Mechanical Systems and Signal Processing 99 (2018) 73–82

Contents lists available at ScienceDirect

Mechanical Systems and Signal Processing

journal homepage: www.elsevier.com/locate/ymssp

Graph-based structural change detection for rotating machinery monitoring

Guoliang Lu [a,*], Jie Liu [b], Peng Yan [a]

[a] Key Laboratory of High-efficiency and Clean Mechanical Manufacture of MOE, School of Mechanical Engineering, Shandong University, Jinan 250061, China
[b] Department of Mechanical & Aerospace Engineering, Carleton University, Ottawa, ON K1S 5B6, Canada

ARTICLE INFO

Article history:
Received 9 December 2016
Received in revised form 14 May 2017
Accepted 4 June 2017

Keywords:
Structural change detection
Machine monitoring
Graph model
Martingale test

ABSTRACT

Detection of structural changes is critically important in operational monitoring of a rotating machine. This paper presents a novel framework for this purpose, where a graph model for data modeling is adopted to represent/capture statistical dynamics in machine operations. Meanwhile we develop a numerical method for computing temporal anomalies in the constructed graphs. The martingale-test method is employed for the change detection when making decisions on possible structural changes, where excellent performance is demonstrated outperforming exciting results such as the autoregressive-integrated-moving average (ARIMA) model. Comprehensive experimental results indicate good potentials of the proposed algorithm in various engineering applications. This work is an extension of a recent result (Lu et al., 2017).

1. Introduction

In operational monitoring of a rotating machine, one major goal is to find unexpected or abnormal process state(s)/behavior(s) in machine operations which can be regarded as detection of statistical changes via monitoring numerical variables such as sound emission, cutting force, vibration, temperature, power consumption and so on [2,3]. This technique can overcome a number of real-world engineering problems ranging from early fault detection/diagnosis, safety protection, and many other process monitoring and control problems. Depending on whether requiring a prior learning/training phase, change-point detection algorithms for machine monitoring can be classified into two categories [4]: *supervised* methods (e.g., [5–9]) and *unsupervised* methods (e.g., [3,10,11]). Supervised detection requires a prior learning/training phase to train a change detector using, e.g., SVM [5,6], neural networks [7,8] and fuzzy logic model [9], by taking advantage of collecting a large number of (or at best a full/complete set of) available training samples. Although it tends to have a good balance between response time and accuracy in detection, the further development and commercialization of this method is presently inhibited by a lack of generalization ability of change detectors [4]. Actually it is reasonable because in real applications, the user cannot simulate various engineering scenarios to collect sufficient samples for training a robust and reliable change detector. On the other hand, unsupervised detection is addressed without making any assumption of available training samples. In a standard unsupervised detection architecture, an auto-regressive (AR) model (e.g., Gaussion [3,12] or relatives of Gaussion [10,13], or a simple linear regression model [2]) is used to fit/model the observed data, and then the change point is detected by minimizing the total errors of local model fitting of segments to the data before and after that

* Corresponding author.
E-mail addresses: luguoliang@sdu.edu.cn (G. Lu), jliu@mae.carleton.ca (J. Liu), yanpeng@sdu.edu.cn (P. Yan).

http://dx.doi.org/10.1016/j.ymssp.2017.06.003
0888-3270/

point with a statistical metric, e.g., CUSUM test [3], F_c metric [14] and GLRT test [11]. A larger detection delay is an essential limitation of these methods for real applications [4]. In our study, we propose a new approach where the change is detected only using the data before that change, and thus change decision can be operated by a *real-time* way.

The work in this paper is an extension of our recent article [1]. In [1], we have proposed a generic framework for detecting changes in a monitored machine operational process, where we used the autoregressive-integrated-moving average (ARIMA) model [15] to learn a statistical regularity from the data, then detect the change points by investigating how much each data is deviated from the regularity using *martingale*. In this paper, we extend the previous framework toward two directions:

(a) Considering typical machines have dynamic natures in monitored variables over time [16,17], we take into account such dynamic information for the design of change detection. We take the graph model instead of the ARIMA model for data modeling. By this replacement, the data is represented by graphs with weighted undirected links, and thus a community that is a subgraph whose members are connected strongly to each other is now a key factor to specify characteristics of the data;
(b) We develop a numerical method for computing temporal anomalies in the constructed graphs which indicates how much the monitored data point is deviated from the already observed data. More specifically, a higher anomaly score indicates a higher probability that a change happens, and vice versa.

Besides the methodological extensions of the proposed method, we also present comprehensive experimental results of the method, which outperforms the previous method [1] in various engineering applications.

The rest of this paper is organized as follows. In the Section 2, we describe the proposed framework in details. Section 3 shows experimental results for change detection using the graph-based model. Section 4 gives two possible techniques of using the proposed method for multi-sensor based machine monitoring. At last, we give concluding remarks in Section 5.

2. Graph-based change point detection for machine monitoring

We consider a data stream collected in an operational process of a monitored machine, each of which is specified by measurable variables such as sound emission, cutting force, vibration and power consumption. Two assumptions are first given to support the proposed framework in this paper: (**A**) the measured time-series data is *periodic*, which allows us to divide the collected data stream into a set of sequential cycles according to the estimated periodicity. Thus, by monitoring the amount of fluctuations in these resulted sequential cycles, we can capture the dynamic natures of a machine in its operations; and (**B**) the structural changes are *detectable*, which guarantees the success of change detection from the collected data. For more details, please refer to [1].

In the following, we will present our algorithm to detect structural changes from a monitored data stream. Section 2.1 introduces the graph model and the computation method for construction of it from the data stream. Section 2.2 gives a method of scoring anomalies in the constructed graphs. Section 2.3 shows the detection by using the measured anomalies with martingale-test. Section 2.4 gives the algorithm of the proposed method for practical implementation.

2.1. Model formulation

Given a periodic data stream denoted by $F = \{f_{nT+v}\}, n = 1, 2, 3\ldots$ where n is the cycle index, T is the period length and v is the phase (timing), obviously this data stream shows a specific and similar structure in each cycle as long as a change does not occur. There have been proposed some data models for capturing such a structure in the data such as using Guassian models and fuzzy logic models mentioned previously. Here, in this paper, we propose to employ the graph model for this purpose.

In general, a graph G consists of a set $\mathscr{V}$ of nodes and a set $\mathscr{E}$ of links, i.e., $G = \{\mathscr{V}, \mathscr{E}\}$, and furthermore, each link has a weight $\omega \in [0, \infty]$ representing the strength of connection. Such a graph can be represented by an adjacency matrix X where each element $x_{i,j}$ indicates the weighted link between the ith node and the jth node (obviously, the adjacency matrix is symmetrical). More specifically, for each cycle of given data, we learn/construct its corresponding graph as follows (see an illustrative example in Fig. 1): the node is the time stamp and the weighted link between two nodes is computed as the Euclidean distance between the measurable values on these nodes. By this modeling, the original stream F is now modeled as a sequence of graphs of cycles, i.e., $F = \{G^1, G^2, \ldots, G^t, \ldots\}$ representing by adjacency matrices $\mathscr{X} = \{X^1, X^2, \ldots, X^t, \ldots\}, X \in \mathbf{R}^{V \times V}, t = 1, 2, \ldots, N$ where G^t is the constructed graph in tth cycle and its corresponding adjacency matric is X^t, N is the number of cycles in F, and V is the number of nodes in one graph (i.e., the number of data in each cycle).

A graph may change its communities in the member and/or in the way of connections because of the dynamic nature of machine in its operations. That is, a change happens (**A**) when the densely connected nodes is separated or sparsely connected nodes have dense connections, and (**B**) when the number of nodes increases or decreases. Such community changes can be analyzed by the spectrum of a matrix representing graph [18]. Let us take one adjacency matrix X^t in F for example. We can decompose the symmetrical X^t into $X^t = \Gamma\Lambda\Gamma'$ where Γ is a matrix whose column is an eigenvector of X^t and Λ is a diagonal matrix whose diagonal elements are the real value eigenvalues. The change **A** of *community structure* can be reflected by monitoring Γ, while the change **B** of *community activity* is by Λ [19,20]. As for the problem of change detection

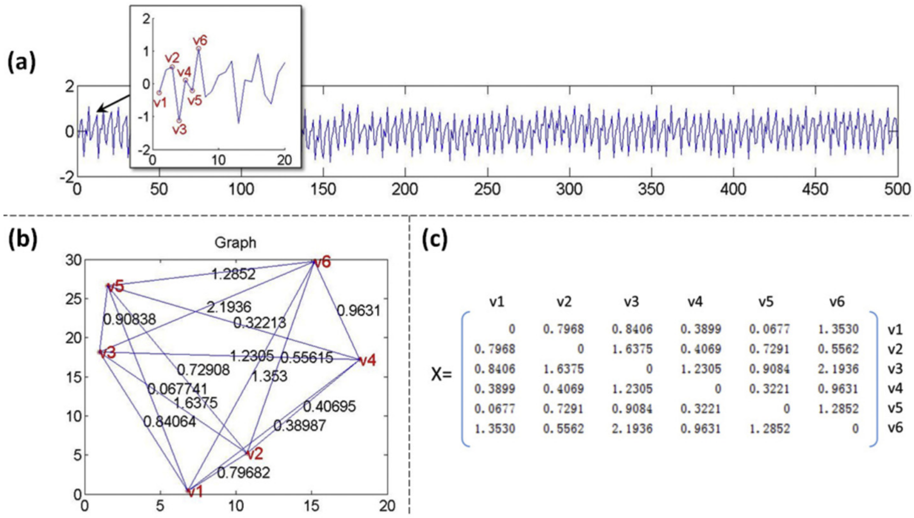

Fig. 1. An illustrative example of graph learning from monitored variables. In this example, we first collected a sequence of sound variables (a) as will be described in the following experiment, and then picked up the beginning 6 points for illustration to learn a graph (b). The corresponding adjacency matrix is given in (c).

in machine monitoring, since we assume the collected data is periodic and have divided the data into individual cycles which implies the number of nodes is unchanged in constructed graphs, we will concentrate on measuring the amount of fluctuation in community structure by Γ for change detection in the following.

The above-presented eigen decomposition should account for the noise which is unavoidably existed in real-world scenarios, as:

$$X^t = \Gamma(\Lambda + \epsilon^t)\Gamma', \quad t = 1, 2, \ldots, N \tag{1}$$

In a normal unchanged state, the structure Γ as well as activity Λ is assumed not to change, while the noise ϵ^t can change as *independent and identically distribution* (I.I.D.) with mean zero, that is $\mathsf{E}[\epsilon^t] = 0$. The expectation of X^t can be thus given by

$$\mathsf{E}[X^t] = \Gamma\Lambda\Gamma' + \Gamma(\mathsf{E}[\epsilon^t])\Gamma' = \Gamma\Lambda\Gamma'. \tag{2}$$

Therefore, assuming ergodicity of X^t for a period of M beginning cycles $t \in [1, M]$, we can estimate $\mathsf{E}[X^t]$ approximately as a sample mean as

$$\bar{X} = \frac{1}{M}\sum_{t=1}^{M} X^t \simeq \Gamma\Lambda\Gamma'. \tag{3}$$

In real usages, we can initialize $\bar{X}$ with the first beginning cycle and iteratively update it with subsequently-arrived cycles. Then, on the basis of Eq. (3), we can estimate Γ and Λ by decomposition of $\bar{X}$. When the data stream F seems to be generated from this model, we regard that F is in a normal state, i.e., no change occurs; otherwise, we consider the data is in an abnormal state, i.e., a change occurs. In the following, we will quantify the data distribution of the graphs from this model and discriminate the normal and abnormal states.

2.2. Anomaly measure

Let us suppose that the model estimated with Eq. (3) has been change from $\{\Gamma, \Lambda\}$ to $\{\Gamma^*, \Lambda^*\}$ at a specific cycle t^*. Since the interested changes in machine monitoring is the ones caused by community structures as discussed already in Section 2.1, we will monitor the amount of fluctuations of corresponding community structure Γ for change detection. The null and alternative hypothesis can be then established as

$$\begin{aligned} &H_0 : \Gamma = \Gamma^*, \ \text{no change occurs}, \\ &H_A : \Gamma \neq \Gamma^*, \ \text{a change occurs}. \end{aligned} \tag{4}$$

Assuming that Γ has been already estimated by decomposition of $\bar{X}$ which is computed from the past $t-1$ cycles as shown in Eq. (3), for X^t we can decompose it by multiplying Γ and Γ' in both sides as

$$X^t = \Gamma Y^t \Gamma' \tag{5}$$

Note that the obtained Y^t is not diagonal in general due to the fact that X^t cannot be kept identical to the past cycles of $X^1, X^2, \ldots, X^{t-1}$ when various noises exist in real data collections. Therefore, we can separate its components into the diagonal part and the non-diagonal part as,

$$X^t = \Gamma Y^t \Gamma' = \Gamma(\text{diag}[Y^t])\Gamma' + \Gamma(\text{non-diag}[Y^t])\Gamma'. \tag{6}$$

Similarly, Eq. (1) can also be re-written as,

$$X^t = \Gamma(\Lambda + \epsilon^t)\Gamma' = \Gamma(\Lambda + \text{diag}[\epsilon^t])\Gamma' + \Gamma(\text{non-diag}[\epsilon^t])\Gamma' \tag{7}$$

By comparing Eqs. (6) and (7), we can see that diag$[Y^t]$ reflects the regular fluctuations within communities, while non-diag$[Y^t]$ reflects the fluctuations between communities. In particular, when a change happens at this time t, the second term in Eq. (6) will be changed accordingly. In this sense, we can consider non-diag$[Y^t]$ as the fluctuations in the monitored community structures. Thanks to the Frobenius norm, we can quantify the fluctuation z^t of the graph at time t by,

$$z^t = ||\text{non-diag}[Y^t]||_f = \text{Tr}[(\text{non-diag}[Y^t])' \times (\text{non-diag}[Y^t])]. \tag{8}$$

Finally, on the basis of $\{z^t\}$, we can measure the anomaly score s^t of each z^t against the already-observed $z^1, z^2, \ldots, z^{t-1}$ by

$$s^t = s(\{z^1, z^2, \ldots, z^{t-1}\}, z^t) = |z^t - H_t|_1, \tag{9}$$

where $H_t = \sum_{i=1}^{t} z^i / t$ and $|\cdot|_1$ is the L_1 norm.

2.3. Change detection by martingale-test

From the sequence of measured anomaly scores $s^t, t = 1, 2, 3\ldots$, we would like to find the time when the community structure is changed. Conventional metrics such as F_c metric [14], the extend F_{max} metric [21] and CUSUM test [3] may possibly be used for such a task, however, they are carried out on the basis of a retrospective analysis for the processed sequence which brings a relatively large detection delay for response. On the contrast, martingale-test has been demonstrated satisfactory performance on real-time change detection in many works (e.g., [22,23]). We, therefore, perform the martingale-test for handing community structure change studied in this paper. Here, we only provide two necessary steps in operation of this test as follows (please refer to [1] for details):

– *Step 1*: On the basis of $s_1, s_2, \ldots, s_t$, the *randomized power martingale* (RPM) [22] is constructed by

$$M(t) = \prod_{i=1}^{t} (\epsilon \hat{p}_i^{\varepsilon-1}), \tag{10}$$

where $\epsilon \in (0, 1)$ and $\hat{p}_i$s are the $\hat{p}$-values computed from $\hat{p}$-value functions:

$$\hat{p}_i(\{s_1, \ldots, s_{i-1}\}, s_i) = \frac{\#\{j : s(j) > s(i)\} + \theta_i \#\{j : s(j) = s(i)\}}{i}, \tag{11}$$

where $\#\{\cdot\}$ is a counting function and θ_i is a random value from a uniformly distribution of $[0, 1], j \in \{1, 2, \ldots, i-1\}$.

– *Step 2*: The *Doob's Maximal Inequality* [24] is then tested for any $t^* \in \{1, 2, \ldots, t\}$ to reject the hull hypothesis in Eq. (4):

$$P(\exists t^* | M(t^*) \geqslant \lambda) \leqslant \frac{1}{\lambda}. \tag{12}$$

That is, if the martingale value $M(t^*)$ is greater than a pre-defined threshold λ, H_A in Eq. (4) is satisfied, i.e., a change occurs on the time t^*. Otherwise, the martingale test continues to operate as long as $0 < M(t^*) < \lambda$, that is, H_0 in Eq. (4) is satisfied.

2.4. Implementation and validation

In order to guarantee the success of proposed method in engineering applications, two practical issues must be addressed: (**1**) in Section 3.1, we assume the periodicity of F has been known, while in practical usages, this prior assumption cannot be always satisfied. To address this problem, we use the periodicity estimation method proposed in [1] as a pre-processing to estimate the underlying periodicity before graph modeling for F; (**2**) some necessary parameters in martingale-test should be set as follows. The ε in Eq. (10) takes responsible for determining the sensitivity for the changes. Specifically, the small ε increases sensitiveness for the change but causes false alarms. According to [23], it is suggested to be in $[0.9, 1)$. We hence set it as 0.9 in our framework. While, the threshold λ in Eq. (12) can be set empirically or estimated by a prior cross-validation. The pseudo code of our change-point detection algorithm is then given in Algorithm 1.

Algorithm 1. Pseudo code of change-point detection.

Require:
Collected data stream $\{f_{nT+v}\}$;
Threshold value λ.
Ensure:
Output change time t^*.
1. Initialize cycle index $t = 1$ and compute the corresponding adjacency matrix of graph X^1;
2. Initialize $\bar{X}$ with X^1 and compute Γ by Eq. (3);
3. Set cycle index $t = 2$;
4. Change detection is performed as
while $t \leqslant$ *stopping-time* **do**
  Compute adjacency matrix of tth graph X^t;
  Compute Y^t by eigen decomposition by Eq. (5);
  Separate Y^t into diag[Y^t] and non-diag[Y^t];
  Compute z^t by the Frobenius form as Eq. (8);
  Compute anomaly score s^t by Eq. (9);
  Construct martingale values by Eq. (10) and test null hypothesis in Eq. (4) as
  if $M(t) \geqslant \lambda$ **then**
    Output time t^*;
    Break and start a new change detection process;
  else
    Update $\bar{X}$ with $\{X^1, \ldots, X^t\}$ and compute Γ by Eq. (3);
    $t = t + 1$;
  end if
end while

We applied the proposed method to non-stationary signals to verity its effectiveness. Considering the condition changes in real machine operations often result in changes of relevant variables in terms of amplitude, frequency, or both of them, the testing signals are formulated respectively as,

$$x_t = \begin{cases} \sin(\omega t), & 1 \leqslant t < c - 1 \\ 2\sin(\omega t), & c \leqslant t \leqslant m \end{cases} \tag{13}$$

$$x_t = \begin{cases} \sin(\omega t), & 1 \leqslant t < c - 1 \\ \sin(2\omega t), & c \leqslant t \leqslant m \end{cases} \tag{14}$$

$$x_t = \begin{cases} \sin(\omega t), & 1 \leqslant t < c - 1 \\ 2\sin(2\omega t), & c \leqslant t \leqslant m \end{cases} \tag{15}$$

to simulate these changes where ω was set as 0.2, c is the change time and m is the length of data. The results are given in Fig. 2 where obviously we can see that, the computed martingale values by our method increase greatly when the change occurs for each testing data, that is, the change can be detected successfully by setting an appropriate value of threshold.

2.5. Comparison with ARIMA model

Data modeling is the first step for change detection in machinery monitoring where many auto-regressive (AR) models can be adopted. In the recent work [1], the ARIMA model was used for this purpose by treating data points as *time-independent* points in individual cycles. While, in real applications, the collected data may exhibit/appear *inter-dependencies* (for example, as shown in Fig. 1(a), the data points show a similar *wave* trend in cycles) which should be accounted when designing change detection. Graphs have been widely used in statistics and data mining because of its flexibility and powerful capabilities in modeling inter-dependent data points with long-range correlations (for review please see [25]). In this paper, we propose to use the graph model for the purpose of data modeling and compare it with the ARIMA model in the three following aspects:

(a) *Modeling of monitoring data*: the effectiveness of the ARIMA model was investigated in [1], where the potential existence of local noise would deteriorate its detection performance since the learning of this model relies on a *point-to-point* strategy. On the other hand, the graph model describes the data from a global view of each cycle by

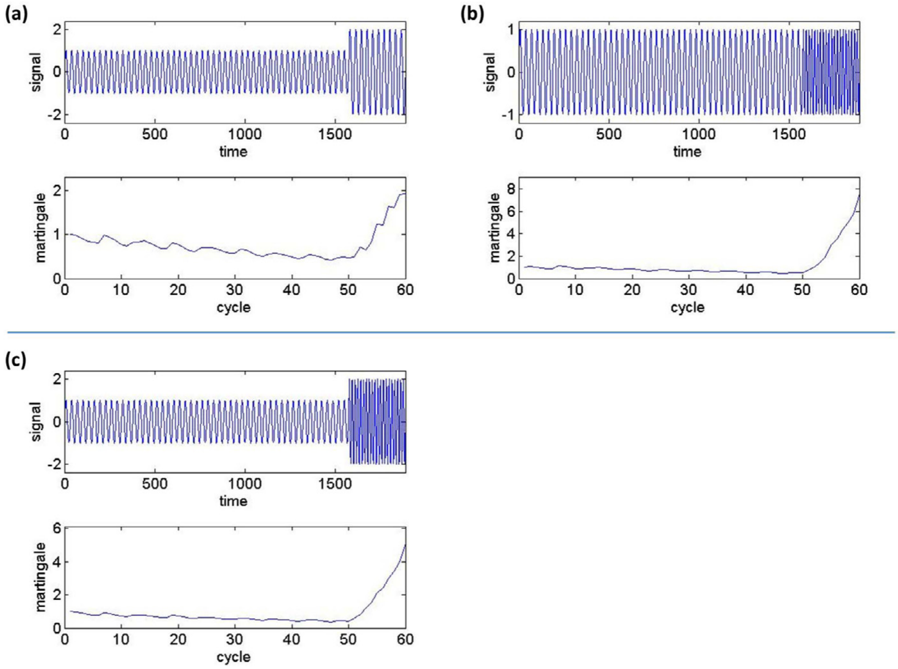

Fig. 2. Results on testing data containing different changes: (a) amplitude change; (b) frequency change; (c) amplitude and frequency change.

using a *cycle-to-cycle* strategy for its learning and updation, which is believed to be more robust against noises [25], and suitable for machine monitoring in real applications.

(b) *Powerful representation*: graphs naturally represent the inter-dependencies by introducing links between each pair of data points in one cycle (see Section 2.1). The multiple paths between these points effectively capture their long-range correlations within individual cycles. Moreover, a graph representation facilitates the data description of data by only using the adjacency matric. As a result, our primary goal is to quantify the temporal anomaly between constructed graphs without knowing the entire characteristic structure and dynamic operations in data, which however is required for the ARIMA model.

(c) *Detection delay*: as mentioned above, the graph model operates a *cycle-to-cycle* learning/updation strategy, which results in cycle-level change decision. This delivers a lower computationally complexity but meanwhile results in a larger detection delay, i.e., a change has to be successfully detected when the underlying cycle is finished. Detailed discussion on this issue will be further given in Section 3.2.

3. Experiment

The purpose of the following experimental investigations is to justify the effectiveness and applicability of the above presented framework in engineering applications. In addition, since the framework is extended from the previous work [1], we also compare it with that method as well.

3.1. Data preparation

Among various information carriers, we use the sound signal as the monitored variable for our investigation due to its relatively large applicability in implementation. One notes that, since the measured variable used in data collection is replaceable, it is feasible to implement the proposed framework with other information carriers (e.g., vibration) as long as the carried information can be measurable and can indicate whether a change occurs.

We used the same experimental setup as used in [1] to collect test data where sound signal was first acquired by a microphone mounted on the gearbox, and then sent to an equipment for a PC. In this collection, the rotating speed of motor was first kept at v, and then changed with an interval of Δv to simulate a change (i.e., $v \rightarrow v + \Delta v$) in real machinery operations. In our implementation, two sets of candidate parameters,

- $v \in \{1000, 1500, 2000\}$ rpm and $\Delta v \in \{0, 150, 300, 450\}$ rpm,

are provided. We thus have 12 parameter combinations obtained by a grid search to generate changes, and then, for each combination (for example, for $v = 1500$ rpm and $\Delta v = 300$ rpm), we collected data sequences at 20 times. At last, we collected 240 test data sequences totally including 180 sequences containing changes and 60 sequences without any change. The collected signal was sampled with a frequency of 4000 Hz lasting around 30 s.

Based on the collected test data sequences, we further formed six scenarios, as described in Table 1. Each scenario also contains the 60 no-change data sequences and thus has 120 data sequences totally in each of them. In the following, we will evaluate the proposed method for each of them.

3.2. Experimental implementation and results

For a given test data sequence, our proposed framework is operated as the following procedures:

– *Step 1*: We first down-sampled the original data by 1:50 to decrease computation burden in process, and then performed periodicity estimation proposed in [1] to confirm the periodicity in the data.
– *Step 2*: Based on the estimated periodicity, we divided the down-sampled version of data into a set of sequential cycles, and furthermore, for the data in each cycle, we modeled it by the graph model (as described above in Section 2.1). Fluctuations of community structure are then measured with the procedures presented in Section 2.2. We last used the martingale-test introduced in Section 2.3 to detect the change.

Fig. 3 shows an example of change detection for a given test data sequence containing a change from 1500 rpm to 2000 rpm where the change is labeled by a human instructor as shown in Fig. 3(a). We first down-sampled the original version of the sequence, and estimated the periodicity in it using the method proposed in [1] by which the sequence was divided into a sequential set of individual cycles (see Fig. 3(b)); temporal anomalies were measured by the procedures as described in Section 3.2. Although the change is hardly identified from the raw data, from Fig. 3(c) we can see that the resulted anomalies reflecting the fluctuations of community structure of graphs increase rapidly after the change occurs, and the change can be detected successfully last by the martingale-test with a pre-defined threshold λ as seen in Fig. 3(d).

Next, we compare the performances of the proposed framework with those by the previous method in terms of the *receiver operating characteristic* (ROC) curves and the area under the ROC curve (AUC) values. The *true positive rate* (TPR) and *false positive rate* (FPR) are defined respectively as follows:

Table 1
Simulated change of $v \to v + \Delta v$ (rpm) in six testing scenarios.

Scenario	s1	s2	s3
$v \to v + \Delta v$	1000 → 1000 + 150 1000 → 1000 + 300 1000 → 1000 + 450	1500 → 1500 + 150 1500 → 1500 + 300 1500 → 1500 + 450	2000 → 2000 + 150 2000 → 2000 + 300 2000 → 2000 + 450
	s4	s5	s6
$v \to v + \Delta v$	1000 → 1000 + 150 1500 → 1500 + 150 2000 → 2000 + 150	1000 → 1000 + 300 1500 → 1500 + 300 2000 → 2000 + 300	1000 → 1000 + 450 1500 → 1500 + 450 2000 → 2000 + 450

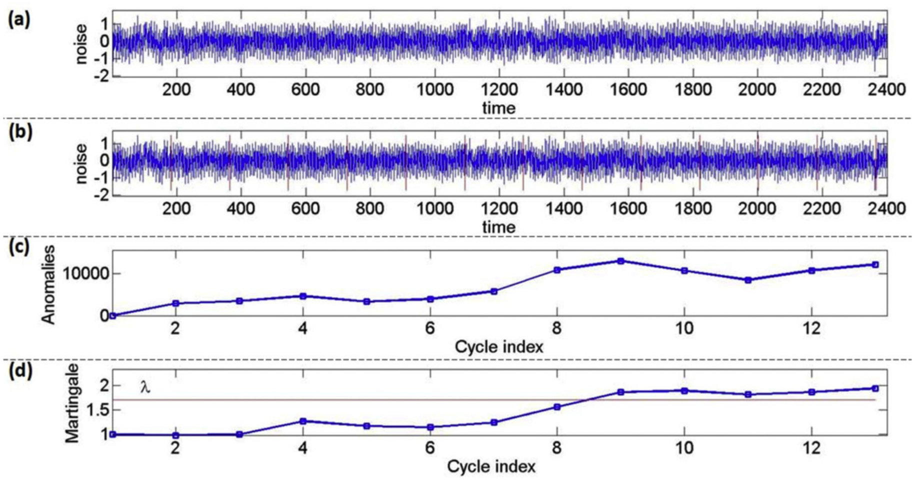

Fig. 3. An example of change detection for a given test sequence by the present method. (a) Down-sampled testing signals, (b) dividing the original data into individual cycles, (c) the computed anomaly scores as described in Section 2.2, and (d) the computed martingale values as described in Section 2.3.

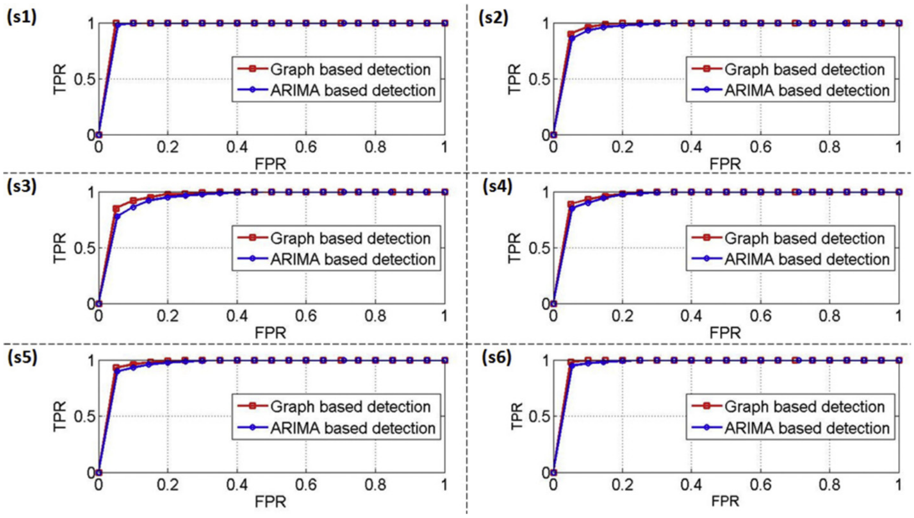

Fig. 4. ROC curves of the graph-based method proposed in this paper and the ARIMA model based method proposed previously in [1] for testing six scenarios.

- True positive rate (TPR): n_{co}/n_{tr},
- False positive rate (FPR): $(n_{al} - n_{co})/n_{al}$,

where n_{co} denotes the number of correctly-detected changes, n_{tr} denotes the number of all true changes, and n_{al} is the number of all detection alarms.

The role of threshold λ in Eq. (12) is to control the balance of precision and recall in change detection. Specifically, with a lower value of λ, the change will be more easily detected while with an increasing false alarm, and it will be in turn for a higher value of λ. In the experiment, we first set up a threshold λ^* to remove all detection alarms, that is, λ^* should be larger than any value of martingales for all testing data sequences, and then decreased the value of λ^* gradually to make sure both TPR and FPR not decrease. By this, we can plot all corresponding pairs of TPR and FPR on the graph, and thus a monotone curve can be drawn.

Fig. 4 illustrate ROC curves averaged over 20 running times for the six testing scenarios, and the corresponding mean and standard deviation of the AUC values are also given in Fig. 5. These results show that the presented graph model based change detection method outperforms the ARIMA model based method proposed previously in [1].

In addition to the above-employed performance indicators, i.e., ROC and AUC, computational complexity and detection delay[1] are two complementary performance indicators which have to be addressed in the design of practical change detectors. Table 2 shows the results of computational complexity and detection delay for two methods. First, for the term of computational complexity, we can see that, in the phase of data modeling, the ARIMA based modeling shows a lower complexity than the graph based modeling, that is, $O(n_c)$ *v.s.* $O(n_c^2)$; while in the detection phase, the graph based method performs better than the compared ARIMA method, that is $O(m_c)$ *v.s.* $O(m_c * n_c)$. The total complexity depends on n_c and m_c: (1) if $n_c > m_c$, the complexity of graph method is $O(n_c^2 + m_c) = O(n_c^2)$ while the ARIMA method is $O(m_c * n_c)$, which means the ARIMA method performs better; (2) if $m_c \gg n_c$, the graph-based method can achieve a much lower computation complexity. This point was demonstrated by an example given in Fig. 6. It can be seen that, in spite that the proposed method takes a more expensive time at the beginning of change detection, while with an increasing number of collected data, the computation time of ARIMA based method increases much more greatly than the present graph-based method. In this sense, the graph-based detection presented in this paper is much more suitable for long-term monitoring applications; while, the ARIMA based detection seems to have a higher computation efficiency for short-term monitoring purposes. For the second term of delay, since the present method detects changes depending on measuring fluctuations of community structure of graphs of cycles, its detection delay is *cycle-leveled* which is larger than a *point-time leveled* delay for the ARIMA method.

In summary, compared with the previous method [1], the graph-based method presented in this paper has the following properties:

- This method can achieve a better detection performance;
- In spite that the method seems to have a larger detection delay, it has a lower complexity in computation which makes it is very suitable for a long-term monitoring.

[1] The detection delay is defined as the time between a true change and its detection.

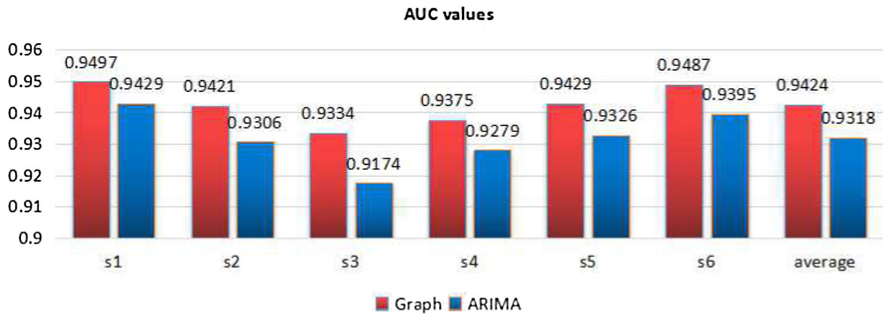

Fig. 5. AUC values corresponding to the ROCs in Fig. 4, where higher values indicate better detection performances. It can be clearly seen that the proposed graph-based detection method in this paper has achieved better performances than the ARIMA model based method [1] for all testing scenarios.

Table 2
Comparison of the graph-model based detection and the ARIMA model based detection. In the table, n_c denotes the number of data points in one cycle and m_c is the number of cycles in a given data sequence.

Model	Computation complexity			Level of delay
	Modeling	Detection	Total	
Graph	$O(n_c^2)$	$O(m_c)$	$O(n_c^2 + m_c)$	Cycle
ARIMA	$O(n_c)$	$O(m_c * n_c)$	$O(n_c + m_c * n_c)$	Point time

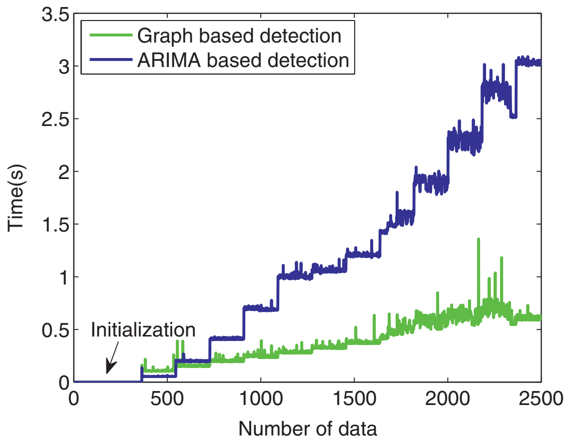

Fig. 6. Computation time of two methods in the phase of change detection on an unchanged test sequence collected at $v = 1500_{rpm}$. Computation environment: MATLAB programming, CPU AMD 2.60 GHz and RAM 4.00 GB.

4. Discussion

Recent experience, for example as seen from [26], shows that the multi-sensor measurements can provide more reliable change decision making, compared with the use of a single sensor, in process condition monitoring. Although we have assumed a single sensor in this study and proposed to use the graph to model the collected monitoring variable, our framework can be utilized even for the case of multi-sensor based machine monitoring. We may implement this by employing one of the following two techniques:

- adopting a data dimension reduction method, e.g., principal component analysis (PCA), kernel principal component analysis (KPCA) and independent component analysis (ICA), with a sliding window or a dynamic strategy on the collected high-dimensional monitoring variables, by which we can project the raw data into low-dimensional space and then use the resulting data to compute the temporal anomalies by the proposed modeling method based on graph, and finally detect the change by martingale test.
- modeling the monitoring data separately in each dimension using the proposed graph model and measuring the temporal anomalies subsequently. For this case, an appropriate aggregating/fusion strategy, e.g., the point-wise average and the

point-wise maximum [27], the point-wise threshold average [28], has to be adopted to aggregate information from all dimensions. After this, the change can be detected by martingale test.

5. Conclusion

In this paper, we proposed a new framework for change detection in machine monitoring where we introduced the graph model instead of the ARIMA model employed previously in [1] to capture/represent the statistical dynamics in a monitored machine operational process. Meanwhile we provided a new numerical computation method for measuring the anomalies in the data by which the changes can be detected from the constructed graphs. Martingale-test is used for making change decision. Experimental results demonstrated that the proposed method can achieve better detection performances than the previous work. Moreover, it has a high computation efficiency which is applicable for long-term monitoring.

Future work, along with the line of this study, will focus on optimizing the proposed method to achieve a higher computational efficiency.

Acknowledgment

This work is supported in part by National Natural Science Foundation of China (61403232, 61327003), Natural Science Foundation of Shandong Province, China (ZR2014FQ025), and Young Scholars Program of Shandong University (YSPSDU, 2015WLJH30). We would like to thank Professor Jie Liu, from the Department of Mechanical & Aerospace Engineering of Carleton University, for his discussions when we were preparing this work.

References

[1] G. Lu, Y. Zhou, C. Lu, X. Li, A novel framework of change-point detection for machine monitoring, Mech. Syst. Signal Process. 83 (2017) 533–548.
[2] A. Reñones, L.J. de Miguel, J.R. Perán, Experimental analysis of change detection algorithms for multitooth machine tool fault detection, Mech. Syst. Signal Process. 23 (7) (2009) 2320–2335.
[3] T.D. Popescu, Blind separation of vibration signals and source change detection–application to machine monitoring, Appl. Math. Model. 34 (11) (2010) 3408–3421.
[4] C. Aldrich, L. Auret, Unsupervised Process Monitoring and Fault Diagnosis with Machine Learning Methods, Springer, 2013.
[5] C.-C. Hsu, M.-C. Chen, L.-S. Chen, Intelligent ICA–SVM fault detector for non-gaussian multivariate process monitoring, Expert Syst. Appl. 37 (4) (2010) 3264–3273.
[6] A. Soualhi, K. Medjaher, N. Zerhouni, Bearing health monitoring based on Hilbert–Huang transform, support vector machine, and regression, IEEE Trans. Instrum. Meas. 64 (1) (2015) 52–62.
[7] M.D. Prieto, G. Cirrincione, A.G. Espinosa, J.A. Ortega, H. Henao, Bearing fault detection by a novel condition-monitoring scheme based on statistical-time features and neural networks, IEEE Trans. Industr. Electron. 60 (8) (2013) 3398–3407.
[8] T. Boukra, A. Lebaroud, G. Clerc, Statistical and neural-network approaches for the classification of induction machine faults using the ambiguity plane representation, IEEE Trans. Industr. Electron. 60 (9) (2013) 4034–4042.
[9] V. Muralidharan, V. Sugumaran, Rough set based rule learning and fuzzy classification of wavelet features for fault diagnosis of monoblock centrifugal pump, Measurement 46 (9) (2013) 3057–3063.
[10] Z. Ge, U. Kruger, L. Lamont, L. Xie, Z. Song, Fault detection in non-gaussian vibration systems using dynamic statistical-based approaches, Mech. Syst. Signal Process. 24 (8) (2010) 2972–2984.
[11] A. Willersrud, M. Blanke, L. Imsland, A. Pavlov, Fault diagnosis of downhole drilling incidents using adaptive observers and statistical change detection, J. Process Control 30 (2015) 90–103.
[12] J. Yu, Machine tool condition monitoring based on an adaptive gaussian mixture model, J. Manuf. Sci. Eng. 134 (3) (2012) 031004.
[13] T.D. Popescu, Signal segmentation using changing regression models with application in seismic engineering, Digital Signal Process. 24 (2014) 14–26.
[14] R. Lund, J. Reeves, Detection of undocumented changepoints: a revision of the two-phase regression model, J. Clim. 15 (17) (2002) 2547–2554.
[15] G.E. Box, G.M. Jenkins, G.C. Reinsel, G.M. Ljung, Time Series Analysis: Forecasting and Control, John Wiley & Sons, 2015.
[16] K.A. Loparo, M. Adams, W. Lin, M.F. Abdel-Magied, N. Afshari, Fault detection and diagnosis of rotating machinery, IEEE Trans. Industr. Electron. 47 (5) (2000) 1005–1014.
[17] A. Heng, S. Zhang, A.C. Tan, J. Mathew, Rotating machinery prognostics: state of the art, challenges and opportunities, Mech. Syst. Signal Process. 23 (3) (2009) 724–739.
[18] S. Fortunato, Community detection in graphs, Phys. Rep. 486 (3) (2010) 75–174.
[19] S. Sarkar, K.L. Boyer, Quantitative measures of change based on feature organization: eigenvalues and eigenvectors, Comput. Vis. Image Understand. 71 (1) (1998) 110–136.
[20] R. Kannan, S. Vempala, A. Vetta, On clusterings: good, bad and spectral, J. ACM (JACM) 51 (3) (2004) 497–515.
[21] X.L. Wang, Comments on detection of undocumented changepoints: a revision of the two-phase regression model, J. Clim. 16 (20) (2003) 3383–3385.
[22] V. Vovk, I. Nouretdinov, A. Gammerman, Testing exchangeability on-line, in: ICML, vol. 12, 2003, pp. 768–775.
[23] S.-S. Ho, H. Wechsler, A martingale framework for detecting changes in data streams by testing exchangeability, IEEE Trans. Pattern Anal. Mach. Intell. 32 (12) (2010) 2113–2127.
[24] J. Doob, Boundary properties of functions with finite Dirichlet integrals, in: Annales de l'institut Fourier, vol. 12, 1962, pp. 573–621.
[25] A. Leman, H. Tong, D. Koutra, Graph based anomaly detection and description: a survey, Data Min. Knowl. Disc. 29 (3) (2015) 626–688.
[26] T. Rato, E. Schmitt, B. De Ketelaere, M. Hubert, M. Reis, A systematic comparison of PCA-based statistical process monitoring methods for high-dimensional, time-dependent processes, AIChE J. 62 (5) (2016) 1478–1493.
[27] J.J. Groen, G. Kapetanios, S. Price, Multivariate methods for monitoring structural change, J. Appl. Econ. 28 (2) (2013) 250–274.
[28] H. Cho, P. Fryzlewicz, Multiple-change-point detection for high dimensional time series via sparsified binary segmentation, J. Roy. Stat. Soc.: Ser. B (Stat. Methodol.) 77 (2) (2015) 475–507.