BAYESIAN SENSOR ESTIMATION FOR MACHINE CONDITION MONITORING

Chao Yuan and Claus Neubauer

Siemens Corporate Research, 755 College Road East, Princeton, NJ, 08540

ABSTRACT

We present a Bayesian framework to tackle the problem of sensor estimation, a critical step of fault diagnosis in machine condition monitoring. A Gaussian mixture model is employed to model the normal operating range of the machine. A Gaussian random vector is introduced to model the possible deviations of the observed sensor values from their corresponding normal values. Different levels of deviations are elegantly handled by the covariance matrix of this random vector, which is estimated adaptively for each input observation. Our algorithm doesn't require faulty operation training data, as desired by previous methods. Significant improvements over the previous methods are achieved in our tests.

Index Terms— Machine condition monitoring, expectation-maximization, Gaussian mixture model.

1. INTRODUCTION

The behavior of a machine is represented by the readings of a set of sensors installed in different parts of the machine. When the machine is working properly, the sensor data should be distributed in a *normal operating range* as shown in Figure 1. A fault is detected if future data deviate much from this range. To diagnose the fault, we additionally need to locate the sensors which cause this deviation. Therefore, fault diagnosis typically consists of two steps: sensor estimation and rule-based decision [1]. In sensor estimation, based on the observed values $\mathbf{y}$ of sensors, we estimate the normal values $\mathbf{x}$ that sensors should have if the machine operates normally. Both $\mathbf{x}$ and $\mathbf{y}$ are d-dimensional vectors, where d is the number of sensors used. In rule-based decision, fault types are determined based on the observations, estimates and residues (the difference between the observation and the estimate) of sensors.

Sensor estimation is our focus. Since a fault usually only affects a small number of sensors, a natural way is to use the observed values of normal sensors to infer the normal values of faulty sensors. However, under many circumstances, we don't know which sensor is faulty. This is assumed throughout this paper, although we note that such information can be very helpful.

Most widely used machine condition monitoring algorithms attempted to establish a deterministic mapping network from $\mathbf{y}$ to $\mathbf{x}$ based on a set of historical operation data. Among them, there is Auto-Associate Neural Networks (AANN), which built a five-layer neural network to model this mapping [1, 2]. The multivariate state estimation technique (MSET) [1] estimated $\mathbf{x}$ as a linear combination of training data via a kernel-based least square method. Both methods learned a d-input-d-output network.

Regression methods were also proposed for sensor estimation; support vector regression (SVR) is one of the most notable [3]. Different from the AANN and the MSET, the SVR estimated a sensor from the other sensors' observed values and this is done for each of the d sensors. Thus, the mapping network consists of d small d-input-one-output mapping network.

The challenge to the above methods is that the deterministic network must be able to map every possible $\mathbf{y}$ close to the corresponding $\mathbf{x}$ in the normal operating range. To achieve this, an input-output pair set $\{\mathbf{y}_i, \mathbf{x}_i\}_{i=1:N}$ representative of all possible deviations is needed for training. Since there are plentiful normal operation data, it is not difficult to obtain training pairs where $\mathbf{y}_i$ is within the normal operating range so that $\mathbf{x}_i = \mathbf{y}_i$. However, it is difficult to acquire training data where $\mathbf{y}_i$ is outside the normal operating range (see Figure 1) for two reasons. First, most of the time, a machine is operating normally; second, the corresponding $\mathbf{x}_i$ is simply unknown. Certain artificial deviations were added to $\mathbf{x}_i$ to form $\mathbf{y}_i$ in the hope of augmenting the training data [2]. However, this is insufficient considering the number of possible deviations combined with the number of sensors.

We employ a Bayesian framework to establish a probabilistic mapping from $\mathbf{y}$ to $\mathbf{x}$, where both $\mathbf{x}$ and $\mathbf{y}$ are random vectors. A Gaussian mixture model (GMM) is used to model the normal operating range of the machine, namely, the probability density function of $\mathbf{x}$. A Gaussian random vector $\boldsymbol{\varepsilon}$ with zero mean and an unknown diagonal covariance matrix $\boldsymbol{\Theta}$ is introduced to model the possible deviation of $\mathbf{y}$ from $\mathbf{x}$ such that:

$$\mathbf{y} = \mathbf{x} + \boldsymbol{\varepsilon}. \tag{1}$$

Our model is able to model different levels of deviations through $\boldsymbol{\Theta}$ which is adaptively estimated for each input $\mathbf{y}$. If the jth sensor value y_j is normal, the corresponding variance Θ_j will be small such that ε_j is close to zero and $x_j \approx y_j$; if a

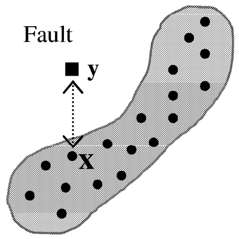


Figure 1. Sensor estimation for machine condition monitoring. The shaded region denotes the normal operating range. Circles are the normal operation data. The square **y** denotes a faulty observation. **x** is the corresponding estimate in the normal operating range.

sensor value y_j is faulty, Θ_j will be large to allow x_j to be different from y_j. The expectation-maximization algorithm [4] offers us a way to estimate both **x** and **Θ** from **y**.

The highlight of our algorithm is that the flexibility of **Θ** eliminates the need for faulty operation training data, as desired by previous methods, and our Bayesian model only requires normal operation training data. Furthermore, there are no ad hoc parameters needed to be preset.

The remainder of this paper is organized as follows. In Section 2, we describe our Bayesian model and how to estimate **x** and **Θ** simultaneously from **y**. We present comparison test results in Section 3. A summary is provided in Section 4.

2. DESCRIPTION OF THE ALGORITHM

2.1. Overview of the algorithm

During the training stage, $P(\mathbf{x})$, the probability density function of **x** is learned from normal operation data, which is described in Section 2.2. During monitoring, in terms of minimizing the mean square error (MSE), the optimal estimate $\mathbf{x}^*$ is given by

$$\mathbf{x}^* = E\,(\mathbf{x} \mid \mathbf{y}, \mathbf{\Theta}), \tag{2}$$

the conditional expectation of **x** given **y**. Since **x** depends on **Θ**, **Θ** is also viewed as one condition. The estimation of **Θ** is elaborated in Section 2.3. For real-time applications, algorithm speedup procedures are provided in Section 2.4.

2.2. Modeling $P(\mathbf{x})$

We use the Gaussian mixture model (GMM) to model the distribution of **x**, since the GMM is capable of modeling complex distributions [5]:

$$P(\mathbf{x}) = \sum_{s=1}^{K} P(\mathbf{x} \mid s) P(s), \tag{3}$$

where s is the label for the sth mixture component. **x** given s has a Gaussian distribution: $\mathbf{x} \mid s \sim N(\mathbf{m}_{\mathbf{x}|s}, \mathbf{C}_{\mathbf{x}|s})$, where $\mathbf{m}_{\mathbf{x}|s}$ and $\mathbf{C}_{\mathbf{x}|s}$ are the mean and covariance of the Gaussian distribution for component s respectively. K is the number of components. The prior probability for a component $P(s)$ is denoted by p_s. $P(\mathbf{x})$ involves parameters: p_s, $\mathbf{m}_{\mathbf{x}|s}$, $\mathbf{C}_{\mathbf{x}|s}$ which are estimated by maximum likelihood estimation (MLE) based on the normal operation training data.

Once $P(\mathbf{x})$ is available, the joint distribution of **x** and **y** given component s is readily computed from (1):

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \Big| s \sim N\left(\begin{bmatrix} \mathbf{m}_{\mathbf{x}|s} \\ \mathbf{m}_{\mathbf{x}|s} \end{bmatrix}, \begin{bmatrix} \mathbf{C}_{\mathbf{x}|s} & \mathbf{C}_{\mathbf{x}|s} \\ \mathbf{C}_{\mathbf{x}|s} & \mathbf{C}_{\mathbf{x}|s} + \mathbf{\Theta} \end{bmatrix} \right). \tag{4}$$

2.3. Estimating x and Θ simultaneously

During monitoring, for an input observation **y**, the goal is to compute (2). However, both **x** and **Θ** are unknown and must be estimated from **y**. We proceed by rewriting (2) as:

$$E(\mathbf{x} \mid \mathbf{y}, \mathbf{\Theta}) = \sum_{s=1}^{K} P(s \mid \mathbf{y}, \mathbf{\Theta}) E(\mathbf{x} \mid \mathbf{y}, s, \mathbf{\Theta}). \tag{5}$$

Every equation on the right side of (5) can be easily computed based on (4) if **Θ** is known.

We resort to the Expectation-Maximization (EM) algorithm [4] for estimation of **x*** and **Θ** simultaneously. The EM is a powerful tool for maximum likelihood estimation in the presence of hidden variables. In our case, **x** is viewed as the hidden variable and **Θ** is the parameter to be estimated. The EM algorithm aims at solving the following MLE problem:

$$\underset{\mathbf{\Theta}}{Max}\, P(\mathbf{y} | \mathbf{\Theta}). \tag{6}$$

We alternate the estimation of **x*** in the E-step and the estimation of **Θ** in the M-step. Suppose that we are now in the nth iteration and the current estimate of **Θ** is denoted by $\mathbf{\Theta}^{(n-1)}$.

In the E-step, the following function is computed:

$$Q\left(\mathbf{\Theta}, \mathbf{\Theta}^{(n-1)}\right) = E\left[log\left(P(\mathbf{y}, \mathbf{x} \mid \mathbf{\Theta})\right) \middle| \mathbf{y}, \mathbf{\Theta}^{(n-1)} \right]. \tag{7}$$

In the M-step, we seek $\mathbf{\Theta}^{(n)}$ which maximizes (7):

$$\mathbf{\Theta}^{(n)} = arg\, \underset{\mathbf{\Theta}}{Max}\, Q\left(\mathbf{\Theta}, \mathbf{\Theta}^{(n-1)}\right). \tag{8}$$

Specifically, we first estimate $\mathbf{\Theta}_s^{(n)}$, the **Θ** estimated solely using the sth Gaussian mixture component:

$$\mathbf{\Theta}_s^{(n)} = diag\left(E\left[(\mathbf{y} - \mathbf{x})(\mathbf{y} - \mathbf{x})^T \middle| \mathbf{y}, s, \mathbf{\Theta}^{(n-1)} \right] \right). \tag{9}$$

Since we assume that **Θ** is a diagonal matrix, we apply *diag*() in (9) to keep the diagonal elements and set the off-diagonal elements to 0. Then the results are combined:

$$\mathbf{\Theta}^{(n)} = \sum_{s=1}^{K} P\left(s \mid \mathbf{y}, \mathbf{\Theta}_s^{(n-1)}\right) \mathbf{\Theta}_s^{(n)}. \tag{10}$$

$\mathbf{x}^* = E\,(\mathbf{x} \mid \mathbf{y}, \mathbf{\Theta}^{(n-1)})$ is just an intermediate result in the E-step while $\mathbf{\Theta}^{(n)}$ is updated in the M-step. The above E and M steps are iterated repeatedly until the algorithm converges (which was guaranteed [4]) and we output the final $\mathbf{x}^*$.

The estimated **Θ** favored by the maximum likelihood estimation requirement in (6) has very nice properties. For y_j which deviates much from x_j, the corresponding Θ_j is large so that the estimated x_j is allowed to be far from y_j and to be located in the normal operating range. On the other hand, for a normal y_j, the corresponding Θ_j is small and x_j is forced to be close to y_j. In contrast to previous methods using the deterministic networks, our algorithm doesn't require any training data containing deviations. The possible deviations are elegantly handled by $\mathbf{\Theta}$, which is adaptive with respect to each input $\mathbf{y}$.

2.4. Algorithm speedup

We now analyze the computational complexity of the above algorithm for each test input $\mathbf{y}$. Assume that the number of iterations of the EM algorithm is $\approx M$. In each iteration, the EM algorithm involves the calculation of the inverse of the covariance matrix $\mathbf{C}_{\mathbf{x}|s} + \mathbf{\Theta}^{(n-1)}$ for each Gaussian component. Thus, the complexity of the above EM algorithm is $O(MKd^3)$. Such complexity obstructs real time monitoring, especially when d (the number of sensors) is large. We propose two ways to speed up the algorithm.

2.4.1 Use of isotropic Gaussian model

We model each mixture component using an isotropic Gaussian model such that $\mathbf{x} \mid s \sim N(\mathbf{m}_{\mathbf{x}|s}, \mathbf{C}_{\mathbf{x}|s} = \sigma^2\mathbf{I}_d)$, where the covariance matrix is the multiplication of a scalar σ^2 and a $d \times d$ identity matrix $\mathbf{I}_d$. σ^2 is the same for all mixture components. Learning $P(\mathbf{x})$ using a mixture of isotropic Gaussian models is also referred to as kernel density estimation [5], which is another popular but simple way to model complex distributions. By doing this, inverse of $\mathbf{C}_{\mathbf{x}|s} + \mathbf{\Theta}^{(n-1)}$ only needs $O(d)$ computational time. This reduces the complexity of the algorithm to $O(MKd)$.

2.4.2 Component selection

During the EM iterations, we often observe that for many mixture components, $P(s \mid \mathbf{y}, \mathbf{\Theta}) \approx 0$ and one can remove the corresponding terms from (5) and (10) without much loss of precision. This motivates us to propose the following component selection algorithm.

During the first iteration of the EM algorithm, we rank the $P(s \mid \mathbf{y}, \mathbf{\Theta}^{(0)})$ for all components in descending order. We select components, starting with the one with the highest $P(s \mid \mathbf{y}, \mathbf{\Theta}^{(0)})$ until the sum of $P(s \mid \mathbf{y}, \mathbf{\Theta}^{(0)})$ of the selected components > 95%. Only the selected components are used in the afterwards EM iterations. In our experience, for $K = 60$, the number of selected components L ranges from 1 to 6. This further reduces the complexity to $\approx O(MLd)$ where $L<<K$.

Through the above two speedup procedures, the proposed method is capable of monitoring real time data with second time resolution.

3. TEST RESULTS

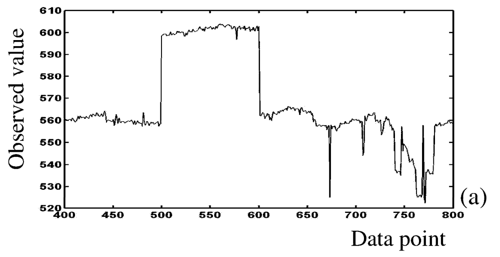
We applied the proposed algorithm to monitor gas turbines. Note that the same methodology is applicable to other devices. The performance of the proposed method abbreviated as GMM (using isotropic Gaussian models) was compared with that of the SVR and the MSET methods. K (the number of Gaussian components) was empirically set to 60 in our tests. There are about 400 sensors installed in each gas turbine under our investigation. 35 sensors which are frequently used in a rule decision system were selected to build our models. They are: power, IGV actuator position, inlet temperature, and 32 blade path temperature (BPTC) sensors.

Real fault cases could be used to test algorithms. However, the normal value of a sensor is unknown and thus it is difficult to evaluate the accuracy of sensor estimation. Thus we consider artificial faults which are simulated by adding deviations to some selected sensors in some time range. By checking the difference (error) between the estimate and the ground truth (original) value of each sensor, we are able to evaluate sensor estimation accuracy quantitatively.

The error measures we use are now introduced. *Estimation error* of a sensor is the absolute difference between the estimate and the ground truth value. We distinguish three types of estimation errors. E_n is the average estimation error of all sensors in the normal time range (without any fault). E_{nf} is the average estimation error of all normal sensors in the faulty time range. E_{ff} is the average estimation error of all faulty sensors in the faulty time range. E_n and E_{nf} indicate the false alarm level while E_{ff} indicates the fault detection sensitivity. Small values are preferred for all these errors. Another way to represent results is to use ROC curves which are mainly used for a detection problem (vs. our estimation problem).

The tests are now detailed. We selected a gas turbine data set, which covers 20 days' normal operation and has a time resolution of 15 minutes. The first 20% (400 data points) of the data set was used for training and the rest for monitoring. Only the original normal training data were used for training the SVR and the MSET. A step fault with a magnitude of 40 was added to two blade path temperature sensors: BPTC6A and BPTC6B between the 500th and the 600th data point. Figure 2a shows one faulty sensor BPTC6A in part of the monitoring time. The mean and standard deviation were calculated for each of the 35 sensors using the training data and were applied to normalize both training and test data.

Figure 2b shows the residue (the deviation between the observed value and the estimate, i.e. $\mathbf{y} - \mathbf{x}$) results of a normal sensor BPTC1B for all three algorithms. For a normal sensor, its residue must be small to avoid false alarms in the rule-based decision step. It is clear that the residues from the SVR and the MSET are much larger than those from our GMM algorithm. It is worth noting that during the faulty time range (between the 500th and 600th data point), the SVR and MSET produced even higher



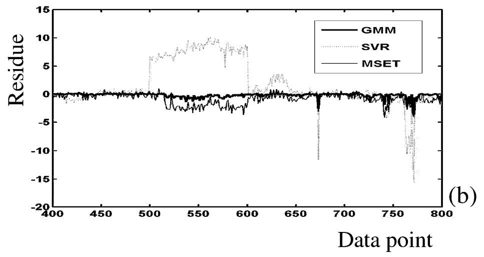
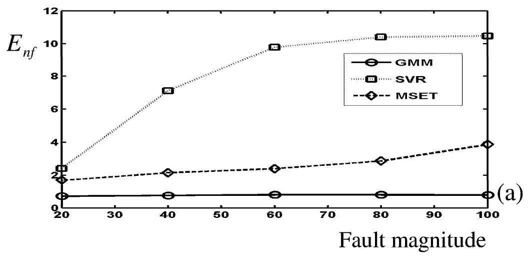


Figure 2. An example of sensor estimation. (a) A faulty sensor BPTC6A, where a step fault with a magnitude of 40 occurred between the 500th and 600th data point. (b) The residues of a normal sensor BPTC1B produced by different algorithms. The residues from our GMM algorithm are the smallest (and the best), which are not affected by the deviation of the faulty sensors .



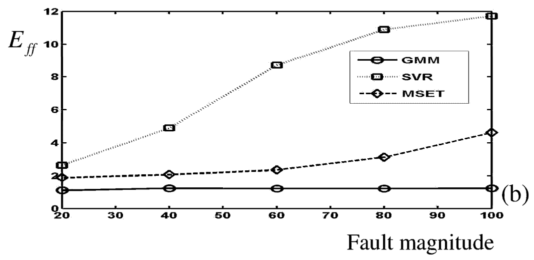


Figure 3. Estimation errors for different algorithms. (a) E_{nf} vs. the step fault magnitude (b) E_{ff} vs. the step fault magnitude. Step fault magnitude was varied from 20 to 100 in increments of 20. Our proposed method GMM produced the lowest estimation errors in both measures.

residues. This is attributed to the fact that such a faulty input **y** cannot be mapped accurately via a deterministic network trained only on a normal training set (as we noted in Section 1).

To test how different algorithms respond to different levels of deviations, we varied the magnitude of the step fault from 20 to 100 in increments of 20 and repeated the above test. Our GMM algorithm produced the smallest estimation error $E_n = 0.60$, followed by 1.24 (MSET) and 1.70 (SVR). Note that E_n doesn't change with fault magnitude. Figure 3 shows the estimation error E_{nf} and E_{ff} for different algorithms vs. the fault magnitude. Our GMM algorithm produced the smallest errors which are relatively constant regardless of the fault magnitude. This confirms that the estimated covariance matrix $\mathbf{\Theta}$ correctly handled different levels of deviations. However, with the increase of fault magnitude, the SVR and MSET produced larger errors since the faulty test input becomes even less represented by the normal training set.

4. SUMMARY

A Bayesian sensor estimation algorithm is presented. A Gaussian random vector $\boldsymbol{\varepsilon}$ is introduced to model the deviation of the observation **y** from the corresponding normal values **x**. The covariance matrix $\mathbf{\Theta}$ of $\boldsymbol{\varepsilon}$ is adaptively estimated together with **x** via the EM algorithm. Test results show the advantage of the new algorithm over previous methods.

The proposed algorithm has its applications to general multivariate signal (e.g. images) reconstruction and denoising. Our work can be extended by using temporal dependency among data points.

REFERENCES

[1] J. W. Hines, A. Gribok and B. Rasmussen (2001), "On-Line Sensor Calibration Verification: A Survey", International Congress and Exhibition on Condition Monitoring and Diagnostic Engineering Management.

[2] D. Wrest, J. W. Hines, and R. E. Uhrig (1996), "Instrument Surveillance and Calibration Verification Through Plant Wide Monitoring Using Autoassociative Neural Networks", the 1996 American Nuclear Society International Topical Meeting on Nuclear Plant Instrumentation, Control and Human Machine Interface Technologies, May 6-9.

[3] A. V. Gribok, J. W. Hines and R. E. Uhrig (2000), "Use of Kernel Based Techniques for Sensor Validation in Nuclear Power Plants", International Topical Meeting on Nuclear Plant Instrumentation, Controls and Human-Machine Interface Technologies.

[4] A. P. Dempster, N. M. Laird and D. B. Rubin (1977), "Maximum-likelihood from Incomplete Data via the EM Algorithm", Journal of the Royal Statistical Society, Series B, 39, pp.1-38.

[5] R. O. Duda, P. E. Hart and D. G. Stork, "Pattern Classification", 2nd Ed. Wiley Interscience, 2001.