Fault Detection Using the k-Nearest Neighbor Rule for Semiconductor Manufacturing Processes

Q. Peter He, *Member, IEEE*, and Jin Wang, *Member, IEEE*

***Abstract*—It has been recognized that effective fault detection techniques can help semiconductor manufacturers reduce scrap, increase equipment uptime, and reduce the usage of test wafers. Traditional univariate statistical process control charts have long been used for fault detection. Recently, multivariate statistical fault detection methods such as principal component analysis (PCA)-based methods have drawn increasing interest in the semiconductor manufacturing industry. However, the unique characteristics of the semiconductor processes, such as nonlinearity in most batch processes, multimodal batch trajectories due to product mix, and process steps with variable durations, have posed some difficulties to the PCA-based methods. To explicitly account for these unique characteristics, a fault detection method using the k-nearest neighbor rule (FD-kNN) is developed in this paper. Because in fault detection faults are usually not identified and characterized beforehand, in this paper the traditional kNN algorithm is adapted such that only normal operation data is needed. Because the developed method makes use of the kNN rule, which is a nonlinear classifier, it naturally handles possible nonlinearity in the data. Also, because the FD-kNN method makes decisions based on small local neighborhoods of similar batches, it is well suited for multimodal cases. Another feature of the proposed FD-kNN method, which is essential for online fault detection, is that the data preprocessing is performed automatically without human intervention. These capabilities of the developed FD-kNN method are demonstrated by simulated illustrative examples as well as an industrial example.**

***Index Terms*—Fault detection, k-nearest neighbor rule, pattern recognition, semiconductor manufacturing, statistical process monitoring.**

I. Introduction

THE semiconductor industry has maintained an average growth of 15% per year over the past few decades. The steady growth is a result of continuous reduction of 25%–30% per year in the cost per function. In the past, costs have been reduced by focusing on factors related to processing technology: reducing feature size, increasing wafer size, and improving yield. Recently, it has been recognized that factory productivity should also be improved in order to keep growth while reducing production cost [1], [2]. In order to meet the increasing demands of the semiconductor industry to improve yield while simultaneously decreasing circuit geometries, recent efforts have focused on characterizing and controlling variability in critical manufacturing processes via advanced process control (APC) and fault detection and classification (FDC) techniques [3]–[9]. The focus of this paper is on the FDC techniques that help semiconductor manufacturers achieve the following goals: 1) reducing scrap; 2) increasing equipment uptime; and 3) reducing the usage of test wafers.

In today's semiconductor industry, a massive amount of trace or machine data is generated and recorded. Traditional univariate statistical process control (SPC) charts, such as the Shewhart, CUSUM (cumulative sum), and EWMA (exponentially weighted moving average) charts, have long been applied to reducing process variability. Although univariate statistical techniques are easy to implement, they often lead to a significant number of false alarms on multivariate processes where the sensor measurements are highly correlated because of physical and chemical principles governing the process operation, such as mass and energy balances [10], [11]. A simple yet illustrative example that shows the misleading nature of the univariate charts is given in [12]. To address this aspect, multivariate statistical fault detection methods such as principal component analysis (PCA) and partial least squares (PLS) have drawn increasing interest in semiconductor manufacturing industry recently [13]–[17]. PCA- and PLS-based methods have been tremendously successful in continuous process applications such as petrochemical processes, and its application to traditional chemical batch processes has been extensively studied in the last decade [18]–[21]. However, some unique characteristics of semiconductor manufacturing processes have posed difficulties to these multivariate statistical methods. Examples of the unique characteristics of semiconductor processes are: nonlinearity in most batch processes, multimodal batch trajectories due to product mix, and process steps with variable durations (often deliberately adjusted). It is well known that data preprocessing is an important step before PCA- or PLS-based fault detection methods can be performed. However, for semiconductor manufacturing processes, the above-mentioned characteristics make data preprocessing difficult, especially when automatic data preprocessing is desirable. Besides, in order to take explicit account of the time sequence, the data has to be unfolded into a 2-D data array before PCA or PLS can be applied. One of the most commonly used unfolding methods is the one used in multiway-PCA (MPCA) where the data is unfolded in such a way that each row represents a batch[1] (see Fig. 1). However, the unfolding increases the number of variables dramatically, which requires a significant amount of

Manuscript received March 19, 2007; revised June 15, 2007.

Q. P. He is with the Department of Chemical Engineering, Tuskegee University, Tuskegee, AL 36088 USA (e-mail: qhe@tuskegee.edu).

J. Wang is with the Department of Chemical Engineering, Auburn University, Auburn, AL 36849 USA (e-mail: wang@auburn.edu).

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/TSM.2007.907607

[1]In this paper, a single wafer processing is assumed for simplicity so that a batch is equivalent to a wafer. However, the techniques discussed in this paper are equally applicable for processes where a lot (25 wafers) or multiple lots of wafers are processed within a batch.

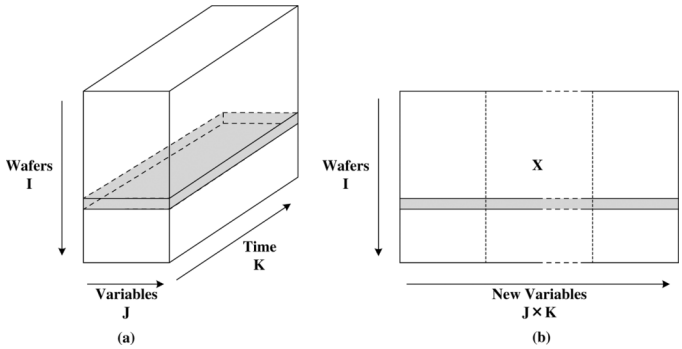


Fig. 1. Data unfolding: (a) Original 3-D data array and (b) MPCA unfolded 2-D data array. (Sample wafer before and after unfolding is highlighted in gray.)

batches to build a reliable PCA model in order to capture the key characteristics of the process.

In this paper, a new fault detection method is developed to explicitly account for the above-mentioned unique characteristics of semiconductor processes. The developed method is based on the k-nearest neighbor (kNN) rule. The kNN rule is an intuitive concept and its basic idea is given as the following: For a given unlabeled sample $\mathbf{x}$, the kNN rule finds the knearest labeled samples in the training data set and assigns $\mathbf{x}$ to the class that appears most frequently within the k-subset (i.e., k-nearest neighbors). Due to its simplicity and flexibility, the kNN rule has been widely applied in pattern classification where unlabeled samples are classified based on their similarities with samples in the training set [22]. Although few commercial software packages, such as ModelWare and Maestria [2] , may have been using the kNN rule or similar techniques for fault detection, very few studies [6], [23] have been published. The main difficulty in utilizing the kNN rule for fault detection is that there is only one class of data, i.e., normal operation data, available as training data, which makes the traditional kNN rule not directly applicable. To circumvent this difficulty, we adapt the traditional kNN rule to make use of normal operation data only. The proposed fault detection method using the kNN rule (FD-kNN) is based on the idea that the trajectory of an incoming normal sample is similar to the trajectories of training samples that consist of no fault (i.e., normal operation data only); on the other hand, the trajectory of an incoming fault sample must exhibit some deviation from the trajectories of normal training samples. In other words, a fault sample's distance to the nearest neighboring training samples must be greater than a normal sample's distance to the nearest neighboring training samples. Again, the training samples contain no fault, only normal samples obtained under normal operation conditions. Therefore, if we can determine the distribution of training samples' distances to their nearest neighboring training samples, we can define a threshold distance for a given confidence level. The incoming sample is considered normal if its distance to its nearest neighboring training samples is below the threshold. Otherwise, a fault is detected.

The remaining part of the paper is organized as follows. In Section II, we give brief reviews on the relevant methods applied in this paper. Section III presents the detailed fault detection method using the kNN rule. Simulation examples to illustrate fault detection capability of the proposed method are presented in Section IV. Section V compares the proposed method and MPCA using an industrial example, and Section VI gives conclusions and some discussions.

[2]ModelWare is a registered trademarks of Triant Technologies Inc. and Maestria is a registered trademark of PDF Solutions, Inc.

II. Methods

In this section, we briefly review relevant methods, namely PCA and the kNN rule, the basis of the proposed fault detection method. In this paper, a scalar is denoted by an italic lowercase character (x), a vector by a bold lowercase character $(\mathbf{x})$, a matrix by a bold uppercase character $(\mathbf{X})$, and a three-way array by an underlined bold uppercase character $(\underline{\mathbf{X}})$. These notations are consistent with notations used by others in the statistical process control community (see e.g., [24]).

A. PCA

Let $\mathbf{X} \in \Re^{n\times m}$ denote the raw data matrix with n samples (rows) and m variables (columns). $\mathbf{X}$ is first scaled to zero mean for covariance-based PCA and further to unit variance for correlation-based PCA. By either the NIPALS [25] or a singular value decomposition (SVD) algorithm, the scaled matrix $\mathbf{X}$ is decomposed as follows:

$$\mathbf{X} = \mathbf{T}\mathbf{P}^T + \tilde{\mathbf{X}} = \mathbf{T}\mathbf{P}^T + \tilde{\mathbf{T}}\tilde{\mathbf{P}}^T = [\mathbf{T} \quad \tilde{\mathbf{T}}][\mathbf{P} \quad \tilde{\mathbf{P}}]^T \quad (1)$$

where $\mathbf{T} \in \Re^{n\times l}$ and $\mathbf{P} \in \Re^{m\times l}$ are the score and loading matrices, respectively. The PCA projection reduces the original set of m variables to l principal components (PCs). The decomposition is made such that $[\mathbf{T}\ \tilde{\mathbf{T}}]$ is orthogonal and $[\mathbf{P}\ \tilde{\mathbf{P}}]$ is orthonormal. The columns of $\mathbf{P}$ are actually eigenvectors of the covariance or correlation matrix of the variables associated with the l largest eigenvalues, and the columns of $\tilde{\mathbf{P}}$ are the remaining eigenvectors. For fault detection in a new sample vector $\mathbf{x}$, the squared prediction error (SPE) and the Hotelling's T^2 are often used. The SPE statistic indicates how well each sample conforms to the model, measured by the projection of the sample vector on the residual space

$$\mathrm{SPE} = ||\tilde{\mathbf{x}}||^2 = ||(\mathbf{I} - \mathbf{P}\mathbf{P}^T)\mathbf{x}||^2. \quad (2)$$

The process is considered normal if

$$\mathrm{SPE} \leq \delta_\alpha^2 \quad (3)$$

where δ_α^2 denotes the upper control limit for SPE with a significance level α. An expression for δ_α^2 has been developed by Jackson and Mudholkar [26] assuming that $\mathbf{x}$ follows a normal distribution.

The Hotelling's T^2 is a measure of the variation in principal component space

$$T^2 = \mathbf{x}^T\mathbf{P}\mathbf{\Lambda}^{-1}\mathbf{P}^T\mathbf{x} \quad (4)$$

where $\mathbf{\Lambda}$ is the diagonal matrix of l largest eigenvalues of $\mathbf{X}\mathbf{X}^T$. The T^2 statistic forms a hyper-ellipsoid, which represents the joint limits of variations that can be explained by a set of common causes. For a given significance level α, the process is considered normal if

$$T^2 \leq T_\alpha^2 \tag{5}$$

where the upper control limit T_α^2 can be calculated or approximated in several ways [27]. If X is obtained by unfolding a 3-D data array $\underline{\mathbf{X}}$, e.g., as shown in Fig. 1, the method is known as multiway-PCA (MPCA). If both process data $\mathbf{X}$ and quality data $\mathbf{Y}$ are available and one wishes to extract variations in $\mathbf{X}$ that contribute to $\mathbf{Y}$, partial least squares (PLS) should be used instead of PCA. PLS attempts to extract the latent variables that not only explain the variations in the process data $\mathbf{X}$, but also the variations in $\mathbf{X}$ which are more predictive of the quality data $\mathbf{Y}$. Since only the process data will be used in this paper, PLS will not be discussed and interested readers should refer to [12], [28], and [29].

B. kNN Rule

In pattern classification, the kNN rule is a method to classify a new object by examining its distances to the nearest neighboring training samples in the feature space [22]. In other words, the kNN rule classifies an unlabeled sample based on its similarity with samples in the training set, just as the saying goes "if it walks like a duck, quacks like a duck, then it is probably a duck." The kNN rule has been used in many applications in the field of data mining, statistical pattern recognition, image processing, and many others. In process industry, the kNN rule has recently been used for fault classification [30]. However, due to the difficulty in identifying and characterizing possible faults and including them in the training data, application of the kNN rule to fault classification has been very limited and only few studies [6], [23] have been published on applying the kNN rule to fault detection. In this section, we review the basic idea of the kNN rule for pattern classification; and in the next section, we discuss its adaptation to fault detection.

For a given unlabeled sample $\mathbf{x}$, the kNN rule finds the k-nearest labeled samples in the training data set based on some distance metric. Although several distance metrics have been proposed for kNN algorithms, the most common metric is the Euclidean distance. After finding $\mathbf{x}$'s k-nearest neighbors, there are two different voting schemes to determine its label, namely majority voting and weighted-sum voting. In majority voting, $\mathbf{x}$ is assigned to the class that appears most frequently within its k-nearest neighbors. In weighted-sum voting, each vote is weighted based on the idea that near neighbors should count more than neighbors far away (e.g., the weighting could be the reciprocal of the squared distance). The difference between the two voting schemes is better illustrated by the following simple example with two variables/features. In Fig. 2, the squares and triangles are training samples from classes A and B respectively; $\mathbf{x}$, shown as the circle, is the sample to be classified. In this example, Euclidean distance metric is used and dashed circles with different radii are distance references for finding

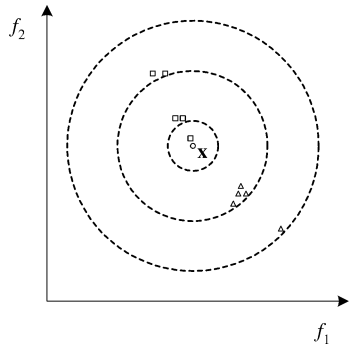


Fig. 2. Example with two features. Squares are training samples from class A; triangles are training samples from class B. x, shown as the circle in center, is sample to be classified.

nearest neighbors. If the majority voting is applied, when $k = 1, \mathbf{x}$ is classified as class A; when $k = 7, \mathbf{x}$ is classified as class B; When $k = 10$, there is no decision. On the other hand, if the weighted-sum voting scheme is applied where each vote is weighted by $1/d^2$ (d is the Euclidean distance), $\mathbf{x}$ is always classified as class A, when $k = 1, 7$, or 10.

For pattern classification, the kNN algorithm only requires an integer k, a set of labeled samples (training data), and a metric to measure distances. Due to its simplicity, the implementation is straightforward.

III. Fault Detection Based on kNN Rule

As discussed in the previous section, the kNN rule classifies an unlabeled sample based on its similarity with samples in the training set. This is the case of fault classification where training data consist of labeled normal samples as well as various fault samples. An incoming unlabeled sample is assigned as normal or one type of known faults using the kNN classification algorithm. The difference between kNN fault detection and kNN fault classification is that in the case of fault detection, faults are not defined or labeled in advance. Therefore, the kNN rule cannot be applied to fault detection directly because the training data contains a single class—normal process data. In this section, the traditional kNN rule is adapted for fault detection.

A. Proposed Fault Detection Method Using kNN Rule

The proposed fault detection method using the kNN rule (FD-kNN) is based on the idea that the trajectory of a normal sample is similar to the trajectories of normal samples in the training data; on the other hand, the trajectory of a fault sample must exhibit some deviation from the trajectories of normal training samples. In other words, a fault sample's distance to the nearest neighboring training samples must be greater than a normal sample's distance to the nearest neighboring training samples. If we can determine the distribution of normal training samples' distances to their nearest neighboring training samples, we can define a threshold with certain confidence level and the unclassified sample is considered normal if its distance to its nearest neighboring training samples is below the threshold. Otherwise, the sample is detected as a fault. The proposed method consists of two parts: model building and fault detection, as shown in Fig. 3.

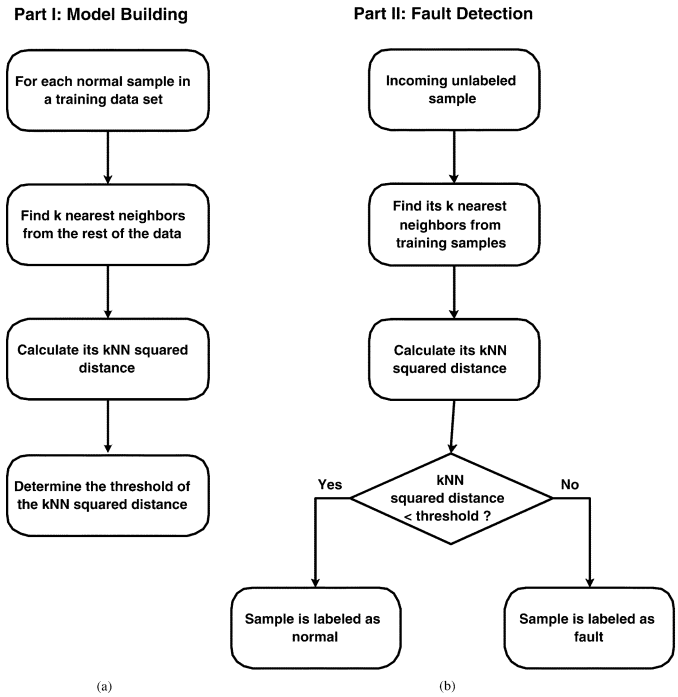


Fig. 3. Flow chart of proposed fault detection method using kNN rule. (a) Part I: model building. (b) Part II: fault detection.

- Part I: model building
  This part consists of three steps.
  1) Finding k-nearest neighbors for each sample in the training data set.[3]
  2) Calculation of the kNN squared distance for each sample.
     In this paper, the kNN squared distance of sample i ($\mathcal{D}_i^2$) is defined as the sum of squared distances of sample i to its k-nearest neighbors

$$\mathcal{D}_i^2 = \sum_{j=1}^{k} d_{ij}^2 \tag{6}$$

     where d_{ij}^2 denotes squared Euclidean distance from sample i to its jth nearest neighbor.
  3) Determination of a threshold for fault detection.
     If we assume that samples are independently identically distributed (i.i.d.) and follow a multivariate normal distribution, it is straightforward to verify that the distances between two random samples follow a chi-square distribution. However, these conditions are seldom satisfied in practice. More importantly, selecting the k smallest distances is not an algebraic operation. Due to these reasons, the distribution of kNN squared distances cannot be obtained through rigorous statistical derivation. Nevertheless, if we assume that the k-nearest neighbor distances d_{ij} among training samples are normally distributed with a nonzero mean, the sum of squares of these distances, if randomly selected, yield a noncentral chi-square distribution. Again, because the calculation of $\mathcal{D}_i^2$ is not a random process, the distribution of $\mathcal{D}_i^2$ can only be approximated by a noncentral chi-square distribution and the threshold $\mathcal{D}_\alpha^2$ with a significance level α can be estimated using the Matlab function *chilimit* in the PLS Toolbox. This is how the threshold is determined in this paper and it is shown in all examples in the following sections that the thresholds determined by this method are reasonable and practical. Another common way to set the threshold is based on the calibration or testing data under normal operation conditions [13], [31]. For example, a 95% confidence limit can be estimated as the value for which 95% of the calibration samples are below the limit.
- Part II: fault detection
  For an incoming unclassified sample $\mathbf{x}$, the fault detection part also consists of three steps.
  1) Finding $\mathbf{x}$'s k-nearest neighbors from the training data set.
  2) Calculation of $\mathbf{x}$'s kNN squared distance $\mathcal{D}_x^2$ (6).
  3) Comparison of $\mathcal{D}_x^2$ against the threshold $\mathcal{D}_\alpha^2$.

  If $\mathcal{D}_x^2 \leq \mathcal{D}_\alpha^2$, it is classified as a normal sample. Otherwise, it is detected as a fault.

[3]The determination of an appropriate k value is discussed later in this paper.

B. Remarks

1) The choice of k is usually uncritical [32]. In general, larger values of k reduce the effect of noise on the fault detection, but make boundaries between normal and fault batches less distinct. A practical approach is to try several different values of k on historical data and choose the one that gives the best cross validation.
2) The kNN rule can be quite effective when the attributes of the data are equally important. However, it can be less effective when many of the attributes are misleading or irrelevant to process characterization. In other words, variable selection is important for the proposed FD-kNN method to work and process knowledge can play a big role in this process. Also, we can define weighted distance to give higher weights on more important variables.
3) Unlike PCA, the developed FD-kNN method makes no assumption of the linearity of the data set. In fact, because the FD-kNN method is based on the kNN rule which is a nonlinear classifier, the FD-kNN method naturally handles process nonlinearity. This is illustrated in a nonlinear case study in the next section.
4) Because the FD-kNN method detects faults based on local neighborhoods of similar batches, it is well suited for multimodal data sets in which batches can be grouped into subsects with different characteristics. This capability is illustrated in a multimodal case study in the next section as well as in an industrial example given in Section V.

IV. Illustrative Examples

In this section, simple examples with two variables (x and y) are given to illustrate the fault detection capabilities of the proposed FD-kNN method in different situations, including the presence of nonlinearity or multimodal distribution, and to compare it with the widely used fault detection method PCA. In each

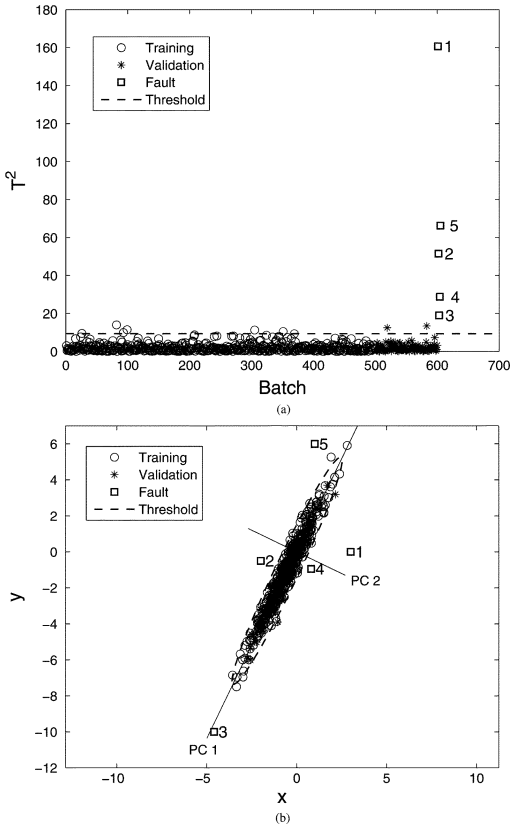


Fig. 4. Fault detection results based on two PCs: (a) T^2 chart and (b) PC's (solid lines) and T^2 threshold (dashed ellipse) in original variable space.

case, before applying a PCA or FD-kNN algorithm, the training data is scaled to zero mean and unit variance for each variable. Validations and faults are scaled accordingly using the mean and variance of the training data.

A. Linear Case

The first simulation example is a linear case with the following process model:

$$y = 2x + \text{noise}. \tag{7}$$

A total of 600 normal runs are conducted and five faults are induced. Among the 600 normal runs, 500 runs are randomly selected as the training data and the rest 100 runs are used as the validation data. The data set is visualized in Fig. 4(b).

PCA is first applied to detect the faults in the data set. If two PCs are chosen to capture 100% of the total variance, all five faults are detected by T^2 as shown in Fig. 4(a). To illustrate the result of PCA in the original space, in Fig. 4(b), two PCs are shown as solid straight lines and the T^2 limit is shown as the dashed ellipse.

The proposed FD-kNN is applied to the same data set and the number of nearest neighbors, k, is set to be 3. The detection result is shown in Fig. 5 and we see that the proposed method successfully detects all five faults. For this linear case, PCA performs similarly to FD-kNN when the maximum number of PCs is chosen to cover the whole feature space. However, it is worth noticing that their metrics of similarity measure are different as shown by the difference between Figs. 4(a) and 5. In Fig. 4(a),

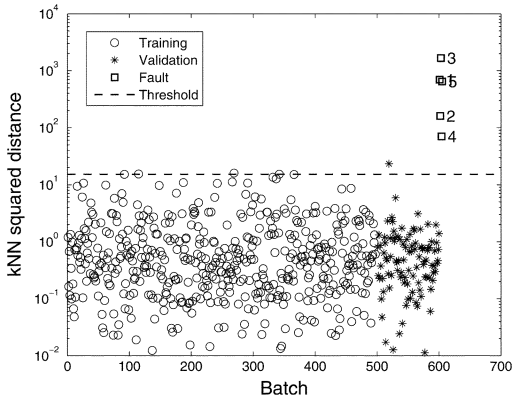


Fig. 5. Fault detection results using proposed FD-kNN method.

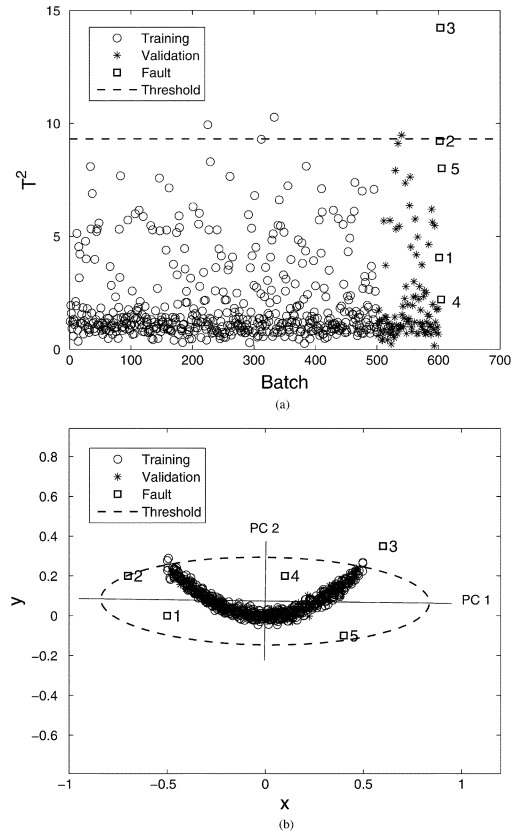


Fig. 6. Fault detection results based on two PCs: (a) T^2 chart and (b) PC's (solid lines) and T^2 threshold (dashed ellipse) in original variable space.

the PCA T^2 metric indicates that fault 1 is a more obvious fault than others and fault 3 is the most subtle fault among all faults. But from Fig. 5, we see that fault 3 is the most obvious fault according to the metric used by FD-kNN.

B. Nonlinear Case

The second simulation example is a nonlinear case with the following process model:

$$y = x^2 + \text{noise}. \tag{8}$$

Similar to the linear case, 500 normal runs are used for training, 100 normal runs are used for validation, and five faults are introduced. Fig. 6(b) shows the scatter plot of the training, validation, and fault samples.

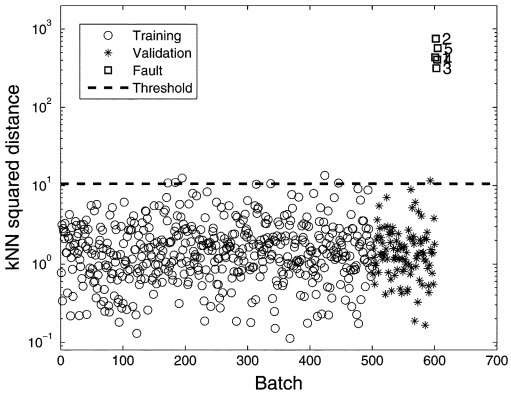


Fig. 7. Fault detection results using proposed FD-kNN method.

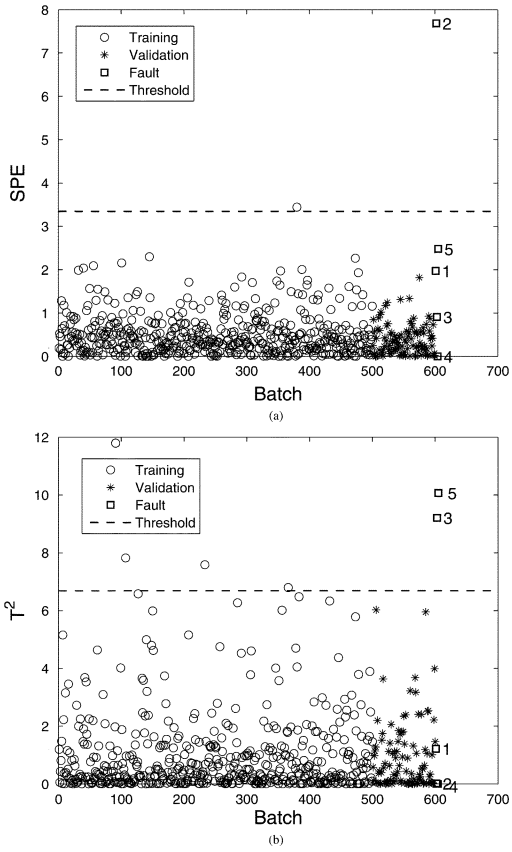


Fig. 8. Fault detection results based on one PC: (a) SPE chart and (b) T^2 chart.

PCA is first applied to detect the faults in the data set. At the confidence level of 99%, the detection results with two PCs are shown in Fig. 6. We see that PCA does not perform well and the majority of the faults are not detected by PCA. From Fig. 6(b), we see that due to the nonlinearity in the process data, the T^2 limit covers a much wider range than it does in the linear case shown in Fig. 4(b). This example illustrates how nonlinearity negatively affects PCA's fault detection effectiveness.

Next, the proposed FD-kNN is applied to the nonlinear case data set. The number of nearest neighbors k is set to be 3 as in the previous example. The detection result is shown in Fig. 7 where all five faults are successfully detected. Notice from Fig. 7 that the false alarm rate of FD-kNN is low and the validation indicates good model consistency.

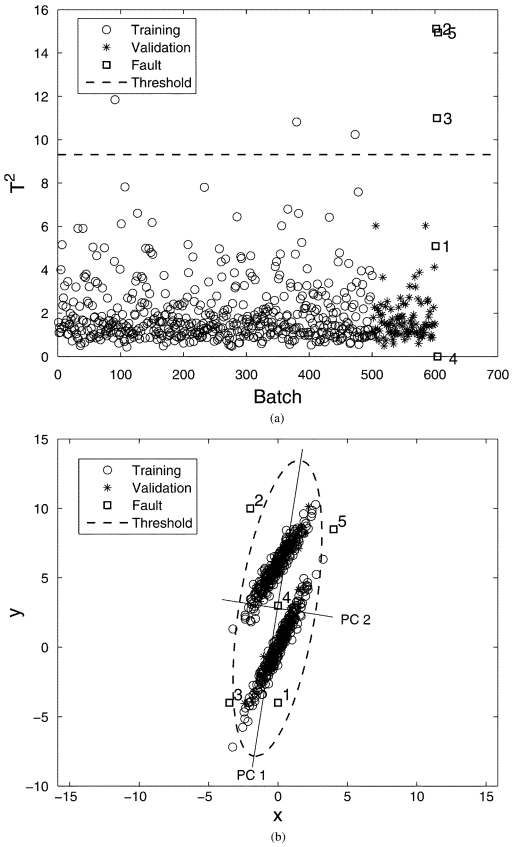


Fig. 9. Fault detection results based on two PC's: (a) T^2 chart and (b) PC's (solid lines) and T^2 threshold (dashed ellipse) in original variable space.

This example illustrates that FD-kNN extracts nonlinear features through selecting nearest neighbors and therefore handles process nonlinearity well.

C. Multimodal Case

The third simulation example is a multimodal case with a similar process model as in the first simulation example. But, in the multimodal case the process is performed on two tools with different process gains and an offset between the tools. Process models are given as follows:

$$\begin{aligned} &\text{Tool A: } y = 2x + \text{noise} \\ &\text{Tool B: } y = 1.5x + 6 + \text{noise}. \end{aligned} \tag{9}$$

The 300 normal runs are conducted on each tool so that totally 600 normal data points are collected. The 500 normal runs are randomly selected from the two tools for training, the remaining 100 normal runs are used for validation, and five faults are introduced. The data set, including the training, validation, and fault samples, is visualized in Fig. 9(b).

We see that not all faults can be detected by PCA because of the bimodal distribution.

Next, the proposed FD-kNN is applied to the multimodal data set. The number of nearest neighbors k is set to be 3 as in the previous examples. The result is shown in Fig. 10(a) and all

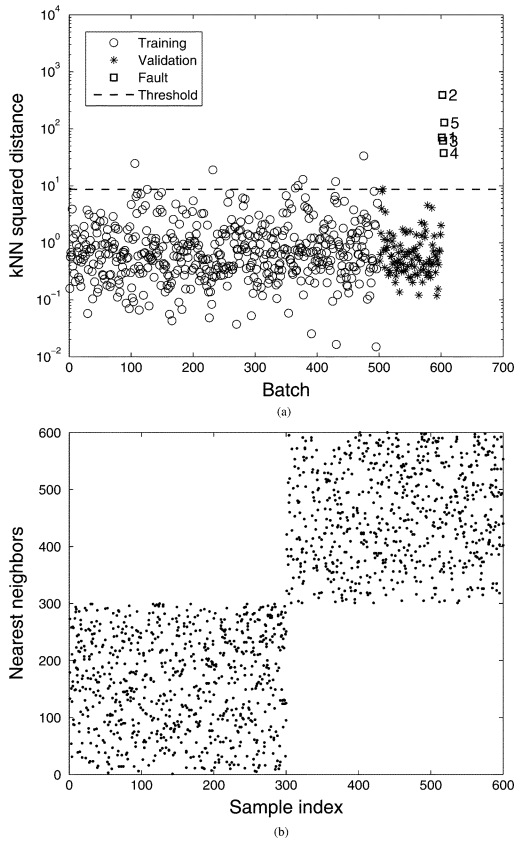


Fig. 10. (a) Fault detection results using proposed FD-kNN method. (b) Mapping of training samples' nearest neighbors.

five faults are detected by the proposed FD-kNN method. Notice in Fig. 10(a) that the validation samples behave essentially the same as the training samples. To track spatial distribution of nearest neighbors, we map the three nearest neighbors of all training samples in the multimodal case and the result is shown in Fig. 10(b). Notice that although the sample indexes range from 1 to 600, because 100 out of 600 normal samples are randomly selected as validation samples, there are actually only 500 training sample points. As we can see from Fig. 10(b), each training sample finds the nearest neighbors in its own mode.

Again, because PCA can only extract linear structure from the data, the multimodal environment imposes limitations on PCA and makes it less effective. For the FD-kNN algorithm, as discussed in Section III, because it focuses on local neighborhoods as shown in Fig. 10(b), FD-kNN does not suffer degradation from multimodal environment.

V. Industrial Examples

In this section, an industrial example is used to demonstrate the performance of the proposed FD-kNN method. The data set is collected from an Al stack etch process performed on a commercial scale Lam 9600 plasma etch tool at Texas Instrument, Inc. [13], [33]. The goal of this process is to etch the TiN/A1-0.5% Cu/TiN/oxide stack with an inductively coupled BCl_3/Cl_2 plasma. The data consists of 108 normal wafers taken during three experiments and 21 wafers with intentionally induced faults taken during the same experiments. Due to large amount of missing data in two batches, only 107 normal wafers and 20 wafers with faults are used in this paper. A more detailed description on the faults can be found in [13]. The standard recipe for the process consists of six steps. The first two are for gas flow and pressure stabilization. Step 3 is a brief plasma ignition step. Step 4 is the etch of the A1 layer terminating at the A1 endpoint, with step 5 acting as the over-etch for the underlying TiN and oxide layers. Step 6 vents the chamber. Since steps 4 and 5 are the main etch steps, as in [13], only the data points from steps 4 and 5 are used in this paper. The data is collected by the machine state sensor system at 1-s intervals during the etch. The original data contains 40 variables including process setpoints, measured variables, and controlled variables such as gas flow rates, chamber pressure, and RF power. Because including irrelevant variables in the data set degrades the performance of both PCA and FD-kNN, in this paper only 19 non-setpoint process variables used in [13] are included for fault detection. The physics of the problem suggests that these variables should be relevant to process and final product state [13].

As pointed out earlier, there are unique characteristics associated with semiconductor processes. These characteristics are also noted in this data set.

1) Unequal batch duration: Like many other batch processes, in the etch data set, different batches have different durations. Among the 107 normal batches, for example, the batch durations range from 95 to 112 s.
2) Unequal step duration: In addition to unequal batch duration, for a specific step, the step duration may vary from batch to batch and time stamps of the step onset are not synchronized. In the etch data set, the duration of step 4, which is one of the main etch steps, varies from 44 to 52 s. In other words, even batches of equal length may not follow exactly the same time trajectory.
3) Process drift and shift: For the etch process, drift and shift in the data are primarily due to the following sources [13]: aging of the etcher over a clean cycle; differences in the incoming materials; drift in the process monitoring sensors; and preventive and corrective maintenances. Because the three experiments were carried out several weeks apart, due to the process drift and shift, data from different experiments have different means and somewhat different covariance structures (see Fig. 11).

A. Data Preprocessing

Data preprocessing is an important aspect of multivariate statistical analysis and can have a significant impact on the overall sensitivity and robustness of the method [13]. In order to get meaningful results, before applying PCA or MPCA, data is usually scaled to zero mean and unit variance. For batch process monitoring, an additional complication involves stretching of the time axis in the data record as discussed previously. One way to approach this is to select a specified number of samples from each step. Another way is to use speech recognition methods such as dynamic time warping (DTW) to map the process response back onto a reference trace [34]. In addition, to discriminate against process drift, a high pass frequency filter can be employed to remove the low frequency drift in the process [35]. Al-

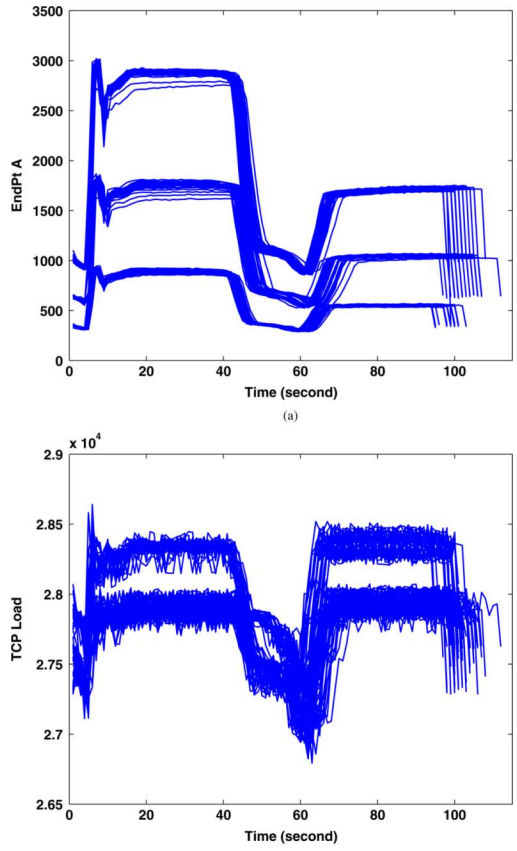


Fig. 11. Variables show mean and covariance change in data set: (a) EndPt A and (b) TCP Load.

though all these data preprocessing techniques are powerful in improving the effectiveness of the fault detection method, they are not desirable in an automated manufacturing environment. This is mainly because they are process specific (e.g., DTW and filtering) which require human interactions and therefore are difficult to automate. One goal of this paper is to maximize the level of automation in fault detection. Therefore, we compare fault detection methods with minimum (and maybe poor) data preprocessing in this paper. The first step is to obtain equal length batch records. In this step, the initial five sample points are removed to eliminate the effect of initial fluctuation in sensors and totally 85 sample points were kept to accommodate shorter batches in the data set. This is done for all training, validation, and test data. Once equal length batch records are obtained, the second step is to scale each variable to zero mean and unit variance for each wafer in the training data set and scale the validation and test data accordingly using the mean and variance values obtained from the training data. Note that the above two-step data preprocessing can be done automatically in the production environment. For MPCA analysis, the data is further unfolded in the way as shown in Fig. 1 where the unfolded 2-D array is denoted by $\mathbf{X}$. Next, we apply MPCA and the proposed FD-kNN method to the preprocessed data sets.

B. Fault Detection

MPCA is first used to analyze the data by applying the Matlab function *pca* in the PLS Toolbox to the unfolded data set $\mathbf{X}$. In

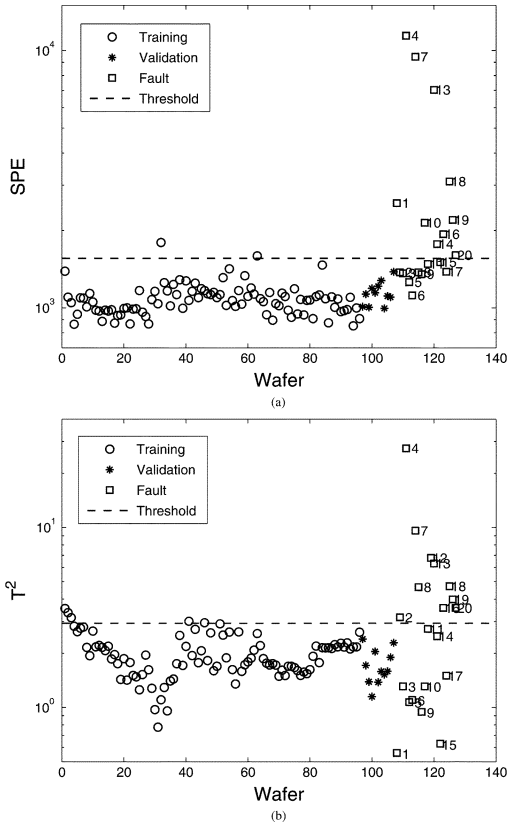


Fig. 12. Fault detection results based on PCA: (a) SPE chart and (b) T^2 chart.

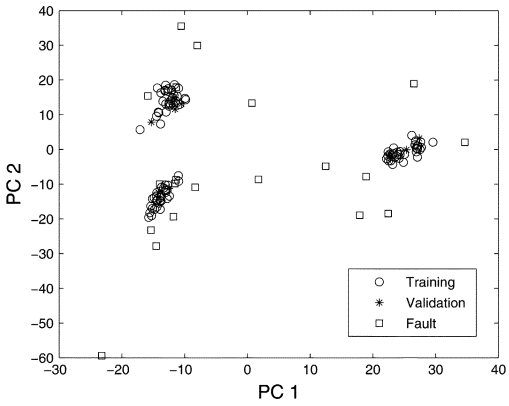


Fig. 13. Score plot of PC 1 and PC 2.

total, three PCs are used to build the PCA model so that the SPE and T^2 values of the validation data are at the same levels as those of the training data. Fig. 12(a) shows the fault detection result based on the SPE index and (b) shows the fault detection result based on the T^2 index. Note that fault 12 is detected by SPE as a fault but not shown in Fig. 12(a) because its SPE value is well above the others. SPE and T^2 charts together detect 13 faults out of 20 total faults, although there is not complete overlap between the faults detected. Detailed fault detection results are listed in Table I. The less efficiency of MPCA in this case can be explained by the characteristic multimodal distribution of the batch trajectories—in the score plot of the first two PCs (Fig. 13), three clusters are observed which correspond to the three experiments. As discussed earlier, the data

TABLE I
FAULT DETECTED BY DIFFERENT METHODS

Fault	PCA-SPE	PCA-T^2	FD-kNN
1	√		√
2		√	√
3			
4	√	√	√
5			
6			
7	√	√	√
8		√	√
9			√
10	√		√
11			√
12	√	√	√
13	√	√	√
14	√		√
15			√
16	√	√	√
17			√
18	√	√	√
19	√	√	√
20	√	√	√

were collected from the three experiments that were run several weeks apart. Due to tool state shift during that period of time, different experiments have different means and covariance structures. Notice that in a manufacturing environment, tool state shift is inevitable due to a variety of reasons such as preventive maintenance (PM) and part replacement.

Next, the proposed FD-kNN method is applied to detecting faults in the etch data set. The number of nearest neighbors is set as 5. The threshold is determined by the *chilimit* function from the Matlab PLS Toolbox. The result is shown as a semi-log plot in Fig. 14(a) where 17 out of 20 faults are detected by the FD-kNN algorithm. Note that fault 12 is detected as a fault but not shown in Fig. 14(a) due to its much higher kNN squared distance than others. To illustrate where each sample's neighbors are distributed, in Fig. 14(b) we draw the mapping of five nearest neighbors for the training wafers. Fig. 14(b) shows the same multimode characteristic as in Fig. 13 that wafers are grouped in three clusters. From Fig. 14(b) we see that each wafer finds its k-nearest neighbors in its own group, which is consistent with our discussion earlier that the FD-kNN method handles multimodal distribution naturally by focusing on local neighborhoods. The detailed fault detection results from MPCA and FD-kNN are shown in Table I.

Again, this example illustrates that FD-kNN is capable of handling multimodal data without additional data preprocessing. Note that if further data preprocessing is performed to eliminate the multimodal group effect, PCA performs similarly to FD-kNN and detects 17 faults as well.

VI. CONCLUSION

In this paper, a new fault detection method using the FD-kNN is developed to explicitly account for some unique characteristics of most semiconductor processes, namely nonlinearity and multimodal trajectories. The traditional kNN algorithm is adapted such that only normal operation data is needed to build a process model, i.e., the distribution of the kNN squared distances of normal samples. Because the developed FD-kNN method makes no assumption about the linearity of the process and it detects abnormality based on local neighborhoods, the developed FD-kNN method naturally handles process nonlinearity and multimodal environment. In addition, data preprocessing required by the proposed FD-kNN method is performed automatically without human intervention, which is essential for online applications. These capabilities are illustrated by simulated examples. Also, the FD-kNN method performs better than PCA in a real industrial example with limited data preprocessing, where multimodal distribution is observed in the data set. In this paper, we highlight the weakness of PCA and the strength of FD-kNN in handling process nonlinearity and multimodal distribution that exist in many semiconductor processes. However, it is not our intention or desire to imply that FD-kNN outperforms PCA in all cases. Instead, we want to provide an alternative fault detection method to the semiconductor research community so that the strengths and weaknesses of the proposed method can be examined in other cases. Also, it would be interesting to explore the fault diagnosis capability of the FD-kNN method once a fault is detected and our work in this direction is underway.

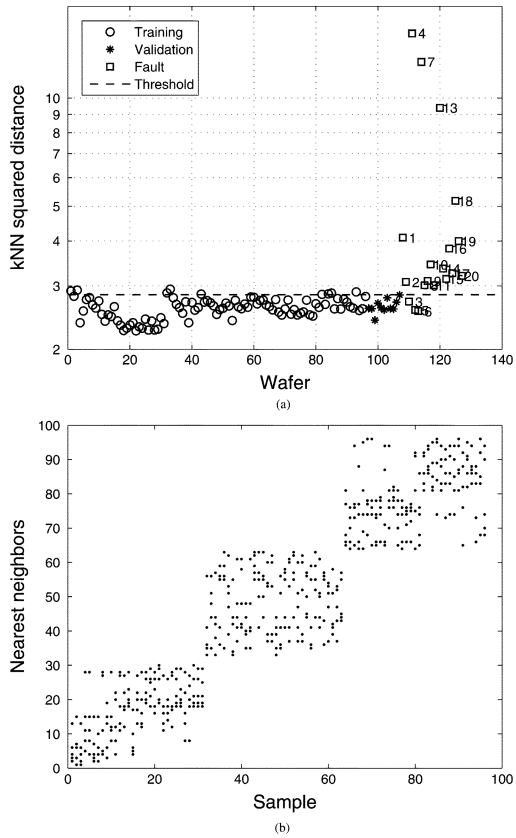


Fig. 14. (a) Fault detection using FD-kNN. (b) Mapping of nearest neighbors for training data with $k = 5$.

REFERENCES

[1] T. F. Edgar, S. W. Butler, W. J. Campbell, C. Pfeiffer, C. Bode, S. B. Hwang, K. S. Balakrishnan, and J. Hahn, "Automatic control in microelectronics manufacturing: Practices, challenges, and possibilities," *Automatica*, vol. 36, no. 11, pp. 1567–1603, 2000.

[2] J. Moyne and B. Van Eck, "Providing APC and integrated metrology input to the international technology roadmap for semiconductors (ITRS)—Factory integration," in *Proc. AEC/APC Symp. XVIII*, Westminster, CO, Sep. 2006.

[3] A. Ison and C. J. Spanos, "Robust fault detection and fault classification of semiconductor manufacturing equipment," in *Proc. Int. Symp. Semiconductor Manufacturing*, Tokyo, Japan, Oct. 1996.

[4] Q. P. He, "Novel multivariate fault detection methods using Mahalanobis distance," in *Proc. AEC/APC Symp. XVII*, Indian Wells, CA, Sep. 2005.

[5] J. Wang and Q. P. He, "A pattern matching approach for fast disturbance detection and classification using Bayesian statistics," in *Proc. AEC/APC Symp. XVII*, Indian Wells, CA, Sep. 2005.

[6] Q. P. He and J. Wang, "Statistical fault detection of batch processes in semiconductor manufacturing," in *Proc. AIChE Annu. Conf.*, San Francisco, CA, Nov. 2006.

[7] J. Wang and Q. P. He, "A new Bayesian approach for fast disturbance detection and classification in microelectronics manufacturing," *IEEE Trans. Semicond. Manuf.,* to be published.

[8] T. Adamson, G. Moore, M. Passow, J. Wong, and Y. Xu, "Strategies for successfully implementing fab-wide FDC methodologies in semiconductor manufacturing," in *Proc. AEC/APC Symp. XVIII*, Westminster, CO, Sep. 2006.

[9] T. Moore, B. Harner, G. Kestner, C. Baab, and J. Stanchfield, "Intel's FDC proliferation in 300 mm HVM: Progress and lessons learned," in *Proc. AEC/APC Symp. XVIII*, Westminster, CO, Sep. 2006.

[10] R. Dunia and S. J. Qin, "Subspace approach to multidimensional fault identification and reconstruction," *AIChE J.*, vol. 44, pp. 1813–1831, 1998.

[11] Q. P. He, J. Wang, and S. J. Qin, "A new fault diagnosis method using fault directions in Fisher discriminant analysis," *AIChE J.*, vol. 51, no. 2, pp. 555–571, 2005.

[12] T. Kourti and J. F. MacGregor, "Process analysis, monitoring and diagnosis, using multivariate projection methods," *Chemometrics Intell. Lab. Syst.*, vol. 28, pp. 3–21, 1995.

[13] B. M. Wise, N. B. Gallagher, S. W. Butler, D. D. White, Jr., and G. G. Barna, "A comparison of principal component analysis, multiway principal component analysis, trilinear decomposition and parallel factor analysis for fault detection in a semiconductor etch process," *J. Chemomotrics*, vol. 13, pp. 379–396, 1999.

[14] B. M. Wise, N. B. Gallagher, and E. B. Martin, "Application of parafac2 to fault detection and diagnosis in semiconductor etch," *J. Chemomotrics*, vol. 15, pp. 285–298, 2001.

[15] G. Cherry, R. Good, and S. J. Qin, "Semiconductor process monitoring and fault detection with recursive multiway pca based on a combined index," in *Proc. AEC/APC Symp. XIV*, Salt Lake City, UT, Sep. 2002.

[16] H. H. Yue and M. Tomoyasu, "Weighted principal component analysis and its applications to improve FDC performance," in *Proc. 43rd IEEE Conf. Decision and Control*, Atlantis, Paradise Island, Bahamas, Dec. 2004, pp. 4262–4267.

[17] J. Wong, "Batch PLS analysis and FDC process control of within lot SiON gate oxide thickness variation in sub-nanometer range," in *Proc. AEC/APC Symp. XVIII*, Westminster, CO, Sep. 2006.

[18] T. Kourti, P. Nomikos, and J. F. MacGregor, "Analysis, monitoring, and fault diagnosis of batch processes using multi-block and multi-way PLS," *J. Proc. Cont.*, vol. 5, pp. 277–284, 1995.

[19] P. Nomikos and J. F. MacGregor, "Multi-way partial least squares in monitoring batch processes," *Chemometrics Intell. Lab. Syst.*, vol. 30, pp. 97–108, 1995.

[20] S. J. Qin, S. Valle-Cervantes, and M. Piovoso, "On unifying multiblock analysis with applications to decentralized process monitoring," *J. Chemomotrics*, vol. 15, pp. 715–742, 2001.

[21] J. A. Westerhuis, T. Kourti, and J. F. MacGregor, "Comparing alternative approaches for multivariate statistical analysis of batch process data," *J. Chemomotrics*, vol. 13, pp. 397–413, 1999.

[22] R. O. Duda, P. E. Hart, and D. G. Stork*, Pattern Classification*, 2nd ed. New York: Wiley, 2001.

[23] K. A. Chamness, "Multivariate fault detection and visualization in the semiconductor industry," Ph.D. dissertation, Univ. Texas, Austin, Dec. 2006.

[24] A. K. Smilde, "Comments on three-way analyses used for batch process data," *J. Chemomotrics*, vol. 15, no. 11, pp. 19–27, 2001.

[25] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics Intell. Lab. Syst.*, vol. 2, pp. 37–52, 1987.

[26] J. E. Jackson and G. Mudholkar, "Control procedures for residuals associated with principal component analysis," *Technometrics*, vol. 21, pp. 341–349, 1979.

[27] S. J. Qin, "Statistical process monitoring: Basics and beyond," *J. Chemomotrics*, vol. 17, pp. 480–502, 2003.

[28] J. V. Kresta, J. F. MacGregor, and T. E. Marlin, "Multivariate statistical monitoring of processes," *Can. J. Chem. Eng.*, vol. 69, no. 1, pp. 35–47, 1991.

[29] B. M. Wise and N. B. Gallagher, "The process chemometrics approach to process monitoring and fault detection," *J. Proc. Cont.*, vol. 6, pp. 329–348, 1996.

[30] C. Schmidt, S. Bartl, M. Speil, J. Straer, G. Ernst, and G. Spitzlsperger, "Fault detection and classification (FDC) for a via-etching-process," in *Proc. 5th Eur. AEC/APC Conf.*, Dresden, Germany, Apr. 2004.

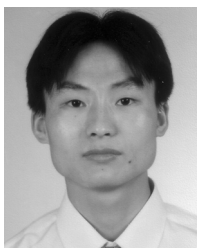
[31] E. L. Russell, L. H. Chiang, and R. D. Braatz, "Fault detection in industrial processes using canonical variate analysis and dynamic principal component analysis," *Chemometrics Intell. Lab. Syst.*, vol. 51, pp. 81–93, 2000.

[32] D. J. Hand*, Kernel Discriminant Analysis*. New York: Research Studies, 1982.

[33] Eigenvector Res. Inc. *Metal Etch Data for Fault Detection Evaluation*, 1999, [Online]. Available: http://software.eigenvector.com/Data/Etch/index.html

[34] D. White, G. G. Barna, S. W. Butler, B. M. Wise, and N. B. Gallagher, "Methodology for robust and sensitive fault detection," *Electrochem. Soc. Proc.*, pp. 55–79, Sep. 1997.

[35] B. E. Goodlin, D. S. Boning, H. H. Sawin, and B. M. Wise, "Simultaneous fault detection and classification for semiconductor manufacturing tools," *J. Electrochem. Soc.*, vol. 150, no. 12, pp. 778–784, 2003.



Q. Peter He (S'01–M'05) received the B.S. degree in chemical engineering from Tsinghua University, Beijing, China, in 1996, and M.S. and Ph.D. degrees in chemical engineering from the University of Texas, Austin, in 2002 and 2005, respectively.

He is currently an Assistant Professor at Tuskegee University, Tuskegee, AL. His research interests include process modeling, monitoring, optimization and control, with special interests in the modeling and optimization, fault detection and classification of batch processes such as semiconductor manufacturing and pharmaceutical processes. He is also interested in molecular dynamic simulation and Monte Carlo simulation of micro/nanoelectronic and biological systems. He has had over three years of experience in semiconductor and chemical industries.

Jin Wang (S'01–M'04) received the B.S. degree in chemical engineering from Tsinghua University, Beijing, China, in 1994, and the M.S. and Ph.D. degrees in chemical engineering from the University of Texas, Austin, in 2001 and 2004, respectively.

From 2002 to 2006, she was a Process Development Engineer and later a Senior Process Development Engineer at Advanced Micro Devices, Inc. Since 2006, she has been with Auburn University, Auburn, AL, as an Assistant Professor. Her research interests include system identification, semiconductor process modeling and control, fault detection and classification, control performance monitoring, and systems biology. She holds ten U.S. patents.