2008 American Control Conference
Westin Seattle Hotel, Seattle, Washington, USA
June 11-13, 2008

WeC11.4

Principal Component based k-Nearest-Neighbor Rule for Semiconductor Process Fault Detection

Q. Peter He and Jin Wang

Abstract— **Fault detection and classification (FDC) has been recognized in the semiconductor industry as an integral component of advanced process control (APC) framework in improving overall equipment efficiency (OEE). To explicitly account for the unique characteristics of the semiconductor processes, such as nonlinearity in most batch processes, multimodal batch trajectories due to product mix, the fault detection method based on the k-nearest-neighbor rule (FD-kNN) has been developed previously for fault detection in semiconductor manufacturing. However, because FD-kNN does not generate a classifier offline, it is computational and storage intensive, which could make it difficult for online process monitoring. To take the advantages of principal component analysis (PCA) in dimensionality reduction and FD-kNN in nonlinearity and multimode handling, a principal component based kNN (PC-kNN) is proposed. Two simulated examples and an industrial example are used to demonstrate the performance of the proposed PC-kNN method in fault detection.**

I. INTRODUCTION

Massive amount of trace or machine data is made available in today's semiconductor industry and fault detection has been one focus of recent efforts to reduce wafer scrap, increase equipment uptime and reduce the usage of test wafers [1-7] . Among all fault detection methods, multivariate statistical fault detection methods such as principal component analysis (PCA), partial least squares (PLS) have drawn increasing interest in semiconductor manufacturing industry recently [8-12] . 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 [13-17] . However, some unique characteristics of semiconductor manufacturing processes, such as multimodal batch trajectories and nonlinearity, have posed difficulties to these multivariate statistical methods. In our previous work [18], a fault detection method based on k-nearest-neighbor (kNN) rule [19] (FD-kNN) was developed to explicitly account for the above mentioned unique characteristics of semiconductor processes. However, there is a drawback associated with the FD-kNN method: it can be computational and storage intensive for large-scale processes where the number of features or variables can easily exceed thousands after batch unfolding. This drawback may prevent it from being implemented for online process monitoring where thousands or tens of thousands of such models are running concurrently. For example, by 2006 there were more than 7,000 active FDC models at IBM [20] and over 30,000 models at Intel [21]. To reduce computation complexity and storage/memory requirements, in this work we propose principal component based kNN method (PC-kNN). Two steps are involved in the proposed method. In the first step, PCA is applied to the original process data set to transform high dimensional data into few principal components (PC's). The few PC's in general capture key features contained by the process data. In the second step, the kNN fault detection method is applied to the scores in the principal subspace to detect potential faults.

II. METHODS

In this section, we briefly review relevant methods: PCA and FD-kNN. Throughout this paper, a scalar is denoted by an italic lower-case character (x), a vector by a bold lower-case character ($\mathbf{x}$), a matrix by a bold upper-case character ($\mathbf{X}$) and a three-way array by an underlined bold upper-case character ($\underline{\mathbf{X}}$). These notations are consistent with notations used by others in the statistical process control community (see e.g., [22]).

A. Principal component analysis (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 [23] or a singular value decomposition (SVD) algorithm, the scaled matrix $\mathbf{X}$ is decomposed as follows

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

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 (PC's). For fault detection in a new sample vector $\mathbf{x}$, the squared prediction error (SPE) and the Hotelling's T^2 are often used.

B. Fault detection using k-nearest-neighbor rule (FD-kNN)

The FD-kNN method [4], [18] is based on the idea that the trajectory of a normal sample is similar to the trajectories of normal samples in the training data; while 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 the distribution of normal training samples' distances to their

---

Q.P. He is with the Department of Chemical Engineering, Tuskegee University, Tuskegee, AL 36088, USA `qhe@tuskegee.edu`

J. Wang is with the Department of Chemical Engineering, Auburn University, Auburn, AL 36849, USA `wang@auburn.edu`

978-1-4244-2079-7/08/$25.00 ©2008 AACC.

nearest neighboring training samples can be determined, a threshold can be defined with certain confidence level and the unclassified sample is considered normal if the distance to its nearest neighboring training samples is below the threshold. Otherwise the sample is detected as a fault. The FD-kNN method consists of two parts: model building and 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[1].
  2) Calculation of the kNN squared distance for each sample. The kNN squared distance of sample i ($\mathcal{D}_i^2$) is defined as

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

     where d_{ij}^2 denotes squared Euclidean distance from sample i to its j-th nearest neighbor.
  3) Determination of a threshold for fault detection. Because the distribution of $\mathcal{D}_i^2$ can be approximated by a noncentral chi-square distribution [18], the threshold $\left(\mathcal{D}_\alpha^2\right)$ with a significance level α can be determined. Another common way to set the threshold is based on the calibration or testing data under normal operation conditions [8], [24]. 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 $\mathcal{D}_x^2$ (Eqn. (2)).
  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.

Because FD-kNN is based on the kNN rule which is a nonlinear classifier, it naturally handles process nonlinearity. Also, because the FD-kNN method detects faults based on local neighborhoods of similar batches, it is well suited for multimodal data set in which batches can be grouped into subsects with different characteristics. More detailed discussion on the FD-kNN method can be found in [4], [18].

III. PRINCIPAL COMPONENT BASED K-NEAREST-NEIGHBOR RULE (PC-KNN)

A disadvantage of the FD-kNN method is that it requires considerable memory resources for large systems as all of the training data must be retained to compute $\mathcal{D}_x^2$ for each incoming sample $\mathbf{x}$. Also, the computational complexity is directly proportional to the dimensionality of the data. Consequently, there is a practical upper limit to both the number of records and the data dimensionality that may be processed if the algorithm is implemented online with thousands or tens of thousands of models running concurrently. Although there are algorithmic techniques that can reduce the computational burden, the implementation of these techniques is nontrivial [19]. To reduce memory requirement and computation time of the FD-kNN method while still keeping its advantage of handling nonlinear and multimodal data, we propose an improved FD-kNN algorithm based on principal component analysis, denoted as PC-kNN. In PC-kNN, we first make use of the dimensionality reduction and information preserving property of PCA to extract principal components (PCs) that contain key information of the data set. For a training dataset $X \in \Re^{n\times m}$ with n samples and m variables, PCs are extracted by Eqn. (1). In the model building phase, instead of applying FD-kNN algorithm directly to the raw training data that usually has high dimensionality, we apply FD-kNN algorithm to the score matrix $\mathbf{T}$ corresponding to the extracted PCs. PC-kNN squared distance of sample i ($\mathbb{D}_i^2$) is defined as

$$\mathbb{D}_i^2 = \sum_{j=1}^{k} \delta_{ij}^2 \tag{3}$$

where δ_{ij}^2 denotes squared Euclidean distance from sample i to its j-th nearest neighbors in the PC subspace, i.e., from $\mathbf{T}(i,:)$ to its j-th nearest neighbor in $\mathbf{T}$. Similar to [18], the distribution of PC-kNN distances can be approximated by a noncentral χ^2 distribution and a threshold $\mathbb{D}_\alpha^2$ can be determined theoretically or practically. For a test sample $\mathbf{x} \in \Re^{1\times m}$, its scores are obtained by projecting onto the loading matrix $\mathbf{P}$: $\mathbf{t} = \mathbf{xP}$. The k-nearest neighbors in the PC subspace can be found by computing distances between $\mathbf{t}$ and $\mathbf{T}$ and its PC-kNN squared distance $\mathbb{D}_x^2$ is calculated using Eqn. (3). $\mathbb{D}_x^2$ is then compared with $\mathbb{D}_\alpha^2$ to determine whether $\mathbf{x}$ is a fault. By doing so, the proposed PC-kNN method can significantly reduce the computation time and memory requirement without sacrificing fault detection capability.

Notice that when PCA is applied to a nonlinear data set, PCA will not be able to extract the nonlinear relations. However, PCA will still find the directions that correspond to the largest variations of the data set. In other words, PCA may not be able to capture all nonlinear features, but the dominant nonlinear features will be captured because of the associated large variations.

IV. ILLUSTRATIVE EXAMPLES

In this section, two simple examples with 12 variables are given to illustrate how the proposed PC-kNN fault detection method works in the presence of nonlinearity and multimodal distribution. In both cases, it is assumed that the first two variables (y_1 and y_2) are dominant modes while others are essentially white noise after subtracting setpoints.

---

[1] The determination of an appropriate k value is discussed later in this work.

A. Nonlinear case

The first simulation example is a nonlinear case with the following dominant process mode:

$$y_1 = y_2^2 + \text{noise} \tag{4}$$

500 normal runs are used for training, 100 normal runs are used for validation, and 5 faults are introduced. Fig. 1 shows the scatter plot of the training, validation and fault samples in the dominant mode.

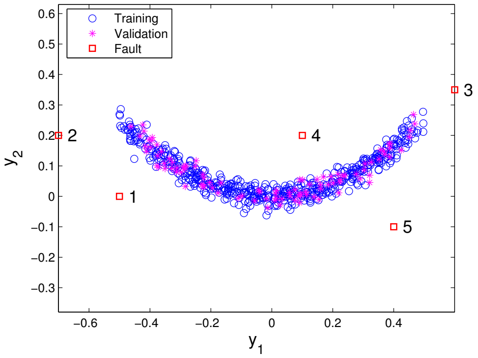
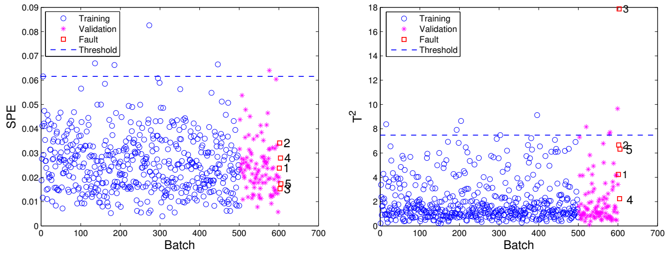


Fig. 1. Scatter plot - nonlinear case: first two characteristic variables

PCA is first applied to detect the faults in the data set. At the confidence level of 99%, the detection results with 2 PC's are shown in Fig. 2. From Fig. 2, we see that PCA does not perform well and the majority of the faults are not detected by PCA. The result is expected because of the nonlinearity in the process data.

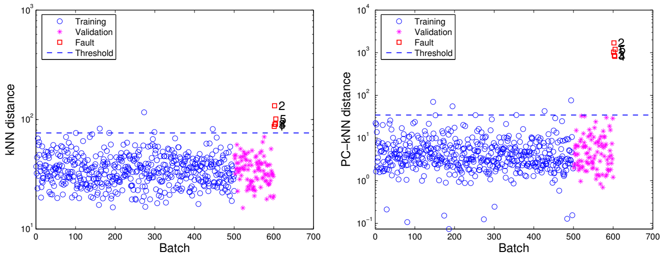


(a) (b)

Fig. 2. Fault detection - nonlinear case: (a) SPE chart, (b) T^2 chart

Next, FD-kNN is applied to the nonlinear case data set. The number of nearest neighbors k is set to be 3. The detection result is shown in Fig. 3 (a) where all 5 faults are successfully detected. If we apply PC-kNN to the scores obtained earlier based on 2 PC's and set k as 3, the result is shown in Fig. 3 (b). Notice that because PCA effectively removes noise by extracting principal components, PC-kNN performs better than FD-kNN.

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



(a) (b)

Fig. 3. Fault detection - nonlinear case: (a) the FD-kNN method, (b) the proposed PC-kNN method

B. Multimodal case

The second simulation example is a multimodal case where the process is performed on two tools with different process gains and an offset between the tools. Again, the process is mainly characterized by two variables (y_1 and y_2) out of 12 variables:

$$\begin{aligned} \text{Tool A: } y_1 &= 2y_2 + \text{noise} \\ \text{Tool B: } y_1 &= 1.5y_2 + 6 + \text{noise} \end{aligned} \tag{5}$$

300 normal runs are conducted on each tool so that totally 600 normal data points are collected. 500 normal runs are randomly selected from the two tools for training, the remaining 100 normal runs are used for validation, and 5 faults are introduced. The data set, including the training, validation and fault samples, is visualized in Fig. 4.

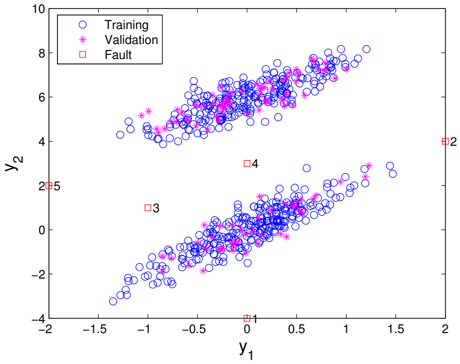


Fig. 4. Scatter plot - multimodal case: first two characteristic variables

Fault detection results using PCA SPE and T^2 charts with 2 PC's are shown in Fig. 5 (a) and Fig. 5 (b). We see that not all faults can be detected by PCA because of the bimodal distribution.

Next, FD-kNN is applied to the multimodal data set. The number of nearest neighbors k is set to be 3 as in the previous example. The result is shown in Fig. 6 (a). If we apply PC-kNN to the scores obtained based on 2 PC's with $k = 3$, the result is shown in Fig. 6 (b). Notice that both FD-kNN and PC-kNN successfully detect all 5 faults.

Again, because PCA can only extract linear structure from the data, multimodal environment imposes limitations on

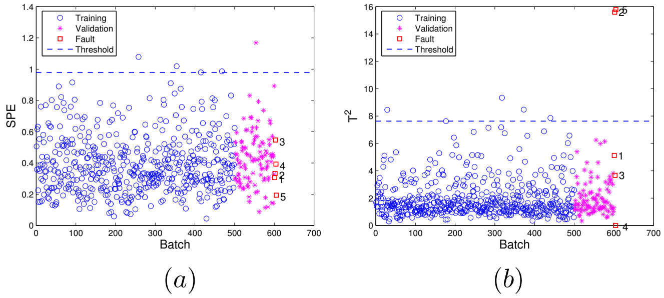


Fig. 5. Fault detection - multimodal case: (a) SPE chart, (b) T^2 chart

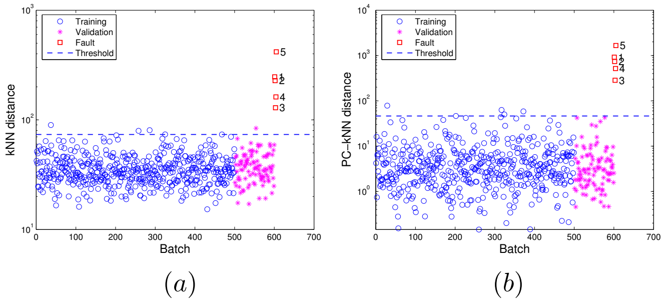


Fig. 6. Fault detection - multimodal case: (a) the FD-kNN method, (b) the proposed PC-kNN method

PCA and makes it less effective. Because FD-kNN and PC-kNN methods focus on local neighborhoods, they do not suffer degradation from multimodal distribution.

V. INDUSTRIAL EXAMPLES

In this section, an industrial example is used to compare different fault detection methods. 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. [8], [25]. The data consists of 108 normal wafers taken during 3 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 study. More detailed description on the faults can be found in [8]. Since steps 4 and 5 are the main etch steps, as in [8], only the data points from steps 4 and 5 are used in this work, and 19 non-setpoint process variables used in [8] are included for fault detection. The physics of the problem suggests that these variables should be relevant to process and final product state [8].

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

- 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 seconds to 112 seconds.
- 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 seconds to 52 seconds. In other words, even batches of equal length may not follow exactly the same time trajectory.
- Process drift and shift: For the etch process, drift and shift in the data are primarily due to the following sources [8]: aging of the etcher over a clean cycle; differences in the incoming materials; drift in the process monitoring sensors; 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. Two examples (variables EndPt A and and TCP Load) illustrating this characteristic are shown in Fig. 7.

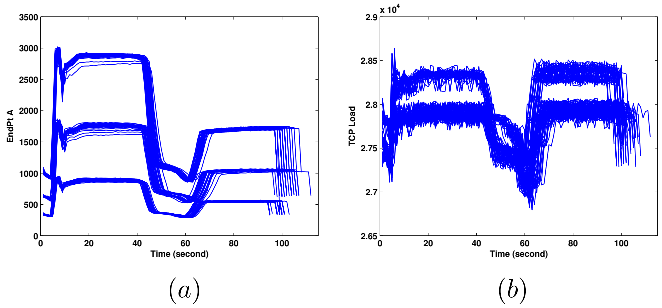


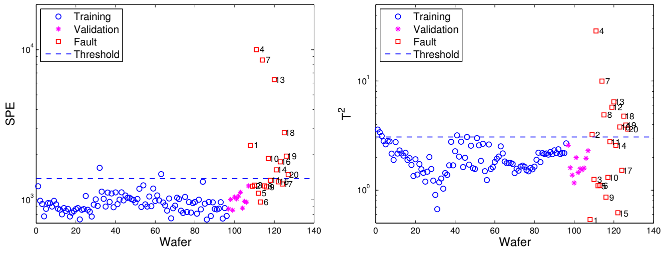
Fig. 7. Mean and covariance change in the data set: (a) EndPt A and (b) TCP Load

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 [8]. 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 [26]. In addition, to discriminate against process drift, a high-pass filter can be employed to remove the low frequency drift in the process [27]. Although 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 work is to maximize the level of automation in fault detection. Therefore, we compare fault detection methods with minimum data preprocessing in this work. The first step is to obtain equal length batch records. In this step, the initial 5 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 above two-step data preprocessing can be done automatically in the production environment. For MPCA analysis, the data is further unfolded batch-wise to obtain a 2D array. Next we apply MPCA, FD-kNN and PC-kNN to the preprocessed data sets.

B. *Fault Detection*

MPCA is first used to analyze the data where the covariance matrix is estimated directly - making no distinction between data from different experiments. Totally 3 PC's 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. 8 (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. 8 (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. As discussed earlier, the data were collected from the 3 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 manufacturing environment, tool state shift is inevitable due to a variety of reasons such as preventive maintenance (PM) and part replacement.



(a) (b)

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

Next, FD-kNN and PC-kNN are applied to detecting fault in the etch data set. The number of nearest neighbors is set as 3 for both methods and 3 PC's are used for PC-kNN. The results are shown as semi-log plots in Fig. 9 and Fig. 10 (a). To illustrate where each sample's neighbors are located, in Fig. 10 (b) we draw the mapping of 3 nearest neighbors for the training wafers obtained by PC-kNN. Fig. 10 (b) shows the same multimodal characteristic as in Fig. 7 that wafers are grouped in 3 clusters. From Fig. 10 (b) we see that each wafer finds its k nearest neighbors in its own group, which is consistent with our discussion earlier that the PC-kNN method handles multimodal distribution naturally by focusing on local neighborhoods. The detailed fault detection results from MPCA, FD-kNN and PC-kNN are shown in Table I.

This example illustrate that both FD-kNN and PC-kNN are capable of handling multimodal data without additional data preprocessing. Note that if further data preprocessing is performed to eliminate the multimodal distribution, PCA performs similarly to FD-kNN and PC-kNN.

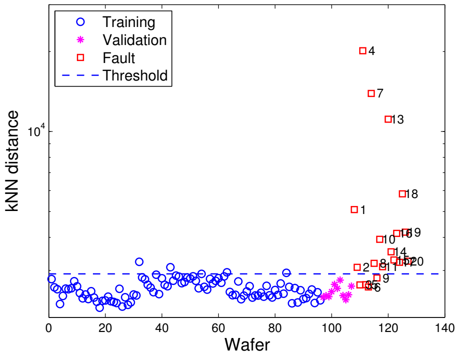
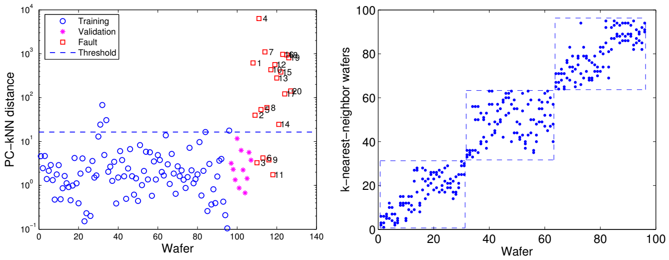


Fig. 9. Fault detection using FD-kNN



(a) (b)

Fig. 10. PC-kNN: (a) fault detection, (b) mapping of the nearest neighbors for $k = 3$

VI. CONCLUSIONS

In this work, a new fault detection method using the principal component based k-nearest-neighbor rule (PC-kNN) is developed to explicitly account for some unique characteristics of most semiconductor processes, namely nonlinearity and multimodal trajectories. PC-kNN overcomes the drawback of the FD-kNN method on computation complexity and memory requirement. The FD-kNN algorithm is adapted such that low dimensional PCA extracted features are used instead of the high dimensional original variables. This strategy significantly reduces computation time and memory consumption. Because the developed FD-kNN method makes no assumption about the linearity of the scores and it detects abnormality based on local neighborhoods, PC-kNN method

TABLE I

FAULT DETECTED BY DIFFERENT METHODS

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

naturally handles process nonlinearity and multimodal distribution. In addition, because noise impact is reduced by PCA, PC-kNN sometimes outperforms FD-kNN in fault detection. The choice of k for FD-kNN and PC-kNN is uncritical. 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. It is worth to note that PC-kNN works in the principal subspace similar to PCA T^2 index except that PC-kNN handles nonlinearity and multimodal distribution in the scores. Therefore, PC-kNN performs essentially the same as T^2 in the linear, unimodal cases while PC-kNN outperforms T^2 in the presence of nonlinearity or multimodal distribution in the scores. Because PC-kNN makes no use of residuals, any fault that occurs in the residual space only will not be detected by PC-kNN. One possible solution is to join faults detected by PC-kNN and faults detected by SPE so that the full space is covered in fault detection.

REFERENCES

[1] A. Ison and C. J. Spanos. Robust fault detection and fault classification of semiconductor manufacturing equipment. In *Proc. Int. Symp. Semiconductor Manufacturing*, Tokyo, Japan, October 1996.

[2] Q.P. He. Novel multivariate fault detection methods using Mahalanobis distance. In *AEC/APC Symposium XVII*, Indian Wells, CA, September 2005.

[3] J. Wang and Q.P. He. A pattern matching approach for fast disturbance detection and classification using Bayesian statistics. In *AEC/APC Symposium XVII*, Indian Wells, CA, September 2005.

[4] Q.P. He and J. Wang. A multivariate fault detection method using k-nearest-neighbor rule. In *Proc. AEC/APC Symp. XVIII*, Westminster, CO, 2006.

[5] J. Wang and Q.P. He. A Bayesian approach for disturbance detection and classification and its application to state estimation in run-to-run control. *IEEE Trans. Semic. Manuf.*, 20(2):126–136, 2007.

[6] T. Adamson, G. Moore, M. Passow, J. Wong, and Y. Xu. Strategies for successfully implementing fab-wide FDC methodologies in semiconductor manufacturing. In *AEC/APC Symposium XVIII*, Westminster, CO, September 2006.

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

[8] 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*, 13:379–396, 1999.

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

[10] G. Cherry, R. Good, and S.J. Qin. Semiconductor process monitoring and fault detection with recursive multiway pca based on a combined index. In *AEC/APC Symposium XIV*, Salt Lake City, September 2002.

[11] H.H. Yue and M. Tomoyasu. Weighted principal component analysis and its applications to improve FDC performance. In *43rd IEEE Conference on Decision and Control*, pages 4262 – 4267, Atlantis, Paradise Island, Bahamas, Dec. 2004.

[12] J. Wong. Batch PLS analysis and FDC process control of within lot SiON gate oxide thickness variation in sub-nanometer range. In *AEC/APC Symposium XVIII*, Westminster, CO, September 2006.

[13] 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.*, 5:277–284, 1995.

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

[15] S.J. Qin, S. Valle-Cervantes, and M. Piovoso. On unifying multi-block analysis with applications to decentralized process monitoring. *J. Chemomotrics*, 15:715–742, 2001.

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

[17] Q.P. He, J. Wang, and S.J. Qin. A new fault diagnosis method using fault directions in Fisher discriminant analysis. *AIChE J.*, 51(2):555–571, 2005.

[18] Q.P. He and J. Wang. Fault detection using k-nearest neighbor rule for semiconductor manufacturing processes. *IEEE Trans. Semic. Manuf.*, 20(4):345–354, 2007.

[19] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. John Wiley & Sons, New York, second edition, 2001.

[20] T. Adamson et al. Strategies for successfully implementing fab-wide fdc methodologies in semiconductor manufacturing. In *Proc. AEC/APC Symp. XVIII*, Westminster, CO, 2006.

[21] T. Moore et al. Intel's fdc proliferation in 300mm hvm: Progress and lessons learned. In *Proc. AEC/APC Symp. XVIII*, Westminster, CO, 2006.

[22] A.K Smilde. Comments on three-way analyses used for batch process data. *J. Chemomotrics*, 15(11):19–27, 2001.

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

[24] 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.*, 51:81–93, 2000.

[25] Eigenvector Research Inc. Metal etch data for fault detection evaluation. `http://software.eigenvector.com/Data/Etch/index.html`, 1999.

[26] D. White, G.G. Barna, S.W. Butler, B.M. Wise, and N.B. Gallagher. Methodology for robust and sensitive fault detection. In *Electrochem. Soc. Proc.*, pages 55–79, September 1997.

[27] B.E. Goodlin, D.S. Boning, H.H. Sawin, and B.M. Wise. Simultaneous fault detection and classification for semiconductor manufacturing tools. *J. The Electrochemical Society*, 150(12):778–784, 2003.