Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)

Unsupervised Machine Condition Monitoring Using Segmental Hidden Markov Models

Chao Yuan
Siemens Corporation, Corporate Technology, Princeton, NJ 08540
yuanchao@yahoo.com

Abstract

The task of machine condition monitoring is to detect machine failures at an early stage such that maintenance can be carried out in a timely manner. Most existing techniques are supervised approaches: they require user annotated training data to learn normal and faulty behaviors of a machine. However, such supervision can be difficult to acquire. In contrast, unsupervised methods don't need much human involvement, however, they face another challenge: how to model the generative (observation) process of sensor signals. We propose an unsupervised approach based on segmental hidden Markov models. Our method has a unifying observation model integrating three pieces of information that are complementary to each other. First, we model the signal as an explicit function over time, which describes its possible non-stationary trending patterns. Second, the stationary part of the signal is fit by an autoregressive model. Third, we introduce contextual information to break down the signal complexity such that the signal is modeled separately under different conditions. The advantages of the proposed model are demonstrated by tests on gas turbine, truck and honeybee datasets.

1 Introduction

We address the problem of machine condition monitoring by analyzing sensor (e.g., temperature and pressure) time series. This can be formulated as a classification problem to distinguish between normal class and each of the failure classes. Supervised machine learning techniques use data annotated by human experts to train their models [Sarkar *et al.*, 2012; Quinn *et al.*, 2009; Smyth, 1993]. However, user annotation can be difficult to acquire. Unsupervised techniques, on the other hand, are more user friendly. They use all data available without annotation by performing segmentation and classification, simultaneously [Eskin, 2000; Antoniadou *et al.*, 2015].

One big challenge for unsupervised approaches is how to model sensor signals or design the observation model. First, hidden Markov models (HMM) [Rabiner, 1989; Smyth, 1993; Parson *et al.*, 2014] and its extension hidden semi-Markov models (HSMM) [Yu, 2010; Johnson and Willsky, 2014] use time implicitly between states. Segmental HMMs (segHMM) improve HSMM by modeling observation explicitly as a function over time [Kim and Smyth, 2006]. This feature fits condition monitoring nicely, because a fault usually develops over time following a trending pattern. Second, previous observations that are often correlated with current observation can be used to predict the latter. This motivates the vector autoregressive HMM models (VAR-HMM) [Lütkepohl, 1991] and its extensions [Fox *et al.*, 2009; Jiang *et al.*, 2012; Chang *et al.*, 2014]. Finally, using contextual information is common in condition monitoring and anomaly detection [Song *et al.*, 2007; Hayes and Capretz, 2014; Valko *et al.*, 2011]. This technique divides all observed variables into two groups: a *contextual* group consisting of input signals to the machine, and a *behavioral* group consisting of output signals of the machine. Prediction is made only for output variables based on input variables.

The above three pieces of information are complementary to each other. Using contextual information helps to break down the signal complexity, because instead of building one single complex model for all variables, we only need to learn a different simple model for output variables under each specific condition (context). Using a function over time and using autoregression information are essentially modeling the non-stationary and stationary parts of the signal, respectively. Despite all the above advances, to the best of our knowledge, we haven't seen a work integrating all above pieces of information in machine condition monitoring.

In this paper, we propose an unsupervised machine learning approach for condition monitoring. We improve the segmental hidden Markov models by a unified observation model with three components. The first component explicitly uses time, as in the original segHMM, to describe the possible trending pattern. The second autoregressive component, as in an AR-HMM, is intended to depict possible temporal dynamics. The last contextual component uses information from input sensors. By unifying these three pieces of important information, we are able to achieve better unsupervised segmentation results for sensor time series. The rest of this paper is organized as follows. In Sect.2, we review previous work. In Sect.3, the proposed approach is described. We present test results in Sect.4 and conclude this paper in Sect.5.

2 Related work

Condition monitoring has been applied to a wide range of machines and applications including aircraft turbine engines [Menon *et al.*, 2003; Clifton *et al.*, 2008; Sarkar *et al.*, 2012], wind turbines [Antoniadou *et al.*, 2015], milling machines [Geramifard, 2013], motor bearings [Marwala, 2012], antenna systems [Smyth, 1993], hydraulic pumps [Dong, 2008], home appliances [Parson *et al.*, 2014] and computers [Eskin, 2000]. A variety of algorithms are used, for example, Gaussian mixture models, self-organizing maps, Markov models, switching Kalman filtering, neural networks, support vector machines, Gaussian process. Most of them are supervised approaches, requiring training data for the normal class and sometimes each of the failure classes. Only few work such as [Eskin, 2000; Antoniadou *et al.*, 2015] proposes unsupervised methods. However, they usually address a two-class anomaly detection problem, distinguishing normality from abnormality. A more detailed discussion about anomaly detection can be found in [Chandola *et al.*, 2009; Pimentel *et al.*, 2014]. In contrast, this paper tackles unsupervised multi-class machine condition monitoring problems.

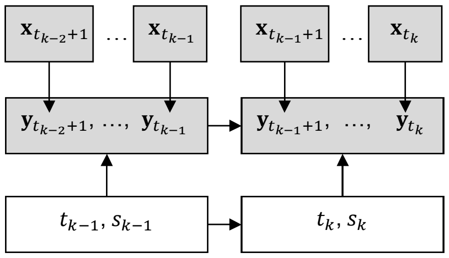
In change point detection, time series are only segmented without classification, because a state is exclusively used for one segment [Xuan and Murphy, 2007; Saatçi *et al.*, 2010]. Time series classification [Sarkar *et al.*, 2012; Parson *et al.*, 2014] contrarily performs classification by assigning a class label to the entire sequence without segmentation. In comparison, we perform segmentation and classification simultaneously, as required by condition monitoring. This is also similarly done in [Hoai and Torre, 2012; Zhou *et al.*, 2013; Hoai *et al.*, 2011; Oh *et al.*, 2008; Quinn *et al.*, 2009; Fox *et al.*, 2009; Jiang *et al.*, 2012].

For many time series segmentation algorithms with a linear observation model [Rabiner, 1989; Yu, 2010; Johnson and Willsky, 2014; 2013] including ours, the learning is typically done using the Expectation-Maximization (EM) algorithm [Dempster *et al.*, 1977]. Markov Chain Monte Carlo (MCMC) is also used in [Oh *et al.*, 2008; Fox *et al.*, 2009; Johnson and Willsky, 2013], but is much more time consuming. Nonlinear models, which use kernel functions to describe the similarity of observations in different segments, are learned by quadratic programming [Hoai *et al.*, 2011] or coordinate descent [Zhou *et al.*, 2013]. These algorithms are also very expensive.

3 The proposed methodology

3.1 Problem definition

Suppose that we have an observation time series $\mathbf{y}_{1:T} = \mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_T$ for a machine. The time stamp t starts from 1 and ends at T. $\mathbf{y}_t$ is a D-dimensional sensor vector consisting of values from D sensors at time t. Our objective is to partition this time series into K consecutive and non-overlapping segments denoted by $\{t_{1:K}, s_{1:K}\}$. t_k indicates that the k-th segment with a state label $s_k = i$ ends at time t_k, where $i = 1, 2, ..., M$ and $k = 0, 1, ..., K$. M indicates the total number of possible states. Another way to represent a partition is simply using a sequence of labels z_t for every time stamp t such that $z_t = s_k$ if t belongs to the k-th segment. This latter representation is useful when we evaluate or visualize the partition results.



$\mathbf{x}_{t_{k-2}+1}$ ... $\mathbf{x}_{t_{k-1}}$
$\mathbf{x}_{t_{k-1}+1}$ ... $\mathbf{x}_{t_k}$
$\mathbf{y}_{t_{k-2}+1}, \dots, \mathbf{y}_{t_{k-1}}$
$\mathbf{y}_{t_{k-1}+1}, \dots, \mathbf{y}_{t_k}$
t_{k-1}, s_{k-1}
t_k, s_k

Figure 1: The proposed model. The gray nodes indicate known variables and white nodes indicate hidden variables.

3.2 Model description

Fig.1 shows the graphical representation of our proposed model. It is a snapshot showing how segment $k-1$ and k are related to each other. Our model consists of four major components.

The initial state model indicates the initial state probability at time 1

$$P(s_1 = i) = \pi_i. \tag{1}$$

The transition model indicates the transition probability from state s_{k-1} to the next state s_k:

$$P(s_k = j | s_{k-1} = i) = A_{ij}. \tag{2}$$

Note that self-transition is not possible $P(s_k = i | s_{k-1} = i) = A_{ii} = 0$, because in that case there is only one segment instead of two. We note that some previous work [Kim and Smyth, 2006] specifies a fixed order of state transition (e.g., state 1 can only switch to state 2). In contrast, we allow arbitrary transitions between states.

The duration model specifies how long a state persists

$$P(t_k | t_{k-1}, s_k = i) = \frac{\exp(-\lambda_i) \lambda_i^{t_k - t_{k-1} - 1}}{(t_k - t_{k-1} - 1)!}. \tag{3}$$

In this work, we consider a poisson distribution. Other choices include geometric distribution and Gaussian distribution [Yu, 2010].

The observation model in general, assumes that the observation given a state follows a Gaussian distribution:

$$P(\mathbf{y}_{t_{k-1}+1:t_k} | t_{k-1}, t_k, s_k = i) = \prod_{t=t_{k-1}+1}^{t_k} \mathcal{N}(\mathbf{y}_t | \mathbf{f}_i(.), \mathbf{V}_i), \tag{4}$$

where $\mathbf{f}_i(.)$ is mean and $\mathbf{V}_i$ is covariance. The main differences of previous approaches lies in the specification of function $\mathbf{f}_i(.)$. For most HMM or HSMM models, $\mathbf{f}_i(.)$ is assumed to be independent of time t (although $\mathbf{f}_i(.)$ can depend on duration $t_k - t_{k-1}$ [Yu, 2010]). The previous segHMM improves this by defining $\mathbf{f}_i(t) = \mathbf{a}_i + \mathbf{b}_i t$ as a function of t [Kim and Smyth, 2006] or $\mathbf{f}_i(t) = \mathbf{a}_i + \mathbf{c}_i \mathbf{x}_t$ as a function of an input variable (covariate) $\mathbf{x}_t$ [Chaubert-Pereira *et al.*, 2010].

We consider a unified observation model defined as follows

$$\mathbf{f}_i(t_{k-1}, t, \mathbf{y}_{t-1:t-Q}, \mathbf{x}_t) = \boldsymbol{\theta}_i \mathbf{h}$$
$$= \sum_{r=0}^{R} \mathbf{a}_{i,r}(t - t_{k-1})^r + \sum_{q=1}^{Q} \mathbf{b}_{i,q}\mathbf{y}_{t-q} + \sum_{l=1}^{L} \mathbf{c}_{i,l} x_{t,l}. \quad (5)$$

Our model consists of three parts. The first part is a polynomial function with a order of R over time, the second part represents an autoregressive component with a order of Q and the last part depends on a L-dimensional input variable $\mathbf{x}_t$. $\mathbf{h} = [1 \ldots (t - t_{k-1})^R \; \mathbf{y}'_{t-1} \ldots \mathbf{y}'_{t-Q} \; x_{t,1}, \ldots, x_{t,L}]'$ is a column vector with all known information at time t and $\boldsymbol{\theta}_i = [\mathbf{a}_{i,0} \ldots \mathbf{a}_{i,R} \; \mathbf{b}_{i,1} \ldots \mathbf{b}_{i,Q} \; \mathbf{c}_{i,1}, \ldots, \mathbf{c}_{i,L}]$ contains all linear coefficients to be estimated. Note that $\boldsymbol{\theta}_i$ is a matrix whose row dimension is equal to D, the dimension of $\mathbf{y}_t$. The autoregressive part is also referred to as vector autoregressive model (VAR) [Lütkepohl, 1991]. We will not distinguish AR and VAR for the rest of this paper.

This unified observation model is the highlight of this paper as we fuse information from time, previous observations and input variables. Using input variables for condition monitoring and anomaly detection is a common practice [Song *et al.*, 2007; Hayes and Capretz, 2014; Valko *et al.*, 2011]. For example, for a gas turbine, gas flow, inlet guide vane (IGV) position (that controls the air flow) and inlet temperature can all be viewed as input sensors $\mathbf{x}_t$. They represent the inputs to the system and determine the values of other output sensors $\mathbf{y}_t$ such as power, compressor temperatures and pressures, which are the main indicators of the performance and status of a gas turbine. The autoregressive part involving previous observations captures the dynamics of a machine, which provides extra information missed by using input variables alone. Finally, a failure often develops over time and exhibits a trending pattern. This is covered by the polynomial function over time in our model. If we exclude this polynomial component, to depict a trending pattern, we may have to use multiple states, one for each stage of the trend. Due to the additive nature of our model, we can easily add a new component if extra information is available or remove an existing component if it is not needed. This makes our model very flexible.

We have assumed a linear relationship between the parameters $\boldsymbol{\theta}_i$ and constants $\mathbf{h}$. The magnitude of the coefficient for a component indicates the importance of this component to the observation. This makes our model very interpretable. It is not difficult to extend $\mathbf{f}_i(.)$ to a nonlinear function. For example, if we consider a Gaussian process [Rasmussen and Williams, 2006] with Gaussian kernel functions, all linear coefficients can be converted into the length scale parameters in the Gaussian kernel. However, in that case, the factorized form of (4) doesn't hold any more, therefore, the training complexity will be higher.

3.3 Training

During training, we are usually given N sequences. The n-th sequence consists of an output variable sequence $\mathbf{Y}_n = \mathbf{y}^{(n)}_{1:T_n}$ and an input variable sequence $\mathbf{X}_n = \mathbf{x}^{(n)}_{1:T_n}$. The task of training is to learn the parameters including initial state probability π_i, transition probability A_{ij}, duration parameter λ_i, observation model coefficients $\boldsymbol{\theta}_i$ and covariance $\mathbf{V}_i$, where $i, j = 1, 2, \ldots, M$. Let $\boldsymbol{\Psi} = \{\boldsymbol{\pi}, \mathbf{A}, \boldsymbol{\lambda}, \boldsymbol{\theta}, \mathbf{V}\}$ denote all parameters. We employ the Expectation-Maximization algorithm [Dempster *et al.*, 1977] to learn $\boldsymbol{\Psi}$ by maximizing the log likelihood of all sequences $\log P(\mathbf{Y}_{1:N}, |\mathbf{X}_{1:N}, \boldsymbol{\Psi})$.

In the E-step, based on the current parameter estimation, we compute the following posterior distribution

$$\alpha(t_k^{(n)}, s_k^{(n)}) = P(\mathbf{y}^{(n)}_{1:t_k^{(n)}}, s_k^{(n)} \text{ ends at } t_k^{(n)}) \quad (6)$$
$$= \sum_{t_{k-1}^{(n)} < t_k^{(n)}, s_{k-1}^{(n)}} \alpha(t_{k-1}^{(n)}, s_{k-1}^{(n)}) P(\mathbf{y}_{t_{k-1}^{(n)}+1:t_k^{(n)}} | s_k^{(n)})$$
$$P(t_k^{(n)} | t_{k-1}^{(n)}, s_k^{(n)}) P(s_k^{(n)} | s_{k-1}^{(n)})$$

$$\beta(t_{k-1}^{(n)}, s_{k-1}^{(n)}) = P(\mathbf{y}^{(n)}_{t_{k-1}^{(n)}+1:T_n} | s_{k-1}^{(n)} \text{ ends at } t_{k-1}^{(n)}) \quad (7)$$
$$= \sum_{t_k^{(n)} > t_{k-1}^{(n)}, s_k^{(n)}} \beta(t_k^{(n)}, s_k^{(n)}) P(\mathbf{y}^{(n)}_{t_{k-1}^{(n)}+1:t_k^{(n)}} | s_k^{(n)})$$
$$P(t_k^{(n)} | t_{k-1}^{(n)}, s_k^{(n)}) P(s_k^{(n)} | s_{k-1}^{(n)})$$

The above procedure is often referred to as forward-backward algorithm [Rabiner, 1989; Yu, 2010]. All probabilities involved in (6) and (7) are defined in (1-4). It is important to view k as any segment (instead of the k-th segment) and $k-1$ as its previous segment, because all parameters are not dependent on the the actual segment number (but only on its associated state s_k). By integrating out the actual number of segments, we also reduce the complexity of the algorithm.

Once $\alpha(t_k^{(n)}, s_k^{(n)})$ and $\beta(t_k^{(n)}, s_k^{(n)})$ are in place, it is straightforward to compute

$$P(t_{k-1}^{(n)}, s_{k-1}^{(n)}, t_k^{(n)}, s_k^{(n)} | \mathbf{Y}_n) \quad (8)$$
$$= P(s_{k-1}^{(n)} \text{ ends at } t_{k-1}^{(n)}, s_k^{(n)} \text{ ends at } t_k^{(n)} | \mathbf{Y}_n)$$
$$\propto \alpha(t_{k-1}^{(n)}, s_{k-1}^{(n)}) P(\mathbf{y}^{(n)}_{t_{k-1}^{(n)}+1:t_k^{(n)}} | s_k^{(n)})$$
$$P(t_k^{(n)} | t_{k-1}^{(n)}, s_k^{(n)}) P(s_k^{(n)} | s_{k-1}^{(n)}) \beta(t_k^{(n)}, s_k^{(n)}).$$

Based on (8), we will compute posterior distributions that are directly used in the M-step.

$$P(t_{k-1}^{(n)}, t_k^{(n)}, s_k^{(n)} | \mathbf{Y}_n) = \sum_{s_{k-1}^{(n)}} P(t_{k-1}^{(n)}, s_{k-1}^{(n)}, t_k^{(n)}, s_k^{(n)} | \mathbf{Y}_n). \quad (9)$$

$$P(s_{k-1}^{(n)}, s_k^{(n)} | \mathbf{Y}_n) = \sum_{t_{k-1}^{(n)} < t_k^{(n)}} P(t_{k-1}^{(n)}, s_{k-1}^{(n)}, t_k^{(n)}, s_k^{(n)} | \mathbf{Y}_n). \quad (10)$$

(9) is used to estimate the parameters for the duration model ($\boldsymbol{\lambda}$ in (3)) and observation model ($\boldsymbol{\theta}$ and $\mathbf{V}$ in (4)). (10) is used to estimate the parameters for the initial state model ($\boldsymbol{\pi}$ in (1)) and transition model ($\mathbf{A}$ in (2)). All parameters are estimated in analytical forms.

Pruning. We start the EM iterations with $M = 10$ states (obtained by randomly sampling training sequences). During EM iterations, we will prune a state if it is not supported by the data. Specifically, if a state i has a very low accumulative posterior probability $\sum_{n=1}^{N} \sum_{t=1}^{T_n} P(z_t^{(n)} = i|\mathbf{Y}_n) < 0.1$, we will prune this state so the M can be reduced gradually. We found that using Poisson distribution for the duration model helps pruning (or merging of states). We initialize λ_i to being the average sequence length. This gives a broad prior for the duration such that the state whose observation model fits better will become more dominant. In contrast, using geometric distribution in the duration model doesn't have this property, because geometric distribution peaks at short duration and thus encourages shorter state (segment). For similar reasons, HMM doesn't have this nice property, either.

Numerical stability. The above algorithm as it often suffers numerical stability problems. The reason is that the EM iterations often involve multiplication of several exponential terms. If two such terms become extremely small or large, we may have an underflow or overflow problem, respectively. To overcome this, we compute both $\alpha(.)$ and $\beta(.)$ in the log domain. The multiplication of two exponential terms $\exp(u)$ and $\exp(v)$ is simply $u + v$ in the log domain. For adding two exponential terms, we take the maximum of u and v, which for example is u, and the result will be $u + \log(1 + \exp(v - u))$. This can be easily extended to operations involving more than two exponential terms. Once $\log P(t_{k-1}^{(n)}, s_{k-1}^{(n)}, t_k^{(n)}, s_k^{(n)}|\mathbf{Y}_n)$ is obtained in this way, we convert it back into the $P(t_{k-1}^{(n)}, s_{k-1}^{(n)}, t_k^{(n)}, s_k^{(n)}|\mathbf{Y}_n)$ to complete the remaining EM computations.

Complexity. The complexity of training is $\mathcal{O}(NT^2M^2)$, where N is number of sequences, T is the maximum length of all sequences and M is the number of states. This is more expensive than $\mathcal{O}(NTM^2)$, the cost of training a HMM.

3.4 Monitoring

Once our model is trained, the standard way to find the best partition of a sequence is by using the Viterbi algorithm [Rabiner, 1989; Yu, 2010]. The complexity is $\mathcal{O}(T^2M^2)$ and it can be done online without iteration. We note that for real-time application, the most recent time t may not be the end of a segment as we have assumed in training. This situation is often referred to being right censored [Yu, 2010], in which we have to consider all possible ending point $\geq t$ for the current segment. The duration model will play a larger role in this situation, because we only have observations up to t. This will demand more training data to assure an accurate duration model. For our applications, relying more on the observation model than the duration model is more reasonable and thus we still assume that the current segment ends at t.

4 Test results

Under the unsupervised setting, many existing condition monitoring algorithms that require labels for training cannot be used. Therefore, we focus on comparing unsupervised HMM-based methods that are widely used and achieve impressive performances. Many such methods can be represented by our proposed unifying model. They differ by which information do we use in the observation component in (5). Let "HSMM", "t", "AR", "C" denote hidden semi-Markov model, using time component, using autoregressive component and using contextual component, respectively. Thus, in a t-AR-HSMM model, only time and autoregressive parts are used, i.e., $\mathbf{c}_{i,l}$ in (5) is preset to zero. In a HSMM model, only $\mathbf{a}_{i,0}$ in (5) is used. We can similarly define other possible combinations. Note that t-HSMM is equivalent to segHMM. Our model can also be easily converted into hidden Markov models (HMM) by removing the duration model (thus no time component) and allowing self transition. For example, C-HMM means that only contextual part is used in the observation model.

Note that if a method, either HMM-based or HSMM-based, does not use input variables, we will append input variables $\mathbf{x}_t$ to the end of $\mathbf{y}_t$. By doing this, different models are using the same data, but in a different way, either conditionally or jointly. All above models are implemented in matlab. For all tests conducted, we set the order of the polynomial function $R = 1$ and order of the autoregressive component $Q = 1$. The initial number of states is set to $M = 10$.

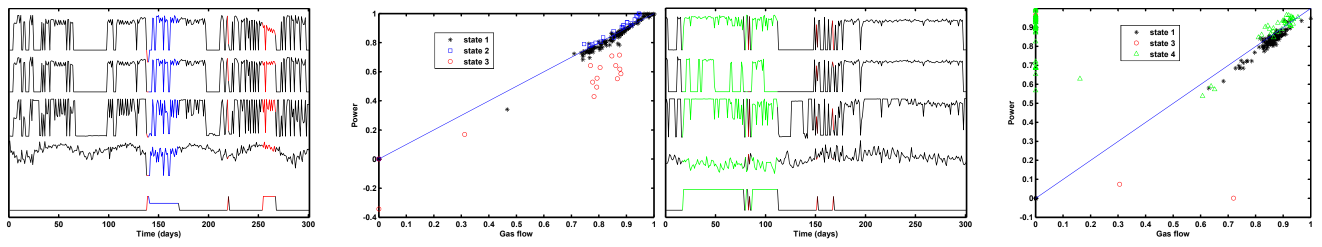
We conduct several tests to evaluate the performance of the proposed algorithm. These include two gas turbine data sets (from Siemens Energy), a truck data set (from Siemens Mobility) and the honeybee data set [Oh *et al.*, 2008]. For test cases where we have ground truth label for every time stamp z_t^*, we can evaluate the performance of an algorithm quantitatively, as similarly done by other unsupervised techniques [Fox *et al.*, 2009]. First, we convert the best partition obtained after the Viterbi algorithm into a label sequence z_t. However, the label may not match that of z_t^* even if they correspond to the same segment. Therefore, for each ground truth label i, we find the corresponding label j produced by an algorithm that has the most overlap with i and map j to i. We repeat this for all ground truth labels. Note that this remapping doesn't change the segmentation result but only re-order its states. Finally, we define the classification rate P_c as $P_c = \frac{\sum_{t=1}^{T} \delta(z_t - z_t^*)}{T}$, where $\delta(z_t - z_t^*) = 1$ if $z_t = z_t^*$ and zero otherwise. Unmatched labels either from ground truth or results all lead to classification errors.

4.1 Gas turbine performance data

We consider six sequences from power generation. Each sequence represents a different gas turbine with about 10 months of operation data. The data resolution is one day. We focus on the power sensor as the output sensor $\mathbf{y}_t$ ($D = 1$), because it is the most important performance indicator for a gas turbine. $L = 3$ input sensors including gas flow, IGV position and inlet temperature (Fig.2) are used.

Since we intend to learn behaviors of different machines jointly, data normalization is needed. One standard approach is to normalize each sensor to zero mean and unit variance. We instead normalize data by dividing each sensor over its corresponding maximum value. By doing this, we maintain the physical meaning of zero, which indicates a machine shutdown and this is true for all machines. If we use the standard normalization method, zero is likely to be converted to a negative value for shutdown period and can be different for different machines.

For this data set, we don't have ground truth labels. Our goal is to apply our t-AR-C-HSMM to blindly partition all six sequences, and find states indicating outliers and performance differences, shared through all units. After the EM algorithm converges, only $M = 4$ states remain while the other 6 are



(a) (b) (c) (d)

Figure 2: Gas turbine performance data and results. (a) displays the output sensor (power) and three input sensors (gas flow, IGV position, inlet temperature) and our partition result for the first gas turbine, respectively. (b) shows the power vs. gas flow plot for the first gas turbine. (c) and (d) show similar contents for the fourth gas turbine. A total of four states are found and shared by all six machines. They are denoted by different colors and symbols.

State i	$a_{i,0}$	$a_{i,1}$	$b_{i,1}$	$c_{i,1}$	$c_{i,2}$	$c_{i,3}$
1	0.002	0.000	0.005	0.954	0.015	-0.007
2	-0.008	0.000	0.012	1.040	0.055	-0.050
3	0.114	-0.029	-0.051	0.180	0.734	0.010
4	0.535	-0.000	0.000	-0.000	0.492	-0.226

Table 1: Learned coefficients of the observation model for the gas turbine performance data. The second through the sixth columns correspond to the coefficients of the time component, autoregressive component and contextual components, respectively.

Gas turbine	state 1	state 2	state 3	state 4
1	0.847	0.100	0.053	0.000
2	0.963	0.027	0.010	0.000
3	0.037	0.957	0.006	0.000
4	0.694	0.000	0.010	0.296
5	0.199	0.000	0.000	0.801
6	0.944	0.000	0.056	0.000

Table 2: Percentage of states for gas turbine performance data. For example, Gas turbine one is in state 1 in 0.847 of its operation time.

pruned by our algorithm automatically. Table 1 shows all estimated observation coefficients. Gas flow appears to have the most impact on power because its corresponding coefficient ($c_{i,1}$ in the fifth column) is the highest for the first two states. Autoregressive coefficient ($b_{i,1}$ in the fourth column) is the lowest, implying that power is quite independent from its previous value. This can be attributed to the low resolution we use: after one day, little dynamics is left.

Figs.2(a) and 2(c) show sensor signals and our partition results for the first and fourth gas turbines, respectively. To make more sense out of the partition result, we show Figs.2(b) and 2(d) as well, where power is plotted against gas flow and different states are marked by different colors and symbols. This is a nice view to indicate the performance of a gas turbine. Typically, a higher power is desired given the same gas flow. A diagonal line is drawn to better visualize the performance difference. State 1 (black stars) appears to represent the most dominant mode (also see Fig.2(a)) of this gas turbine when the performance is slightly lower than the diagonal line. This is confirmed by its corresponding $c_{1,1} = 0.954$ ($c_{1,1} = 1$ representing the diagonal line) in Table 1. State 2 (blue squares) represents a higher efficiency with most data points above the diagonal line because of its $c_{2,1} = 1.040$.

State 3 clearly indicates an under-performing case. Power is much less correlated with gas flow and most data points are well below the diagonal line (red circles in Fig.2(b)). In addition, the first order polynomial coefficient $a_{3,1} = -0.029$ captures the downward trending of power around day 260 in the first plot of Fig.2(a). State 3 also covers some outlier time stamps when power is negative. This is unrealistic and could be due to sensor reading error.

State 4 is not shown in the first gas turbine, but has a significant presence in the fourth gas turbine (green curves or triangles in Figs.2(c) and 2(d)). A key characteristic of state 4 is that we have power even when gas flow is zero (e.g., day 60 in Fig.2(c)). This again is likely due to sensor reading error. However, even when gas flow reading is normal (e.g., day 40 in Fig.2(c)), that data point is still classified as state 3. This can be attributed to the duration model that has a tendency to merge small segments into a big one if the resulting model error is tolerable.

Table 2 shows the percentage of four states in each of the six gas turbines. Based on the above analysis, we can conclude that the third gas turbine is the most efficient, because it has the most presence (0.957) of the most efficient state 2. The second gas turbine is mostly operating in the slightly less efficient state 1. Both the first and sixth gas turbine have about 0.05 of their time running in the very inefficient state 3. Therefore, a maintenance inspection should be suggested. The data from the fifth gas turbine appear to have many sensor reading errors because 0.801 of its time is in the unrealistic state 4. So a data cleaning is recommended.

4.2 Gas turbine failure data

In this section, we use gas turbine data containing a common failure due to cracks in the blade path component. This is usually reflected by a drop of the blade path temperatures. Typically, multiple (in our case eight) temperature sensors are installed in different locations of the blade path. The sensor closest to the crack will show the most obvious symptom and thus guide engineers to inspect that particular location. In this data set, there is one sequence with a resolution of 45 minutes. The beginning part of the sequence is normal and the failure shows in the last part. Therefore, we have ground truth with two states, one for normality and one for the failure.

We use $D = 8$ output sensors, all blade path temperature sensors as noted above and $L = 3$ input sensors including power, inlet temperature and the average of all output sensors. The last input variable is a calculated value, which is widely

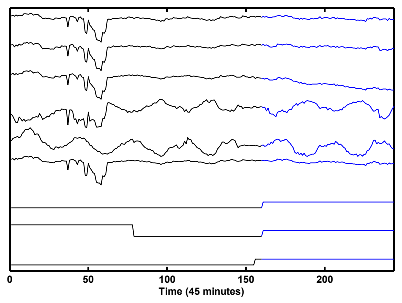


Figure 3: Gas turbine failure data and results. The first six plots shows the three output variables and three input variables. The last three plots show the ground truth partition, the partition found by t-AR-HSMM model and t-AR-C-HSMM model, respectively.

used in power generation. The basic assumption is that even if one temperature has deviation, the average temperature should still be stable and can be used to represent the normal behavior.

Fig.3 shows three output sensors and three input sensors. The third output sensor carries the failure symptom of a downward trending pattern at the end of the period. The ground truth labels, the partition results from a t-AR-HSMM and our model are also shown at the bottom. Classification rates for all models are shown in the third row of Table 3. Our model achieves the highest 0.980. We detect the failure slightly earlier than the ground truth annotated by the human expert (Fig.3), which may be more preferable. The t-AR-HSMM model creates an extra state at the begin of the sequence, which leads to a poor $P_c = 0.680$. This can be attributed to the following. Without using contextual component, the first part of the sequence (especially around day 60 in Fig.3) does look different from the rest. However, in the context of input sensors, the output sensors still follow the same state. This shows the benefit of using contextual information.

4.3 Truck data

We use operation data from a truck. We consider two types of failures. The first type is caused by the failure of the exhaust filter, which results in a hike in exhaust air pressure. The second type is due to blocking of the grill, which leads to an increase of engine intake manifold temperature. Therefore, we use $D = 2$ input sensors including the exhaust air pressure and engine intake manifold temperature to identify these two failures. $L = 7$ other sensors including charge air cooler temperature, road speed, throttle, engine load, cooling temperature, ambient air temperature and altitude are used as input sensors.

There are $N = 3$ sequences used in this test, representing normality (ground truth state 1), exhaust filter failure (ground truth state 2), grill failure (ground truth state 3), respectively. Therefore, there is only one segment in each sequence. Each sequence has a resolution of one minute. Fig.4 shows all three sequences with two output sensors and the first four input sensors. The exhaust filter failure appears to be apparent: when it occurs, the exhaust air pressure (first plot in Fig.4(b)) is much higher than that in normal condition (first plot in Figs.4(a) and 4(c)). The grill blocking failure seems to be more subtle, because only a slight increase in engine intake manifold temperature is observable (second plot in Fig.4(c) compared to those in Figs.4(a) and 4(b)).

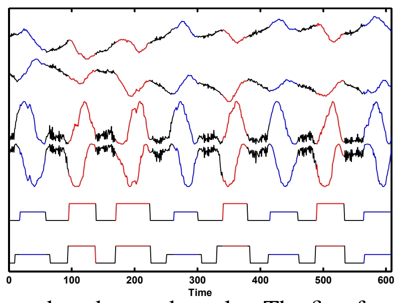


Figure 5: Honeybee data and results. The first four plots shows the honeybee's x, y coordinates and $sine$, $cosine$ of its head angle. The last two plots show the ground truth partition, the partition found by our AR-HSMM model, respectively.

Our model and most other HSMM-based models are able to get perfect $P_c = 1.000$ for all three sequences (last three rows in Table 3). However, we note that t-HSMM gets $P_c = 0.000$ for the first sequence. A detailed check turns out that this model mis-classifies all data points in the first sequence to state 3. This suggests that using more evidence such as contextual information can increase modeling accuracy.

4.4 Honeybee data

This dataset has $N = 6$ sequences, each with $D = 4$ signals indicating the honeybee's x, y coordinates and $sine$, $cosine$ of its head angle. Fig.5 shows all four signals of a sample sequence. There are three states to be identified: waggle, left turn and right turn, denoted by black, blue and red, respectively. This data set is not from machine condition monitoring field, but is well studied by many time series segmentation algorithms.

Since there is no trending pattern within a state and no concept of contextual variables, we use AR-HSMM as our model. There are two settings to conduct the test. The first setting is unsupervised as we have been doing so far: an algorithm is trained using all six sequences without using any ground truth labels. The results are then evaluated against the ground truth. The second setting is supervised with a leave-one-out strategy: five sequences with the ground truth labels are used for training and the remaining sequence is for testing. Following [Fox *et al.*, 2009], we fix the partitions for all five training sequences and learn our model. Then we apply the learned model with fixed parameter $\mathbf{\Omega}$ to find the partition of the test sequence.

Table 4 shows the classification rate for all algorithms. In the supervised setting, SVM [Hoai *et al.*, 2011] performs the best, possibly due to the nonlinear nature of a honeybee's trajectories. In the unsupervised setting, our results (last column) are better than those of [Fox *et al.*, 2009; Hoai and Torre, 2012] but worse than that of [Zhou *et al.*, 2013]. However, we note that both [Zhou *et al.*, 2013] and [Oh *et al.*, 2008] use domain knowledge specific to this problem in their model, which helps to improve their results.

Models	HMM	AR-HMM	C-HMM	AR-C-HMM	HSMM	t-HSMM	AR-HSMM	C-HSMM	t-AR-HSMM	t-C-HSMM	AR-C-HSMM	t-AR-C-HSMM
Gas	0.557	0.533	0.631	0.612	0.910	0.750	0.939	0.926	0.680	0.828	0.885	0.980
Truck 1	0.847	0.943	1.000	0.838	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000
Truck 2	0.758	0.847	0.718	0.758	1.000	1.000	0.919	1.000	0.919	1.000	1.000	1.000
Truck 3	0.634	0.927	0.945	0.512	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

Table 3: Classification rates for different models on different datasets. "HMM" represents hidden Markov model and "HSMM" represents hidden semi-Markov model. "t", "AR" or "C" indicates whether we use time component, autoregressive component or contextual component in the observation model.

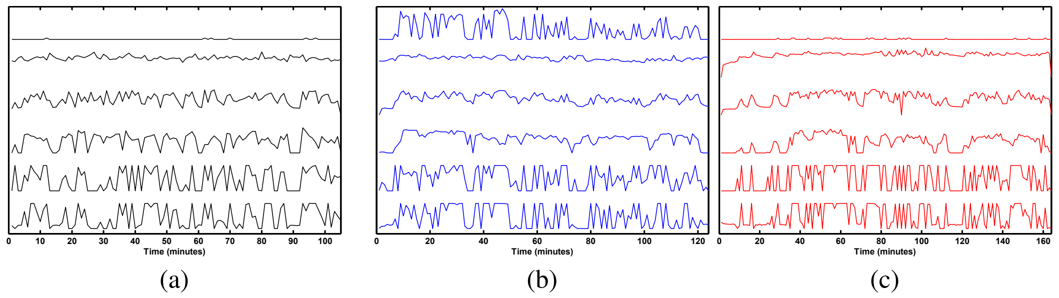


Figure 4: Truck data and results. (a),(b) and (c) show the normal, exhaust filter failure and grill blocking failure sequence, respectively. Each figure displays two output sensors and four input sensors, respectively.

Test	[Oh]	[Hoai'11]	s-[Fox]	s-ours	[Fox]	[Hoai'12]	[Zhou]	Ours
1	0.759	0.859	0.659	0.784	0.465	0.510	0.845	0.635
2	0.924	0.926	0.885	0.844	0.441	0.666	0.925	0.638
3	0.831	0.813	0.792	0.846	0.456	0.483	0.600	0.620
4	0.934	0.923	0.869	0.918	0.832	0.916	0.922	0.930
5	0.904	0.906	0.923	0.905	0.932	0.912	0.878	0.920
6	0.910	0.931	0.891	0.893	0.887	0.888	0.928	0.900
Ave	0.877	0.893	0.837	0.865	0.669	0.729	0.850	0.774

Table 4: Classification rates P_c for different algorithms on the honeybee dataset. The first four methods are supervised and the last four are unsupervised. s-[Fox] and s-ours represent the supervised version of [Fox *et al.*, 2009] and our method, respectively. The last row is the average P_c for all algorithms.

5 Summary

This paper tackles the machine condition monitoring problem under the unsupervised and multi-class setting, which is often neglected but becomes increasingly important in the era of big data. We present a new time series segmentation approach based on segmental hidden Markov model. We model the sensor observation as a unified function of three components including time, previous observations and contextual variables, which is proven to be suitable for machine condition monitoring. One big challenge of unsupervised approaches is how to interpret the results. Based on the intuitive meaning associated with the parameters, we show how the differences between states can indicate different operating modes and efficiencies of a machine. A variety of tests are conducted to show the effectiveness of the proposed model. Due to the additive nature of our observation model, we can easily introduce a new component if extra information is available or remove an existing component if it is not needed. This makes our model very flexible to be easily extended to other applications.

In this work, all coefficients in the observation model are free parameters. Overfitting doesn't appear to be a concern so far for two reasons. First, the parameters for each state are shared across segments and sequences (like a multi-task setting). Second, we keep the time and AR components rather simple by setting $R = Q = 1$. However, in some other applications, overfitting can occur with increasing model complexity or decreasing training set size. Parameter reduction techniques similar to LASSO [Jiang *et al.*, 2012] and principal component analysis [Chang *et al.*, 2014] have been introduced recently in several VAR-based approaches. One possible future direction of our model is to introduce regularization for variable selection as similarly done in [Jiang *et al.*, 2012]. Our optimization objective will then consist of the log likelihood for the standard EM algorithm and this new regularization term. Another future direction is to extend our linear model to a nonlinear model such as Gaussian process [Rasmussen and Williams, 2006], as noted earlier. Gaussian process has a built-in mechanism for relevance determination.

Finally, although our training via the EM algorithm is much faster than MCMC-based algorithms and nonlinear segmentation algorithms, our algorithm still has a complexity proportional to T^2, the square of a sequence length. This may become prohibitive for long sequences. One possible improvement is a multi-resolution coarse-to-fine approach that breaks an original sequence into many sub-sequences with multiple levels.

References

[Antoniadou *et al.*, 2015] I. Antoniadou, N. Dervilis, E. Papatheou, A. E. Maguire, and K. Worden. Aspects of structural health and condition monitoring of offshore wind turbines. *Philosophical Transactions of the Royal Society A*, 373, 2015.

[Chandola *et al.*, 2009] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):15:1–15:58, 2009.

[Chang *et al.*, 2014] J. Chang, B. Guo, and Q. Yao. Segmenting multiple time series by contemporaneous linear transformation. In *arXiv:1410.2323*, 2014.

[Chaubert-Pereira *et al.*, 2010] F. Chaubert-Pereira, Y. Guedon, C. Lavergne, and C. Trottier. Markov and semi-Markov switching linear mixed models used to identify forest tree growth components. *Biometrics*, 66:753–762, 2010.

[Clifton *et al.*, 2008] D. A. Clifton, L. A. Clifton, P. R. Bannister, and L. Tarassenko. Automated novelty detection in industrial systems. *Advances of Computational Intelligence in Industrial Systems Studies in Computational Intelligence*, 116:269–296, 2008.

[Dempster *et al.*, 1977] 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, Series B*, 39:1–38, 1977.

[Dong, 2008] M. Dong. A novel approach to equipment health management based on auto-regressive hidden semi-Markov model (AR-HSMM). *Science in China Series F: Information Sciences*, 51:1291–1304, 2008.

[Eskin, 2000] E. Eskin. Anomaly detection over noisy data using learned probability distributions. In *International Conference on Machine Learning*, 2000.

[Fox *et al.*, 2009] E. B. Fox, A. S. Willsky, E. B. Sudderth, and M. I. Jordan. Nonparametric Bayesian learning of switching linear dynamical systems. In *Advances in Neural Information Processing Systems 21*, 2009.

[Geramifard, 2013] O. Geramifard. *Hidden Markov model-based methods in condition monitoring of machinery systems*. PhD Thesis, National University of Singapore, 2013.

[Hayes and Capretz, 2014] M. A. Hayes and M. A. M. Capretz. Contextual anomaly detection in big sensor data. In *IEEE International Congress on Big Data*, 2014.

[Hoai and Torre, 2012] M. Hoai and F. De La Torre. Maximum margin temporal clustering. In *International Conference on Artificial Intelligence and Statistics*, 2012.

[Hoai *et al.*, 2011] M. Hoai, Z.-Z. Lan, and F. De la Torre. Joint segmentation and classification of human actions in video. In *Proceedings of IEEE Conf. on Computer Vision and Pattern Recognition*, 2011.

[Jiang *et al.*, 2012] H. Jiang, A. C. Lozano, and F. Liu. A bayesian Markov-switching model for sparse dynamic network estimation. In *SIAM Conference on Data Mining*, 2012.

[Johnson and Willsky, 2013] M. J. Johnson and A. S. Willsky. Bayesian nonparametric hidden semi-Markov models. *Journal of Machine Learning Research*, 14(1), 2013.

[Johnson and Willsky, 2014] M. J. Johnson and A. S. Willsky. Stochastic variational inference for Bayesian time series models. In *International Conference on Machine Learning*, 2014.

[Kim and Smyth, 2006] S. Kim and P. Smyth. Segmental hidden Markov models with random effects for waveform modeling. *Journal of Machine Learning Research*, 7, 2006.

[Lütkepohl, 1991] H. Lütkepohl. *Introduction to Multiple Time Series Analysis*. Springer-Verlag, Berlin, 1991.

[Marwala, 2012] T. Marwala. *Condition monitoring using computational intelligence methods: applications in mechanical and electrical systems*. Springer, 2012.

[Menon *et al.*, 2003] S. Menon, O. Uluyol, K. Kim, and E. O. Nwadiogbu. Incipient fault detection and diagnosis in turbine engines using hidden Markov models. In *ASME Turbo Expo.*, 2003.

[Oh *et al.*, 2008] S. M. Oh, J. M. Rehg, T. Balch, and F. Dellaert. Learning and inferring motion patterns using parametric segmental switching linear dynamic systems. *International Journal of Computer Vision*, 77:103–124, 2008.

[Parson *et al.*, 2014] O. Parson, S. Ghosh, M. Weal, and A. Rogers. An unsupervised training method for non-intrusive appliance load monitoring. *Artificial Intelligence*, 217:1–19, 2014.

[Pimentel *et al.*, 2014] M. A. F. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko. A review of novelty detection. *Signal Processing*, 99:215–249, 2014.

[Quinn *et al.*, 2009] J. A. Quinn, C. K. I. Williams, and N. McIntosh. Factorial switching linear dynamical systems applied to physiological condition monitoring. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 31:1537–1551, 2009.

[Rabiner, 1989] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 257–286, 1989.

[Rasmussen and Williams, 2006] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[Saatçi *et al.*, 2010] Y. Saatçi, R. Turner, and C. E. Rasmussen. Gaussian process change point models. In *International Conference on Machine Learning*, 2010.

[Sarkar *et al.*, 2012] S. Sarkar, K. Mukherjee, S. Sarkar, and A. Ray. Symbolic transient time-series analysis for fault detection in aircraft gas turbine engines. In *American Control Conference*, 2012.

[Smyth, 1993] P. Smyth. Markov monitoring with unknown states. *IEEE Journal On Selected Areas In Communications*, 12:1600–1612, 1993.

[Song *et al.*, 2007] X. Song, M. Wu, C. Jermaine, and S. Ranka. Conditional anomaly detection. *IEEE Trans. on Knowledge and Data Engineering*, 19(5):631–645, 2007.

[Valko *et al.*, 2011] M. Valko, B. Kvetony, H. Valizadeganz, G. F. Cooperx, and M. Hauskrecht. Conditional anomaly detection with soft harmonic functions. In *IEEE International Conference on Data Mining*, 2011.

[Xuan and Murphy, 2007] X. Xuan and K. Murphy. Modeling changing dependency structure in multivariate time series. In *International Conference on Machine Learning*, 2007.

[Yu, 2010] S. Yu. Hidden semi-Markov models. *Artificial Intelligence*, 174(2):215–243, 2010.

[Zhou *et al.*, 2013] F. Zhou, F. De la Torre, and J. K. Hodgins. Hierarchical aligned cluster analysis for temporal clustering of human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(3), 2013.