FAULT DETECTION AND DIAGNOSIS IN TURBINE ENGINES USING HIDDEN MARKOV MODELS

Sunil Menon
Önder Uluyol
Kyusung Kim
Honeywell Engines, Systems & Services
3660 Technology Drive
Minneapolis, MN 55418
E-mail: sunil.menon@honeywell.com, onder.uluyol@honeywell.com, kyusung.kim@honeywell.com

Emmanuel O. Nwadiogbu
Honeywell Engines, Systems & Services
111 S. 34th Street
P.O. Box 52181
Phoenix, AZ 85034
E-mail: emmanuel.nwadiogbu@honeywell.com

NOMENCLATURE

EGT Exhaust Gas Temperature
\( x_i(t) \) HMM hidden state at time \( t \)
\( y(t) \) Sensor measurements
\( P(\cdot) \) Conditional probability function
\( a_{ij} \) HMM hidden state transition probability
\( b_{jk} \) HMM output transition probability
L Log-likelihood value

ABSTRACT

Accurate fault detection and diagnosis in turbine engines is key to effective maintenance and improved availability of systems dependent on these engines. In this paper, we present a novel method for accurate fault detection and diagnosis using Hidden Markov Models (HMMs). In particular, we focus on engine faults that are manifest in transient operating conditions such as engine startup and acceleration. HMMs are stochastic signal models that are effective in modeling transient signals. They are developed with engine data collected under nominal operating conditions. Engine data representing different fault conditions are used to develop the fault HMMs; a separate model is developed for each of the faults. Once the nominal and fault HMMs are developed, new engine data collected from the engine are evaluated against the HMMs and a determination is made whether a fault is indicated. Here, we demonstrate our HMM-based fault detection and diagnosis approach on engine speed profiles taken from a real engine. Further, the effectiveness of the HMM-based approach is compared with a neural-network-based approach and a method based on using principal component analysis in conjunction with a neural network approach.
INTRODUCTION

Accurate turbine engine fault detection and diagnosis (FDD) is vitally important to reducing airline operating costs and improving safety. Incipient fault detection in turbine engines can reduce costs significantly due to in-flight shutdowns (IFSDs), delays and cancellations (Ds & Cs), unscheduled engine removals (UERs), or take-off aborts (TOAs). Accurate incipient fault detection is necessary for a more effective preventive maintenance program.

Turbine engine FDD involves the early detection of incipient as well as sudden faults. Detection is followed by diagnosis of a sensor, actuator, or system component fault. Many approaches to FDD are discussed in the literature. Broadly, the two main approaches to FDD are model-based and data-driven [1]. Model-based FDD methods use a physical or empirical model of the system to generate residuals (difference between the model outputs and actual system measurements). These residuals are passed through a decision logic system to determine a diagnosis. Data-driven methods use historical data collected from the system to construct quantitative (neural networks, statistical models) or qualitative (trends-based, rule-based) FDD methods.

Most engine FDD is performed using measurement data from engines that are in steady-state conditions. There are several reasons for this; most notably, system transients can confuse FDD methods, resulting in reporting of incorrect results. A more robust approach to developing FDD methods that explicitly account for transient data is required. Furthermore, most turbine engine FDD methods are developed with engine performance models that have been validated only under steady-state conditions or with actual engine data at steady-state conditions. Engine models that accurately represent the system in transient conditions are difficult to develop [2]. Nevertheless, developing FDD methods designed to operate on system data during transient as well as steady-state operation has several important advantages. First, certain system faults have a distinct signature during system transient conditions that normally would not be discernible during steady-state conditions. Second, the effect of feedback control action is less dominant during transient conditions than during steady-state conditions. Because feedback control suppresses the effect of sensor and system faults, faults are more evident during transient conditions. Finally, certain engine component incipient faults are manifest only during transient conditions such as startup and shutdown (e.g., starter and igniter system faults).

Here, we present an approach to engine FDD using Hidden Markov Models (HMMs). HMMs are stochastic signal models whose parameters are determined using well-defined methods [3]. The theory of HMMs was first introduced in the late 1960s by Baum and colleagues [4] and has been widely used in speech processing algorithms since the 1970s. HMMs have been shown to be very good at modeling transient signals such as those occurring in speech recognition and communication systems [3,5]. Owsley et al. have used HMMs as a classification method in a machine-tool monitoring application [6].

In this work, HMMs are used to model the engine startup characteristics. Data representing the engine startup under nominal conditions are used to train the HMMs or, in other words, determine the parameters of the HMMs. This is analogous to system identification methods used to develop empirical models such as neural networks (NNs).

Engine data is input to the nominal HMM model, and the output log-likelihood values (posterior probabilities) represent how close the nominal engine, as represented by the HMM, is to the engine data. In a similar way, HMM models representing the engine startup under different faults can be developed; these are the so-called fault HMM models. The performance of HMM models in detecting engine startup faults using startup data is compared with NN-based FDD approaches. One of the latter approaches uses the principal component analysis (PCA) method to determine the important features in the data. This smaller feature set is then fed into a NN classifier [7]. Another NN-based approach used for comparison with the HMM-based approach is to use the NN directly as a classifier without any preprocessing [8].

The following sections present turbine engine startup characteristics and a brief overview of HMMs, followed by a description of the application of HMMs to engine FDD. Results of using the HMM engine FDD approach with actual engine startup data are given. The HMM-based engine FDD approach is compared with a conventional approach using NNs and a method that combines PCA and NN classifiers. These results are then discussed, followed by a summary and conclusions that can be drawn from this work.

ENGINE STARTUP CHARACTERISTICS

In this paper, we demonstrate the potential of developing an engine FDD using the engine startup as a specific engine transient condition. A typical engine speed plot during startup is shown in Figure 1. The startup sequence proceeds as follows: The starter begins to rotate the engine compressor and the igniter is switched on (igniter plugs start firing). At around 7% of full engine speed (hereafter, all references to engine speed are percentages of
full engine speed), the igniter system successfully completes light-off (when the combustor is able to maintain combustion). The engine's exhaust gas temperature (EGT), a key engine parameter, rises sharply at light-off. This rapid increase in EGT is often the only indication that light-off has occurred. The engine starter continues to provide rotational torque to the engine as the engine speed continues to increase. The engine turbine now begins to provide rotational energy to the system. At around 40% engine speed, the starter system is shut off (at time $t_{off}$ in Figure 1). There is a momentary drop in engine speed, as shown in Figure 1, due to the drop in rotational torque when the starter is shut off. The engine power section now takes complete responsibility for bringing the engine to full speed and does so within a prescribed amount of time.

During the first phase of startup, until light-off has occurred, igniter system degradation will be manifest more strongly than at other times during startup (of course, if the igniter system has a developed fault, light-off is not likely to occur and can be easily detected). During the next startup phase, from 7%-40% speed, starter faults are manifest more strongly than at other times during startup. In the final phase, from the time the starter shuts off and the engine rotates on its own power, the effects of the power section degradation are most evident. The effects of starter degradation will propagate (in terms of start times) forward in time during startup, but the engine power section effects dominate after the 40% speed mark.

HIDDEN MARKOV MODEL BASICS

This section gives a short introduction to HMMs. The reader is referred to Rabiner [3] for more details on HMMs. Consider a system that can be described at any time by a state that is influenced by a state at a previous time instant. This temporality is well described by HMMs. The HMM shown in Figure 2 describes a system with three states, $x_i(t)$. A sequence of states at successive times, up to time $T$, is $x^T = \{x(1), x(2), \ldots, x(t), \ldots, x(T)\}$, where $x(t)$ is the state at time $t$. 

![Figure 2. Hidden Markov Model with Three States](image-url)
The state sequence is described by state transition probabilities, $a_{ij}$:

$$a_{ij} = P(x_{j}(t+1) \mid x_i(t)), \quad 1 \leq i, j \leq N,$$  \hspace{1cm} (1)

where, $a_{ij}$ is the time-independent probability of having state $x_i$ at time $(t+1)$ given that the state at time $t$ was $x_j$ and $N$ is the number of distinct system states. To calculate the probability that a particular model, specified by the state transition probabilities $a_{ij}$, generated a particular state sequence, the successive probabilities are multiplied.

In HMMs, the states are not available—hence the “hidden” in the name—so the model external measurements $y$ are distinct from the hidden states. There is a probability, $b_{jk}$, associated with a particular output being emitted by a particular state defined by

$$b_{jk} = P(y_j(t) \mid x_j(t)), \quad 1 \leq j \leq N, 1 \leq k \leq M,$$  \hspace{1cm} (2)

where $M$ is the total number of measurable outputs. The HMM outputs are depicted in Figure 2.

The three main problems associated with HMMs are (1) given an HMM with transition probabilities, $a_{ij}$ and $b_{jk}$, defined, determine the probability that a particular set of observations, $y(t)$, was generated by this model; (2) given an HMM as well as a set of measurements, $y(t)$, determine the most likely sequence of hidden states, $x(t)$, that led to these observations; and (3) given a set of training observations and the approximate structure of the HMM, the number of hidden states, and the number of observations, determine the probabilities $a_{ij}$ and $b_{jk}$. We are concerned mainly with problems (1) and (3) in this work.

Problem (1), to calculate the probability of a particular observation sequence given a specified HMM, is represented by the equation

$$P(Y^T) = \sum_{s=1}^{S} P(Y^T \mid x^T_s) P(x^T_s),$$  \hspace{1cm} (3)

where $s$ is the index of the state sequence $x^T_s$ of $T$ hidden states. Since the output probabilities depend only on the hidden states, which is assumed here, the first term in equation (3) can be written as

$$P(Y^T \mid x^T_s) = \prod_{j=1}^{T} P(y(t) \mid x(t)),$$  \hspace{1cm} (4)

which are essentially the products of the output transition probabilities, $b_{jk}$. Also, since we assume that the state dynamics is described by a first-order Markov process, the second term in equation (3) can be described by the equation:

$$P(x^T_s) = \prod_{j=1}^{T} P(x(t) \mid x(t-1)),$$  \hspace{1cm} (5)

which is essentially a product of the state transition probabilities, $a_{ij}$. Problem (1) as represented by equation (3) is solved using the forward procedure [3].

The solution to problem (3), to determine the HMM parameters, is analogous to the problem of determining the weights of a NN given the training data set and network model structure. In the case of the HMM, this is accomplished by the Baum-Welch or forward-backward procedure, which is a specific instance of the Expectation-Maximization (EM) method [3].

Finally, we have thus far described HMMs with the assumption that the output $y(t)$ is a discrete variable. In many applications, including the one discussed in this paper, the output is a continuous variable. This is the continuous HMM and is modeled by assuming that the output has a Gaussian or a Gaussian mixture distribution [9].

**TURBINE ENGINE DIAGNOSTICS USING HMMS**

The turbine engine data collected for this work is from an auxiliary-power-unit-type (APU) system for an aircraft. The APU system provides bleed air for the environmental control system, as well as to start the aircraft main engine. In addition, the APU runs the electric generator that supplies electric power to the aircraft. The engine is started using a system of pressurized air tanks that are modulated by a system of valves. Using
compressed air to provide mechanical energy to the engine turbine is analogous to using an electric starter motor in commercial APU systems.

The engine data are recorded, from startup to full operating speed, at a sampling rate of 1 s. The recorded engine parameters include engine speed, exhaust gas temperature (EGT), and the compressed gas pressures, both at the source and after valve modulation. The engine startup data corresponding to engine starts in which systems functioned normally are referred to as “good” starts. In a certain number of recorded engine starts, the air pressure modulating valve had mechanical failures that resulted in engine starts characterized as “bad.” Eighteen good starts and 11 bad starts are used for analysis in this work. It should be noted that when the air pressure modulating valve failures were advanced, the engine would not start at all—so-called “hung” starts. The bad engine starts used in this work are with failures that are not yet advanced, but are at a beginning to intermediate stage.

The HMMs used in this work are the left-right type, an architecture widely used in applications involving dynamical systems modeling. The engine variables modeled are the engine speed and exhaust gas temperature (EGT). HMMs that model just the engine speed and then both the engine speed and EGT are developed in this work. The number of states (nodes) assigned to the HMM model is equal to the number of measurements representing the engine startup—the number of HMM outputs is equal to the number of hidden states. Here, the number of HMM nodes is fixed at 20, which represents the first 20 s of engine startup (the sampling rate is 1 Hz). The training and test data sets used to develop the HMM are generated from the good start data in two ways: in the first method, a multiplicative normal noise term is added to the original data and additional data points are thus generated; in the second method, a variation of the bootstrap method [8] is applied to the data at each time instant. The bootstrap method is a technique of repeatedly drawing data, with replacement, from a data set to generate additional data sets. These additional “bootstrap” data sets are then used for HMM training. The data set generation methods are evaluated for their efficacy in developing HMMs for FDD. The HMM models are trained using the training data sets and then tested using test data with different levels of additive noise.

The HMMs developed in this work are nominal models; i.e., the models represent the engine startups with no faults. Figure 5 shows a block diagram of the development of the HMM-based engine FDD method. The startup training data set, which comprises multiple sets of good startup data, is used to determine the HMM parameters. The HMM structure, such as the number of outputs and states, is prespecified. The HMM parameters are determined using the Baum-Welch algorithm as discussed in the previous section. The HMM parameters now specify the HMM nominal model. During normal operation, the startup data are input to the nominal HMM model. The model calculates the probability that the input measurement sequence came from the HMM model. This log-likelihood value is a measure of how close the input data is to a nominal engine system, subject to the accuracy of the HMM model, of course. The fault startup test data would yield log-likelihood values that are very different from those generated by good startup data. Therefore, the HMM output log-likelihood values can be used as a fault indicator for engine FDD.

![Figure 5. HMM-based Engine FDD Method](image)

The HMM-based FDD method is compared with the results obtained from a NN approach and a PCA approach in conjunction with a NN approach. In the latter approach, the PCA method is used to map the engine startup data into a smaller set of principal components that capture the essence of the startup data. The principal components are then input into a NN classifier [9]. The NN is a feedforward type with one hidden layer. The backpropagation training method is used to develop the NN classifiers. A more detailed explanation of the methods used is given in [10].
**RESULTS AND DISCUSSION**

The HMM-based FDD methods are developed with just the engine speed as the model input, as well as with both the engine speed and EGT. Two different training data sets are used in this work; the first training data set is obtained by adding normal noise to the original set of startup data, and the second set is based on a variation of the bootstrap method of resampling data. The bootstrap method, as used here, is applied to the data set at each particular time instant. The HMMs are tested on test data sets developed with normal noise added to the original data set. The levels of noise are zero mean and 10% and 20% (of the original signal value) variance.

**Table 1. Comparison of the HMM and Neural Network Classification Results with Engine Speed (Percentage Values)**

<table>
<thead>
<tr>
<th>Actual</th>
<th>HMM</th>
<th>Predicted</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Good</td>
<td>Bad</td>
<td>Good</td>
<td>Bad</td>
<td>Good</td>
<td>Bad</td>
</tr>
<tr>
<td>Bad</td>
<td>18.2</td>
<td>81.8</td>
<td>19.7</td>
<td>80.3</td>
<td>19.7</td>
<td>80.3</td>
</tr>
<tr>
<td>Good</td>
<td>94.6</td>
<td>5.4</td>
<td>91.7</td>
<td>8.4</td>
<td>90.6</td>
<td>9.4</td>
</tr>
</tbody>
</table>

**Table 2. Comparison of the HMM and Neural Network Classification Results with Engine Speed and EGT (Percentage Values).**

<table>
<thead>
<tr>
<th>Actual</th>
<th>HMM</th>
<th>Predicted</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Good</td>
<td>Bad</td>
<td>Good</td>
<td>Bad</td>
<td>Good</td>
<td>Bad</td>
</tr>
<tr>
<td>Bad</td>
<td>9.1</td>
<td>90.9</td>
<td>9.1</td>
<td>90.9</td>
<td>9.1</td>
<td>90.0</td>
</tr>
<tr>
<td>Good</td>
<td>88.9</td>
<td>11.1</td>
<td>87.4</td>
<td>12.6</td>
<td>87.3</td>
<td>12.7</td>
</tr>
</tbody>
</table>

Table 1 lists the classification results of the HMM FDD method, the NN method, and the PCA plus NN method using only the engine speed. The table lists the method (HMM, NN, PCA+NN) classification percentages when a good start is classified as such or when a good start is classified as a bad start etc. These classification results are given with the actual data, and data with additive noise added. The performance of these methods is evaluated with test data with different additive noise levels. The results in Table 1 show that the HMM classifier method is more accurate than the NN-based classifier and the PCA plus NN method in classifying the nominal startup data. The HMM classifier method performance is about the same as compared to the NN method and PCA plus NN method in classifying the bad startup data. Table 2 lists the HMM, NN method, and the PCA plus NN method classification results when both the engine speed and EGT are used. In this case, it can be seen that the NN and PCA plus NN methods perform better at classifying the good startup data. However, the HMM method performs better than the NN method at classifying bad startup data. It is observed in Tables 1 and 2 that the NN and PCA+NNs classification results are identical in some cases and the classification method is insensitive to the level of additive noise. This is due to a combination of factors: the robustness of the NN and PCA+NN methods and the particular selection of the test data cases. In addition, the actual cases where the NN and PCA+NN methods classify correctly/incorrectly are different.
SUMMARY AND CONCLUSIONS

This paper describes a method for detecting engine startup faults using HMMs. The HMMs are developed using good engine startup data. To generate a sufficient engine data population for training, the original data are resampled using a variation of the bootstrap method. In addition, HMMs are developed using training data created by adding normal noise to the original data set. The engine speed and the engine speed and the EGT combined are used to develop the HMM models. The performance of the HMM in fault detection and diagnosis is evaluated by calculating the probabilities (log-likelihood values) that the HMM generated the observed data. The log-likelihood values are a direct indication of whether the engine data should be classified as good or bad. The HMMs developed are tested extensively with good and bad engine startup data, and the classification results are reported. A NN method and a PCA in combination with a NN classification method is also developed, and the performance of the HMM with these two FDD methods is compared. The HMM is shown to perform well in classifying good and bad engine startup data. The HMM bad startup data classification is better compared to the NN and PCA plus NN methods when both the engine speed and EGT are used. While the actual classification results using the different methods, percentage-wise, are close and in some cases equal, the actual test cases that are correctly classified are different. This suggests that a better approach to fault classification would be to fuse the output results of the different methods using an appropriate knowledge fusion method [11].

This work demonstrates the utility of using transient data for engine FDD. Although this paper demonstrates engine FDD using engine startup data, other operational transient modes such as ground idle to full power can be used as well for FDD. The results of the HMM-based FDD methods indicate that the HMM technique is very promising in FDD with transient data. Further work in HMM training methods and architecture for FDD is currently underway and the results will be reported in future papers.

REFERENCES


