Distributed Parallel PCA for Modeling and Monitoring of Large-Scale Plant-Wide Processes With Big Data

Jinlin Zhu, Zhiqiang Ge, *Member, IEEE*, and Zhihuan Song

***Abstract*—In order to deal with the modeling and monitoring issue of large-scale industrial processes with big data, a distributed and parallel designed principal component analysis approach is proposed. To handle the high-dimensional process variables, the large-scale process is first decomposed into distributed blocks with *a priori* process knowledge. Afterward, in order to solve the modeling issue with large-scale data chunks in each block, a distributed and parallel data processing strategy is proposed based on the framework of MapReduce and then principal components are further extracted for each distributed block. With all these steps, statistical modeling of large-scale processes with big data can be established. Finally, a systematic fault detection and isolation scheme is designed so that the whole large-scale process can be hierarchically monitored from the plant-wide level, unit block level, and variable level. The effectiveness of the proposed method is evaluated through the Tennessee Eastman benchmark process.**

***Index Terms*—Bayesian fusion, big data, distributed and parallel principal component analysis (dpPCA), hierarchical process monitoring, MapReduce.**

I. Introduction

PROCESS modeling and monitoring have become one of the most popular academic trends due to the great advancement of data generation and storage abilities in industrial systems over the past few decades [1]–[3]. The aim of process modeling and monitoring is to enhance process safety and production quality of the manufacturing processes. For small-scale systems with explicit process details, traditional model-based first principal methods could be preferred [4]. However, the model-based mechanism may encounter a big challenge in large-scale plant-wide modeling situations due to the complex manufacturing procedures and implicit process kinetic mechanisms [5]. As feasible alternatives, data-based methods with fewer requirements on process knowledge have attracted widespread attentions [6]–[11].

Unlike traditional small-scale process with centralized monitoring techniques, those decentralized or distributed methods with multiblock decompositions are usually specified for plant-wide processes [12]. Several advantages can be speculated [13]. First, the process decomposition can largely reduce the computational loadings since each modeling subblock is employed by a smaller set of relevant variables. Second, by dividing multiple blocks, one can detect faulty events in a distributed manner and the most responsible faulty sections as well as faulty variables can be isolated, respectively. Third, through building multiple distributed blocks, one may in turns enhance the robustness of the entire statistic model against local process maintenance [5]. In other words, regular maintenance and update in certain blocks may have a little impact on other constructed blocks. On the contrast, those data-based models constructed with centralized schemes have to be totally discarded once some minor changes have been made on even one local unit. In this framework, several data-based models based on multiblock and hierarchical principal component analysis (PCA) and partial least squares have been provided and explored in previous studies [14], [15]. These studies show that the multiblock methods can deal with those plant-wide processes in a more flexible manner.

It should be mentioned that the current multiblock methods have been focused on the variable-wise decomposition to deal with the large-scale plant-wide industrial process. However, the sample-wise decomposition has been rarely considered. Almost all the previously studied multiblock methods are based on single-line algorithms that are aimed at deriving one-shot solutions for all data samples. Therefore, the entire data mass should be stored in local computing node, preprocessed, and then, loaded in memory before user-defined numerical calculations. This procedure works for situations when data sizes are moderate for computer memory. With the development of the internet of things, wireless communication technologies, and also the advancement of smart manufacturing, huge amount of data have been collected and stored [16]. Accordingly, the whole industry has been overwhelmed with the increasing data size and has entered the era of big data [17], [18]. On one hand, in the era of big data, all involved traditional methods, such as locality data storage, centralized data preprocessing, and single-line statistical modeling methods are based on memory residency, and,

Manuscript received October 20, 2016; revised December 22, 2016; accepted January 21, 2017. Date of publication January 26, 2017; date of current version August 1, 2017. This work was supported in part by the National Natural Science Foundation of China (61573308). Paper no. TII-16-1189. *(Corresponding author: Z. Ge.)*

The authors are with the State Key Laboratory of Industrial Control Technology, College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: wx_zjl@126.com; gezhiqiang@zju.edu.cn; zhsong@iipc.zju.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2017.2658732

hence, should become cumbersome. For this reason and in order to provide fast and cost-effective solutions to deal with the storage and analytic issue, both distributed and parallel mechanisms should be taken into consideration. On the other hand, the monitoring mechanism should also be improved. Under the plant-wide process background, it should be extremely inconvenience to monitor hundreds or thousands of variables in a synchronous manner [5]. As a matter of fact, one should consider the best visualization of most important and relevant information once the abnormal event has taken place so as to inform the operators and take immediate responses to target malfunctions.

The motivation of this study is to propose a distributed and parallel big data modeling and hierarchical monitoring framework based on the PCA. As the most popular distributed and parallel infrastructure, Hadoop MapReduce has been widely studied [19]. On one hand, for big data distributed storage, the Hadoop distributed file system (HDFS) can be commonly established. On the other hand, for parallel computation in the distributed environment, the MapReduce paradigm is especially desirable [16]. MapReduce can be viewed as a simplified data processing platform for dealing with big data with a cluster of compute nodes. In previous works, several popular algorithms have been adapted and parallelized with MapReduce model, such as support vector machine [20], neural network [21], and random forest [22]. However, for statistical plant-wide process modeling and monitoring, few investigations can be found. An intuitive idea is to combine MapReduce with multiblock PCA. MapReduce can be integrated for constructing PCA in each block with big data chunk, while the multiblock PCA can be applied for monitoring large-scale industrial process. It should be mentioned that for a successful application of the industrial process modeling, some issues still need to be considered. First, one should also design PCA models for each distributed block so that the entire statistical modeling procedure can be assigned to distributed compute node. Second, monitoring results from the constructed distributed and parallel principal component analysis (dpPCA) should be fused with proper designed scheme so as to provide hierarchical operating views from plant-wide level, block level, and variable level.

In this study, a distributed and parallel data statistical modeling algorithm is implemented within the MapReduce framework. Based on that the big data in a certain unit block can be assigned into several distributed compute nodes, and, hence, the speed should be increased and the memory resident issue could be significantly alleviated. Subsequently, a statistic combination strategy is induced so that the intermediate results from each block can be combined into the global result of the entire dataset. Afterward, we show that dpPCA can be easily developed based on the derived calculated statistics. Therefore, the big data modeling issue can be easily realized on one or several connected personal computers. For monitoring purpose, based on the constructed dpPCA for each distributed block, the distributed monitoring results have been fused with a Bayesian fusion strategy so as to provide operating status of both plant-wide level and block level. For the variable level contribution, the reconstruction-based contribution (RBC) method has been incorporated [23]. Therefore, under the proposed distributed and parallel framework, PCA can be applied for modeling large-scale plant-wide process with big data. Besides, the hierarchical monitoring mechanism can also provide informative views on multilevel industrial process status in a data-friendly way.

The rest of this paper is organized as follows. Section II presents some preliminaries includes PCA and MapReduce. Sections III and IV provide distributed block decomposition, parallel PCA modeling, and hierarchical monitoring mechanisms. Simulation results on the Tennessee Eastman process (TE process) are shown in Section V. Section VI gives out conclusions and outlooks.

II. Preliminaries

A. PCA Revisit

PCA seeks an orthogonal transformation procedure that extracts a set of linearly uncorrelated principal components from the possibly correlated original variables. Suppose a set of observations have been collected and are denoted with data matrix $\mathbf{X} \in R^{N\times D}$, the structure of PCA can be given as [24]:

$$\mathbf{X} = \mathbf{T}\mathbf{P}^T + \mathbf{E} \tag{1}$$

where $\mathbf{T} \in R^{N\times d}$ is the matrix for orthogonal principal components, $\mathbf{P} \in R^{D\times d}$ is the transform matrix, d is the selected number of principal components, and the offset term $\mathbf{E}$ is the modeling residuals. Typically, several numerical ways can be found for calculating the transform matrix: eigenvalue decomposition of observation covariance matrix or correlation matrix, or simply conduct singular value decomposition (SVD) on the original data matrix $\mathbf{X}$. Afterward, all eigenvalues or singular values which representing variance levels are then sorted in descending sequence to determine the most informative principal components by the criterion called cumulative percent variance (CPV) [24].

B. MapReduce Framework

MapReduce is a programming platform for modeling and analyzing those massive amounts of data in the parallel manner [19]. The compute nodes in MapReduce contain one master node (namenode) and several slave nodes (datanode). The master node takes in charge of the entire file system, while each slave node is a worker node. In the meanwhile, the designed runtime system based on HDFS is designed with implicit mechanisms and can automatically deal with data splitting, parallel task scheduling/monitoring, parallel compute node communication management, and also provide data redundancy and fault tolerance mechanisms [25]. In other words, the platform is highly functional and the developers only need to focus on the explicit expressions of two functions: Map and Reduce. Specifically, in each slave node, the computations with respect to Map and Reduce phases both take the data structure in the organized form of <*key, value*> pairs, following the commonly steps:

$$Map : < key1,\ value1 > \rightarrow\ list < key2,\ value2 >$$

$$Reduce : < key2,\ list\,(value2) > \rightarrow list < key2,\ value3 >$$

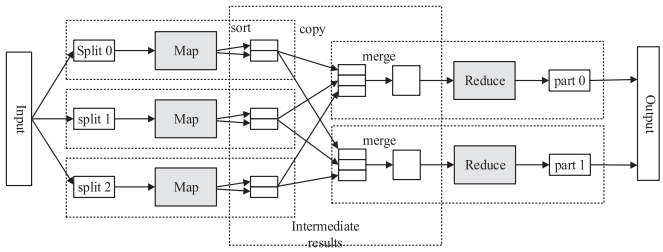


Fig. 1. Illustration for a data flow in MapReduce with three map workers and two reduce workers [19].

With the Map function, each worker node takes the initial organized <*key1, value1*> pairs and produces a list of intermediate <*key2, value2*> pairs. Then, the activated Reduce function takes the pairs to compute the expected <*key2, value3*> pair lists. It should be noted that the MapReduce system shuffles those intermediate results by lists of those same-key pairs with an implicit set of procedures including sort, copy, and merge steps. The shuffled lists of pairs are grouped by the specific keys denoted as <*key2, list(value2)*> and then passed down to the Reduce function. Once the Reduce has been accomplished, the MapReduce system gets Reduce-output results by *key2* and outputs the final outcome. For illustration, the above data flow in MapReduce has been shown in Fig. 1. For more details about MapReduce, one can refer to the relevant literature [19].

III. Distributed and Parallel PCA for Plant-Wide Process Modeling

In this section, a brief discussion is first given on process decomposition. Afterward, parallel computation algorithms regarding parallel big data statistics calculation and dpPCA modeling are proposed.

A. Plant-Wide Process Decomposition

As previously mentioned, the large-scale plant-wide process needs to be decomposed into distributed blocks so as to be beneficial to at least three specific phases: the computing loadings reduction in modeling phase, the distinguished faulty isolation ability in monitoring phase, and the systematic modeling robustness enhancement in the subsequent process/model maintenance phase. The prerequisite for the second and last virtues, however, should be achieved by minimizing the potential simultaneous failures of various variable blocks. In previous studies, the distributed blocks could be derived based on the data-driven methods, such as the principal component decomposition proposed in [24] and the mutual-information-based method proposed in [26]. The former directly conduct variable blocks based on the orthogonal directions of principal components, while the latter intends to compare all mutual information among the entire variable space so as to cluster variables with strong relations into the same block. Although with some merits, such as simple and flexible, a typical issue for these pure data-based methods is the fact that they cannot guarantee the robustness of the constructed statistical model against local maintenance. The variables in each block could be potentially extracted from different unit blocks in industrial process. Therefore, potential simultaneous failures of various variable blocks may result in the issue of multiple model maintenance. In practical, process domain knowledge, such as process unit connections and control feedback constraints from the system flowchart, can be available in most cases. From this perspective, this study conducts process decomposition based on the fact that a reasonable decomposition could be conducted directly on the basis of physical operating unit blocks.

B. Distributed and Parallel Modeling for PCA

In a traditional PCA method, before one can perform PCA construction, the data normalization should be required due to the fact that PCA is sensitive to the relative scaling of various variables. In practice, each process variable should be normalized to zero means and unit variance to get rid of the scale differences among different variables. For this purpose, one has to compute the mean and variance for each process attribute. However, in the era of big data, due to the massive data size for each attribute, the batch computation could be infeasible and the parallel computation should be considered. Please recall that the MapReduce framework considers data splitting so as to assign distributed nodes for parallel computation works. Hence, the critical task of MapReduce is the design of proper <key, value> pairs for the Map and Reduce procedures, respectively. In other words, one should take mean and variance calculations in the Map phase and then consider proper global combination in the Reduce step.

Specifically, first assume the process has been decomposed into Q blocks 1, 2, ..., Q. In addition, a set of S_q variables have been collected for block q as $V_q = \{X_{q1}, X_{q2}, ..., X_{qS_q}\}$. For block q, the data splits which should be originally stored with the HDFS are called into the Map workers as a sequence of <key, value> pairs. Here, the default "key" is commonly referred to the offset in bytes of each data sample to the start point of the data file and the "value" is the data sample value. Afterward, assume a total of M Map functions have been activated and Map work node m is assigned with data chunk $\mathbf{X}_{qm} \in R^{N_{qm} \times S_q}$ where N_{qm} is the sample number. The intermediate result for a Map node m is designed as a pair with the key defined as sample number, variable mean vector, variable covariance matrix, and the corresponding values are $\{N_{qm}, \bar{\mathbf{x}}_{qm}, \Sigma_{qm}\}$, where $\bar{\mathbf{x}}_{qm} = (\bar{x}_{qm1}, \bar{x}_{qm2}, ..., \bar{x}_{qmS_q})$ is the mean vector for variable set V_q and Σ_{qm} is the corresponding covariance matrix. In the Reduce work, all these intermediate results should be combined to the global statistics. Specifically, we needs to consider the following problem: Given two set of values $\{N^1_{qm}, \bar{\mathbf{x}}^1_{qm}, \Sigma^1_{qm}\}$ and $\{N^2_{qm}, \bar{\mathbf{x}}^2_{qm}, \Sigma^2_{qm}\}$ analyzed from two data split $\mathbf{X}^1_{qm}$ and $\mathbf{X}^2_{qm}$, how to compute the global results $\{N^c_{qm}, \bar{\mathbf{x}}^c_{qm}, \Sigma^c_{qm}\}$ of $\{\mathbf{X}^1_{qm}, \mathbf{X}^2_{qm}\}$. For the combined samples number and combined mean vector, one can directly induce as

$$N^c_{qm} = N^1_{qm} + N^2_{qm} \tag{2}$$

$$\bar{\mathbf{x}}_{mc} = \frac{N^1_{qm}\bar{\mathbf{x}}^1_{qm} + N^2_{qm}\bar{\mathbf{x}}^2_{qm}}{N^c_{qm}}. \tag{3}$$

TABLE I
ALGORITHM 1

Algorithm1: (Map/Reduce function for parallel computation of sample number, mean and variance).

Map function

Input: <key, value> pairs by the offset (key) and the data sample (value).

Output: <key, value> pairs with 'keys' refer to data sample number, variable mean, variable variance and 'values' are assigned with values of keys.

Start:

Step 1. Compute number of data samples;

Step 2. Compute mean value for each variable;

Step 3. Compute variance for each variable;

Step 4. Construct key-values as data split sample number, data split sample mean value and data split sample variance;

Step 5. Emit intermediate <key, value> pairs.

End

Reduce function

Input: <key, value> pairs from Map function.

Output: <key, value> pairs with 'keys' denoted as combined sample number, combined sample mean vector, combined sample variances and 'values' are the exact numerical results of keys.

Start:

Step 1. While has next intermediate <key, value> pairs;

Compute combined sample number by Eq.(2);

Compute combined sample mean vector by Eq.(3);

Compute combined sample variance by Eq.(4);

End while

Step 2. Construct output key-values as global sample number; global sample mean and global sample variance.

Step 3. Emit output <key, value>pairs.

End

For variance, we have (please see Appendix for details)

$$\Sigma_{qm}^{c} = \frac{1}{N_{qm}^{c}} \left\{ N_{qm}^{1} \left[\Sigma_{qm}^{1} + \left(\bar{\mathbf{x}}_{qm}^{1} - \bar{\mathbf{x}}_{mc} \right)^{T} \left(\bar{\mathbf{x}}_{qm}^{1} - \bar{\mathbf{x}}_{mc} \right) \right] + N_{qm}^{2} \left[\Sigma_{qm}^{2} + \left(\bar{\mathbf{x}}_{qm}^{2} - \bar{\mathbf{x}}_{mc} \right)^{T} \left(\bar{\mathbf{x}}_{qm}^{2} - \bar{\mathbf{x}}_{mc} \right) \right] \right\}. \tag{4}$$

Please notice that the squared variance for each variable can be derived by extracting diagonal elements as $\Lambda_{qm}^{c} = \mathrm{diag}(\Sigma_{qm}^{c})$ where $\mathrm{diag}(.)$ keeps all off-diagonal elements to zeros. With the defined parallel computation and combining mechanism, the pseudocode of the Map function and Reduce function for each data split is given in Table I.

With the above Map and Reduce procedures have been implemented and combined in multiple nodes, the mean vector and covariance matrix for the entire big data split in each distributed block can be obtained. Assume that the mean vector and covariance matrix for the big data mass in block q are $\bar{\mathbf{X}}_{qo}$ and Σ_{qo}, respectively. One can transform the covariance matrix Σ_{qo} in process block q into correlation matrix Σ_{q} as

$$\Sigma_{q} = \left[\mathrm{diag}\left(\Sigma_{qo} \right) \right]^{-\frac{1}{2}} \Sigma_{qo} \left\{ \left[\mathrm{diag}\left(\Sigma_{qo} \right) \right]^{-\frac{1}{2}} \right\}^{T}. \tag{5}$$

Based on the transformed correlation matrix, one can perform eigendecomposition to find the principal component directions and construct the PCA model for each block. Meanwhile, the CPV can be used for determining the reasonable latent space dimensionalities for each individual block. As an illustration, the entire distributed and parallel modeling flowchart for dpPCA is given in Fig. 2.

IV. HIERARCHICAL MONITORING WITH DPPCA

Once the dpPCA models have been established on various blocks, one can apply into the plant-wide process monitoring. However, for a large-scale process, it should be far from enough to simply detect and report the occurrence of a fault since the troubleshooting over the plant-wide process could be tremendous and time consuming. Therefore, it should be meaningful to provide hierarchical monitoring charts from plant wide to variable level so as to provide the informative visualization of operating conditions over hundreds or even thousands of variables. For a given data sample $\mathbf{x}_n$, one should first allocate the variables into variable blocks in accordance with the process decomposition as $\mathbf{x}_n = \{\mathbf{z}_{n,1}, \mathbf{z}_{n,2}, ..., \mathbf{z}_{n,Q}\}$ and $\mathbf{z}_{n,q}$ is the sample vector assigned for block q.The score vector in each block can be calculated as

$$\mathbf{t}_{n,q} = P_{n,q}^{T} \mathbf{z}_{n,q},\ q = 1, 2, ..., Q. \tag{6}$$

After then, $T_{n,q}^{2}$ and $SPE_{n,q}$ statistics can be constructed as

$$T_{n,q}^{2}\left(\mathbf{z}_{n,q} \right) = \sum_{j=1}^{K_q} \frac{t_{n,q,j}^{2}}{\lambda_{q,j}} \tag{7}$$

$$SPE_{n,q}\left(\mathbf{z}_{n,q} \right) = \mathbf{z}_{n,q} \left(I - \mathbf{P}_{q} \mathbf{P}_{q}^{T} \right) \mathbf{z}_{n,q}^{T} \tag{8}$$

where $\lambda_{q,j}$ is the jth eigenvalue of Σ_q, $t_{n,q,j}$ is the jth element of vector $\mathbf{t}_{n,q}$, K_q is the selected number of principal components for block q, and $\mathbf{P}_q$ refers to the PCA projection matrix for block q. The control limits are

$$T_{q,\mathrm{lim}}^{2} = \frac{K_q \left(N - 1 \right)}{N - K_q} F_{K_q, (N - K_q), \alpha} \tag{9}$$

$$SPE_{q,\mathrm{lim}} = \theta_1 \left[1 + \frac{c_\alpha \sqrt{2\theta_2 h_0^2}}{\theta_1} + \frac{\theta_2 h_0 \left(h_0 - 1 \right)}{\theta_1^2} \right]^{1/h_0} \tag{10}$$

where $h_0 = 1 - (2\theta_1\theta_3)/(3\theta_2^2)$, $\theta_r = \sum_{j=K_q+1}^{S_q} \lambda_{q,j}^{r}$ for $r = 1, 2, 3$, and α is the significance level, and c_α is the normal deviation with upper $1 - \alpha$ percentile.

By far, the monitoring statistics for the block level has been established. However, in order to combine the local statistics into global plant-wide level statistics, one needs to perform Bayesian fusion. First, for a new data vector $\mathbf{z}_{q,\mathrm{new}}$, the T^2 and SPE statistics can be transformed to the normal (denoted by "N") and faulty (denoted by "F") likelihoods, respectively, by

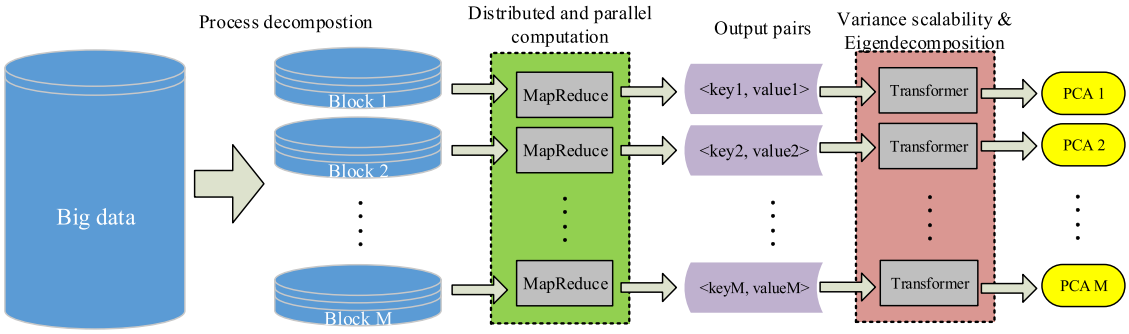


Fig. 2. Distributed and parallel modeling of dpPCA for plant-wide process with big data.

the defined control limits as

$$P^q_{T^2}\left(\mathbf{z}_{q,\text{new}}|N\right) = \exp\left\{-\frac{T^2_{q,\text{new}}}{T^2_{q,\text{lim}}}\right\}, P^q_{T^2}\left(\mathbf{z}_{q,\text{new}}|F\right) = \exp\left\{-\frac{T^2_{q,\text{lim}}}{T^2_{q,\text{new}}}\right\} \tag{11}$$

$$P^q_{SPE}\left(\mathbf{z}_{q,\text{new}}|N\right) = \exp\left\{-\frac{SPE_{q,\text{new}}}{SPE_{q,\text{lim}}}\right\}, P^q_{SPE}\left(\mathbf{z}_{q,\text{new}}|F\right) = \exp\left\{-\frac{SPE_{q,\text{lim}}}{SPE_{q,\text{new}}}\right\}. \tag{12}$$

Then, the Bayes rule can be involved to calculate the posterior probabilities as [5]:

$$P^q_{T^2}\left(F|\mathbf{z}_{q,\text{new}}\right) = \frac{P^q_{T^2}\left(\mathbf{z}_{q,\text{new}}|F\right)P^q_{T^2}\left(F\right)}{P^q_{T^2}\left(\mathbf{z}_{q,\mathbf{new}}|F\right)P^q_{T^2}\left(F\right)+P^q_{T^2}\left(\mathbf{z}_{q,\text{new}}|N\right)P^q_{T^2}\left(N\right)} \tag{13}$$

$$P^q_{SPE}\left(F|\mathbf{z}_{q,\text{new}}\right) = \frac{P^q_{SPE}\left(\mathbf{z}_{q,\text{new}}|F\right)P^q_{SPE}\left(F\right)}{P^q_{SPE}\left(\mathbf{z}_{q,\text{new}}|F\right)P^q_{SPE}\left(F\right)+P^q_{SPE}\left(\mathbf{z}_{q,\text{new}}|N\right)P^q_{SPE}\left(N\right)} \tag{14}$$

where priors can be given in advance by defining a significance level α, and $P^q_{T^2}(N) = P^q_{SPE}(N) = 1-\alpha$, $P^q_{T^2}(F) = P^q_{SPE}(F) = \alpha$.

Once all the fault probabilities have been derived for all blocks, one should consider the combination of each block into the global plant-wide monitoring statistics. Specifically, the likelihood of fault in each block can be regarded as the weight and the plant-wide faulty index (PFI) in latent space and residual space can be calculated as

$$\text{PFI}_{T^2,\mathbf{z}_{\text{new}}} = P_{T^2}\left(F|\mathbf{z}_{\text{new}}\right) = \sum_{q=1}^{Q}\frac{P^q_{T^2}\left(\mathbf{z}_{q,\text{new}}|F\right)P^q_{T^2}\left(F|\mathbf{z}_{q,\text{new}}\right)}{\sum_{q=1}^{Q}P^q_{T^2}\left(\mathbf{z}_{q,\text{new}}|F\right)} \tag{15}$$

$$\text{PFI}_{SPE,\mathbf{z}_{\text{new}}} = P_{SPE}\left(F|\mathbf{z}_{\text{new}}\right) = \sum_{q=1}^{Q}\frac{P^q_{SPE}\left(\mathbf{z}_{q,\text{new}}|F\right)P^q_{SPE}\left(F|\mathbf{z}_{q,\text{new}}\right)}{\sum_{q=1}^{Q}P^q_{SPE}\left(\mathbf{z}_{q,\text{new}}|F\right)}. \tag{16}$$

Note that the control limit for the above PFI equals to the predefined significance level α. Once a fault has been detected, one needs to consider the fault isolation. Fault isolation tries to find the most responsible blocks as well as variables for the abnormal process so that operators can take the corresponding measurements. Please note that different with small-scale processes which only consider variable contributions, one has to consider both block contribution and variable contribution based on the distributed monitoring mechanism of plant-wide processes.

Since the previous monitoring mechanism has been based on the block level, the contribution for each block can be easily constructed. Suppose a series of N potential faulty samples $\{\mathbf{x}_n\}_{n=1}^N$ have been recorded, let $\Gamma_{N,q}$ counts the number of times block q exceeded the given control limit, then one gets

$$\Gamma_{N,q} = \sum_{n=1}^{N}\tau_{n,q},\ \tau_{n,q} = \begin{cases}1, & P^q_{T^2}\left(F|\mathbf{z}_{q,\text{new}}\right)\geq\alpha \ \text{ or } \ P^q_{SPE}\left(F|\mathbf{z}_{q,\text{new}}\right)\geq\alpha\\ 0, & \text{otherwise}\end{cases}. \tag{17}$$

Therefore, block faulty index BFI_q can be defined by normalization as

$$\text{BFI}_q = \Gamma_{N,q}\left(\sum_{q=1}^{Q}\Gamma_{N,q}\right)^{-1}. \tag{18}$$

For variable contribution, the RBC method can be used. Specifically, the contribution $\text{RBC}_{i,n}$ for variable i in sample $\mathbf{x}_n$ should be

$$\text{RBC}^{T^2}_{i,n} = \left(\xi_i^T\Phi\mathbf{x}_n\right)^2\left(\xi_i^T\Phi\xi_i\right)^{-1} \tag{19}$$

where ξ_i is the ith column off the identity matrix $\mathbf{I}_{K_q}$ (assume variable i is allocated in block q). Please note that

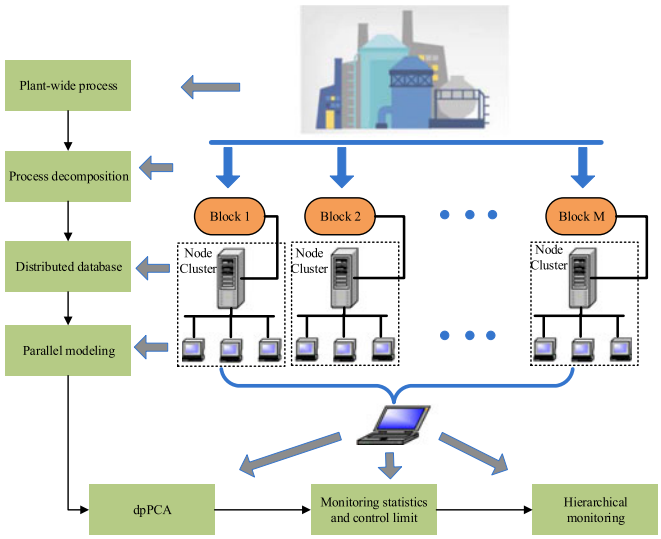


Fig. 3. Outline of modeling and monitoring flowchart.

$\Phi = \mathbf{P}_q \Lambda_q^{-1} \mathbf{P}_q^T$ for T^2 and $\Phi = \mathbf{I}_{K_q} - \mathbf{P}_q \mathbf{P}_q^T$ for *SPE*, Λ_q is the diagonal matrix containing enginvalues for Σ_q. For more details about the RBC method, one can refer to literatures [23].

As a summarization, the corresponding schematic modeling and monitoring flowchart illustration is depicted in Fig. 3.

V. Industrial Application

In this section, the effectiveness of the proposed method is investigated on the plant-wide TE process [27]. The TE process has been a benchmark based on a real industrial plant for evaluation of multiple traditional data-driven modeling and monitoring methods [28]. The process working flowchart can be found in the corresponding literature [27]. Specifically, the entire process contains five units including a reactor, a condenser, a compressor, a separator, and a stripper. The process consists of eight components and two products [29]. The entire process is involved by 41 measured process variables and 12 manipulated variables. In this study, a total of 31 variables as listed in Table II. Other variables are constants or quality variables and are not considered. According to the process flowchart, all variables have been allocated into four blocks according to the units, as given in Table III. Units like condenser and compressor only have one sampling process variable, and, hence, one may integrate them into the nearest separator block.

For model validation, a large amount of data samples from 10^5 continuous running hours have been collected under the normal operation condition from the model provided by [30]. The sampling interval of the data sample is 36 s. Therefore, the data contains 10^7 (10 million) data samples, while the whole file size is 2.01 GB which is very challenging task for traditional modeling methods. For such a big data, the centralized PCA modeling with SVD procedures will easily get stuck into the modeling failure issue due to out of memory. In this case, dpPCA based on MapReduce should be performed. Since the TE process in fact is not very large in size (the decomposed variable set size in each unit is also small). Therefore, MapReduce in this study is established on the host node (Intel Core i5 2.2 GHz, 8-GB memory, 500-GB hard disk with Windows 7 as OS). In this case, the host acts as both the master compute node and the slave compute node.

To validate the dpPCA modeling scheme, the Algorithm 1 is called for parallel computation of sample number, mean, and variance in each block. The averaged modeling time lists for each block are given in Table IV. One can infer that the modeling time for each block with 10 million data size is around 1 min which should be very efficient. From Table IV, it can be confirmed that the big data modeling issue can be practically accomplished in the proposed distributed and parallel framework with basic-performance personal computers. In other words, one does not have to resort to the high-performance computers, and, thus, the equipment cost can also be significantly reduced.

To investigate the performance of hierarchical monitoring mechanism for the large-scale plant-wide process, all 28 faults have been considered and the fault detection rates are given in Table V, the descriptions for faults can be referred to [30]. One can speculate that dpPCA can detect most faults. Among them, two disturbances including IDV4 and IDV13 have been considered as illustrations. Among them, IDV4 is the step error in reactor cooling water inlet temperature; IDV13 is the slow drift error of reaction kinetics. Note that both test scenarios contain 1000 data samples and the faults are introduced by the 301th sampling time and the significance level is set as $\alpha = 0.01$. Specifically, the hierarchical monitoring results for all faults including PFI, BFI, and RBC (for T^2) indexes based on the 400th sampling time are given in Figs. 4 and 5, respectively.

One can speculate that the hierarchical monitoring mechanism provides convenient and fault-oriented information for monitoring and isolation. Specifically, one can infer from Fig. 4(a) that the dpPCA can detect the faults immediately once the fault has been introduced. In the meanwhile, the block contribution in Fig. 4(b) significantly indicates that the reactor block should be most responsible. Based on the block result, the operators can further refer to the variable contribution in Fig. 4(c), which reveals that the reactor cooling water (reactor cooling water outlet temperature and reactor cooling water flow) should be potentially investigated. Similarly, for the case of IDV13, shown in Fig. 5(a) and (b), one can easily determine that the reactor block should be problematic. The pressure abnormal shown in Fig. 5(c) implies that the kinetic procedure within reactor should be out of the normal expectation, and, hence, the isolation results should be compliant with the fault settings.

From the above modeling and monitoring illustrations, two important results can be concluded. On one hand, the feasibility of the distributed and parallel modeling approach is demonstrated. The large-scale numerical calculation issue of PCA can be decomposed within the distributed and parallel computation environment with MapReduce where local statistics can be reasonably combined for the global solution. In this way, one can

TABLE II
MONITORING VARIABLES IN THE TE PROCESS

No	Measured variables	No	Measured variables	No	Manipulated variables
1	A feed	12	Product separator level	23	D feed flow valve
2	D feed	13	Product separator pressure	24	E feed flow valve
3	E feed	14	Product separator underflow	25	A feed flow valve
4	Total feed	15	Stripper level	26	Total feed flow valve
5	Recycle flow	16	Stripper pressure	27	Purge valve
6	Reactor feed rate	17	Stripper underflow	28	Separator pot liquid flow valve
7	Reactor pressure	18	Stripper temperature	29	Stripper liquid product flow valve
8	Reactor level	19	Stripper steam flow	30	Reactor cooling water flow
9	Reactor temperature	20	Compressor work	31	Condenser cooling water flow
10	Purge rate	21	Reactor cooling water outlet temperature		
11	Product separator temperature	22	Separator cooling water outlet temperature		

TABLE III
DIVIDED BLOCKS OF THE PROCESS VARIABLES

Block	Variables	Description
1	1, 2, 3, 5, 6, 23, 24, 25	Input
2	7, 8, 9, 21, 30	Reactor
3	10, 11, 12, 13, 14, 20, 22, 27, 28, 31	Separator and Compressor and Condenser
4	4, 15, 16, 17, 18, 19, 26, 29	Stripper

TABLE IV
MODELING TIME FOR EACH BLOCK WITH TEN MILLION SAMPLES (IN S)

Block	Block 1	Block 2	Block 3	Block 4
Time	62.858	60.529	64.373	62.409

TABLE V
FAULT DETECTION RATES WITH DPPCA

Fault	Block 1	Block 2	Block 3	Block 4	PFI
1	0.954	0.981	0.984	0.991	0.990
2	0.691	0.556	0.944	0.940	0.957
3	0.030	0.054	0.024	0.010	0.066
4	0.030	0.999	0.066	0.047	0.999
5	0.009	0.014	0.031	0.011	0.043
6	0.999	0.997	0.999	0.997	0.997
7	0.067	0.214	0.379	0.999	0.999
8	0.856	0.830	0.870	0.896	0.893
9	0.030	0.159	0.131	0.049	0.190
10	0.026	0.029	0.191	0.870	0.873
11	0.013	0.959	0.573	0.409	0.980
12	0.031	0.304	0.603	0.429	0.589
13	0.779	0.951	0.940	0.943	0.950
14	0.019	0.990	0.023	0.023	0.989
15	0.009	0.007	0.017	0.013	0.033
16	0.021	0.030	0.081	0.026	0.074
17	0.036	0.851	0.390	0.280	0.854
18	0.023	0.096	0.641	0.217	0.640
19	0.026	0.046	0.923	0.853	0.954
20	0.354	0.800	0.834	0.824	0.839
21	0.021	0.029	0.079	0.026	0.073
22	0.041	0.084	0.030	0.030	0.110
23	0.024	0.039	0.087	0.030	0.080
24	0.696	0.574	0.717	0.487	0.887
25	0.907	0.061	0.306	0.204	0.921
26	0.021	0.049	0.140	0.914	0.913
27	0.023	0.923	0.074	0.064	0.923
28	0.023	0.016	0.020	0.016	0.036

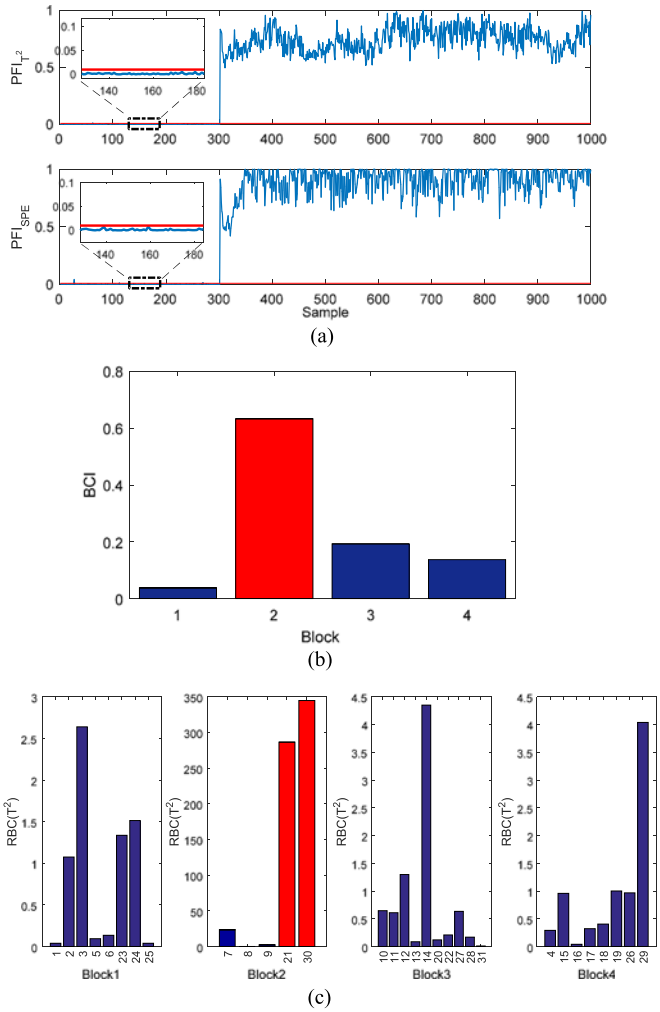


Fig. 4. Hierarchical monitoring results for IDV4: (a) PFI; (b) block contribution index (BFI); and (c) reconstruction-based variable contribution (RBC) for T^2.

realize the large-scale plant-wide process modeling on the network platform of several connected low-cost personal computers. On the other hand, the hierarchical monitoring mechanism from the plant level, block level, and variable level is also very instructive and can be significantly helpful to fault isolation,

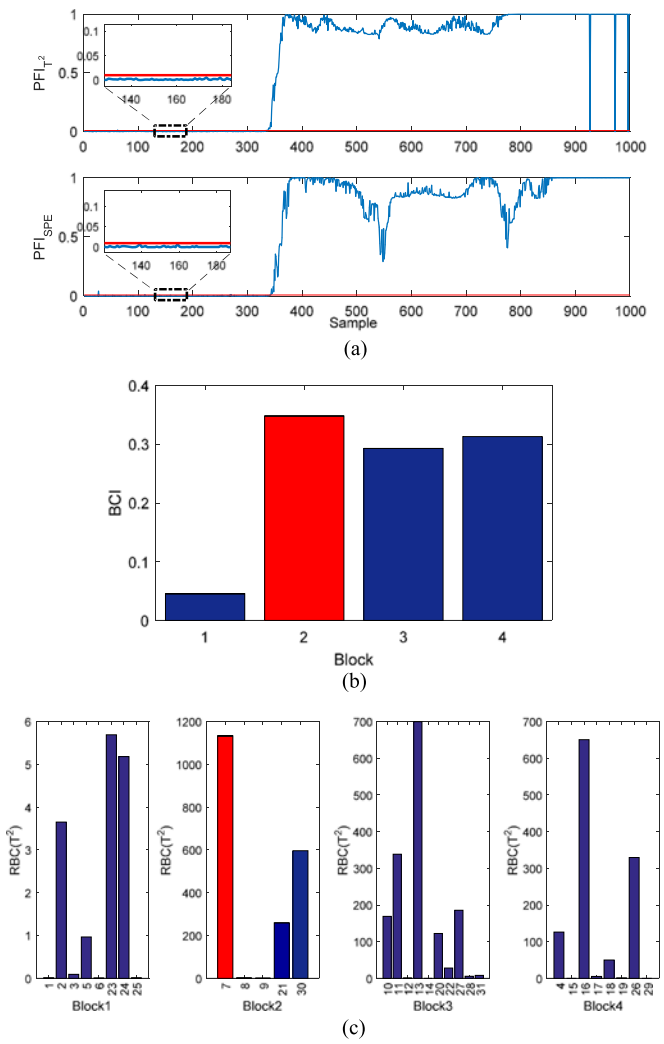


Fig. 5. Hierarchical monitoring results for IDV13: (a) PFI; (b) block contribution index (BFI); and (c) reconstruction-based variable contribution (RBC) for T^2.

decision making, and troubleshooting for plant-wide industrial systems with a large set of variables.

VI. Conclusion

In this study, a systematic framework with distributed and parallel modeling and hierarchical monitoring based on PCA is proposed for large-scale plant-wide processes with big data. First, process variables have been allocated into different distributed blocks in accordance with the physical units. Then, the MapReduce platform is established on multiple blocks so as to perform distributed and parallel computations. Afterward, the combination rules for local block statistics have been designed and then we show PCA for each block can be developed by variance transformation and then taking SVD for correlation matrix. Finally, a hierarchical monitoring flowchart for the plant-wide level, block level, and variable level has been proposed. Case studies on the plant-wide TE benchmark show that dpPCA can be efficiently constructed in about 1 min for 10 million samples. Furthermore, the monitoring results on different unit faults also demonstrate the effectiveness of the proposed hierarchical monitoring mechanism.

Appendix

For the computational convenience, in this study, the population variance is used. Given a sample consisting of N independent observations $\mathbf{x}_n \in R^{1\times D}$, we have

$$\Sigma = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})^T(\mathbf{x}_n - \bar{\mathbf{x}}). \tag{20}$$

Based on the above assumption, now consider the combined situation. Given two sets of samples $\{\mathbf{x}_n\}_{n=1}^{N_1}$ and $\{\mathbf{y}_n\}_{n=1}^{N_2}$, the combined covariance should be

$$\begin{aligned}
\Sigma &= \frac{1}{N_1+N_2}\left[\sum_{n=1}^{N_1}(\mathbf{x}_n-\bar{\mathbf{u}})^T(\mathbf{x}_n-\bar{\mathbf{u}})\right.\\
&\quad\left.+\sum_{n=1}^{N_2}(\mathbf{y}_n-\bar{\mathbf{u}})^T(\mathbf{y}_n-\bar{\mathbf{u}})\right]\\
&= \frac{1}{N_1+N_2}\left\{\sum_{n=1}^{N_1}\left[\mathbf{x}_n-\bar{\mathbf{x}}-\frac{N_2}{N_1+N_2}(\bar{\mathbf{y}}-\bar{\mathbf{x}})\right]^2\right.\\
&\quad\left.+\sum_{n=1}^{N_2}\left[\mathbf{y}_n-\bar{\mathbf{y}}-\frac{N_1}{N_1+N_2}(\bar{\mathbf{x}}-\bar{\mathbf{y}})\right]^2\right\}\\
&= \frac{1}{N_1+N_2}\left\{\sum_{n=1}^{N_1}\left[(\mathbf{x}_n-\bar{\mathbf{x}})^T(\mathbf{x}_n-\bar{\mathbf{x}})\right.\right.\\
&\quad-\frac{N_2}{N_1+N_2}(\mathbf{x}_n-\bar{\mathbf{x}})^T(\bar{\mathbf{y}}-\bar{\mathbf{x}})\\
&\quad-\frac{N_2}{N_1+N_2}(\bar{\mathbf{y}}-\bar{\mathbf{x}})^T(\mathbf{x}_n-\bar{\mathbf{x}})\\
&\quad\left.+\left(\frac{N_2}{N_1+N_2}\right)^2(\bar{\mathbf{y}}-\bar{\mathbf{x}})^T(\bar{\mathbf{y}}-\bar{\mathbf{x}})\right]\\
&\quad+\sum_{n=1}^{N_2}\left[(\mathbf{y}_n-\bar{\mathbf{y}})^T(\mathbf{y}_n-\bar{\mathbf{y}})-\frac{N_1}{N_1+N_2}(\mathbf{y}_n-\bar{\mathbf{y}})^T(\bar{\mathbf{x}}-\bar{\mathbf{y}})\right.\\
&\quad-\frac{N_1}{N_1+N_2}(\bar{\mathbf{x}}-\bar{\mathbf{y}})^T(\mathbf{y}_n-\bar{\mathbf{y}})\\
&\quad\left.\left.+\left(\frac{N_1}{N_1+N_2}\right)^2(\bar{\mathbf{x}}-\bar{\mathbf{y}})^T(\bar{\mathbf{x}}-\bar{\mathbf{y}})\right]\right\}\\
&= \frac{N_1}{N_1+N_2}\Sigma_1+\frac{N_1}{N_1+N_2}(\bar{\mathbf{x}}-\bar{\mathbf{u}})^T(\bar{\mathbf{x}}-\bar{\mathbf{u}})\\
&\quad+\frac{N_2}{N_1+N_2}\Sigma_2+\frac{N_2}{N_1+N_2}(\bar{\mathbf{y}}-\bar{\mathbf{u}})^T(\bar{\mathbf{y}}-\bar{\mathbf{u}}).
\end{aligned}\tag{21}$$

Note that the last equation is derived based on the fact that $\bar{\mathbf{x}}-\bar{\mathbf{u}} = -\frac{N_2}{N_1+N_2}(\bar{\mathbf{y}}-\bar{\mathbf{x}}), \bar{\mathbf{y}}-\bar{\mathbf{u}} = -\frac{N_1}{N_1+N_2}(\bar{\mathbf{x}}-\bar{\mathbf{y}})$, while those additional terms $\sum_{n=1}^{N_2}(\mathbf{x}_n-\bar{\mathbf{x}})^T(\bar{\mathbf{y}}-\bar{\mathbf{x}}) =$

$\sum_{n=1}^{N_2} (\bar{\mathbf{y}} - \bar{\mathbf{x}})^T (\mathbf{x}_n - \bar{\mathbf{x}}) = \mathbf{0}$ and so as $\sum_{n=1}^{N_2} (\mathbf{y}_n - \bar{\mathbf{y}})^T (\bar{\mathbf{x}} - \bar{\mathbf{y}}) = \sum_{n=1}^{N_2} (\bar{\mathbf{x}} - \bar{\mathbf{y}})^T (\mathbf{y}_n - \bar{\mathbf{y}}) = \mathbf{0}$.

REFERENCES

[1] Z. Ge, Z. Song, and F. Gao, "Review of recent research on data-based process monitoring," *Ind. Eng. Chem. Res.*, vol. 52, no. 10, pp. 3543–3562, 2013.

[2] S. J. Qin, "Survey on data-driven industrial process monitoring and diagnosis," *Annu. Rev. Control*, vol. 36, no. 2, pp. 220–234, 2012.

[3] S. Yin, S. X. Ding, X. Xie, and H. Luo, "A review on basic data-driven approaches for industrial process monitoring," *IEEE Trans. Ind. Electron.*, vol. 61, no. 11, pp. 6418–6428, Nov. 2014.

[4] S. Ding, *Model-Based Fault Diagnosis Techniques: Design Schemes, Algorithms, and Tools*. New York, NY, USA: Springer, 2008.

[5] Z. Ge and J. Chen, "Plant-wide industrial process monitoring: A distributed modeling framework," *IEEE Trans. Ind. Informat.*, vol. 12, no. 1, pp. 310–321, Feb. 2016.

[6] J. Yu and M. M. Rashid, "A novel dynamic Bayesian network-based networked process monitoring approach for fault detection, propagation identification, and root cause diagnosis," *AIChE J.*, vol. 59, no. 7, pp. 2348–2365, 2013.

[7] S. Yin, L. Liu, and J. Hou, "A multivariate statistical combination forecasting method for product quality evaluation," *Inf. Sci.*, vols. 355/356, pp. 229–236, 2016.

[8] D. Zhou, G. Li, and S. J. Qin, "Total projection to latent structures for process monitoring," *AIChE J.*, vol. 56, no. 1, pp. 168–178, 2010.

[9] S. Maleki, C. Bingham, and Y. Zhang, "Development and realization of changepoint analysis for the detection of emerging faults on industrial systems," *IEEE Trans. Ind. Informat.*, vol. 12, no. 2, pp. 1180–1187, Jun. 2016.

[10] W. Jiang, Z. Zhang, F. Li, L. Zhang, M. Zhao, and X. Jin, "Joint label consistent dictionary learning and adaptive label prediction for semisupervised machine fault classification," *IEEE Trans. Ind. Informat.*, vol. 12, no. 1, pp. 248–256, Feb. 2016.

[11] J. Zhu, Z. Ge, and Z. Song, "HMM-driven robust probabilistic principal component analyzer for dynamic process fault classification," *IEEE Trans. Ind. Electron.*, vol. 62, no. 6, pp. 3814–3821, Jun. 2015.

[12] S. J. Qin, S. Valle, and M. J. Piovoso, "On unifying multiblock analysis with application to decentralized process monitoring," *J. Chemometrics*, vol. 15, no. 9, pp. 715–742, 2001.

[13] J. F. MacGregor, C. Jaeckle, C. Kiparissides, and M. Koutoudi, "Process monitoring and diagnosis by multiblock PLS methods," *AIChE J.*, vol. 40, no. 5, pp. 826–838, 1994.

[14] Q. Liu, S. J. Qin, and T. Chai, "Multiblock concurrent PLS for decentralized monitoring of continuous annealing processes," *IEEE Trans. Ind. Electron.*, vol. 61, no. 11, pp. 6429–6437, Nov. 2014.

[15] J. A. Westerhuis, T. Kourti, and J. F. MacGregor, "Analysis of multiblock and hierarchical PCA and PLS models," *J. Chemometrics*, vol. 12, no. 5, pp. 301–321, 1998.

[16] S. J. Qin, "Process data analytics in the era of big data," *AIChE J.*, vol. 60, no. 6, pp. 3092–3100, 2014.

[17] S. Yin and O. Kaynak, "Big data for modern industry: Challenges and trends," *Proc.. IEEE*, vol. 103, no. 2, pp. 143–146, Feb. 2015.

[18] K. Kambatla, G. Kollias, V. Kumar, and A. Grama, "Trends in big data analytics," *J. Parallel Distrib. Comput.*, vol. 74, no. 7, pp. 2561–2573, 2014.

[19] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[20] N. K. Alham, M. Li, Y. Liu, and S. Hammoud, "A MapReduce-based distributed SVM algorithm for automatic image annotation," *Comput. Math. Appl.*, vol. 62, no. 7, pp. 2801–2811, 2011.

[21] V. J. Hodge, S. O'Keefe, and J. Austin, "Hadoop neural network for parallel and distributed feature selection," *Neural Netw.*, vol. 78, pp. 24–35, 2016.

[22] S. del Río, V. López, J. M. Benítez, and F. Herrera, "On the use of MapReduce for imbalanced big data using random forest," *Inf. Sci.*, vol. 285, pp. 112–137, 2014.

[23] C. F. Alcala and S. J. Qin, "Reconstruction-based contribution for process monitoring," *Automatica*, vol. 45, no. 7, pp. 1593–1600, 2009.

[24] Z. Ge and Z. Song, "Distributed PCA model for plant-wide process monitoring," *Ind. Eng. Chem. Res.*, vol. 52, no. 5, pp. 1947–1957, 2013.

[25] T. White, *Hadoop: The Definitive Guide*. Sebastopol, CA, USA: O'Reilly Media, Inc., 2012.

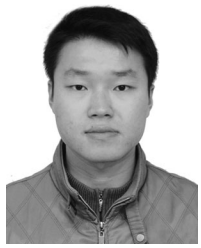
[26] Q. Jiang and X. Yan, "Monitoring multi-mode plant-wide processes by using mutual information-based multi-block PCA, joint probability, and Bayesian inference," *Chemometrics Intell. Lab. Syst.*, vol. 136, no. 9, pp. 121–137, 2014.

[27] J. J. Downs and E. F. Vogel, "A plant-wide industrial process control problem," *Comput. Chem. Eng.*, vol. 17, no. 3, pp. 245–255, 1993.

[28] S. Yin, S. X. Ding, A. Haghani, H. Hao, and P. Zhang, "A comparison study of basic data-driven fault diagnosis and process monitoring methods on the benchmark Tennessee Eastman process," *J. Process Control*, vol. 22, no. 9, pp. 1567–1581, 2012.

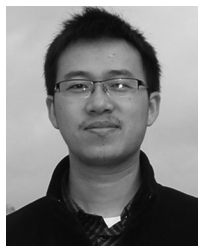
[29] X. Xie, W. Sun, and K. C. Cheung, "An advanced PLS approach for key performance indicator-related prediction and diagnosis in case of outliers," *IEEE Trans. Ind. Electron.*, vol. 63, no. 4, pp. 2587–2594, Apr. 2016.

[30] A. Bathelt, N. L. Ricker, and M. Jelali, "Revision of the Tennessee Eastman process model," in *Proc. 9th Int. Symp. Adv. Control Chem. Processes*, 2015, pp. 309–314.



Jinlin Zhu received the B.Eng. and M.Sc. degrees from Jiangnan University, Wuxi, China, in 2010 and 2013, respectively, and the Ph.D. degree from the College of Control Science and Engineering, Zhejiang University, Hangzhou, China, in 2016.

His research interests include industrial process monitoring and fault diagnosis, soft sensor modeling, process data analysis, and machine learning.



Zhiqiang Ge (M'13) received the B.Eng. and Ph.D. degrees from the Department of Control Science and Engineering, Zhejiang University, Hangzhou, China, in 2004 and 2009, respectively.

He was a Research Associate with the Department of Chemical and Biomolecular Engineering, Hong Kong University of Science Technology, from July 2010 to December 2011, and a Visiting Professor with the Department of Chemical and Materials Engineering, University of Alberta, from January 2013 to May 2013. He was an Alexander von Humboldt Research Fellow with the University of Duisburg-Essen during November 2014 to January 2017. He is currently a Full Professor with the College of Control Science and Engineering, Zhejiang University. His research interests include data mining and analytics, process monitoring and fault diagnosis, machine leaning, and industrial big data.

Zhihuan Song received the B.Eng. and M.Eng. degrees in industrial automation from Hefei University of Technology, Anhui, China, in 1983 and 1986, respectively, and the Ph.D. degree in industrial automation from Zhejiang University, Hangzhou, China, in 1997.

Since 1997, he has been in the Department of Control Science and Engineering, Zhejiang University, where he was first a Postdoctoral Research Fellow, then an Associate Professor, and currently a Professor. He has published more than 200 papers in journals and conference proceedings. His research interests include modeling and fault diagnosis of industrial processes, embedded control systems, and advanced process control technologies.