Abstract: In the field of Process Condition Monitoring (PCM), fault detection has been a fundamental and challenging problem. Particularly in the situation where new kind of operating mode is launched, detecting anomalies is quickly becoming difficult due to lack of sufficient prior knowledge of the newly imported operation. This article introduces a naive and low-complexity approach, Binary classifier For Fault detection (BaFFle), which extract the operating mode in a naive representation and identify process conditions in unconfirnity with expectancy. The proposed method is devised on the basis of Gaussian model assumption and Principal Component Analysis (PCA). The validation of BaFFle is experienced on process data from Cranfield University. Experiments show that, BaFFle can reasonably handle the task of differentiation between normal and abnormal operations.

Keywords: Process condition monitoring, fault detection, PCA, Gaussian model

1. INTRODUCTION

Industrial process monitoring has been a vital role in supervising the health of working plants, especially conducting online, since quick reaction to the abnormal behaviours contributes to immediate stop of unexpected operations, sequentially mitigating further risk of damaging systems. In addition, regular monitoring operations not only indicates the anomalies, but also relates to diagnosis and maintenance functions (Jardine et al. (2006) and Reis and Gins (2017)).

Fault detection is a preliminary step in Process Condition Monitoring (PCM), also of crucial importance. A number of papers spread a comprehensive discussion on the topic of fault detection. Frank (1992) concluded three directions to handle fault detection: model-based methods which engineer specific models, such as first principle models and time-series models, to spot any measurements not fitting into the given models; knowledge-based methods which statistically and analytically infer the process conditions according to data themselves; signal-based methods which extract frequency features of observations for analysing operating modes. In terms of model-based approaches, a major challenge is attributed to modelling normal operating modes, particularly for plants with multiple sub-systems, which is costly and complex. A promising work direction, data analytics, gradually grows into popularity, leading recent researches to seeking proper representation of normal operating conditions analytically. These work can be seen in a large quantity of literatures: Jin et al. (2011a,b) proposed an algorithm based on Goodness-of-Fit to detect the changes occurring in the time series of data, and further expands the algorithm suitable to analysis in frequency domain; a multivariate statistical method, Canonical Variate Analysis with Kernel Density Estimations (CVA-KDE), proposed by Ruiz-Cárcel et al. (2015), is able to fault detection ability in varying operation conditions. With respect to signal-based fault detection algorithms, Ahmar et al. (2010) present a research of comparison between classical periodogram, Welch periodogram, Short-Time Fourier Transform and Wavelet Transform. However, fault detection with frequency-domain features are less practical versus general data-driven methods.

In terms of solving large-scale data issues, data-driven techniques, according to the labelling extend of data, can be broadly categorised into supervised learning and unsupervised learning, respectively serving for classification and clustering task. Fault detection is a special case in classification problems, only having one group of labeled observations; one-class Support Vector Machines (SVMs) (Schölkopf et al. (2001), Ratsch et al. (2002)) is designed to prepare for one-label classification, however a commonly open question in SVMs is about how to determine kernel parameters. Considering the limits of traditional classification algorithms and particularity of anomaly detection, an optimal method is to present operating modes by learning from acquired measurements in an unsupervised learning manner, noting that this method is feasible as measurements of normal operating mode are cheaply and easily obtained. One obvious characteristic of unsupervised learning is that the learning works in a without-human-guide fashion (Barlow (1989), Becker and Plumbley (1996)), representative unsupervised learning algorithms including k-means (Kanungo et al. (2002)) and Independent Component Analysis (Hyvärinen and Oja (2000)).
The algorithm introduced in this paper aims to solve a more special detection scenario in which the detection of the appearances of anomalies is conducted in a new system, without prior preparation of collecting a large amount of measurements. The core of such kind of situation is to quickly extract operating patterns using a small amount of observations. To this end, the analysis of signals in the work of Boashasha et al. (2012) sheds light on the development of process condition monitoring using image processing techniques. Images and process measurements are distinct types of data, however, techniques can be shared by each other given similar context and objectives. For instance, background subtraction is an important research topic in computer vision, aiming to detect objects of interest; likewise, analysis of fault occurrences is to detect the changes in the process measurements. Through the reading of literatures, a background subtraction tool for sequencing video (ViBe) proposed by Barnich and Van Droogenbroeck (2011) offers an unsupervised method of devising the image into foreground and background, and possesses the ability of building up background model with a single image. The proposed fault detection algorithm borrows the thought and workflow of ViBe, conceptualising normal and abnormal operating modes respectively as 'Background' and 'Foreground'.

The contributions of this paper are listed as follows:

1. Develop a low-complexity fault detection algorithm which possesses the capability of quickly constructing a naive model of operating mode
2. Design a detection mechanism using two sets of monitoring limits, one with fixed monitoring coefficient, while another with an adjustable monitoring coefficient
3. Develop an estimation of probability distribution in an adaptive way
4. Propose an averaging filter to reduce the negative effect caused by missed fault
5. Validate proposed algorithm on process measurements collected by Cranfield University

The remainder of this paper consists of three sections. Section 2 introduces and details three main steps of the proposed algorithm. Validation experiments for the proposed algorithm is presented in Section 3, as well as a discussion relating to the experiment results. The last part concludes the study of fault detection using BaFFle as well as future work.

2. METHODOLOGY

The fault detection method introduced in this article is devised for stationary systems, being capable of extracting the working patterns from a new system and performing analysis of fault occurrences. BaFFle algorithm involves three main steps which are initialisation, classification and model updating; the details of algorithm are introduced and explained as follows.

2.1 Initialisation

In the phase of initialisation, the function of monitoring process condition is temporarily and shortly blocked. During this period, a set of observations, \( \{X_1, X_2, \ldots, X_l\} \), are collected; each observation at timestamp \( t \) is a vector, expressed in the form of \( X_t = [x_t^1, x_t^2, \ldots, x_t^n]^T \), where \( n \) is the number of variables. Monitoring function starts working at timestamp \( t + 1 \), called lagged time. It is of note that the period of lagged time is predefined without anomalies. Training data for modelling normal operating mode is \( B = [X_1, X_2, \ldots, X_l] \in \mathbb{R}^{nxl} \).

Here, PCA technique is applied to, from raw process measurements extract principle components as input for parameter estimation of Gaussian distribution. The reason of using PCA is because PCA can minimise the linear correlations between variables and maximum the information entropy over distinct principal extractions; also PCA provides a straightforward way of dimension reduction. The operation of PCA is illustrated as

\[
V = T^TP
\]

where \( T \in \mathbb{R}^{n \times l} \) is the normalised original observations, \( P \in \mathbb{R}^{n \times r} \) is the projection matrix calculated from eigenvalue decomposition of \( T^TP \), and \( V \in \mathbb{R}^{r \times r} \) is the corresponding extracted principal components.

In the proposed algorithm, each principle component follows a one-dimension Gaussian distribution. Then, one kind of operating mode with multivariate can be written as \( Model = [M, Std] \), where \( M = [\mu_1, \mu_2, \ldots, \mu_r]^T \), \( Std = [\sigma_1, \sigma_2, \ldots, \sigma_r]^T \) and \( r = 1, 2, \ldots, k \). \( k \) is the remained number of dimensions. Each element in \( M \) and \( Std \) corresponds to the mean and standard deviation of each column of \( V \).

Starting from the time instant \( t + 1 \), distinct components of observations are obtained as follows

\[
Y_t = T^TP
\]

where \( T \) is normalised observation at time stamp \( t \) and \( Y_t = [y_t^1, y_t^2, \ldots, y_t^r]^T \) is the projection result.

2.2 Classification

In Gaussian distribution, \( 3\sigma \) distance away from the mean is extensively used for drawing 99.7% confidence interval; additionally, a larger interval defined by \( m \pm (k + 3)\sigma \), where \( k \in [0, 1] \), is adopted in this algorithm collaborating with \( 3\sigma \).

In course of monitoring process condition, \( m \pm 3\sigma \) threshold plays a role in warning the occurrences of unsure anomalies. Furthermore, warnings generated by \( m \pm 3\sigma \) determines how to tune the value of \( k \), leading to the fluctuations of \( m \pm (k + 3)\sigma \) threshold. The definitive determination of current health state of system depends on \( m \pm (k + 3)\sigma \) limit. The discrimination of operating mode can be expressed as

\[
E_t^r = \begin{cases} 1, & |y_t^r - \mu_r| > (k + 3)\sigma_r \\ 0, & \text{otherwise} \end{cases}
\]

\[
W_t^r = \begin{cases} 1, & |y_t^r - \mu_r| > 3\sigma_r \\ 0, & \text{otherwise} \end{cases}
\]

where \( W_t^r \) and \( E_t^r \) are the detection results of \( m \pm 3\sigma \) limit and \( m \pm (k + 3)\sigma \) limit respectively.
On each principal component direction, \( \mu_r \) and \( \sigma_r \) draw out the centre and spread of distribution of possible observations. Let \( c \) denote the counts of \((y^r_t - \mu_r) > (3 + k)\sigma_r\) in all the principal components. \( c > \frac{k}{2} \) implies that over half components support the identification of an anomaly, saying that system at time \( t \) is in unhealthy state.

2.3 Model updating

The estimation of probability distribution is a continuously learning process. New observations will make contributions to re-estimating parameters of Gaussian distribution. Meanwhile, the oldest data will be dropped. Considering the possibility of missed fault, it is hardly worth using full value of observations. An averaging filter is formulated for deciding to accept or reject the values of new observations. Eq. 5 details the expressions of the averaging filter in different cases

\[
\hat{y}_t^r = \begin{cases} \alpha y_{t-1}^r + (1 - \alpha)\hat{y}_{t-1}^r, & \text{if } E_t^r = 0, W_t^r = 0 \\ \alpha y_{t-1}^r + (1 - \alpha)b^r, & \text{if } E_t^r = 0, W_t^r = 1 \\ b^r, & \text{if } E_t^r = 1, W_t^r = 1 \\ \end{cases} \tag{5}
\]

where \( b^r \in \{x_{t-1}^r, x_{t-2}^r, \ldots, x_1^r\} \), \( \hat{y}_t^r \) and \( \hat{y}_t^r \) are respectively the filtered and extracted principal components at time \( t \), \( \hat{y}_{t-1}^r \) is extracted principal component at time \( t - 1 \) and \( \alpha \) is an influence factor. Eq. 5 suggests that when \( E_t^r \) and \( W_t^r \) both return no-fault answers, \( \hat{y}_t^r \) combines the information from previous and current state in a certain ratio; when only the lower monitoring threshold captures the fault occurrence, \( (W_t^r = 1) \), averaging filter brings solid data information from training data to mix with \( \hat{y}_t^r \) to generate \( \hat{y}_t^r \); when \( E_t^r = 1 \), apparently \( W_t^r = 1 \), the value of \( \hat{y}_t^r \) will be updated with a random sample from training set.

After filtering, elements in \( M \) and \( Std \) are renewed as

\[
\mu_r = \frac{1}{\omega - 1} \sum_{k=t-\omega+1}^{t} \hat{y}_t^r \tag{6}
\]

\[
\sigma_r = \sqrt{\frac{\sum_{k=t-\omega+1}^{t-1} (\hat{y}_t^r - \mu_r)^2}{\omega}} \tag{7}
\]

where \( \omega \) is window size.

3. EXPERIMENTS AND RESULTS

The validation of algorithm is verified by applying it on the data collected from a multiphase flow rig of Cranfield University who designed fault scenarios to simulate the possible malfunctions in real industrial process. Datasets acquired under steady-state operation conditions are used for experiments, involving four types of faults, respectively air line blockage (A), water line blockage (B), top separator input blockage (C) and open direct bypass (D). Specifications of Cranfield experiments can be found in Ruiz-Cárcel et al. (2015). Table 1 lists out the details of datasets.

### Table 1. Operation conditions for datasets

| Fault types | Datasets | Fault Duration [s] | Fault start [s] | Fault end [s] | Dim. |
|-------------|----------|-------------------|----------------|--------------|------|
| A           | 1-1      | 1447              | 657            | 3777         | 3    |
|             | 1-2      | 4221              | 691            | 3691         | 2    |
| B           | 2-1      | 3496              | 476            | 2656         | 4    |
|             | 2-2      | 3421              | 331            | 2467         | 3    |
| C           | 3-1      | 6272              | 333            | 5871         | 3    |
|             | 3-2      | 10764             | 596            | 9566         | 4    |
| D           | 4-1      | 4451              | 851            | 3851         | 4    |
|             | 4-2      | 3661              | 241            | 3241         | 4    |

### 3.1 Parameters settings

There are four hyper parameters to be specified before running algorithm, as illustrated as follows

1. in theory, longer lag time \( l \) would be helpful with smoothing the influence on the \( M \) and \( \sigma \) caused by system noise and instability at early stage, however, considering the sudden appearance of faults, lag time is supposed to be short. Therefore, lag time is set as \( l = 150 \).

2. window size \( \omega = 150 \) is set as equal as the length of training set in order to calculate \( \mu_r \) and \( \sigma_r \) fairly.

3. the reserved number of principal components, \( r \), is associated with the proportion of total variance explained. The greater cumulative variance means higher information entropy. According to the cut-off rule in Jolliffe (2002), the search of a proper value for \( r \) is in descending order of component variance, stopping at the \( r \)-th component where the cumulative variance accounts for 70% total variance. Also, the determination of \( r \) value is accomplished in initialisation phase. The last column of Table 1 shows the remained number of principal components for each data set.

4. a default value of influence factor \( \alpha \) is 0.5 in experiments. In addition, the investigation of \( \alpha \) value is discussed in section 3.

### 3.2 Metrics

Three metrics are used to evaluate detection performance. Note that in experiments, anomalies are treated as positive, whereas constant operation is as negative. According to the given conditions and detection results, four terminologies are given, true positive(TP), true negative(TN), false positive(FP) and false negative(FN). Here, the calculation of detection accuracy (ACC) and precision (PR) are illustrated as

\[
ACC = \frac{TN + TP}{TN + TP + FN + FP} \tag{8}
\]

\[
PR = \frac{TP}{TP + FP} \tag{9}
\]

The third metric, DT, is used to measure the time gap between fault start and first detection. Additionally, OT introduced in Ruiz-Cárcel et al. (2015) indicates the situation when a false positive occurs before fault starts.

This subsection investigates how \( \alpha \) impacts on the performance of the algorithm. The research range of the values of \( r \) is from 0 to 1, an increment in 0.1. \( alpha = 0 \) implies that filtered principal component \( \hat{y}_t^r \) is completely dependent on the value of previous timestamp, with the increase in
Fig. 1. Experiment results of case 3-1. Subfigures (a)-(c) are the plots for each principal component, and subfigure (d) is the plots of fault detection results.

α value, being less influenced by the previous timestamp. Experiment results are shown in Table. 2, coming with explanations as follows:

1) Dataset 1-1: as α ≥ 0.3 , OT phenomena disappears. The values of ACC and PR are maximum at α = 0.3, falling slightly with the increase in α, ACC from 28.01% to 27.43% and PR from 50.47% to 49.56%. Meanwhile, detections with larger α values take longer time.

2) Dataset 1-2: The metric of ACC goes up and down between 39.15% and 39.32%, while PR has a rising tendency. DT metric performs constantly except at α = 0.1 and α = 0.2.

3) Dataset 2-1: this dataset has trends as similar as dataset 1-1. However, the detection results of dataset 2-1 are more sensitive to the changes of α than data 1-1.

4) Dataset 2-2: Different values of α has less influence on the detection performance in terms of ACC and PR.

5) Dataset 3-1: When α has smaller values, faults can be detected drastically earlier.

6) Dataset 3-2: Except for the OT situations under the condition of α = 0.1, 0.2, 0.3, ACC, PR and DT all perform constantly, achieving high level results.

7) Dataset 4-1: Staring at α = 0.3, the results of ACC, PR and DT have stabilised.

8) Dataset 4-2: Both of ACC and PR show upward trends as the value of α goes up. DT presents good performance at α = 0.1, 0.2, 0.3.

As 0 ≤ α ≤ 0.3, the phenomena of false positives before fault start occurs in most cases, whereas 0.3 ≤ α ≤ 1, ACC, PR and DT stay relatively stable.
Table 2. Fault detection performance under different settings of the value of influence factor \( \alpha \), with respect to Accuracy (ACC), Precision (PR) and Detection Time (DT).

| Influence factor \( \alpha \) | 0.0 | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 1.0 |
|-----------------------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Dataset 1-1 ACC (%) | OT | OT | OT | 28.01 | 27.87 | 27.73 | 27.70 | 27.70 | 27.66 | 27.63 | 27.43 |
| PR (%) | OT | OT | OT | 50.47 | 50.25 | 50.04 | 50.00 | 50.00 | 49.93 | 49.89 | 49.56 |
| DT (s) | OT | OT | OT | 203 | 2429 | 2429 | 2430 | 2430 | 2430 | 2430 | 2434 |
| Dataset 1-2 ACC (%) | 39.32 | 39.30 | 39.32 | 39.37 | 39.27 | 39.17 | 39.15 | 39.22 | 39.20 | 39.20 | 39.20 |
| PR (%) | 88.80 | 89.15 | 89.43 | 89.40 | 89.66 | 89.73 | 89.71 | 90.31 | 90.29 | 90.29 | 90.29 |
| DT (s) | 2432 | 73 | 73 | 2432 | 2423 | 2422 | 2423 | 2422 | 2432 | 2432 | 2432 |
| Dataset 2-1 ACC (%) | OT | 44.14 | 44.11 | 43.75 | 43.18 | 42.22 | 41.99 | 41.81 | 41.78 | 41.81 | 41.81 |
| PR (%) | OT | 83.48 | 83.44 | 83.00 | 82.26 | 80.85 | 80.46 | 80.31 | 80.26 | 80.47 | 80.47 |
| DT (s) | OT | 157 | 75 | 157 | 167 | 1835 | 1835 | 1871 | 1872 | 1872 | 1872 |
| Dataset 2-2 ACC (%) | OT | 53.68 | 52.27 | 52.09 | 51.82 | 51.51 | 51.45 | 51.33 | 51.33 | 51.33 | 51.33 |
| PR (%) | OT | 91.58 | 90.91 | 91.19 | 91.31 | 91.18 | 91.15 | 91.10 | 91.10 | 91.10 | 91.10 |
| DT (s) | OT | 7 | 1060 | 1287 | 1290 | 1523 | 1524 | 1524 | 1524 | 1524 | 1524 |
| Dataset 3-1 ACC (%) | OT | 68.86 | 68.70 | 68.46 | 68.36 | 68.34 | 68.34 | 68.34 | 68.34 | 68.34 | 68.43 |
| PR (%) | OT | 98.87 | 98.87 | 98.86 | 98.86 | 98.86 | 98.89 | 98.89 | 98.91 | 98.99 | 99.07 |
| DT (s) | OT | 339 | 338 | 1887 | 1894 | 1894 | 1894 | 1895 | 1895 | 1895 | 1899 |
| Dataset 3-2 ACC (%) | 95.97 | OT | OT | OT | OT | 97.69 | 97.71 | 97.72 | 97.75 | 97.75 | 97.75 |
| PR (%) | 96.70 | OT | OT | OT | OT | 98.53 | 98.58 | 98.61 | 98.64 | 98.65 | 98.66 |
| DT (s) | 108 | 0 | 0 | 0 | 107 | 107 | 107 | 107 | 107 | 107 | 107 |
| Dataset 4-1 ACC (%) | OT | OT | OT | 81.21 | 81.23 | 81.24 | 81.19 | 81.17 | 81.17 | 81.17 | 81.17 |
| PR (%) | OT | OT | OT | 84.56 | 84.61 | 84.61 | 84.60 | 84.61 | 84.61 | 84.64 | 84.63 |
| DT (s) | OT | 318 | 318 | 318 | 318 | 318 | 318 | 318 | 318 | 318 | 318 |
| Dataset 4-2 ACC (%) | 86.04 | 86.76 | 87.64 | 87.61 | 87.50 | 87.70 | 87.70 | 87.75 | 87.81 | 87.95 | 88.01 |
| PR (%) | 93.82 | 94.43 | 95.40 | 95.46 | 95.49 | 95.73 | 95.73 | 95.80 | 95.86 | 96.04 | 96.10 |
| DT (s) | 313 | 49 | 49 | 313 | 312 | 312 | 312 | 312 | 312 | 312 | 312 |

† OT in the table represents a situation when monitoring indicates the appearance of abnormally before fault seeded. In the framework of the aforementioned BaFFle algorithm, an experiment is designed to have a sight of the relations between influence factor \( \alpha \) and detection performance. Eight datasets of process data of multiphase flow rig from Cranfield University was experienced by setting \( \alpha \) value ranging from 0 to 1, corresponding detection performance showing that small values of \( \alpha \) frequently cause the phenomena of false positive before fault start (marked by OT), even if sometimes generates best performance. In order to avoid the failure resulted from OT, an empirical value of \( \alpha \) is 0.5.

3.3 Experiment Results

Ruiz-Cárcel et al. (2015) provided a benchmark to demonstrate the detection ability of a varies of methods; among these detection techniques, CVA-KDE outperforms the others. Therefore, BaFFle algorithm will be evaluated by comparing with CVA-KDE. Table 3 lists out the results of BaFFle, CVA-KDE (involving two indicators, \( T^2 \) and \( Q \)). In terms of ACC, results of BaFFle are inferior to the ones of CVA-KDE in datasets 1-1 and 1-2, but turn out comparable results in datasets 2-1, 2-2 and 3-2, and outperform in datasets 3-1, 4-1 and 4-2. Particularly, classification accuracy demonstrates significant increases when \( \hat{y}_t \) takes less ratio of previous time instant. Despite succeed in achieving early detection with lower value of influence factor, the other detection metrics present a loss performance. The reason is because faults accumulate over time, which requires a period of time resulting in physical manifestation. To balance relations between DT and the other metrics, an empirical influence faces is recommended as 0.5.
Table 3. Experiment results for fault detection with BaFFle, CVA-KDE($T^2$) and CVA-KDE(Q). Values in bold are the best performed results.

| Datasets | Algorithm  | Accuracy [%] | Precision [%] | Detection Time [s] |
|----------|-----------|--------------|---------------|-------------------|
| 1-1      | BaFFle    | 27.73        | 50.04         | 2429              |
|          | CVA-$T^2$ | 64.23        | 97.70         | 1506              |
|          | CVA-Q     | **70.88**    | **97.75**     | **1213**          |
| 1-2      | BaFFle    | 39.18        | 89.73         | 2432              |
|          | CVA-$T^2$ | **55.91**    | **97.47**     | **1808**          |
|          | CVA-Q     | 55.69        | 96.76         | 1808              |
| 2-1      | BaFFle    | 42.23        | 80.85         | 1835              |
|          | CVA-$T^2$ | **45.22**    | **94.62**     | **1812**          |
|          | CVA-Q     | 32.64        | 4.94          | **1811**          |
| 2-2      | BaFFle    | 51.82        | 91.31         | **1290**          |
|          | CVA-$T^2$ | **53.23**    | **97.20**     | 1512              |
|          | CVA-Q     | 34.12        | 30.43         | 1508              |
| 3-1      | BaFFle    | **68.34**    | 98.86         | 1894              |
|          | CVA-$T^2$ | 50.52        | **99.02**     | 2924              |
|          | CVA-Q     | 46.85        | 98.89         | **662**           |
| 3-2      | BaFFle    | 97.71        | 98.58         | 107               |
|          | CVA-$T^2$ | **98.54**    | **99.45**     | **106**           |
|          | CVA-Q     | 96.17        | 98.88         | 309               |
| 4-1      | BaFFle    | **81.24**    | 84.61         | **319**           |
|          | CVA-$T^2$ | 43.43        | 89.34         | 436               |
|          | CVA-Q     | 61.17        | **93.58**     | 430               |
| 4-2      | BaFFle    | **87.70**    | **95.73**     | **312**           |
|          | CVA-$T^2$ | 36.83        | 90.82         | 380               |
|          | CVA-Q     | 62.18        | 95.64         | 371               |

in datasets 3-1, 3-2 and 4-1, respectively from 46.85% to 68.34%, from 43.43% to 81.24% and from 36.83% to 87.70%. As for PR and DT, although BaFFle cannot totally beat CVA, in certain cases, it can still approach a level close to the performances of CVA-KDE. Here, dataset 3-1 is taken as an example to have a further understanding of the behaviours in this binary classifier algorithm.

Fault detection results of dataset 3-1 with $\alpha = 0.5$ present in Figure 1, subfigures 1(a)-1(c) plotting out means, monitoring thresholds and values of principle components on distinct components, and subfigure 1(d) shows analysis of detection results. From subfigure 1(d), it can be seen that determination mechanism efficiently eliminates false positives before faults start and equivocal fault detection on second components. Particularly comparing with the results plots of CVA-KDE in Fig. 3, BaFFle performs persistent detection ability. Nevertheless, because of weak sensitivity to fault end resulting from the windowing function, determination mechanism can only gently mitigate the false positives after fault end.

4. CONCLUSION

This paper provides a new algorithm for fault detection in the industrial process monitoring. According to existing background subtraction framework, we devised a binary classifier for on-line fault detection. One important advantage of this approach is that it possesses the ability of fast learning operating mode and achieving a naive representation. Thus, BaFFle 1) drastically reduces the workload of data collection and high cost of training, 2) succeeds in lowering the barrier to adapt to strange systems; and requires low online computation work. Above, this algorithm by simply equipped with PCA and Gaussian model presents reasonably well-performed detection work, furthermore showing potentials to cooperate with other detection approaches by embedding into existing monitoring system to improve detection accuracy. However, there are still some limits existing in the proposed BaFFle algorithm: 1) BaFFle is a fault detection algorithm more about signalling the occurrences of abnormalities, a challenge boiling down to how to improve the algorithm for providing reliable decision support, and 2) the naive modelling of BaFFle algorithm underperforms with high complexed systems since BaFFle converts nonlinear measurements to linear models in a brute-force way, losing the capacity of capturing the multivariate correlations. Future work will focus on solving following questions. Firstly, we shall explore the methods of accurately representing operating mode instead of Gaussian model. Secondly, we shall solve the long-memory problem caused by windowing operation, thereby reducing the rate of false positives after fault end. Thirdly, we shall expand current work to varying operation conditions.

REFERENCES

Ahmar, E.E., Choqueuse, V., Benbouzid, M., Amirat, Y., Assad, J.E., Karam, R., and Farah, S. (2010). Advanced signal processing techniques for fault detection and diagnosis in a wind turbine induction generator drive train: A comparative study. *IEEE Energy Conversion Congress and Exposition*, 3576–3581.

Barlow, H. (1989). Unsupervised learning. *Neural Computation*, 1, 295–311.

Barnich, O. and Van Droogenbroeck, M. (2011). ViBe: A universal background subtraction algorithm for video sequences. *IEEE Transactions on Image Processing*, 20(6), 1709–1724.

Becker, S. and Plumbley, M. (1996). Unsupervised neural network learning procedures for feature extraction and classification. *International Journal of Applied Intelligence*, 6, 185–203.

Boashasha, B., Noubchir, L., and Azemi, G. (2012). A methodology for time-frequency image processing applied to the classification of non-stationary multichannel
signals using instantaneous frequency descriptors with
application to newborn eeg signals. EURASIP Journal
on Advances in Signal Processing.

Frank, P. (1992). Principles of model-based fault detection.
Proceedings of International Symposium on AI in Real-
time Control, 363–370.

Hyvärinen, A. and Oja, E. (2000). Independent component
analysis: Algorithm and applications. Neural Netw., 13,
411–430.

Jardine, A.K., Lin, D., and Banjevic, D. (2006). A review
on machinery diagnostics and prognostics implementing
condition-based maintenance. Mechanical Systems and
Signal Processing, 1483–1510.

Jin, Y., Tebekaemi, E., Berges, M., and Soibelman, L.
(2011a). Robust adaptive event detection in non-
intrusive load monitoring for energy aware smart fac-
cilities. Proceedings of the Signal Processing, Sensor
Fusion, and Target Recognition XX, 4340–4343.

Jin, Y., Tebekaemi, E., Berges, M., and Soibelman, L.
(2011b). A time-frequency approach for event detection
in non-intrusive load monitoring. Proceedings of the Sig-
nal Processing, Sensor Fusion, and Target Recognition
XX, 80501–80501u.

Jolliffe, I. (2002). Principal Component Analysis. Springer,
New York.

Kanungo, T., Mount, M., D., Netanyahu, S., N., Piatko,
D., C., Silverman, R., and Wu, Y.A. (2002). An efficient
k-means clustering algorithm: Analysis and implementa-
tion. IEEE Transactions on Pattern Analysis and
Machine Intelligence, 24(7), 881–892.

Ratsch, G., Mika, S., Scholkopf, B., and R., M.K. (2002).
Constructing boosting algorithms from svms: An appli-
cation to one-class classification. IEEE Transactions on
Pattern Analysis and Machine Intelligence 24, 9, 1184–
1199.

Reis, M.S. and Gins, G. (2017). Industrial process moni-
toring in the big data/industry 4.0 era: From detection,
to diagnosis, to prognosis. Processes, 5, 35.

Ruiz-Cárce1, C., Cao, Y., Mba, D., Lao, L., and Samuel, R.
(2015). Statistical process monitoring of a multiphase
flow facility. Control Engineering Practic1e.

Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J.,
and Williamson, R.C. (2001). Estimating the support of
a high-dimensional distribution. Neural Compute, 13(7),
1443–1471.