over-temperatures, and loss of efficiency within the compressor[1, 2]. Therefore, considerable effort is expended to ensure that there is sufficient safety margin between the estimated stall line and the operating line. Due to the need for a safety margin, the compressor is operated at conditions less than optimum efficiency and reduced power generation capability. Because the recovery from stall and surge is difficult once an event has begun, we focus on the detection of the precursors to stall and surge in order that sufficient time will be available to use existing low bandwidth controls to protect the compressor before a stall or surge event is encountered. We define stall/surge precursor diagnostics as the determination that the precursors to an impending compressor event have occurred at the current time. Figure 1 illustrates normal operating pressure, stall pressure, and the surge pressure response for a compressor undergoing a transition from apparently normal operation to two consecutive compressor events. In this case, stall is shown to last 40 seconds before surge occurs, but in general, this is not always true as surge can occur without any perceptible indication of stall.

* krok@research.ge.com; phone 1 518 387-5123; fax 1 518 387- 5164; GE Global Research, KWD 209A
+ goebelk@research.ge.com; phone 1 518 387-4194; fax 1 518 387-6104; http://best.me.berkeley.edu/~goebel/kai.html; GE Global Research, K1-5C4A; 1 Research Circle, Niskayuna, NY 12309, USA

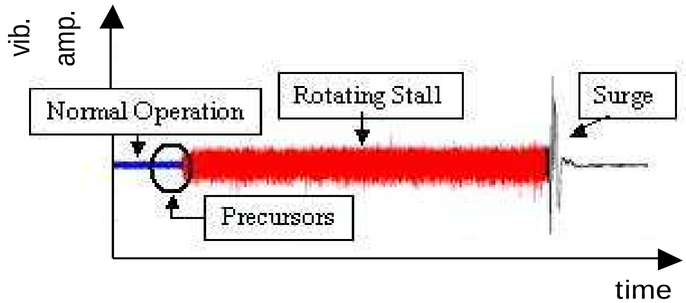

Figure 1: Compressor with Flow Instability Events

Accurate detection is complicated by small magnitude of the event precursors, precursor observability, and sensor noise. Advanced PHM systems need to use engine models as a way to enhance fault observability. In these cases, model fidelity will also impact accurate detection. For reliable detection, precursors large enough to be separated from sensor and model uncertainty need to be observed. We define stall/surge precursor prognostics as the prediction of entering the control critical time prior to a stall or surge event. The period is defined by a window 6 – 60 seconds before the stall. Accurate prediction is complicated by lack of knowledge of the current compressor state, past state history, the time horizon, future operating scenarios, and the occurrence of stall/surge precursors. In this paper, we will focus on diagnostic information fusion technology as applied to a key subsystem of a gas turbine. This technology is a continuation of information fusion work[3, 4] applied to aircraft engine diagnostics. We will demonstrate accurate and reliable compressor diagnostic and prognostic flow instability event assessments.

2. DIAGNOSTIC INFORMATION FUSION

There are different ways to fuse information from different sources. A plurality of approaches can be found including majority voting, weighted majority voting, Behavior-Knowledge Space (BKS), Bayesian Belief Functions, Borda count, weighted Borda count, averaging, order statistics (max, min, med), meta-classifiers (knn, neural nets, support vector machines), Dempster-Shafer, etc. Clearly, the level of complexity varies a lot between the different classifiers. Regrettably, there is no one method that will perform best for all problems.

2.1 Prognostic Performance Measures

In contrast to diagnostic performance evaluation, prognostic performance evaluation exhibits a few specific characteristics. Whereas in diagnostics the aim is to classify a fault or precursor of a fault, a prognostics problem tries to make a judgment about the remaining life of a component. This has repercussions for the performance criteria used to measure the goodness of a tool. Confusion matrices typically establish performance of a diagnostic problem. For prognostic problems, new performance metrics have to be established. While it would seem a reasonable assumption to assess the performance by whether the estimate was on target or not, there will rarely be an estimate that it does that with complete accuracy. However, this is in most cases not required anyhow. The question than is what prognostic period is acceptable for the problem at hand. Clearly the best prediction occurs when the error between the real and the predicted time to failure is zero. However, the utility of the error is not symmetric with respect to zero. For instance, assume the desired earliest prediction for an event is 60 seconds. Assume also that the event will occur in 60 seconds. If the prediction is too early (e.g., 30 seconds earlier), it means that the control critical zone is reached 90 seconds before the failure, while expecting the failure to occur in 60 seconds. This early alarm forces more lead-time than needed to verify the potential for failure, monitor the various process variables, and perform a corrective action. In the case of a compressor, early corrective action would reduce power output and efficiency earlier than was warranted. On the other hand, if the failure is predicted too late (e.g., 30 seconds later), it means that red zone is reached 30 seconds before the failure, while expecting the failure to occur in 60 seconds. This error reduces the time available to assess the situation and take a corrective action. The situation deteriorates completely if the prediction is 60 seconds late (since the failure will occur at the same time as the red zone is reached). Clearly, given the same error size, it is preferable to have a positive bias (early prediction), rather than a negative one (late prediction). On the other hand, one needs to define a limit on how early a prediction can be and still be useful.

Therefore, two different boundaries for the maximum acceptable late prediction and the maximum acceptable early one were established. Any prediction outside of the boundaries will be considered either a false positive or a false negative.

We define the prediction error[5] as

$$E(t) = [Actual \quad time \quad to \quad failure \quad (t) - Predicted \quad time \quad to \quad failure \quad (t)] \tag{1}$$

and we will report prediction results in terms of a histogram of the prediction error E(t). In particular, focus will be on two instances of *E(t)*:

- $E(t_r)$ - prediction error at the time when the red zone is reached, and
- $E(t_0)$ - prediction error at the time when the failure occurs.

Incorrect classifications are typically classified as false negatives (FN) and false positive (FP). In the context of late or early predictions, these categorizations are based on the magnitude of deviation from true time of failure. Therefore, we will define the following limits as the maximum allowed deviations from the origin:

False Negatives A prediction is considered a false negative if one fails to correctly predict a failure more than t_{fn} seconds later than the actual time to failure, i.e., $E(t_r) < -t_{fn}$ seconds. Note that a prediction that is late more than t_r seconds is equivalent to not making any prediction and having the failure occurring.

False Positives A prediction is considered a false positive if we fail to correctly predict a failure if the prediction is more than t_{fp} seconds earlier than the actual time to failure, i.e., $E(t_r) > t_{fp}$ seconds. We consider this to be excessive lead time, which may lead to unnecessary corrections, for example a slow-down of the engine speed.

Although these are subjective boundaries, they seem quite reasonable and reflect the greater usefulness in having earlier rather then later warning/alarms.

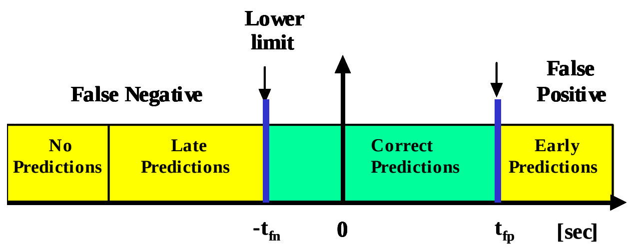

Figure 2: Depiction of False Positives and False Negatives

2.2 Approach

The primary goal of the fusion scheme is to establish time to failure confidence factors, severity and priority while reducing the number of early predictions sent to the controller by integrating the outputs from the different diagnostic tools as well as information from evidentiary sources. Secondary domain specific information is also utilized to further improve the alarm reliability. Critical design requirements are: 1.) Diagnostic Confidence; 2.) Tool Reliability; 3.) Evidentiary Confidence; and 4.) Persistence Indication.

Unlike previous approaches such as Bagging and boosting[6], Dempster-Shafer[7], Model-based approaches[8], fuzzy fusion [9] or statistics based approaches[10], we employ a sequential and parallel multi-layered configurations strategy[11]. In particular, we propose a hierarchical, multi-layer architecture[12] to implement the fusion concept. It manipulates the information from the diagnostic tools (initially) and the fused estimate (later) sequentially until the most likely candidate class has been refined. This process increases and decreases the weight given to the classes according to the strategies implemented in the respective layers of the fusion process. This implies also that it is possible for some classes to emanate as winners from one layer, only to be overruled in the next layer. The architecture displayed in Figure 3 gives an overview over how the fusion is performed conceptually. In particular, there are a total of 5 modules. They are grouped into 3 major layers : 1.) Pre-processing ; 2.) Aggregation ; and 3.) Post-Processing. Details of each module will be given in the following section.

2.2.1 Preprocessing Layer

The Preprocessing layer deals with manipulations of the diagnostic output before the first fusion is performed. It addresses temporal issues such as diagnostic tool disagreement when tools are subject to different update rates as well as scaling the output according to tool performance, and conditioning of the tool output such as filtering.

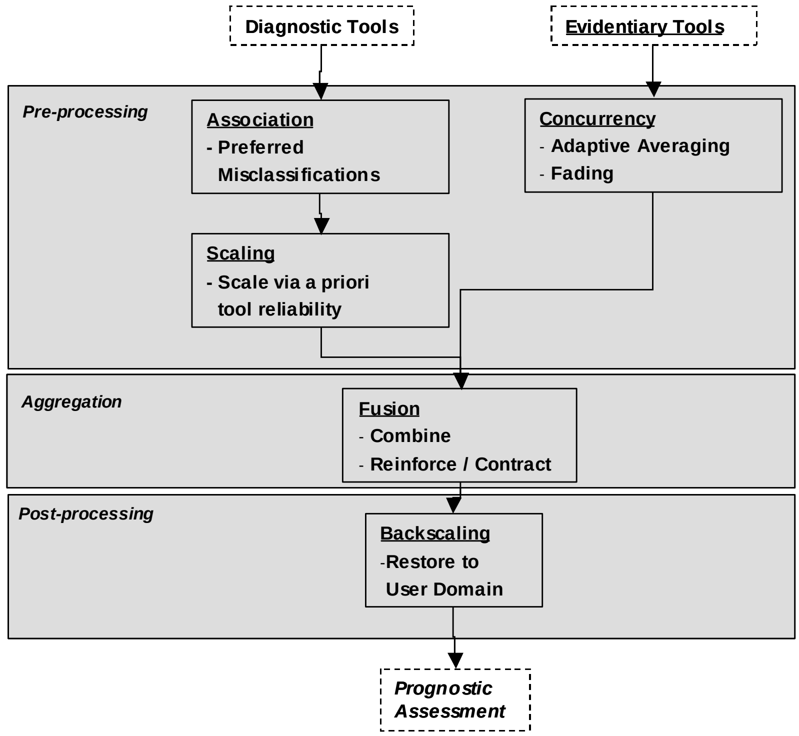

Figure 3 : Fusion Process Flow

Adaptive Averaging

As mentioned before, the challenge in dynamic systems is in providing a different reaction to situations where decisions agree and situations where decisions disagree. When decisions agree, and in the absence of evidence to the contrary, we postulate there is no reason for the fusion agent to change the collective opinion within that time step (there is still a minute chance of joint incorrect classification). However, if tools disagree, the fusion main module has to decide whether one tool is correct and the other is not (and which) or whether an event has occurred between the two tools. To help in this situation, we try to support the fusion main module by removing outliers and generally by smoothing decisions of individual tools in situations of agreement and by updating decisions quickly when a changed event is indicated. This necessitates the need to provide a system status recognition tool that allows the two different strategies to work side by side. We implement the concept of decision smoothing via an exponential averaging time series filter with adaptive smoothing parameter[13, 14]. Let *x(k)* and *x(k-1)* be the filtered decisions at time *k* and *k*-1, respectively, α be the smoothing parameter bounded between a lower and upper value to ensure that the filter becomes neither too sluggish nor too responsive, i.e., $\alpha \in [lowerbound, upperbound]$, where *lowerbound* and *upperbound* are tunable parameters), and *y(k)* is the new incoming decision. Then the filtered decision can be expressed as

$$x(k) = \alpha x(k-1) + (1-\alpha) y(k) \tag{2}$$

Changes of the smoothing parameter will allow weeding out noise and outliers when no fault has occurred but reacting quickly to changes from one event to another. Therefore, if everything is calm, the smoothing parameter is large (remove noise); if something is happening, the strategy is to be more cautious, and to reduce the smoothing parameter value. The reduction is accomplished with the help of an intermittency factor. This factor can be interpreted as a sentinel that monitors system changes. The sentinel checks if the status of a decision changes and keeps track of these changes. First, a change counter establishes whether there is a change of opinion compared to the last one. If the change counter is greater than zero, an intermittency factor is computed dividing the change counter by some user defined sentinel window, i.e.,

$$intermittency = \frac{change\_counter}{sentinel\_window} \tag{3}$$

The sentinel window is a running window that is small initially to allow operation even when there is only one measurement. This implies that initially, the parameter will be more receptive to changes. When more values become available, the window increases and the decision smoothing becomes more temperate. This also means that in general, more information is better under this paradigm because longer and more refined smoothing can take place. The smoothing parameter α is calculated by

$$\alpha = (1 - intermittency)^{constriction\_exponent} \tag{4}$$

where the *constriction_exponent* scales α and is a value > 1

Finally, a forgetting factor is employed that has the effect of reducing the sensitivity of the parameter when no changes have been observed for a while thus allowing the algorithm to settle to its smoothing task. Let the forgetting factor be $forgetting\_factor \in \left]0,1\right[$. Then *change_count* is computed over time as

$$change\_count = change\_count \cdot forgetting\_factor \tag{5}$$

Decision fading

So far we have discussed how to reduce variations in decisions during decision agreement and how the smoothing is automatically turned off when the decisions disagree. However, we still need a mechanism to deal with the disagreement where one tool makes a decision d_1 about a system state at time t_1 and another tool comes to a different conclusion d_2 at a later time t_2. It is then necessary to account for the fact that d_2 may have occurred between t_1 and t_2. We postulate that the later decision needs to be given more weight in case of decision disagreement to account for the possibility of occurrence of event d_2. The question is how much more weight d_2 (or how much less weight d_1) should have. We further propose that the discounting is a function of time passed between d_1 at time t_1 and d_2 at time t_2. The reason is that there is a higher possibility for event d_2 to occur when the period t_2-t_1 is larger and vice versa. In addition, the tools must have information about their a priori performance. Again, we assume that the decisions are scaled between 0 and 1[15] and propose to change the forgetting factor as the confidence value increases. The idea of decision fading is to discount information as it ages when tools disagree at different times (and no new update of both tools can be obtained). We force the older information to "fade" as a function of time passed. If the conditions outlined are met, the decision will be subjected to a fading exponent. The fading exponent must be tuned to the system at hand. The fading factor is close to 1 for small tool confidence and rises to, say, 1.1 as tool confidence increases. The particular slope and upper saturation is system dependent. The governing equation is

$$new\_decision = old\_decision^{fading\_exponent} \tag{6}$$

Evidence updating

The evidence updating incorporates information that does not originate from a diagnostic tool. Besides primary diagnostic information, many systems are rich with secondary evidential information, much of which is encoded as expert knowledge. Evidential information is information that, in conjunction with the primary diagnostic finding, confirms this finding. However, it is not diagnostic in nature. That is, no action would be taken based on evidential information alone. Another reason for classification of a tool as evidential information is if vital information required for the fusion process (such as the confusion matrix) cannot be provided. We implemented this strategy by using evidential information only to support a diagnostic opinion. If the evidential information is found not to corroborate the diagnostic finding, it will not be used to discount the diagnostic finding. Take, as an example, the thermal history of a compressor. A compressor fault might make more sense in light of information indicating many cycles on the compressor. However, few cycles per se do not discount a compressor fault indicated by the diagnostic tools. We propose the use of an evidence matrix that relates evidence to faults. In particular, in the presence of *n* evidence items, a $n \times f$ matrix is formed where *f* is the number of faults considered with "1" and "0" entries for relevance and no relevance, respectively. An evidence item may have relevance for more than one fault but may must have at least one relevance entry. The evidence matrix is then post multiplied by the evidence input. The evidence input is a *1xn* vector containing the current evidence scaled between zero and one, depending on the strength of the evidence. While desired, the scaling is not required for the operation of this layer and can be omitted where strength of evidence information is hard to come by. To realize the boosting character of the evidence, the fused value is increased by the entries of the evidence matrix and the evidence input, weighted by a scaling factor greater than 1. The scaling factor is tuned to keep the effect of the evidence within bounds. 5%-10% was a good value for our system, but other values may be chosen, depending on the application. The operative equation is

$$x_i^f = x_i^f\left(1+e_i\cdot e_b\right) \tag{7}$$

where x_i^f is the fused value for the i^{th} state, e_i is evidence for the i^{th} state, and e_b is the tunable booster.

Associating

We postulate that there is information in a preferred misclassification. That is, if it is known beforehand that a tool B misclassifies fault 1 often as fault 2, then it appears to be prudent to integrate that information into the reasoning process if fault 1 is indicated by tool A and fault 2 is observed by tool B. This information is contained in the asymmetric entries of the confusion matrix. This relation can be utilized by correlating (or associating) the classifier output according to those preferred misclassifications. The correlation is performed by pre-multiplying the classifier output with the respective tool specific association matrix. The latter is calculated by using the negative normalized confusion matrix with unity diagonal elements set to one[15]. The effect of this operation is that side-effects of cross-correlation are factored out before the first fused estimate is calculated. Let the output vector from the tool in question be $[v_1, v_2, v_3]$. The modified outputs *w* corresponding to the three classes are computed as $w = \left(Av^T\right)^T$. For a certain tool in a 3 class situation, *A* is calculated by using the negative normalized confusion matrix with the diagonal elements set to one. Let the confusion matrix *C*

$$C = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} \tag{8}$$

Then *A* is obtained as

$$A = \begin{bmatrix} 1 & -c_{12} & -c_{13} \\ -c_{21} & 1 & -c_{23} \\ -c_{31} & -c_{32} & 1 \end{bmatrix} \tag{9}$$

Scaling

The scaling layer implements functionality allowing a tool that is more reliable to be weighted more heavily than a tool with a worse track record[16]. Clearly, there should be higher trust in a more reliable tool, even when a less reliable tool indicates a strong confidence in its own opinion. However, even the more reliable tool fails occasionally and when the less reliable tool happens to be correct, it would be a mistake to always use the more reliable tool's output exclusively. Then again, it is not known when a particular tool is correct and when it is not. Therefore, we propose to use each tool's information scaled by the degree of its reliability. This reliability is encoded as a priori information in the confusion matrix. More specifically, it is the diagonal entries of the confusion matrix which we will use to weigh each output. In cases where there are considerably less tools for a class than for other classes and the reliabilities for this class are small, there is a chance that the output is biased to the other classes. Therefore, we boost the former cases to allow a more balanced comparison. The test for that scenario is whether the sum of the reliabilities for a decision (given they are relevant) is less than a threshold multiplied by the relevances. Relevance is defined as the ability of a tool to provide output for particular decision. The boosting operation scales the pertinent reliability values to the power of the sum of the reliabilities for that fault normed by the relevances. Let *C* be the confusion matrix of a diagnostic tool, *raw_output* the raw output, *relevance* the binary information granule whether a tool is able to give output for a particular decision (i.e., *relevance*=1 if the tool can give output, 0 otherwise), then the governing equation for scaling is:

$$scaled\_output = raw\_output \cdot \left(\prod_{classes} C(i,i) \cdot \left(trace(C)\right)^{\frac{\sum_{classes} trace(C)}{\sum_{classes} relevance}} \right)^{scaling\_exponent} \tag{10}$$

2.2.2 Aggregation

Diagnostic opinions expressed by different tools which all agree should lead to a more confident assessment of the system state. This is the trivial case where coinciding opinions are rewarded. The rewarding scheme is accomplished by calculating the fused value as the sum of the outputs that are in agreement[16]. The strengthening equation

$s: C' \rightarrow [0,1]$ aggregates the scaled diagnostic outputs that are greater than or equal to threshold t_s i.e., $c'_i \geq t_s$, where $c'_i = \begin{cases} c_i & \text{if } c_i \geq t_s \\ 0 & \text{otherwise} \end{cases}$ produces the fused value c_f, i.e.,

$$c_{f\,i} = \sum_{j=1}^{t} \left(c'_{i_j}\right)^p \qquad (11)$$

p performs the contraction and dilation operation. During contraction information is discounted if several tools disagree on the diagnostic state. It also discounts information if any tool indicates several conditions at the same time. The contraction performs part of the conflict resolution in cases where tools disagree on the diagnostic state of the system. Discounting was chosen because conflicting information lowers the conviction in a correct outcome. Therefore the fused value for the conflicting states will be decreased. Similarly, if several states are indicated at the same time, their contribution will also be discounted (for systems with a one-state assumption). The contraction is accomplished in both cases by an exponent greater than 1, *p>1*. In contrast, dilation is desired when the fused value should be reinforced. This is the case, for example, when all tools agree. Dilation is accomplished with *p<1*. The operative equation for p is

$$p = \begin{cases} p_{upper} & \text{for } c < t_C \\ \dfrac{p_{upper} - p_{lower}}{t_C - t_R} c + p_{upper} - \dfrac{p_{upper} - p_{lower}}{t_C - t_R} & \text{for } t_C \leq c \leq t_R \\ p_{lower} & \text{for } c > t_C \end{cases} \qquad (12)$$

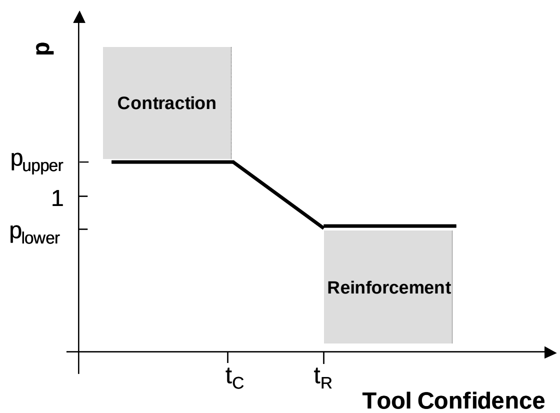

Figure 4: Piecewise Linear Fusion Exponents

2.2.3 Back-Scaling

The fused value has undergone several rounds of modifications in the previous layers which leaves it – while internally coherent – in a state which may not be easily interpretable by a user. To that end, a sequence of backscaling steps is employed. First, the output for each fault is scaled by the number of tools for that class. This increases the relative weight of faults for which there are only few classifiers. This is necessary because those cases are undervalued in the strengthening and weakening layers. Next, the fused value undergoes a dilating operation. The final result is expressed within the 0-1 domain and the faults are ranked from more likely to less likely.

2.4 Results

We benchmarked the algorithm against different standard aggregation tools. In particular, we investigate the performance of the scheme in comparison to an averaging scheme. Averaging, while simple, can be powerful when the tools are sufficiently uncorrelated[17]. However, like in majority voting, a bias can easily be taint the fusion results. We tested individual modules as well as the complete of the fusion algorithm via extensive Monte Carlo simulation within an *n*-class diagnostic context (here *n*=7). A performance index was defined as:

$$performance\_index = 0.6 \cdot (1 - FP) + 0.3 \cdot (1 - FN) + 0.1 \cdot (1 - FC) \qquad (13)$$

where *FC* are the false classified cases that are neither FP nor FN.

The performance of the individual tools varied from 73.41% to 95.85% as summarized in Table 1.

Table 1: Performance indices of individual tools

Tool	Performance index
Tool 1	85.98%
Tool 2	73.41%
Tool 3	95.85%

Averaging a fairly high degree of performance improvement (compared to individual tool performance) with little computational expense. Specifically, averaging performance was 99.67% and 99.87% for bivariate and univariate distributions[18], respectively. In contrast, the overall fusion performance index was $99.91\% \pm 0.02\%$ with 95% confidence and $99.73\% \pm 0.06\%$ with 95% confidence for univariate and bivariate distributions, respectively. A performance gain was established as the percent improvement from the baseline performance. Contributions of some of the individual modules are compiled in Table 2.

Table 2 : Individual layer performance gain

Module	Performance gain
Concurrency	10%
Association	9%
Scaling	6%
Fusion	39%

The overall fused system performance gain compared to the averaging baseline was 30.77% and 18.18% for univariate and bivariate distributions, respectively.

Table 3: Performance indices for averaging and fusion tools using univariate and bivariate distributions

	Univariate distribution		Bivariate distribution	
	Performance Index	Index gain	Performance index	Index Gain
Averaging	99.87% +/- 0.02%	0	99.67% +/- 0.03%	0
Fusion	99.91% +/- 0.02%	30.77%	99.73% +/- 0.06%	18.18%

3. APPLICATION

3.1 Compressor PHM Functionality

The compressor PHM system design includes input signal validation, a real-time compressor model, model-based trending and diagnostics, diagnostic reasoning, and external communication. The system concept is illustrated Figure 5, and each major component will be briefly discussed.

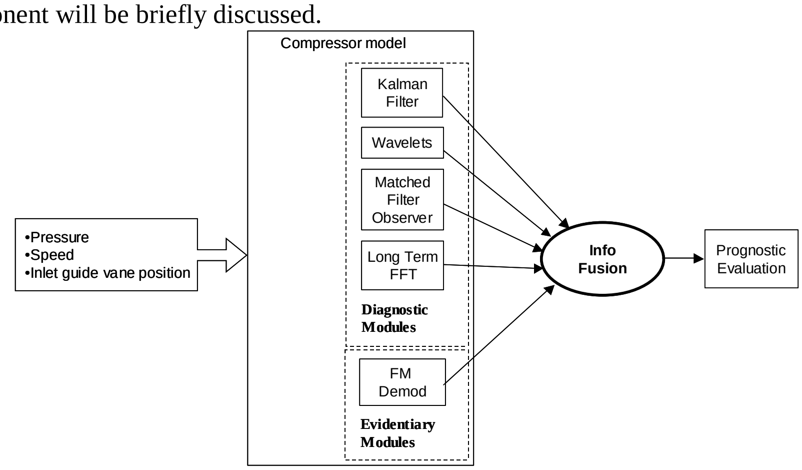

Figure 5: Stall Precursor Detection Concept

3.1.1 Compressor Model

The compressor map model provides the enabling technology for model-based diagnostics, model-based trending and on-line life usage functions. This model is designed to be a real-time, steady state compressor representation, with realistic sensitivities to operating conditions, and control variable inputs. The compressor map is designed to match a nominal, new compressor given the appropriate inputs (environmental conditions, control inputs, and compressor quality). Due to variations in sensor noise and compressor component quality or deterioration levels, the model output will not be accurate for all gas turbines in a fleet given identical inputs and environmental conditions. In order to adjust the model to match any specific compressor, a quality estimator is used.

3.1.2 Stall Precursor Detection Algorithms

The compressor map and sensor information is used to identify changes in compressor performance and relate these changes to specific stages of the compressor hardware. Stall/surge identification techniques employed are based upon extracting perturbations in the sensor response in both the low and high frequency regimes of compressor operation. The various detection algorithms include:

1) Kalman Filter; It acts as a predictor and corrector and models precursor frequency band operating in the 10 – 100 Hz range.
2) Wavelets; It gains frequency composition at a particular time operating in the 0-30 Hz regime.
3) Matched Filter Observer; It adapts to the first few dominant modes of frequency, amplitude and phase of a real quasi periodic signal operating in 10-45 Hz regime by separating signal perturbations in precursor frequency regime from the signal perturbations in the "good" frequency regime.
4) Long Term FFT; It looks for signal perturbations in a wide band about the blade passing frequency of the compressor stage being monitored.
5) FM Demodulator; It looks for perturbations about the blade passing frequency and transforms the blade passing to low frequency.

3.2 Stall Precursor Reasoner

The function of the SPR is to combine all relevant diagnostic and other information, including current compressor state, operational conditions and recent history, to produce a stall/surge prognosis in real-time that is more reliable and robust than any of the individual sources of information. The architecture displayed in Figure 5 gives an overview over the SPR components. In this implementation, there are two classes to consider, No Event, and Event. Differences among diagnostic detection modules can occur for example due to dissimilar compressor map coverage. However, when different diagnostic tools have overlap in the fault coverage, the performance of the diagnostic tools will most likely not be identical. Rather, some diagnostic tools will do better at recognizing an event at certain locations in the compressor map than other locations. Finally, differences in the operating rate at which the diagnostic opinion of a tool is expressed will vary. Some tools perform their detection at a low rate (only once every 8 seconds), others do it at a fairly high rate of 2 KHz (or more).

First, the separation of the inputs into primary and evidential information is made. The Model Based Diagnostic tools (Kalman filter, wavelets, matched filter observer, and long-term FFT) were selected as the primary sources of event detection and FM Demodulator was selected as the evidential input. The evidential diagnostic input to the stall precursor reasoner input mapping function is similar to that for the primary tool, but magnitudes below 50% (or those which represent a likely event estimate) are set to zero since the *a priori* reliability of the diagnosis is not known. The second interface requirement imposed on the primary detection modules is the confusion matrix which represents the *NxN* detection performance of the diagnostic tool across the compressor map. If a detection tool does not possess an assessment of its reliability across the compressor map, then the diagnostic tool is deemed evidential - information that in itself does not constitute diagnostic information but can be used to support a diagnostic finding.

Next, to solve the issue of combining different input types into a unified domain, we imposed an interface requirement that the input to the reasoner had to take the shape of a confidence level for each individual fault scaled between zero and one. The interpretation is that a level closer to one means that an event is increasingly more likely while a confidence level less than 0.5 means an event is increasingly not likely. It was then the task of the designer of the detection tool to provide the mapping from individual detection tool output to diagnostic confidence level. The mapping is performed in a different fashion for each tool, using an appropriate function, which is assumed to be monotonically increasing in shape.

3.3 Simulation and Test Results

Simulation testing of the SPR has been completed and overall system performance is encouraging. A number of stall/surge cases was collected in the field and used for training and testing of the various algorithms. Guided by these data, the diagnostic assessments representing compressors of different component qualities, deterioration levels, and sensor errors were simulated. Multiple variations of the number of primary and evidential tools with different levels of diagnostic reliability were evaluated. A total of three primary diagnostic tools were used and provided the level of diagnostic performance with a reliability of 90%. One evidential tool was used to provide supporting diagnostic information. The confidence values produced by the evidential tool was normally distributed with mean of 0.60 and sigma of 0.25, yielding a reliability of approximately 66%. Figures 6 and 7 illustrate of how the SPR transforms the detection inputs to a overall stall/surge assessment, where Figure 7 is a magnification of the assessment to the last 60 seconds of the test. SPR indicates that there is approximately an 8 second warning of an impending surge before the actual surge occurs. The situation represents testing on a compressor rig where a given compressor's stall line is to be located relative to the design estimate. In this case, the surge occurs earlier than expected and represents ~ 1.2% of deterioration. Shown are the outputs of three stall precursor detection algorithms and the output of the SPR. At the start of the simulation, the compressor is at a relatively safe distance from the stall line. Correspondingly, the stall/surge likelihood estimates are low and different from each other. The SPR concludes that the overall likelihood is lower than all three stall precursor detection algorithms. Holding speed and the IGV angle constant, the backpressure on the compressor was increased in order to drive it towards the stall line. As surge is approached, the stall precursors increase in magnitude. Each detection algorithm responds differently, extracting information pertinent to precursor activity at different frequency regimes within the sensor measurement response. The surge event occurs at 230 sec but the SPR indicates an impending surge (based on the 0.9 confidence threshold) about 8 sec earlier at 222 sec. In each case, the detection algorithms correctly identified the surge, but at an incipient level of surge response. The sharp upward SPR transition 195 sec into the test occurs when all three algorithms are reinforcing the overall stall assessment. Before that only two of the three were reinforcing the assessment that the precursors were becoming dangerously high. The overall SPR assessment allows the low bandwidth controls to change the compressor's state and move it away from the impending surge conditions, without having the compressor experience incipient surge. In general, but depending upon the vintage of the controls and actuators, a 6-10 sec advance warning is sufficient to allow the controls to be fully active, and transition the compressor to a safe operating region.

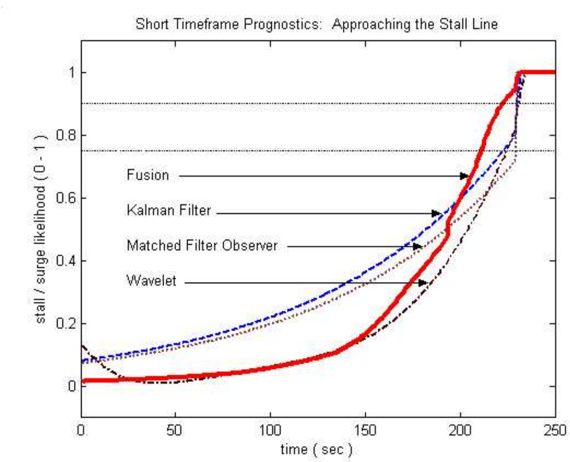

Figure 6: Last 250 Seconds of Compressor Mapping Test

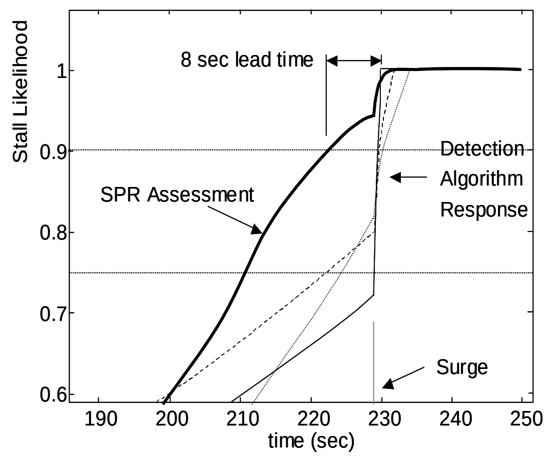

Figure 7: Last 60 Seconds of the Test Point

Precursors were initially assumed to be constant. However, they are not. According to the rating scheme outlined in section 2, within the scope of this work, we offer a real-time failure prediction using an expanded traffic-light metaphor. The red zone is characterized by stall likelihood $\geq$ 0.9; the orange zone is characterized by 0.75 $\leq$ stall likelihood < 0.9; the yellow zone is characterized by stall likelihood within the corner points defined by 0.35 and 0.75; and, the green zone is everything < 0.35. The variability of the stall likelihood is +/- 0.005 except in those rare situations when a particular algorithm has a false positive, while each individual algorithm has a variability in its own assessment that ranges from +/- 0.02 to +/- 0.035. This variability can be considered noise that varies about the mean assessment. Figure 8 illustrates the detection algorithm and SPR time responses during an actual turbine/compressor test that captured various power levels and operating states of the compressor. The SPR typically has a variability that is four time less than an individual SP algorithm response variability. The figure also illustrates what happens when a False Positive occurs. One algorithm has incorrectly indicated an imminent surge, but the SPR decides that the compressor remains in a safe state, and no protective controls are necessary.

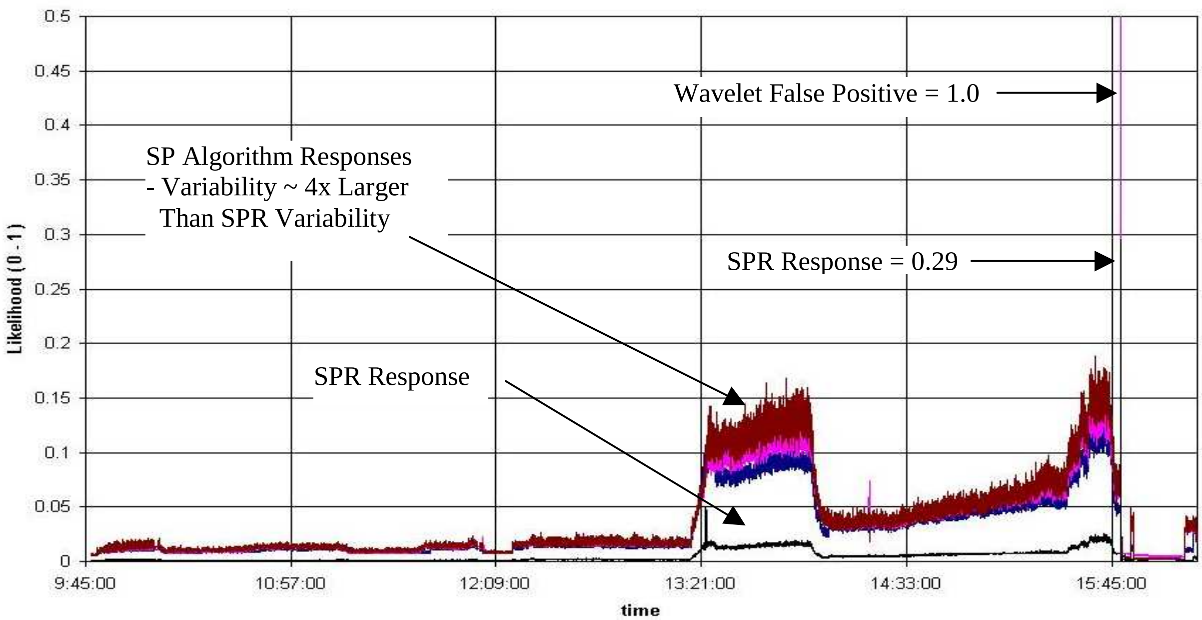

Figure 8: Stall Precursor Reasoner and Associated Algorithm Responses During Turbine Testing

4. SUMMARY AND CONCLUSIONS

A condition-based, diagnostic Stall Precursor Reasoner for gas turbine compressors has been developed. The SPR is a key module of the overall Compressor PHM system and integrates continuous data from various diagnostic and usage sources to provide a more accurate assessment of compressor condition than available from any individual sources of information. The design has been accomplished in a highly modular form such that application to other compressor models and gas turbine platforms can be readily achieved. Through a systematic flowdown of performance and functional requirements into the system design, the combined effects of all the diagnostic tools were considered in order to achieve the desired high level of accuracy for compressor health monitoring capability. We developed a capability that fuses multiple assessments together to provide a short timeframe prognosis of an impending compressor stall/surge event.

We focused in particular on issues which arise from a real-time implementation of compressor stall/surge detection across the entire compressor operating regime. We solve the issue of transient anomaly detection and the inclusion of diagnostic information whose a priori performance is unknown. Simulation and gas turbine testing of SPR has been completed and overall system performance is very encouraging. The resulting GPR performance meets the system level requirements and provides a multiple seconds of advance warning before a stall/surge occurs allowing the low bandwidth controllers to take appropriate protective action. The benefits are that less stall margin could be set either in the factory in a new machine or later as an upgrade to an installed machine to allow more power output and efficiency. Future work should address a continuous mapping to time to failure.

REFERENCES

1. Greitzer, E. M.; "REVIEW – Axial Compressor Stall Phenomena"; Transactions of the ASME; Vol. 102; June 1980.
2. Gu, Guoxiang; Sparks, Andrew; Banda, Siva S.; "An Overview of Rotating Stall and Surge Control for Axial Compressors"; IEEE Transactions on Control Systems Technology, Vol. 7, No.6, November 1999.
3. M. Ashby, W. Scheuren, *Intelligent Maintenance Advisor for Turbine Engines (IMATE),* Proceedings of the IEEE Aerospace Conference, April 2000
4. Goebel, K., Krok, M. and Sutherland, H., "Diagnostic Information Fusion: Requirements Flowdown and Interface Issues" *Proceedings of the IEEE 2000 Aerospace Confence – Advanced Reasoner and Information Fusion Techniques*, vol. 6, pp. 155-162,, 2000.
5. Bonissone, P., Goebel, K. and Chen, Y. "Predicting Wet-End Web Breakage in Paper Mills", *Working Notes of the 2002 AAAI symposium: Information Refinement and Revision for Decision Making: Modeling for Diagnostics, Prognostics, and Prediction,* Technical Report SS-02-03, pp. 84-92, AAAI Press, Menlo Park, CA, 2002.
6. Y. Freund and R. Schapire, "A Short Introduction to Boosting", J. Japanese Society Artificial Intelligence, Vol. 14, No.5, pp. 771-780. 1999.
7. P. Smets, "What is Dempster-Shafer's model?" Advances in the Dempster-Shafer Theory of Evidence, Yager, R., Fedrizzi, M., and Kacprzyk, J., (Eds.), John Wiley & Sons, New York, pp. 5-34, 1994.
8. M. Nelson and K. Mason, "A Model-Based Approach to Information Fusion". Proc. Information, Decision, and Control, pp. 395-400, 1999.
9. A. Loskiewicz-Buczak, R. Uhrig, "Decision Fusion by Fuzzy Set Operations", Proc. third IEEE Conf. Fuzzy Systems, Vol. 2, pp.1412-1417, 1994.
10. N. S. V. Rao, Finite sample performance guarantees of fusers for function estimators, Information Fusion, Vol. 1, no. 1, pp. 35-44, 2000.
11. A. Rahman and M. Fairhurst, "Towards a Theoretical Framework for Multi-Layer Decision Fusion", Proc. IEE Third Europ. Workshop on Handwriting Analysis and Recognition, pp. 7/1-7/7, 1998.
12. K. Goebel, Architecture and Design of a Diagnostic Information Fusion Tool, AIEDAM: special edition AI in Equipment Service, vol. 15 no. 4, pp. 335-348, 2001.
13. Khedkar, P., and Keshav, S., "Fuzzy Prediction of Time Series", Proceedings of the IEEE International Conference on Fuzzy Systems, San Diego, 1992.
14. K. Goebel, "Decision Forgetting and Decision Smoothing for Diagnostic Information Fusion in Systems with Redundant Information", Proc. SPIE: Sensor Fusion: Architectures, Algorithms, and Applications IV, Vol. 4051, pp. 438-445, 2000.
15. K. Goebel and S. Mysore, "Taking Advantage of Misclassifications to Boost Classification Rate in Decision Fusion", Proc. SPIE: Sensor Fusion: Architectures, Algorithms, and Applications V, pp. 11-20, 2001.
16. K. Goebel, "Conflict Resolution using Strengthening and Weakening Operations in Decision Fusion", Proc. 4th Annual Conf. Information Fusion, Fusion 2001, pp. ThA1-19 - ThA1-25, 2001.
17. A Method to Calculate Classifier Correlation for Decision Fusion, K. Goebel, W. Yan, and W. Cheetham, pp. 135-140, Proceedings of IDC 2002, Adelaide, 11-13 February, 2002.
18. Goebel, K, "Sensitivity of fusion performance to classifier model variations" *Proceedings of SPIE Sensor Fusion: Architectures, Algorithms, and Applications VII,* this conference, 2003.