Chemometrics and Intelligent Laboratory Systems 96 (2009) 132–143

Contents lists available at ScienceDirect

Chemometrics and Intelligent Laboratory Systems

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

Moving window kernel PCA for adaptive monitoring of nonlinear processes

Xueqin Liu [a,b], Uwe Kruger [c,*], Tim Littler [a], Lei Xie [b,*], Shuqing Wang [b]

[a] School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Ashby Building, Stranmillis Road, Belfast BT9 5AH, UK
[b] State Key Lab of Industrial Control Technology, Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou 310027, PR China
[c] Department of Electrical Engineering, The Petroleum Institute, P.O. Box 2533, Abu Dhabi, United Arab Emirates

ARTICLE INFO

Article history:
Received 13 October 2008
Received in revised form 25 December 2008
Accepted 7 January 2009
Available online 23 January 2009

Keywords:
Nonlinear process
Kernel PCA
Moving window
Multivariate statistical process control
Adaptive
Numerically efficient

ABSTRACT

This paper discusses the monitoring of complex nonlinear and time-varying processes. Kernel principal component analysis (KPCA) has gained significant attention as a monitoring tool for nonlinear systems in recent years but relies on a fixed model that cannot be employed for time-varying systems. The contribution of this article is the development of a numerically efficient and memory saving moving window KPCA (MWKPCA) monitoring approach. The proposed technique incorporates an up- and downdating procedure to adapt (i) the data mean and covariance matrix in the feature space and (ii) approximates the eigenvalues and eigenvectors of the Gram matrix. The article shows that the proposed MWKPCA algorithm has a computation complexity of $O(N^2)$, whilst batch techniques, e.g. the Lanczos method, are of $O(N^3)$. Including the adaptation of the number of retained components and an l-step ahead application of the MWKPCA monitoring model, the paper finally demonstrates the utility of the proposed technique using a simulated nonlinear time-varying system and recorded data from an industrial distillation column.

1. Introduction

The demand for effective quality monitoring and safe operation in the petrochemical industry has propelled research into statistical-based fault detection and diagnosis methods over the past few decades. Multivariate statistical methods such as principal component analysis (PCA) [1,42,47], partial least squares [24,32,46] and more recently independent component analysis [10,27,30] have been developed and applied for this purpose. Among them, PCA is the most popular one, which relates to its conceptual simplicity. However, such methods collectively assume linear variable interrelationships, which hamper their application if these relationships are nonlinear.

Kramer [22] developed one of the first nonlinear extensions of PCA, which rely on auto-associative neural networks (ANNs). Reference [9] proposed a simplification that incorporated principal curves into this neural network structure. However, the nonlinear function is approximated by a linear combination of several univariate nonlinear functions, which represents a restriction of generality [19]. This was addressed by introducing input-training neural networks [19,38]. A detailed review in Reference [25] suggested that neural network based techniques may not represent a generic nonlinear extension of PCA.

The analysis in Reference [25] also yielded that KPCA [37] is a generic nonlinear PCA extension, which can efficiently compute PCs in a high-dimensional feature space using integral operators and nonlinear kernel functions. The core idea of KPCA is to first map the data space into a feature space using a nonlinear mapping and then compute the PCs in the feature space. It should also be noted that KPCA only requires the solution of an eigenvalue problem, and, since it can incorporate different kernel functions, KPCA can handle a wide range of nonlinearities. In addition, KPCA does not require a pre-estimate of the number of retained PCs.

Despite recently reported KPCA-based monitoring applications [8,16,26,45], the following problems arise: (i) the identification of a KPCA monitoring model requires the storage of the symmetric *kernel matrix*, whose dimension is given by the number of reference samples and (ii) the monitoring model is fixed which may produce false alarms if the process is naturally time-varying. The latter problem has been addressed by a recursive KPCA formulation [5,21,44], similar in approach to the work on linear recursive PCA [29,43]. However, the kernel matrix grows in size each time a new data point becomes available which is practically problematic (memory and computational requirements).

Implementing a moving window approach, as proposed by Hoegaerts et al. [17], overcomes this problem and produces a constant size of the kernel matrix and a constant speed of adaptation [42]. The adaptation of the kernel matrix relies on a simultaneous up- and downdating, which is memory efficient compared to a recalculation of the kernel matrix. It is important to note, however, that alterations in the sample mean of the original as well as the transformed variable set in the feature space were not considered in [17]. This, however, can

* Corresponding authors. Kruger is to be contacted at Department of Electrical Engineering, The Petroleum Institute, P.O. Box 2533, Abu Dhabi, United Arab Emirates. Xie, State Key Lab of Industrial Control Technology, Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou 310027, PR China. Tel.: +971 2607 5150; fax: +971 2607 5200.
E-mail addresses: ukruger@pi.ac.ae (U. Kruger), leix@iipc.zju.edu.cn (L. Xie).

0169-7439/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.chemolab.2009.01.002

reduce the sensitivity for detecting fault conditions as discussed in Reference [44]. An adaptive monitoring scheme is therefore required for the mean of the transformed variable set in the feature space and for incorporating this change into the kernel matrix prior to the adaptation of the KPCA model.

This article proposes the adaptation of the mean and covariance matrix in the feature space and addresses the following issues, which the literature has not considered yet: (i) how to adapt the eigendecomposition of the Gram matrix numerically efficient, (ii) how to determine changes in the number of retained PCs, (iii) how to adapt the statistical confidence limits based on the adapted KPCA model and (iv) how to prevent the KPCA model from adapting incipiently developing faults. Compared to conventional methods, such as Rank-1 modification [3,11], inverse iteration [12] or the Lanzcos method [34,35] which this article shows to be of $O(N^3)$, the proposed method is of $O(N^2)$.

The paper is organized as follows. Preliminaries of KPCA-based monitoring are presented prior to the adaptation of the mean and covariance matrix in the feature space. Section 4 then presents the adaptation of the MWKPCA model. This is followed by summarizing the adaptation of the number of retained PCs and the univariate monitoring statistics. Next, a simulation example (Section 6) and the analysis of recorded data from an industrial distillation unit (Section 7) are given to demonstrate the effectiveness of the MWKPCA technique. Finally, Section 8 provides a concluding summary of this article.

2. Preliminaries

KPCA maps a set of M observations $\mathbf{x}\in\mathbb{R}^n$, n and $M\in\mathbb{N}$, $M>n$, into a high-dimensional feature space $\boldsymbol{\Phi}(\mathbf{x})\in F$ and subsequently performs a PCA on $\boldsymbol{\Phi}(\mathbf{x})$. Let $\mathbf{x}_i$ be the ith sample, the covariance matrix for $\boldsymbol{\Phi}(\mathbf{x})$ is:

$$\mathbf{C}_\Phi = \frac{1}{M-1}\sum_{i=1}^{M}(\boldsymbol{\Phi}(\mathbf{x}_i)-\mathbf{m}_\Phi)(\boldsymbol{\Phi}(\mathbf{x}_i)-\mathbf{m}_\Phi)^T = \frac{1}{M-1}\overline{\boldsymbol{\Phi}}(\mathbf{X})\overline{\boldsymbol{\Phi}}T(\mathbf{X}), \quad (1)$$

where $\mathbf{m}_\Phi = \frac{1}{M}\boldsymbol{\Phi}(\mathbf{X})1_M$ is the sample mean in the feature space, 1_M is an M-dimensional vector of ones, $\boldsymbol{\Phi}(\mathbf{X})=[\boldsymbol{\Phi}(\mathbf{x}_1),\boldsymbol{\Phi}(\mathbf{x}_2),\ldots,\boldsymbol{\Phi}(\mathbf{x}_M)]$, $\overline{\boldsymbol{\Phi}}(\mathbf{X}) = \boldsymbol{\Phi}(\mathbf{X}) - \frac{1}{M}\boldsymbol{\Phi}(\mathbf{X})\mathbf{E}_M$, $\mathbf{E}_M = 1_M\times 1_M^T$, is the mean centered feature matrix and $\mathbf{X}=[\mathbf{x}_1\ \mathbf{x}_2\ \cdots\ \mathbf{x}_M]$. Next, an eigendecomposition of $\mathbf{C}_\Phi$ is computed:

$$\mathbf{C}_\Phi\mathbf{u}_k = \frac{1}{M-1}\overline{\boldsymbol{\Phi}}(\mathbf{X})\overline{\boldsymbol{\Phi}}^T(\mathbf{X})\mathbf{u}_k = \lambda_k\mathbf{u}_k \qquad k=1,2,\ldots,M, \quad (2)$$

where λ_k and $\mathbf{u}_k$ represent the kth eigenvalue-eigenvector pair of $\mathbf{C}_\Phi$. Given that the explicit mapping function $\boldsymbol{\Phi}(\mathbf{x})$ is unknown, KPCA circumvents the use of $\boldsymbol{\Phi}(\mathbf{x})$ by utilizing the eigendecomposition of the centered Gram matrix $\mathbf{G}=\overline{\boldsymbol{\Phi}}^T(\mathbf{X})\overline{\boldsymbol{\Phi}}(\mathbf{X})\in\mathbb{R}^{M\times M}$:

$$\overline{\boldsymbol{\Phi}}^T(\mathbf{X})\overline{\boldsymbol{\Phi}}(\mathbf{X})\mathbf{v}_k = \zeta_k\mathbf{v}_k, \quad (3)$$

where $\zeta_k\in\mathbb{R}$ and $\mathbf{v}_k\in\mathbb{R}^M$ represent the kth eigenvalue-eigenvector pair of $\mathbf{G}$. Introducing the kernel definition $K(\mathbf{x}_i,\mathbf{x}_j)=\boldsymbol{\Phi}^T(\mathbf{x}_i)\boldsymbol{\Phi}(\mathbf{x}_j)$, $\mathbf{G}$ can be computed from the kernel matrix $\mathbf{K}=\boldsymbol{\Phi}^T(\mathbf{X})\boldsymbol{\Phi}(\mathbf{X})\in\mathbb{R}^{M\times M}$

$$\mathbf{G} = \mathbf{K} - \frac{1}{M}\mathbf{K}\mathbf{E}_M - \frac{1}{M}\mathbf{E}_M\mathbf{K} + \frac{1}{M^2}\mathbf{E}_M\mathbf{K}\mathbf{E}_M. \quad (4)$$

After constructing the PCA model in the feature space, the KPCA score vector, $\mathbf{t}\in\mathbb{R}^r$, for a new sample $\mathbf{x}\notin\mathbf{X}$ is given by:

$$t = \mathbf{U}^T\overline{\boldsymbol{\Phi}}(\mathbf{x}) = \mathbf{A}^T\boldsymbol{\Phi}^T(\mathbf{X})\left(\boldsymbol{\Phi}(\mathbf{x}) - \frac{1}{M}\boldsymbol{\Phi}(\mathbf{X})1_M\right) = \mathbf{A}^T\left(\mathbf{k}(\mathbf{X},\mathbf{x}) - \frac{1}{M}\mathbf{K}1_M\right). \quad (5)$$

Here, $\overline{\boldsymbol{\Phi}}(\mathbf{x})=\boldsymbol{\Phi}(\mathbf{x})-\mathbf{m}_\Phi$, r is the number of retained PCs, $\mathbf{U}=[\mathbf{u}_1\,\mathbf{u}_2\ldots\mathbf{u}_r]\in F$, $\mathbf{A}=\left[\mathbf{I}-\frac{1}{M}\mathbf{E}_M\right]\mathbf{V}\in\mathbb{R}^{M\times r}$, $\mathbf{V}=\left[\frac{\mathbf{v}_1}{\sqrt{\zeta_1}}\frac{\mathbf{v}_2}{\sqrt{\zeta_2}}\cdots\frac{\mathbf{v}_r}{\sqrt{\zeta_r}}\right]\in\mathbb{R}^{M\times r}$ and $\mathbf{k}(\mathbf{X},\mathbf{x})\in\mathbb{R}^M$ represents the kernel vector constructed from $\mathbf{X}$ and $\mathbf{x}$:

$$\mathbf{k}(\mathbf{X},\mathbf{x}) = \begin{pmatrix}K(\mathbf{x}_1,\mathbf{x}) & K(\mathbf{x}_2,\mathbf{x}) & \cdots & K(\mathbf{x}_M,\mathbf{x})\end{pmatrix}^T. \quad (6)$$

For process monitoring, KPCA relies on Hotelling's T^2 and Q statistics:

$$T^2 = \mathbf{t}^T\boldsymbol{\Lambda}^{-1}\mathbf{t} \qquad Q = \overline{\boldsymbol{\Phi}}(\mathbf{x})^T\left[\mathbf{I}-\mathbf{U}\mathbf{U}^T\right]\overline{\boldsymbol{\Phi}}(\mathbf{x}), \quad (7)$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix storing the variances of the score variables and $Q = K(\mathbf{x},\mathbf{x}) - \frac{2}{M}1_M^T\mathbf{k}(\mathbf{X},\mathbf{x}) + \frac{1}{M^2}1_M^T\mathbf{K}1_M - \mathbf{t}^T\mathbf{t}$.

3. Adaptation of mean and covariance matrix

This section describes the first contribution of this article and relates to the adaptation of the mean and covariance matrix in the feature space. Conventional moving window PCA (MWPCA) work [42] utilizes a two-step procedure that involves removing the oldest sample, further referred to as downdating, and then adding the newly available sample, defined as updating. This is followed by recomputing the PCA decomposition including the adapted variable mean and covariance matrix in a numerically efficient way, e.g. by applying inverse iteration or Lanczos method [29,42].

For the transformed data in the feature matrix, $\boldsymbol{\Phi}(\mathbf{X})$, applying the same two-step procedure involves the utilization of an intermediate window, which discards the effect of the oldest sample, and the "new" window that includes the impact of the newly available sample. Setting the window length to $N\in\mathbb{N}$ and defining the feature matrices that stores the transformed data of the intermediate window and the new window by $\boldsymbol{\Phi}(\widetilde{\mathbf{X}})$ and $\boldsymbol{\Phi}(\widehat{\mathbf{X}})$, respectively, the adaptation of the mean and covariance matrix of the transform variable set conceptually relies on the following procedure: $\boldsymbol{\Phi}(\mathbf{X})\Rightarrow\boldsymbol{\Phi}(\widetilde{\mathbf{X}})\Rightarrow\boldsymbol{\Phi}(\widehat{\mathbf{X}})$, where $\boldsymbol{\Phi}(\widetilde{\mathbf{X}})=[\boldsymbol{\Phi}(\mathbf{x}_2),\cdots,\boldsymbol{\Phi}(\mathbf{x}_N)]$, $\boldsymbol{\Phi}(\widehat{\mathbf{X}})=[\boldsymbol{\Phi}(\mathbf{x}_2),\cdots,\boldsymbol{\Phi}(\mathbf{x}_N),\boldsymbol{\Phi}(\mathbf{x}_{N+1})]$. Here, $\widetilde{\mathbf{X}}=[\mathbf{x}_2\,\mathbf{x}_3\cdots\mathbf{x}_N]$, $\widehat{\mathbf{X}}=[\mathbf{x}_2\cdots\mathbf{x}_N\,\mathbf{x}_{N+1}]$, and $\boldsymbol{\Phi}(\mathbf{x}_{N+1})$ is the newly recorded sample transformed into the feature space.

As shown in Eq. (1), a KPCA model is constructed from the covariance matrix of the process data transformed into the feature space. Its adaptation requires the adaptation of the mean vector, $\mathbf{m}_\Phi$, and the covariance matrix, $\mathbf{C}_\Phi$, by the following two-step procedure [42]. It should be noted that the development of the adaptation algorithms for the mean, covariance matrix and the subsequent KPCA model relies on the first shift of the moving window, that is when the first "new" sample, $\mathbf{x}_{N+1}$, becomes available. This is for convenience only and does not represent a restriction of generality.

Step 1. *Downdating* ($\boldsymbol{\Phi}(\mathbf{X})\rightarrow\boldsymbol{\Phi}(\widetilde{\mathbf{X}})$): The mean vector in the feature space of the intermediate window, $\widetilde{\mathbf{m}}_\Phi$, can be expressed by that of the "old" window, $\mathbf{m}_\Phi$, and the removal of the impact of the oldest sample, $\boldsymbol{\Phi}(\mathbf{x}_1)$:

$$\widetilde{\mathbf{m}}_\Phi = \frac{N}{N-1\mathbf{m}_\Phi - \frac{1}{N-1}\boldsymbol{\Phi}(\mathbf{x}_1)} \quad (8)$$

Incorporating Eq. (8) into the definition of the covariance matrix gives rise to:

$$\widetilde{\mathbf{C}}_\Phi = \frac{N-1}{N-2}\left[\mathbf{C}_\Phi - \frac{N}{(N-1)^2}(\boldsymbol{\Phi}(\mathbf{x}_1)-\mathbf{m}_\Phi)(\boldsymbol{\Phi}(\mathbf{x}_1)-\mathbf{m}_\Phi)^T\right]. \quad (9)$$

Step 2. *Updating* $\boldsymbol{\Phi}(\widetilde{\mathbf{X}})\rightarrow\boldsymbol{\Phi}(\widehat{\mathbf{X}})$: The mean vector in the feature space of the new window, $\widehat{\mathbf{m}}_\Phi$, can be computed from the intermediate window matrix, $\widetilde{\mathbf{m}}_\Phi$, and the new observations, $\boldsymbol{\Phi}(\mathbf{x}_{N+1})$:

$$\widehat{\mathrm{m}}_\Phi = \frac{N-1}{N}\widetilde{m}_\Phi + \frac{1}{N}\boldsymbol{\Phi}(\mathbf{x}_{N+1}) \quad (10)$$

Using the above equation, the covariance matrix of the transformed data in the new window becomes:

$$\widehat{\mathbf{C}}_\Phi = \frac{1}{N-1}\overline{\boldsymbol{\Phi}}(\widehat{\mathbf{X}})\overline{\boldsymbol{\Phi}}(\widehat{\mathbf{X}})^T = \frac{N-2}{N-1}\widetilde{\mathbf{C}}_{\boldsymbol{\Phi}} + \frac{1}{N}(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\widetilde{\mathbf{m}}_\Phi)(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\widetilde{\mathbf{m}}_\Phi)^T,$$

where $\overline{\boldsymbol{\Phi}}\left(\widehat{\mathbf{X}}\right) = \boldsymbol{\Phi}\left(\widehat{\mathbf{X}}\right) - \frac{1}{N}\boldsymbol{\Phi}\left(\widehat{\mathbf{X}}\right)\mathbf{E}_N$, $\mathbf{E}_N = 1_N \times 1_N^T$, is the mean centered feature matrix. The derivation of $\widehat{\mathbf{C}}_\Phi$ is given in Appendix A.

Step 3. *Moving window* $\boldsymbol{\Phi}(\mathbf{X}) \rightarrow \boldsymbol{\Phi}(\widehat{\mathbf{X}})$: Combining Steps 1 and 2, the mean vector in the feature space of the new window matrix $\boldsymbol{\Phi}(\widehat{\mathbf{X}})$ can be computed using the old window matrix, $\mathbf{m}_\Phi$, removing the contribution of the oldest sample, $\boldsymbol{\Phi}(\mathbf{x}_1)$, and adding the impact of the newest observations, $\boldsymbol{\Phi}(\mathbf{x}_{N+1})$:

$$\widehat{\mathbf{m}}_\Phi = \mathbf{m}_\Phi + \frac{1}{N}\left[\boldsymbol{\Phi}(\mathbf{x}_{N+1}) - \boldsymbol{\Phi}(\mathbf{x}_1)\right] \tag{11}$$

The combination of Steps 1 and 2 for determining an adapted covariance matrix for the new window yields:

$$\begin{aligned}\widehat{\mathbf{C}}_\Phi = \mathbf{C}_\Phi &- \frac{N}{(N-1)^2}[\boldsymbol{\Phi}(\mathbf{x}_1) - \mathbf{m}_\Phi][\boldsymbol{\Phi}(\mathbf{x}_1) - \mathbf{m}_\Phi]^T \\ &+ \frac{1}{N}\left[\boldsymbol{\Phi}(\mathbf{x}_{N+1}) - \frac{N}{N-1}\mathbf{m}_\Phi + \frac{1}{N-1}\boldsymbol{\Phi}(\mathbf{x}_1)\right] \\ &\times \left[\boldsymbol{\Phi}(\mathbf{x}_{N+1}) - \frac{N}{N-1}\mathbf{m}_\Phi - \frac{1}{N-1}\boldsymbol{\Phi}(\mathbf{x}_1)\right]^T\end{aligned} \tag{12}$$

4. Adapting the eigendecomposition of $\widehat{\mathbf{C}}_\Phi$

The previous section showed how to adapt $\mathbf{C}_\Phi$ to become $\widehat{\mathbf{C}}_\Phi$. The second contribution of this article is to incorporate a numerically efficient procedure for computing the eigendecomposition of $\widehat{\mathbf{C}}_\Phi$ which, in turn, describes the adapted KPCA model. Based on the adapted model, Section 5 outlines how to complete the adaptation of the KPCA-based monitoring model.

Adapting the KPCA model, in fact, requires a recalculation of the eigenvalues and eigenvectors of $\widehat{\mathbf{C}}_\Phi$, which span the KPCA model plane and the complementary residual subspace. Existing methods include a Rank-1 modification [3,11], the Lanczos method [34,35], an inverse iteration [12] and an adaptive computation of the eigenspace [14,15].

Reference [29] showed that the complexity for recomputing a linear PCA model is of $O(N^3)$ for the Rank-1 modification and of $O(N^2)$ for the Lanczos approach. The Rank-1 modification is not suitable here, since $\widehat{\mathbf{G}}$ cannot be decomposed into $\mathbf{G} + \boldsymbol{\eta}\boldsymbol{\eta}^T$ where $\boldsymbol{\eta}$ represents an update vector. Furthermore, the Lanczos approach requires a recalculation of the Gram matrix $\mathbf{G}$ (Eq. (4)), which, in turn, is of $O(N^3)$ and would consequently render this approach to be of $O(N^3)$ too.

In contrast, the method by Hall et al. [14,15] does not require a complete adaptation of $\mathbf{G}$. More precisely, this technique adapts the first $p \ll N$ non-zero eigenvalues of $\mathbf{G}$ instead of the complete set of eigenvalues. Section 5 highlights how to select p and how to determine the number of retained PCs, r, adaptively. The analysis in SubSection 4.3 shows that recomputing $p > r$ eigenvalues is computationally efficient and of $O(N^2)$. As shown in Subsections 4.1 and 4.2, incorporating the method in [14,15] produces adaptations of the eigenvalues and the matrix $\mathbf{A}$. It should be noted, however, that these are approximations and Sections 6 and 7 study their accuracy.

Next, the steps for adapting the eigendecomposition of a symmetric positive definite matrix using the technique in References [14,15] are extended into the feature space and discussed in Subsections 4.1 (downdating) and 4.2 (updating). Subsection 4.3 finally contrasts the various adaptation techniques and demonstrates that the extended version of the technique by [14,15] is of $O(N^2)$.

4.1. Downdating the current eigendecomposition

The "downdated" eigenvectors represent a rotation of the old eigenvectors in the feature space:

$$\widetilde{\mathbf{U}} = \mathbf{U}\mathbf{R}_1 \tag{13}$$

where $\mathbf{U} = \boldsymbol{\Phi}(\mathbf{X})\ \mathbf{A} \in F$, $\mathbf{A} \in \mathbb{R}^{N \times p}$ and $\mathbf{R}_1 \in \mathbb{R}^{p \times p}$ is an orthogonal rotation matrix. The next step is to define the diagonalization of $\mathbf{C}_\Phi$ and the downdated kernel matrix $\widetilde{\mathbf{C}}_\Phi$:

$$\mathbf{U}^T\mathbf{C}_\Phi\mathbf{U} = \boldsymbol{\Lambda} \qquad \widetilde{\mathbf{U}}^T\widetilde{\mathbf{C}}_\Phi\widetilde{\mathbf{U}} = \widetilde{\boldsymbol{\Lambda}} \tag{14}$$

It should be noted that $\boldsymbol{\Lambda}$ and $\widetilde{\boldsymbol{\Lambda}}$ only contain $p \ll N$ eigenvalues. Section 5 shows how to obtain p such that the contribution of the $N-p$ discarded eigenvalues and eigenvectors to both covariance matrices is negligible, for example less than 10^{-5} to their Frobenius norm. Combining Eqs. (9), (13) and (14) gives rise to:

$$\frac{N-1}{N-2}\boldsymbol{\Lambda} - \frac{N}{(N-1)(N-2)}\mathbf{g}_1\mathbf{g}_1^T = \mathbf{R}_1\widetilde{\boldsymbol{\Lambda}}\mathbf{R}_1^T \tag{15}$$

where $\mathbf{g}_1 = \mathbf{U}^T[\boldsymbol{\Phi}(\mathbf{x}_1) - \mathbf{m}_\Phi] = \mathbf{A}^T\left[\mathbf{k}(\mathbf{X}, \mathbf{x}_1) - \frac{1}{N}\mathbf{K}1_N\right] \in \mathbb{R}^p$.

Abbreviating the product $\mathbf{R}_1\widetilde{\boldsymbol{\Lambda}}\mathbf{R}_1^T$ by $\mathbf{D}_1 = \frac{N-1}{N-2}\boldsymbol{\Lambda} - \frac{N}{(N-1)(N-2)}\mathbf{g}_1\mathbf{g}_1^T$, the downdated eigenvalues of $\widetilde{\boldsymbol{\Lambda}}$ are equivalent to those of $\mathbf{D}_1$. Now, comparing

$$\widetilde{\mathbf{U}} = \boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\widetilde{\mathbf{A}} \tag{16}$$

with

$$\widetilde{\mathbf{U}} = \mathbf{U}\mathbf{R}_1 = \boldsymbol{\Phi}(\mathbf{X})\mathbf{A}\mathbf{R}_1 = \left[\boldsymbol{\Phi}(\mathbf{x}_1)\ \ \boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\right]\begin{bmatrix}\mathbf{A}_1 \\ \mathbf{A}_2\end{bmatrix}\mathbf{R}_1 \tag{17}$$

yields that the updated matrix $\widetilde{\mathbf{A}}$ can be approximated by:

$$\widetilde{\mathbf{A}} \approx \mathbf{A}_2\mathbf{R}_1 \tag{18}$$

This approximation is required since $\boldsymbol{\Phi}(\mathbf{x}_1)$ is unknown. The analysis in Sections 5 and 6, however, highlights that this approximation has a negligible effect upon the accuracy of the adapted eigenvalues and the matrix $\mathbf{A}$.

4.2. Subsequent updating of the eigendecomposition

The updating commences by projecting a new observation $\boldsymbol{\Phi}(\mathbf{x}_{N+1})$ onto the downdated eigenspace and the complementary residual subspace to produce $\mathbf{g}_2 \in \mathbb{R}^p$ and $\mathbf{h} \in F$, respectively:

$$\mathbf{g}_2 = \widetilde{\mathbf{U}}^T(\boldsymbol{\Phi}(\mathbf{x}_{N+1}) - \widetilde{\mathbf{m}}_\Phi) \qquad \mathbf{h} = \left[\mathbf{I} - \widetilde{\mathbf{U}}\,\widetilde{\mathbf{U}}^T\right](\boldsymbol{\Phi}(\mathbf{x}_{N+1}) - \widetilde{\mathbf{m}}_\Phi) \tag{19}$$

Reference [13] showed that the new eigenspace can be incrementally computed. The eigendecomposition, after including the new sample $\boldsymbol{\Phi}(\mathbf{x}_{N+1})$ to the feature matrix $\boldsymbol{\Phi}(\widetilde{\mathbf{X}})$, is given by:

$$\widehat{\mathbf{C}}_\Phi\,\widehat{\mathbf{U}} = \widehat{\mathbf{U}}\,\widehat{\boldsymbol{\Lambda}}, \tag{20}$$

The adapted eigenvectors are rotated by $\mathbf{R}_2 \in \mathbb{R}^{(p+1)\times(p+1)}$ of the current eigenvectors plus the residue vector, $\mathbf{h}$ scaled to unity:

$$\widehat{\mathbf{U}} = \left[\widetilde{\mathbf{U}}\ \hat{\mathbf{h}}\right]\mathbf{R}_2, \tag{21}$$

where

$$\hat{\mathbf{h}} = \begin{cases}\dfrac{\mathbf{h}}{|\,\mathbf{h}\,|_2} & \text{if } |\,\mathbf{h}\,|_2 \neq 0 \\ 0 & \text{otherwise}\end{cases} \tag{22}$$

and $||\cdot||_2$ is the squared norm of a vector. Next, substituting Eqs. (19) and (16) into Eq. (22) yields:

$$\hat{\mathbf{h}} = \frac{\boldsymbol{\Phi}(\mathbf{x}_{N+1}) - \boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\mathbf{b}}{\sqrt{K(\mathbf{x}_{N+1}, \mathbf{x}_{N+1}) - \mathbf{k}\left(\mathbf{x}_{N+1}, \widetilde{\mathbf{X}}\right)\mathbf{b} - \mathbf{b}^T\mathbf{k}\left(\widetilde{\mathbf{X}}, \mathbf{x}_{N+1}\right) + \mathbf{b}^T\widetilde{\mathbf{K}}\mathbf{b}}}, \tag{23}$$

Table 1
Comparison of computation cost (flops) between batch method (with Lanczos approach and standard SVD) and MWKPCA.

Method	Flops	Calculation complexity
Lanczos approach	$8N^3+4pN^2+3N^2-1.5Np^2+6.5Np-2p^2$	$O(N^3)$
Standard SVD	$30N^3+4N^2$	$O(N^3)$
MWKPCA	$(2p+8)N^2+18p^3+33p^2-4N+33p+9$	$O(N^2)$

where $\widetilde{\mathbf{K}}=\boldsymbol{\Phi}^T\left(\widetilde{\mathbf{X}}\right)\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\in\mathbb{R}^{(N-1)\times(N-1)}$. The derivation of $\hat{\mathbf{h}}$ is given in Appendix B, and $\mathbf{b}\in\mathbb{R}^{N-1}$ is defined as:

$$\begin{aligned}\mathbf{b}&=\frac{1}{N-1}1_{N-1}+\widetilde{\mathbf{A}}\,\widetilde{\mathbf{A}}^T\boldsymbol{\Phi}^T\left(\widetilde{\mathbf{X}}\right)\boldsymbol{\Phi}(\mathbf{x}_{N+1})+\frac{1}{N-1}\widetilde{\mathbf{A}}\,\widetilde{\mathbf{A}}^T\boldsymbol{\Phi}^T\left(\widetilde{\mathbf{X}}\right)\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)1_{N-1}\\&=\frac{1}{N-1}1_{N-1}+\widetilde{\mathbf{A}}\,\widetilde{\mathbf{A}}^T\left(\mathbf{k}\left(\widetilde{\mathbf{X}},\mathbf{x}_{N+1}\right)-\frac{1}{N-1}\widetilde{\mathbf{K}}1_{N-1}\right).\end{aligned}\tag{24}$$

In a similar fashion to the downdating procedure, combining Eqs. (12), (20) and (21) leads to the definition of $\mathbf{D}_2$:

$$\mathbf{D}_2=\begin{bmatrix}\frac{N-2}{N-1}\widetilde{\boldsymbol{\Lambda}}+\frac{1}{N}\mathbf{g}_2\mathbf{g}_2^T & \frac{1}{N}\gamma\mathbf{g}_2\\ \frac{1}{N}\gamma\mathbf{g}_2^T & \frac{1}{N}\gamma^2\end{bmatrix}=\mathbf{R}_2\hat{\boldsymbol{\Lambda}}\mathbf{R}_2^T\tag{25}$$

where

$$\mathbf{g}_2=\widetilde{\mathbf{U}}^T\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\widetilde{\mathbf{m}}_\Phi\right)=\widetilde{\mathbf{A}}^T\left(\mathbf{k}\left(\widetilde{\mathbf{X}},\mathbf{x}_{N+1}\right)-\frac{1}{N-1}\widetilde{\mathbf{K}}1_{N-1}\right)\tag{26}$$

and

$$\begin{aligned}\gamma=\hat{\mathbf{h}}^T\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\widetilde{\mathbf{m}}_\Phi\right)&=\frac{\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\widetilde{\mathbf{m}}_\Phi\right)^T\left[\mathbf{I}-\widetilde{\mathbf{U}}\widetilde{\mathbf{U}}^T\right]\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\widetilde{\mathbf{m}}_\Phi\right)}{\left|\left[\mathbf{I}-\widetilde{\mathbf{U}}\widetilde{\mathbf{U}}^T\right]\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\widetilde{\mathbf{m}}_\Phi\right)\right|_2}\\&=\left|\left[\mathbf{I}-\widetilde{\mathbf{U}}\widetilde{\mathbf{U}}^T\right]\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\widetilde{\mathbf{m}}_\Phi\right)\right|_2=|\,\mathbf{h}\,|_2\\&=\sqrt{K(\mathbf{x}_{N+1},\mathbf{x}_{N+1})-2\mathbf{k}\left(\mathbf{x}_{N+1},\widetilde{\mathbf{X}}\right)\mathbf{b}+\mathbf{b}^T\widetilde{\mathbf{K}}\mathbf{b}}\end{aligned}\tag{27}$$

Following an eigendecomposition of $\mathbf{D}_2$, the eigenvectors, stored in $\mathbf{R}_2$, and the eigenvalues, stored in $\hat{\boldsymbol{\Lambda}}$, can be obtained. Next, substituting Eqs. (16) and (23) into Eq. (21) gives rise to:

$$\hat{\mathbf{U}}=\left[\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\ \ \boldsymbol{\Phi}(\mathbf{x}_{N+1})\right]\begin{bmatrix}\widetilde{\mathbf{A}} & \frac{-\mathbf{b}}{||\mathbf{h}||_2}\\ \mathbf{0}^T & \frac{1}{||\mathbf{h}||_2}\end{bmatrix}\mathbf{R}_2=\left[\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\ \ \boldsymbol{\Phi}(\mathbf{x}_{N+1})\right]\hat{\mathbf{A}}\tag{28}$$

where $\hat{\mathbf{A}}=\begin{bmatrix}\widetilde{\mathbf{A}} & \frac{-\mathbf{b}}{||\mathbf{h}||_2}\\ \mathbf{0}^T & \frac{1}{||\mathbf{h}||_2}\end{bmatrix}\mathbf{R}_2$, and $\mathbf{0}^T\in\mathbb{R}^{1\times p}$. Combining the above equation with Eq. (18) finally yields:

$$\hat{\mathbf{A}}=\begin{bmatrix}\mathbf{A}_2\mathbf{R}_1 & \frac{-\mathbf{b}}{||\mathbf{h}||_2}\\ \mathbf{0}^T & \frac{1}{||\mathbf{h}||_2}\end{bmatrix}\mathbf{R}_2.\tag{29}$$

Reference [13] showed that compared to a singular value decomposition, where all right singular vectors must be retained in order to update the model, the eigendecomposition of $\mathbf{D}_2$ is computationally more efficient and requires less memory storage which is particularly important for large window sizes. This is further discussed in the next subsection. It should also be noted that the updating and downdating step cannot be combined, unlike the mean vector, $\widehat{\mathbf{m}}_\Phi$ and covariance matrix $\widehat{\mathbf{C}}_\Phi$.

4.3. Computational comparison for adapting the eigendecomposition

This section compares the proposed MWKPCA algorithm with that of batch KPCA, using a standard recalculation and the Lanczos method. This comparison is based on the number of floating point operations (flops). The proposed MWKPCA method can be separated into the following operations:

(1) The calculation of $\mathbf{D}_1$ requires $4p^2$ flops;
(2) The eigendecomposition of $\mathbf{D}_1$ consumes $9p^3$ flops;
(3) The construction of $\mathbf{D}_2$ involves $2p^2+6p$ operations;
(4) The eigendecomposition of $\mathbf{D}_2$ consumes $9(p+1)^3$ flops; and
(5) The calculation $\mathbf{g}_1$, $\mathbf{g}_2$, $\mathbf{b}$ and γ comprises of $2N^2+(2p+2)N$, $2(N-1)(N+p-1)$, $2(N-1)^2(p+1)$ and $2(N-1)(N+2)$ flops, respectively.

The overall complexity is therefore equal to $(2p+8)N^2+18p^3+33p^2-4N+33p+9$. This compares favorably to batch KPCA, which initially requires $8N^3+4N^2$ operations to compute the Gram matrix $\mathbf{G}$. According to Reference [29], applying the fast Lanczos approach to the Gram matrix $\mathbf{G}$, involves $4pN^2-1.5Np^2+6.5Np-N^2-2p^2$ flops and the overall complexity is therefore $8N^3+4pN^2+3N^2-1.5Np^2+6.5Np-2p^2$. Moreover, using the standard singular value decomposition procedure on the Gram matrix $\mathbf{G}$ to compute the eigendecomposition consumes $22N^3$ flops, which results in a total of $30N^3+4N^2$ flops, as listed in Table 1 and illustrated in Fig. 1.

It is also important to note that for large window sizes N, the number of non-zero eigenvalues p of $\mathbf{G}$ can be considerably less than N, that is $p\ll N$, which is discussed in Sections 6 and 7. This advantage can be exploited to reduce the computational effort for adapting the KPCA model (including, that only p eigenvalues and eigenvectors are adapted). Table 1 and Fig. 1 show that small p over N ratios, e.g. $p/N\leq 1/10$, can offer a substantial reduction in the number of flops compared to the Lanczos approach. This analysis therefore outlines that the proposed adaptation method is of $O(N^2)$, whilst alternative methods will, by default, be of $O(N^3)$ since they rely on the adaptation of the Gram matrix. It is important to note that this computational benefit is at the expense of an approximated solution for the eigenvalues and the $\mathbf{A}$ matrix. The analysis in Sections 6 and 7, however, shows that the approximation error is negligible for the first few hundred adaptations.

5. Adaptation of the KPCA-based monitoring model

This section utilizes the adapted KPCA model and develops an adaptive monitoring scheme for fault detection. The next subsection shows how to adapt the number of retained PCs and how to determine

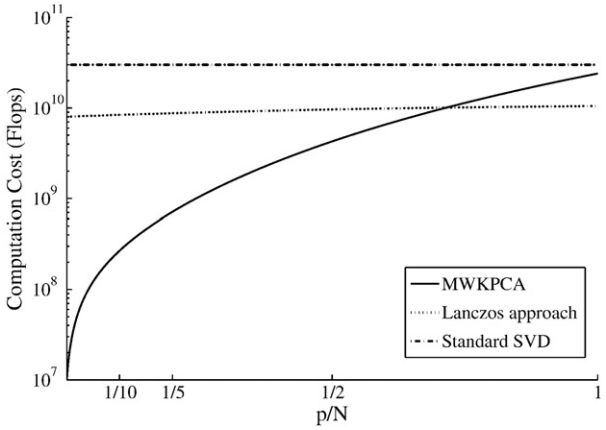

Fig. 1. Comparison of computation cost(flops) among a number of ratios of p/N (N is fixed at 1000).

the number of non-zero, or negligible, eigenvalues. This is followed by a summary of how to compute the adapted PCs as well as the adapted Hotelling's T^2 and Q statistics and their associated confidence limits in Subsection 5.2. Finally, Subsection 5.3 summarizes the proposed adaptive KPCA-based monitoring scheme.

5.1. Adaptation of the number of retained PCs

Since the number of significant principal components may vary over time [29], it is necessary to determine this number adaptively. There are numerous methods for determining the number of PCs, including cumulative percent variance [31], the scree test [4], average eigenvalues, and the variance of reconstruction error [36]. Good overviews on methods for selecting this number may be found in References [18,20,40]. As the KPCA model only requires an update of the eigenvalues of **G** and the **A** matrix, the eigenvectors of $\mathbf{C}_\Phi$ are not available. As outlined in Section 2, the nonlinear mapping $\boldsymbol{\Phi}(\cdot)$ is, in fact, unknown. This implies that most methods for determining the number of retained PCs cannot be employed here.

This article proposes the use of a method introduced in Anderson [2], Chapter 11, which is a hypothesis test for testing whether the sum of the last $p-r$ eigenvalues is smaller relative to the sum of all p eigenvalues. Consider the null hypothesis:

$$H : f_r = \frac{\sum_{i=r+1}^{p} \lambda_i}{\sum_{i=1}^{p} \lambda_i} \geq \delta, \tag{30}$$

which is rejected if $H_{f_r} = \sqrt{N}(f_r - \delta) < H_{g_r}$, the upper significance limit of the standard normal distribution, determined with a confidence of α, times $\sqrt{2\left(\frac{\delta}{\sum_{i=1}^{p}\lambda_i}\right)^2 \sum_{i=1}^{r}\lambda_i + 2\left(\frac{1-\delta}{\sum_{i=1}^{p}\lambda_i}\right)^2 \sum_{i=r+1}^{p}\lambda_i}$.

The number of retained PCs, r, is chosen by a predetermined limit, for example $\delta = 0.05$ which implies that at least 95% of the information for reconstructing the covariance matrix in the feature space is contained in the retained PCs. By increasing the number of PCs iteratively, r is selected when H_{fr} is smaller than H_{gr} for the first time.

This method is also used in this article to determine p using the initial window. The sum of the $p+1, \ldots, N$ eigenvalues can be negligible as the threshold is considerably smaller than 0.05, for example $\varepsilon = 10^{-4}$ or $\varepsilon = 10^{-5}$:

$$H : f_p = \frac{\sum_{i=p+1}^{N} \lambda_i}{\sum_{i=1}^{N} \lambda_i} \geq \varepsilon, \tag{31}$$

This test is to be conducted in the same way as above and since $\varepsilon \ll \delta$, it follows that $p > r$.

5.2. Computing the scores

Utilizing the adapted matrix $\hat{\mathbf{A}}_r$, which stores the first $r<p$ column vectors of $\hat{\mathbf{A}}$, the adapted KPCA model will be applied to a data point recorded l samples later. Reference [42] demonstrated that this can prevent incipiently developing faults from being adapted. The score vector of the centered sample $\mathbf{x}_{k+l}$, $\mathbf{t}_{k+l}$, can be computed as follows:

$$\mathbf{t}_{k+l} = \hat{\mathbf{A}}_r^T \left(\mathbf{k}\left(\hat{\mathbf{X}}, \mathbf{x}_{k+l}\right) - \frac{1}{N}\hat{\mathbf{K}}\mathbf{1}_N \right) \tag{32}$$

where $\hat{\mathbf{K}} = \boldsymbol{\Phi}^T(\hat{\mathbf{X}})\boldsymbol{\Phi}(\hat{\mathbf{X}})$.

5.3. Adaptation of the T^2 and Q statistics

For time-varying system, the confidence limits for the T^2 and Q statistics vary over time [29,43]. Using the adapted eigenvalue matrix, $\hat{\boldsymbol{\Lambda}}_r$ which stores the first r eigenvalues of $\hat{\boldsymbol{\Lambda}}$, the T^2 statistic for $\mathbf{x}_{k+l}$ is:

$$T^2_{k+l} = \mathbf{t}^T_{k+l} \hat{\boldsymbol{\Lambda}}_r^{-1} \mathbf{t}_{k+l} \tag{33}$$

This statistic follows an F-distribution [39] and the confidence limit, T^2_β, is given by:

$$T^2_\beta = \frac{r\left(N^2-1\right)}{N(N-r)} F_{r,N-r,\beta}, \tag{34}$$

where β is the confidence level. This upper confidence limit needs to be updated since r can vary over time. In order to apply this confidence limit for T^2_{k+l}, the number of retained PCs must be determined for the data window storing the samples between $(k-N+1)th$ and $(k+1)th$ instance.

The Q statistic for $\mathbf{x}_{k+l}$ is given by:

$$Q_{k+l} = K(\mathbf{x}_{k+l}, \mathbf{x}_{k+l}) - \frac{2}{N}\mathbf{1}_N^T \mathbf{k}\left(\hat{\mathbf{X}}, \mathbf{x}_{k+l}\right) - \frac{1}{N^2}\mathbf{1}_N^T \hat{\mathbf{K}}\mathbf{1}_N - \mathbf{t}^T_{k+l}\mathbf{t}_{k+l}. \tag{35}$$

Based on the work in [33], this statistic can be approximated by a central χ^2-distribution. The confidence limit for Q, Q_β, can be approximated by:

$$Q_\beta = g\chi^2(h) \qquad g = \frac{\rho^2}{2\mu} \qquad h = \frac{2\mu^2}{\rho^2} \tag{36}$$

where μ and ρ^2 are the mean and variance of the Q statistic, respectively. For the values of the Q statistic inside the sliding window between the $(k-N+1)th$ and $(k+1)th$ sample, the mean μ and variance ρ^2 can be obtained, which enables the calculation of the adaptive confidence limit using Eq. (36).

5.4. Process monitoring using l-step-ahead prediction

The process monitoring procedure using l-step-ahead prediction is as follows:

(1) Build the initial KPCA model on the first N data points.
(2) For a new sample $\mathbf{x}_k$, use the fast MWKPCA algorithm to update the eigenvalues and matrix **A** for computing the updated KPCA model.
    This, however, requires a discrimination between the following two cases:
    - Case 1: If $k<N+l$, the monitoring statistics for $\mathbf{x}_k$ are calculated using the initial model. The confidence limits remain the same as before;
    - Case 2: If $k>N+l$, the monitoring statistics for $\mathbf{x}_k$ and the KPCA model are calculated l steps earlier. The confidence limits for this sample are recalculated based on the statistics for the data used to update the KPCA model.
(3) If the statistics for $\mathbf{x}_{k+l}$ exceed the confident limits, that is $T^2_{k+l} > T^2_\beta$ and/or $Q_{k+l} > Q_\beta$, consideration should be given as to whether $\mathbf{x}_{k+l}$ should be included for model adaptation;
(4) After a number of adaptations, for example 200, check whether Eq. (32) still holds, that is if $\frac{\sum_{i=p+1}^{N}\lambda_i}{\sum_{i=1}^{N}\lambda_i} \geq \varepsilon$ go to (1) and recompute the entire KPCA model from the data inside current window and if $\frac{\sum_{i=p+1}^{N}\lambda_i}{\sum_{i=1}^{N}\lambda_i} < \varepsilon$ go to (2); it should be noted that the sum of eigenvalues can be computed as $\|\hat{\mathbf{G}}\|_2^2$.

5.5. Summary of the developed adaptation method

The application of the proposed adaptive algorithm for the KPCA-based monitoring model can be summarized by the following 13 steps:

(1) Compute a SVD of the initial Gram matrix **G**, determine the number of non-zero eigenvalues p, as discussed in Subsection 5.1, and calculate the p eigenvector-eigenvalue pairs ζ_k and $\mathbf{v}_k$;
(2) Establish the original eigenvalue matrix $\boldsymbol{\Lambda}$, which stores the p eigenvalues λ_k as diagonal elements, and compute **A**;

(3) Calculate the vector $\mathbf{g}_1$ and the matrix $\mathbf{D}_1$;
(4) Carry out the eigendecomposition of $\mathbf{D}_1$ to produce $\widetilde{\mathbf{\Lambda}}$ and $\mathbf{R}_1$;
(5) Compute $\widetilde{\mathbf{A}}$;
(6) Calculate the vector $\mathbf{g}_2$, the variable γ and the matrix $\mathbf{D}_2$;
(7) Carry out the eigendecomposition of $\mathbf{D}_2$ to produce $\widehat{\mathbf{\Lambda}}$ and $\mathbf{R}_2$;
(8) Compute $\widehat{\mathbf{A}}$;
(9) Recompute the Gram matrix after, for example several hundred adaptations, and work out the sum of the eigenvalues as $||\widehat{\mathbf{G}}||_2^2$. If $f_p \geq \varepsilon$ recompute the eigendecomposition of $\widehat{\mathbf{G}}$, redetermine p and obtain $\widehat{\mathbf{\Lambda}}$ and $\widehat{\mathbf{A}}$, otherwise utilize $\widehat{\mathbf{\Lambda}}$ and $\widehat{\mathbf{A}}$ computed in Steps 7 and 8, respectively.
(10) Determine the score vector for the sample $\mathbf{x}_{k+l}$ using the KPCA model adaptation obtained at sampling instance $k+1$, $\mathbf{t}_{k+l}$;
(11) Determine the Hotelling's T^2 and residual Q statistics;
(12) Check whether $T^2_{k+l} > T^2_\beta$ and/or $Q_{k+l} > Q_\beta$ exceeds the significance level $1-\beta$; if so, the hypothesis that the process is out-of-statistical-control is accepted, otherwise the process is in-statistical-control.
(13) Go to Step 3.

6. Simulation example

This section shows the utility of the proposed adaptive nonlinear monitoring scheme using a simulated process that includes a total of 3 process variables. The adaptive KPCA monitoring approach is compared here to conventional KPCA using a fixed monitoring model. Details of this process are given next, followed by a discussion of how the KPCA model was established in Subsection 6.2. Finally, Subsection 6.3 shows that the adaptive KPCA monitoring was capable of adapting slow time-varying process behavior whilst it was still able to detect anomalous process behavior.

6.1. Process description

The process under investigation comprised 3 process variables, x_1, x_2 and x_3 that depended on a parameter t. This simulation example is from the literature and has been extensively used for nonlinear monitoring applications, e.g. [6,9,28]. The following equation details the nonlinear variable interrelationships between x_1, x_2 and x_3:

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} t \\ t^2 - 3at \\ -t^3 + 3at^2 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} = \boldsymbol{\theta}(t) + \mathbf{e}, \tag{37}$$

where e_1, e_2 and e_3 were independent and identically-distributed Gaussian sequences of zero mean and variance 0.01, that is $\mathbf{e}^T = (e_1\, e_2\, e_3) \sim \mathcal{N}(0, 0.01\mathbf{I})$, which represented measurement uncertainty, t was a uniformly distributed stochastic sequence within the range of [0.01 2] and a was a time varying parameter that gradually increased from an initial value of 1 with a slope of 0.001 per sample. This consequently introduced time varying process behavior for the process variable x_2 and x_3 and hence $\boldsymbol{\theta}(t)$. From the above process, a data set of 1500 samples was generated.

6.2. Determination of a KPCA monitoring model

Section 2 has shown that establishing a KPCA model requires the selection of a kernel function including the kernel parameter, σ, the number of retained PCs, r, the window size, N, and the delay in applying the adapted KPCA monitoring model, l. It should be noted that the selection of σ has only been sporadically touched upon in the research literature. Whilst this is not a conceptually important point, it is of fundamental practical significance.

On the basis of discussions in Reference [41], which has outlined that the Gaussian kernel function $K(\mathbf{x}_i, \mathbf{x}_j) = exp\left(-\frac{|\mathbf{x}_i - \mathbf{x}_j|_2^2}{\sigma^2}\right)$ gives a good performance under general smooth conditions, this work relied on the use of Gaussian kernels for both application studies. Selecting the threshold $\varepsilon = 10^{-4}$ in Eq. (31), Fig. 2 shows p versus the kernel parameter σ. This figure outlines a decrease in p with an increase in σ, which, as discussed in Section 4, reduces the computational effort in adapting the KPCA model.

That the $\sigma(p)$ curve is decreasing is not surprising given the following analysis. Knowing that $K(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\Phi}^T(\mathbf{x}_i)\boldsymbol{\Phi}(\mathbf{x}_j) = ||\boldsymbol{\Phi}(\mathbf{x}_i)|| \, ||\boldsymbol{\Phi}(\mathbf{x}_j)|| \times \cos(\phi) = \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_j||_2^2}{\sigma^2}\right)$, if $\sigma \to \infty$ then $\exp(0) = 0$, $\phi = 0$ and $\boldsymbol{\Phi}(\mathbf{x}_i) = \boldsymbol{\Phi}(\mathbf{x}_j)$; the larger σ becomes, the more similar the elements in $\mathbf{G}$ are and hence the fewer retained PCs, r, are required to reconstruct $\mathbf{G}$. For the other extreme case, $\sigma \to 0$, it follows that $\exp(-\infty) = 0$, $\phi = \frac{\pi}{2}$ and $\boldsymbol{\Phi}(\mathbf{x}_i) \perp \boldsymbol{\Phi}(\mathbf{x}_j)$. In this case, the elements of $\mathbf{G}$ will, by default, differ significantly which, in turn, requires more retained PCs for reconstructing $\mathbf{G}$.

It is also important to note that the former case, large σ values, implies that each of the scalar products fall within the same sphere which renders incipient faults to be difficult to detect. In contrast, a small σ amplifies different directions between $\mathbf{x}_i$ and $\mathbf{x}_j$ in the feature space and therefore increases the sensitivity in detecting incipiently developing faults. The selection of σ is therefore a tradeoff between sensitivity and computational efficiency, since the smaller p the smaller the sizes of $\mathbf{U}$ and $\mathbf{\Lambda}$, hence the lower the computational costs according to Fig. 1. In contrast, with a larger σ, the monitoring scheme will be less sensitive. Based on the recommendations in Reference [7], σ was determined to be 2. For $\varepsilon = 10^{-4}$, $p = 25$, whilst an $\varepsilon = 10^{-5}$ yielded $p = 40$.

A window size of 1000 samples was chosen that represented naturally occurring changes in the process. Inspecting the adaptation performance of the proposed technique revealed that a window length of $N = 500$ was able to adapt the gradual change in the parameter a. A larger window size led to an increase in the number of Type 1 errors even for a relatively short delay between the adaptation and the application of the adapted KPCA monitoring scheme. On the other hand, a smaller window size might have rendered the adaptive monitoring approach insensitive in detecting incipiently developing fault conditions, which is discussed in [42].

Next, the initial number of retained PCs had to be determined. Applying Eq. (30) for $\delta = 0.001$ yielded an initial value for $r = 4$. An increase in this value to $\delta = 0.01$ or $\delta = 0.05$ suggested r to be 2 and 1, respectively. This, however, produced a sharp reduction in N, which is undesired as it reduces sensitivity in detection incipient faults. We therefore selected δ to be 0.001, which implied that $\mathbf{G}$ would have been reconstructed with an accuracy of 99.9% using 4 retained PCs. The final step was the determination of the delay in applying the currently updated KPCA model, l. Reference [42] suggested that a large l means the monitoring scheme is more sensitive to incipiently developing faults. Selecting l to be 400 showed that the number of

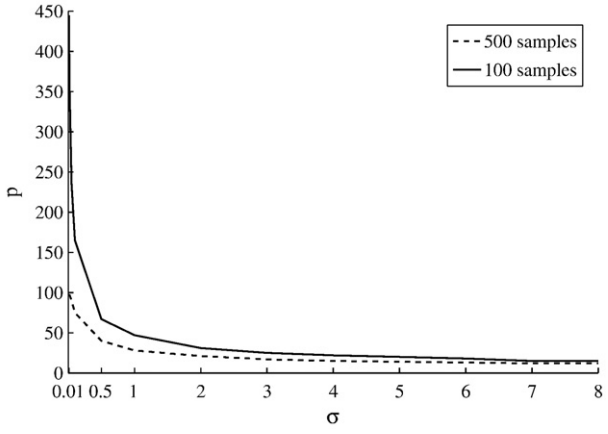

Fig. 2. Kernel parameter vs. number of non-zero eigenvalue, determined for a threshold 10^{-4} for window sizes 100 and 500.

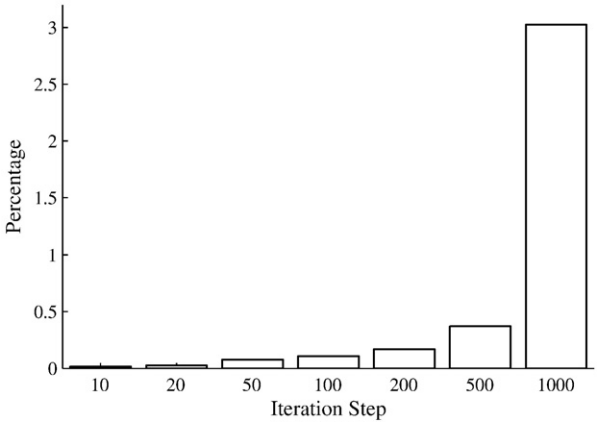

Fig. 3. Accuracy for approximating eigenvalues using proposed adaptation method.

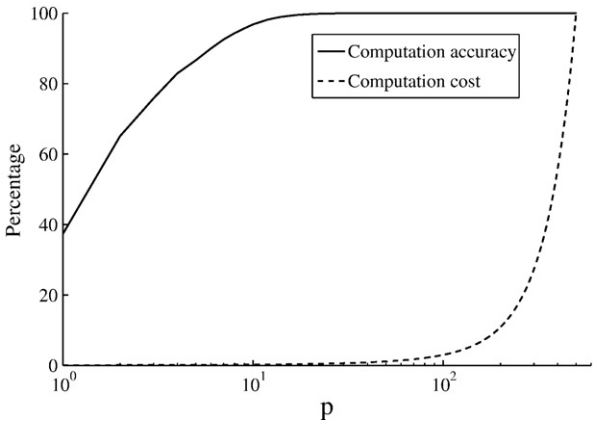

Fig. 5. Computation cost and accuracy vs. number of non-zero eigenvalues.

Type 1 errors did not exceed the significance level of 1%, which Fig. 7 confirms.

Section 4 has outlined that the downdating procedure to produce $\widetilde{\mathbf{A}}$ in Eq. (18) and the subsequent updating which relies on this matrix represents an approximation. It is therefore imperative to examine the influence of this approximation upon the accuracy of the adaptation. Appropriate measures for this comparison are the following relative differences between the adapted **A** matrices, $\Delta||\hat{\mathbf{A}}||_2^2$, and the adapted eigenvalues $\Delta||\hat{\mathbf{\Lambda}}||_2^2$:

$$\Delta\left|\left|\hat{\mathbf{A}}\right|\right|_2^2 = \left|\left|\hat{\mathbf{A}}_L - \frac{\hat{\mathbf{A}}_H\left|\left|_2^2\right.\right.}{\left|\left|\hat{\mathbf{A}}_L\right|\right|_2^2} \times 100\% \quad \Delta\left|\left|\hat{\mathbf{\Lambda}}\right|\right|_2^2 = \frac{\left|\left|\hat{\mathbf{\Lambda}}_L - \hat{\mathbf{\Lambda}}_H\left|\left|_2^2\right.\right.\right.}{\left|\left|\hat{\mathbf{\Lambda}}_L\right|\right|_2^2} \times 100\%, \tag{38}$$

where the subscripts *L* and *H* refer to the adaptation using the Lanczos method and the approximation based on the method proposed by Hall et al. [14,15]. As stated before, utilizing the Lanczos method yields an accurate adaptation that is of $O(N^3)$, whilst the proposed approximation is of $O(N^2)$. Figs. 3 and 4 show the results of applying both criteria in Eq. (38) for a total of 1000 consecutive adaptations. Whilst Fig. 3 confirms that significant departures arose only after 500 iterations, Fig. 4 outlines that a recalculation of the Gram matrix should be considered after 200 adaptation, as otherwise substantial errors arise between the true and approximated **A** matrix.

Fig. 5 gives a clear account of the benefit of using the proposed adaptation technique. In this figure, the relative computational accuracy is plotted (solid line), which shows the accuracy for reconstructing the Gram, $\Delta||\mathbf{G}||_2^2$, matrix vs. the number of included component matrices $\mathbf{v}_k\mathbf{v}_k^T\zeta_k$:

$$\mathbf{G} = \sum_{k=1}^{N} \mathbf{v}_k\mathbf{v}_k^T\zeta_k \quad ||\mathbf{G}||_2^2 = \sum_{k=1}^{N} \zeta_k \quad \Delta||\mathbf{G}||_2^2 = \frac{\sum_{k=1}^{p} \zeta_k}{\sum_{k=1}^{N} \zeta_k} \times 100\% \tag{39}$$

Given that $N=500$ and with $p=25$ an accuracy of 99.99% can be achieved. This can be increased to 99.999% with $p=40$. Therefore, it can be concluded that relatively few component matrices are required to recompute **G**. Fig. 5 also depicts the computational cost (dashed line) vs. *p*. According to Table 1, this cost is of $O(N^2)$ and includes *p* as a parameter. Applying the analysis in Section 4.3 for an accuracy of 99.99%, $p=25$, yields that the number of floating points consumed only amounts to 0.5% compared to $p=N$, whilst an accuracy of 99.999%, $p=40$, increases this number to 0.8796%. Moreover, a comparison of the proposed adaptation for $p=25$ with the Lanczos method (including only *p* component matrices) shows that it only consumes 1.5702% of flops. This number increases to 2.5% for $p=40$. Next, we contrast the performance of the proposed adaptive with the fixed KPCA-based monitoring scheme.

6.3. Process monitoring applications

Using the generated 1500 samples, this subsection first contrasts the application of adaptive nonlinear monitoring scheme with one involving a fixed KPCA model. Whilst Fig. 6 shows that, as expected, an increasing number of violations of the 99% confidence limit (solid line)

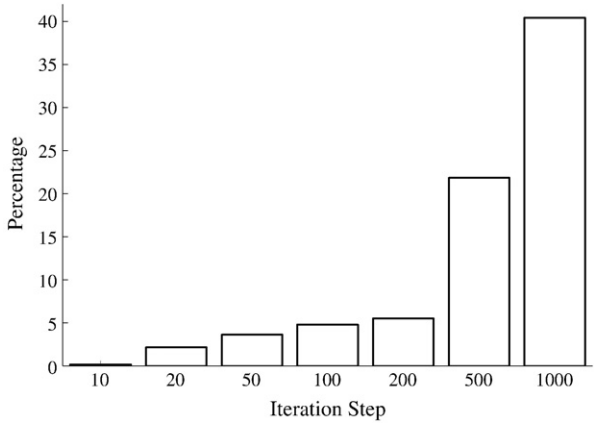

Fig. 4. Accuracy for approximating **A** matrix using proposed adaptation method.

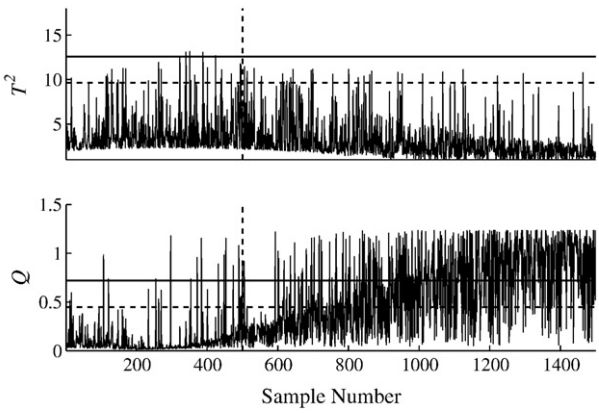

Fig. 6. T^2 and *Q* statistics calculated by fixed model for fault-free data set.

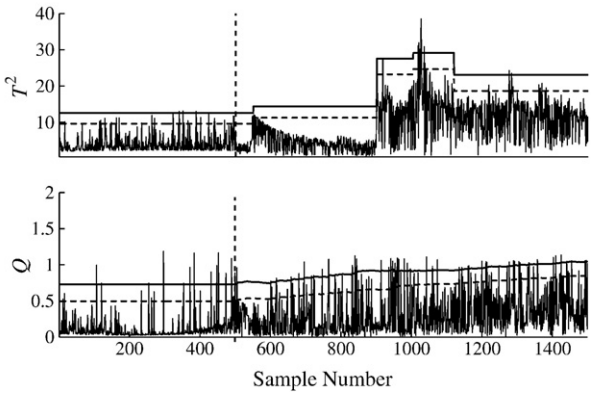

Fig. 7. Adaptively computed T^2 and Q statistics using the moving window model for fault-free data.

arose just after 600 samples into the data set, Fig. 7 confirms that the proposed adaptive technique did adapt the impact of the time-varying process behavior. It should be noted that the number of retained PCs varied from an initial value of $r=4$ to a maximum of $r=13$ between around the 1000th and 1150th sample. In addition, the 99% confidence limit increased slightly over time for the Q statistic.

Now, the first 1000 samples were used to demonstrate the effect of time-varying process behavior, whilst the last 500 samples served as a test case for a fault condition. This fault condition was simulated to be a ramp-type fault that was superimposed onto the process variable x_1 and had a slope of 0.05 between two samples. As before, the first 500 data points were used to identify the initial KPCA model and to determine the control limits for both statistics. Fig. 8 shows the monitoring result for this second data set. The Q statistic detected the injected fault after the 1031*th* sample after which the adaptation was stopped to prevent corruption of the on-line monitoring model. This simulation example has therefore illustrated that slowly developing time-varying behavior could be accommodated by the adaptive KPCA-model, whilst incipient but more significant changes caused by a fault condition could be detected using the delayed application of the adapted monitoring model.

7. Industrial application study

This application study involved a distillation process. The process is designed to purify butane from a fresh feed comprising a mixture of hydrocarbons, mainly butane and hexane and impurities of propane. The distillation tower includes 33 trays with which the separation is achieved. A purified butane stream, including any lighter gases such as propane impurities, leaves the distillation process as the top product and consequently, the hexane and heavier hydrocarbons leave the distillation process with the bottom stream. The process operates at a single operating point and the product quality refers to the hexane concentration in the top stream. This concentration should be kept below a predefined limit. In order to achieve an economic operation, it is desirable to maintain the butane concentration of the bottom draw also below a specified level.

The analyzed variable set consisted of 15 process variables that could be divided into 11 output variables, 2 input variables and 2 measured disturbances variables that also affected the output variables but could not be manipulated. This variable set was recorded at a sampling interval of 30 s and Table 2 shows that the output variables included temperature readings along the distillation unit, vessel levels, flow rates, and concentrations. The input variables were the reflux set point and the reboiler steam flow, whilst the measured disturbances included the feed flow and temperature of the input flow.

The application of the proposed adaptive KPCA-based monitoring scheme was contrasted here again with conventional KPCA to verify the need for an adaptive monitoring scheme. This requirement also followed from the time-varying nature of the fresh feed flow as well as the fresh feed temperature, which alter the variable interrelationships between the temperature, flow rate and concentration readings. In order to justify the requirement for a nonlinear monitoring model, the nonlinearity test proposed in References [23,25] was applied to a reference data set including a total of 2000 samples representing normal process operation. This test yielded that a linear PCA model could not be applied to accurately represent the variable interrelationships.

Testing the performance of the proposed adaptive KPCA-based monitoring scheme for detecting abnormal process behavior relied on the analysis of a second data set. This set described the effect of a significant drop in fresh feed from a 25 t/h to around 18 t/h, which upset the material and energy balance in the unit and eventually produced a substantial and undesired increase in the C5 in C4 concentration in the top draw. Fig. 9 shows the time-based plots of the 15 variables for this second data set.

Investigating the recorded signals for each variable in Fig. 9 revealed a sharp and prolonged reduction in fresh feed from around the 400 sample into the data set. The feed drop ($D2$) is accompanied by an initial slight drop in feed temperature ($D1$) and a subsequent increase in this temperature. This event caused an almost instant response in the reflux flow ($A1$), the reboiler steam flow ($A2$) and the bottom draw ($O11$) and a minor alteration in the reflux vessel level

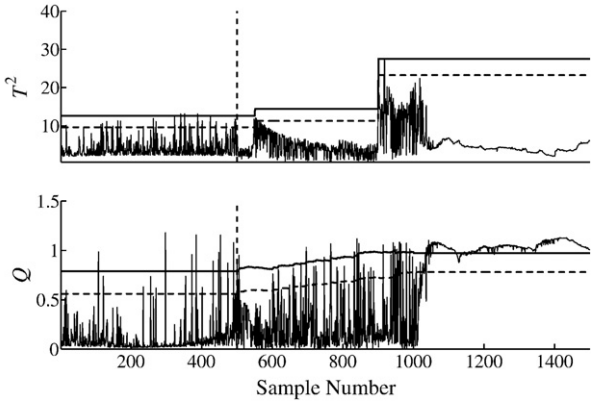

Fig. 8. Adaptively computed T^2 and Q statistics using the moving window model for ramp-type fault.

Table 2
List of recorded process variables for industrial distillation unit.

Variable tag	Variable type	Description	Unit
$O1$	output	Tray 14 temperature	[°C]
$O2$	output	Tray 2 temperature	[°C]
$O3$	output	Reflux vessel level	[%]
$O4$	output	Butane product flow	[t/h]
$O5$	output	Tray 31 temperature	[°C]
$O6$	output	$C3$ in $C4$	[%]
$O7$	output	$C4$ in $C5$	[%]
$O8$	output	$C5$ in $C4$	[%]
$O9$	output	Reboiler vessel level	[%]
$O10$	output	Reboiler temperature	[°C]
$O11$	output	Bottom draw	[t/h]
$A1$	input	Reflux flow	[t/h]
$A2$	input	Reboiler steam flow	[t/h]
$D1$	measure disturbance	Feed temperature	[°C]
$D2$	measured disturbance	Feed flow	[t/h]

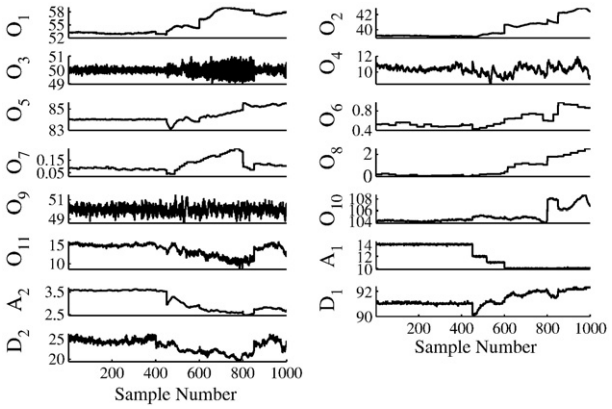

Fig. 9. Time-based plots of recorded variables for second data set (describing the effect of a drop in fresh feed).

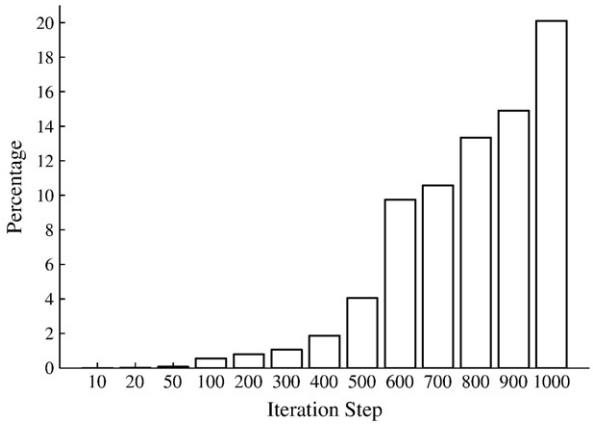

Fig. 11. Accuracy of eigenvalues with proposed adaptation method.

(*O*3). A response of the temperature readings arose with a delay of approximately 10 min and affected the tray 14 temperature (*O*1), the tray 2 temperature (*O*2), the tray 31 temperature (*O*5) and the reboiler temperature (*O*10). The effect on the temperature readings was a consequence of a reduced liquid level at the bottom of the unit that was heated by the reboiler and therefore increased in temperature. An additional effect is that the vapor rose within the distillation unit with a higher enthalpy and hence more *C*5 evaporated and remained in the top draw (variable *O*8). In contrast, the *C*3 in *C*4 and *C*4 in *C*5 concentrations (variables *O*6 and *O*7) only showed to be slightly affected.

7.1. Determination of a KPCA monitoring model

As before, the determination of a KPCA-based monitoring model required the selection of the kernel parameter, σ, the initial number of retained PCs, r, the window length, N, and the delay for applying the adapted monitoring model, l. Using Gaussian kernel functions, as advocated in Reference [41], the sum of nonzero eigenvalues, $N-p$, was determined by Eq. (31) for a threshold of $\varepsilon = 10^{-5}$. For $N = 100$ and $N = 1000$, Fig. 10 shows the resultant p vs. the kernel parameter σ. Given that smaller values of σ increase the sensitivity of the adaptive monitoring scheme but also increase the computational effort in adapting the eigenvalues and the **A** matrix, the parameter σ represents a trade-off between sensitivity and computational efficiency. Following the discussion in Reference [7], σ was determined to be 25, which produced a value for $p = 100$. With $\frac{p}{N} = 0.1$, the adaptation of the p eigenvalues and the **A** matrix consumed 1.1% flops when compared with the proposed method for $p = N$ and 3.2% when compared with the Lanczos method. This was a substantial reduction in computational expense for one adaptation.

Varying the window sizes between $N = 10$ and $N = 1500$ suggested that $N = 1000$ could adapt changes in the variable interrelationships within the reference data set. A smaller N might have rendered the adaptive scheme insensitive to incipient fault condition and a larger value for N produced a large number of Type 1 errors even for a short delay in applying the adapted KPCA-based monitoring model. The next step was to determine the initial number of retained PCs. Applying Eq. (30) yielded $r = 13$ for $\delta = 0.01$. Finally, the delay horizon for applying the adaptive monitoring scheme was selected to be $l = 100$. Again, a larger delay would have increased the number of Type 1 errors to exceed 1% and a smaller delay would have compromised sensitivity.

Given that the proposed algorithm for adapting the p eigenvalues and the **A** matrix represents an approximation, we next examined how accurate this approximation was compared to the use of the computationally more expensive Lanczos method. Using Eq. (38), Fig. 11 shows the accuracy with which the eigenvalues were determined and Fig. 12 provides the accuracy for determining the **A** matrix vs. the number of adaptations. Both figures suggest that a significant deterioration in accuracy emerged from 300 adaptations onwards.

Using Eq. (39) and the calculation of the number of floating point operations in Table 1, Fig. 13 finally shows the selection of $p = 100$ gave, as expected, a very high accuracy for reconstructing the Gram matrix,

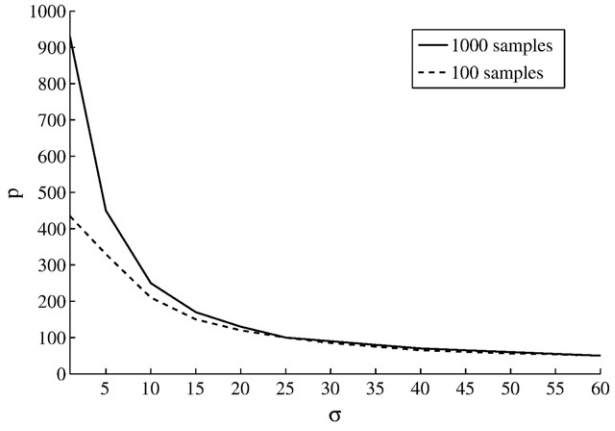

Fig. 10. Kernel parameter σ vs. number of non-zero eigenvalues.

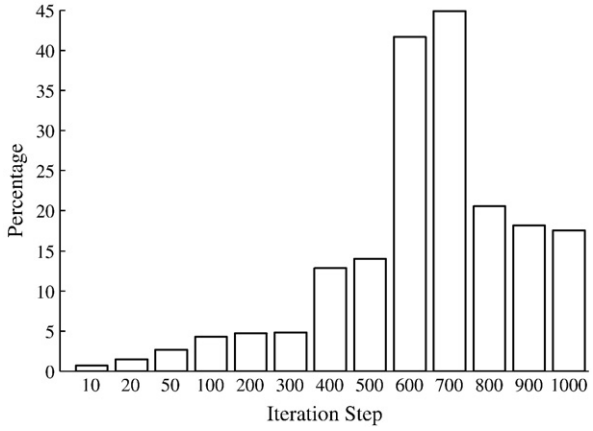

Fig. 12. Accuracy of matrix **A** with proposed adaptation method.

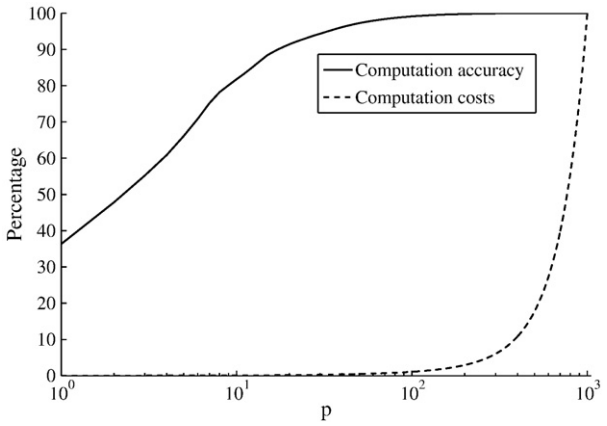

Fig. 13. Computation cost (flops)(solid line) and computation accuracy (dashed line) vs. the number of non-zero eigenvalues p.

G, at a low computational expense. Note that the accuracy as well as the computational cost is given in percentage.

7.2. Process monitoring results

To justify the use of an adaptive monitoring model, the adaptive KPCA-based monitoring scheme was contrasted with its fixed counterpart, the KPCA model. The number of retained PCs and the confidence limits for both univariate monitoring statistics remained constant. Fig. 14 presents the monitoring charts for KPCA. Whilst the T^2 statistic suggested in-statistical-control, the Q statistic produced a substantial number of violations after about 250 samples into the second half of the reference data. Since this data set was not used to identify the KPCA model and the process presented time-varying behavior, this was not surprising.

In contrast, Fig. 15, which represents the results of applying the proposed adaptive KPCA-based monitoring scheme to the same data set, confirms the capability of this technique to adapt to naturally occurring time-varying behavior. During the adaptation, within the latter half of the reference data set, the number of retained PCs varied between 13 and 8, which, in turn, produced changes in the 99% confidence limit for the T^2 statistic. An adaptation for the confidence limit of the Q statistic can also be noted in the lower plot.

Finally, Fig. 16 summarizes the application of the adaptive monitoring scheme to the "faulty" data set. The fault could be detected after around 450 samples into the data set and hence, the adaptation of the nonlinear monitoring model was suppressed thereafter. The fault could clearly be detected and the fact that violations of the Q statistics persisted until the end the recording period coincides with the previous analysis of the recorded data (Fig. 9) in Subsection 7.1.

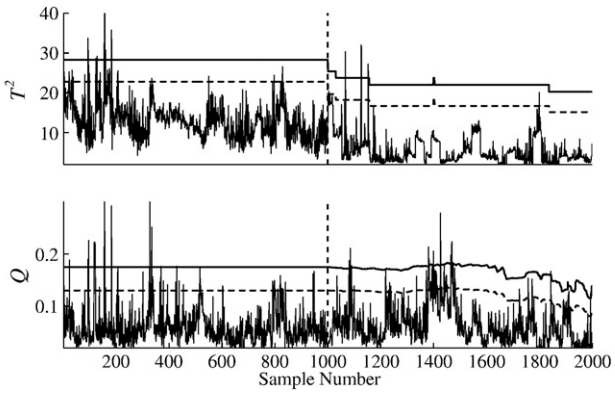

Fig. 15. MWKPCA monitoring for normal data set in SVT.

8. Concluding summary

This article has studied the application of Kernel PCA for monitoring nonlinear and time-varying systems, which are frequent occurrences in the chemical industry. Since a fixed KPCA-based process monitoring model is not applicable in this scenario, the paper has proposed a moving window KPCA formulation. The contributions of this article propose (i) the adaptation of the Gram matrix instead of the Kernel matrix, (ii) incorporation of a recently introduced adaptation method for the eigendecomposition of the Gram matrix, (iii) an adaptation of the KPCA-based on-line monitoring model including the number of retained PCs and the confidence limits for univariate monitoring statistics and (iv) to apply the adapted monitoring model with a delay to increase the sensitivity of the proposed scheme.

The paper has argued that adapting the Gram matrix instead of the Kernel matrix increases the sensitivity for detecting incipient fault conditions. For the second contribution, a detailed analysis has demonstrated that integrating the adaptation of the eigendecomposition of the Gram matrix is of $O(N^2)$, whilst comparative techniques, such as the Lanczos method or the use of an SVD are of $O(N^3)$. A comparison between both the Lanczos technique and the proposed adaptation has shown that the latter method consumed 97.5% and 98.2% fewer floating point operations. The proposed method, however, relies on an approximation. An examination of the accuracy of this approximation has revealed that a recalculation of the Gram matrix was required after 200 to 300 adaptations to ensure insignificant departures in both application studies. Given that the number

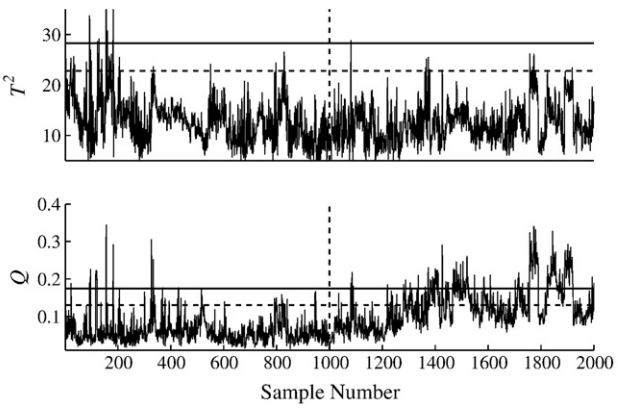

Fig. 14. KPCA monitoring for normal data set in SVT.

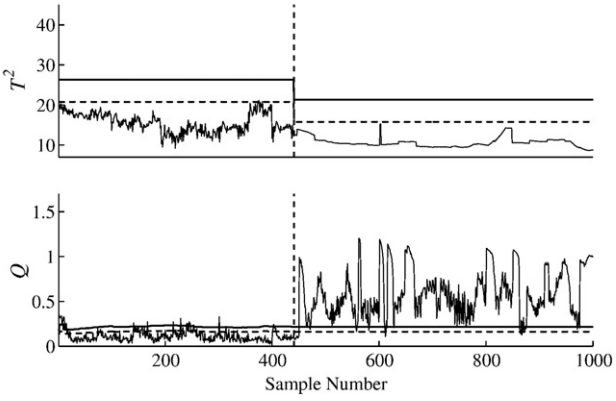

Fig. 16. Application of proposed adaptive KPCA-based monitoring scheme to complete data set including a fault in the last quarter of the data set.

of retained PCs can vary over time, the work utilized a standard method for determining this number on the basis of the eigenvectors. The paper also proposed an adaptation of the statistical confidence limits for univariate monitoring statistics.

To evaluate the working of the proposed nonlinear adaptive monitoring scheme, the paper presented two example studies involving a simulation and the analysis of recorded data from an industrial distillation unit. Both examples confirmed that the proposed adaptive method has been effective in adapting time varying process behavior, whilst being able to detect abnormal events. Both application studies have exemplified how to implement the proposed technique, that is how to determine the kernel parameter, the number of non-zero eigenvalues, and the number of retained PCs, the window size, and the delay for applying the adapted model. Further work will concentrate on how to incorporate missing or corrupted data into the adaptive monitoring scheme and how to diagnose abnormal events. It is also proposed to apply this method to the analysis of power system data, with special regard to quality and reliability.

Acknowledgement

Xueqin Liu is very grateful for the financial support from the Engineering and Physical Science Research Council (EPSRC), Grant Number GR/S84354/01. The work was also partly supported by the National Natural Science Foundation of China, Grant Number 60721062, the National High Technology Research and Development Program of China (863 Program), Grant Number 2007AA04Z162 and the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry of China.

Appendix A. The derivation of $\hat{\mathbf{C}}_{\Phi}$

$$\begin{aligned}
\hat{\mathbf{C}}_{\Phi} &= \frac{1}{N-1}\overline{\boldsymbol{\Phi}}\left(\widehat{\mathbf{X}}\right)\overline{\boldsymbol{\Phi}}\left(\widehat{\mathbf{X}}\right)^{T} \qquad (A.1)\\
&= \frac{1}{N-1}\left\{\sum_{i=2}^{N}\left(\boldsymbol{\Phi}(\mathbf{x}_i)-\hat{\mathbf{m}}_{\Phi}\right)\left(\boldsymbol{\Phi}(\mathbf{x}_i)-\hat{\mathbf{m}}_{\Phi}\right)^{T}\right.\\
&\quad\left.+\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\hat{\mathbf{m}}_{\Phi}\right)\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\hat{\mathbf{m}}_{\Phi}\right)^{T}\right\}\\
&= \frac{1}{N-1}\left\{\sum_{i=2}^{N}\left(\boldsymbol{\Phi}(\mathbf{x}_i)-\tilde{\mathbf{m}}_{\Phi}+\frac{1}{N}\tilde{\mathbf{m}}_{\Phi}-\frac{1}{N}\boldsymbol{\Phi}(\mathbf{x}_{N+1})\right)\right.\\
&\quad\times\left(\boldsymbol{\Phi}(\mathbf{x}_i)-\tilde{\mathbf{m}}_{\Phi}+\frac{1}{N}\tilde{\mathbf{m}}_{\Phi}-\frac{1}{N}\boldsymbol{\Phi}(\mathbf{x}_{N+1})\right)^{T}\\
&\quad+\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\tilde{\mathbf{m}}_{\Phi}+\frac{1}{N}\tilde{\mathbf{m}}_{\Phi}-\frac{1}{N}\boldsymbol{\Phi}(\mathbf{x}_{N+1})\right)\\
&\quad\left.\times\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\tilde{\mathbf{m}}_{\Phi}+\frac{1}{N}\tilde{\mathbf{m}}_{\Phi}-\frac{1}{N}\boldsymbol{\Phi}(\mathbf{x}_{N+1})\right)^{T}\right\}\\
&= \frac{1}{N-1}\left\{\sum_{i=2}^{N}\left[\left(\boldsymbol{\Phi}(\mathbf{x}_i)-\tilde{\mathbf{m}}_{\Phi}\right)\left(\boldsymbol{\Phi}(\mathbf{x}_i)-\tilde{\mathbf{m}}_{\Phi}\right)^{T}\right.\right.\\
&\quad+\frac{1}{N^2}\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\tilde{\mathbf{m}}_{\Phi}\right)\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\tilde{\mathbf{m}}_{\Phi}\right)^{T}\\
&\quad-\frac{1}{N}\left(\boldsymbol{\Phi}(\mathbf{x}_i)-\tilde{\mathbf{m}}_{\Phi}\right)\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\tilde{\mathbf{m}}_{\Phi}\right)^{T}\\
&\quad\left.-\frac{1}{N}\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\tilde{\mathbf{m}}_{\Phi}\right)\left(\boldsymbol{\Phi}(\mathbf{x}_i)-\tilde{\mathbf{m}}_{\Phi}\right)^{T}\right]\\
&\quad\left.+\frac{(N-1)^2}{N^2}\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\tilde{\mathbf{m}}_{\Phi}\right)\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\tilde{\mathbf{m}}_{\Phi}\right)^{T}\right\}\\
&= \frac{1}{N-1}\sum_{i=2}^{N}\left(\boldsymbol{\Phi}(\mathbf{x}_i)-\tilde{\mathbf{m}}_{\Phi}\right)\left(\boldsymbol{\Phi}(\mathbf{x}_i)-\tilde{\mathbf{m}}_{\Phi}\right)^{T}\\
&\quad+\frac{1}{N}\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\tilde{\mathbf{m}}_{\Phi}\right)\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\tilde{\mathbf{m}}_{\Phi}\right)^{T}\\
&= \frac{N-2}{N-1}\tilde{\mathbf{C}}_{\Phi}+\frac{1}{N}\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\tilde{\mathbf{m}}_{\Phi}\right)\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\tilde{\mathbf{m}}_{\Phi}\right)^{T},
\end{aligned}$$

Appendix B. The derivation of $\hat{\mathbf{h}}$

$$\begin{aligned}
\hat{\mathbf{h}} &= \frac{\mathbf{h}}{||\mathbf{h}||_2} \qquad (B.1)\\
&= \frac{\left(\mathbf{I}-\tilde{\mathbf{U}}\tilde{\mathbf{U}}^{T}\right)\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\tilde{\mathbf{m}}_{\Phi}\right)}{\left\|\left(\mathbf{I}-\tilde{\mathbf{U}}\tilde{\mathbf{U}}^{T}\right)\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\tilde{\mathbf{m}}_{\Phi}\right)\right\|_2}\\
&= \frac{\left(\mathbf{I}-\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^{T}\boldsymbol{\Phi}^{T}\left(\widetilde{\mathbf{X}}\right)\right)\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\tilde{\mathbf{m}}_{\Phi}\right)}{\left\|\left(\mathbf{I}-\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^{T}\boldsymbol{\Phi}^{T}\left(\widetilde{\mathbf{X}}\right)\right)\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\tilde{\mathbf{m}}_{\Phi}\right)\right\|_2}\\
&= \frac{\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\frac{1}{N-1}\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)1_{N-1}-\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^{T}\boldsymbol{\Phi}^{T}\left(\widetilde{\mathbf{X}}\right)\boldsymbol{\Phi}(\mathbf{x}_{N+1})+\frac{1}{N-1}\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^{T}\boldsymbol{\Phi}^{T}\left(\widetilde{\mathbf{X}}\right)\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)1_{N-1}}{\left\|\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\frac{1}{N-1}\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)1_{N-1}-\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^{T}\boldsymbol{\Phi}^{T}\left(\widetilde{\mathbf{X}}\right)\boldsymbol{\Phi}(\mathbf{x}_{N+1})+\frac{1}{N-1}\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\tilde{\mathbf{A}}\tilde{\mathbf{A}}^{T}\boldsymbol{\Phi}^{T}\left(\widetilde{\mathbf{X}}\right)\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)1_{N-1}\right\|_2}\\
&= \frac{\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\mathbf{b}}{\left\|\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\mathbf{b}\right\|_2}\\
&= \frac{\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\mathbf{b}}{\sqrt{\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\mathbf{b}\right)^{T}\left(\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\mathbf{b}\right)}}\\
&= \frac{\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\mathbf{b}}{\sqrt{\boldsymbol{\Phi}^{T}(\mathbf{x}_{N+1})\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\boldsymbol{\Phi}^{T}(\mathbf{x}_{N+1})\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\mathbf{b}-\mathbf{b}^{T}\boldsymbol{\Phi}^{T}\left(\widetilde{\mathbf{X}}\right)\boldsymbol{\Phi}(\mathbf{x}_{N+1})+\mathbf{b}^{T}\boldsymbol{\Phi}^{T}\left(\widetilde{\mathbf{X}}\right)\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\mathbf{b}}}\\
&= \frac{\boldsymbol{\Phi}(\mathbf{x}_{N+1})-\boldsymbol{\Phi}\left(\widetilde{\mathbf{X}}\right)\mathbf{b}}{\sqrt{K(\mathbf{x}_{N+1},\mathbf{x}_{N+1})-\mathbf{k}\left(\mathbf{x}_{N+1},\widetilde{\mathbf{X}}\right)\mathbf{b}-\mathbf{b}^{T}\mathbf{k}\left(\widetilde{\mathbf{X}},\mathbf{x}_{N+1}\right)+\mathbf{b}^{T}\widetilde{\mathbf{K}}\mathbf{b}}}
\end{aligned}$$

References

[1] A. AlGhazzawi, B. Lennox, Monitoring a complex refining process using multivariate statistics, Control Engineering Practice 16 (3) (2008) 294–307.
[2] T.W. Anderson, An introduction to multivariate statistical analysis, 3rd edition. John Wiley & Sons, 2003.
[3] J.R. Bunch, C.P. Nielsen, D.C. Sorensen, Rank-one modification of the symmetric eigenproblem, Numerische Mathematik 31 (1978) 31–48.
[4] R.B. Cattell, The scree test for the number of factors, Multivariate Behavioral Research 1 (2) (1966) 245–276.
[5] T.J. Chin, K. Schindler, D. Suter, Incremental kernel svd for face recognition with image sets, Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition, IEEE Computer Society, 2006, pp. 461–466.
[6] J.-H. Choi, J.-M. Lee, S.W. Choi, D. Lee, I.-B. Lee, Fault identification for process monitoring using kernel principal component analysis, Chemical Engineering Science 60 (1) (2005) 279–288.
[7] S.W. Choi, C. Lee, J.-M. Lee, J.H. Park, I.-B. Lee, Fault detection and identification of nonlinear processes based on kpca, Chemometrics and Intelligent Laboratory Systems 75 (1) (2005) 55–67.
[8] P. Cui, J. Li, G. Wang, Improved kernel principal component analysis for fault detection, Expert Systems With Applications 34 (2) (2008) 1210–1219.
[9] D. Dong, T.J. McAvoy, Nonlinear principal component analysis-based on principal curves and neural networks, Computers and Chemical Engineering 20 (1) (1996) 65–78.
[10] Z. Ge, Z. Song, Process monitoring based on independent component analysis–principal component analysis (ica–pca) and similarity factors, Industrial & Engineering Chemistry Research 46 (7) (2007) 2054–2063.
[11] G.H. Golub, Some modified matrix eigenvalue problems, SIAM Review 15 (2) (1973) 318–334.
[12] G.H. Golub, C.F. van Loan, Matrix computation, 3rd edition. John Hopkins, Baltimore, 1996.
[13] P. Hall, D. Marshall, R. Martin, Incrementally computing eigenspace models, Proceeding of British Machine Vision Conference. Southampton, 1998, pp. 286–295.
[14] P. Hall, D. Marshall, R. Martin, Merging and splitting eigenspace models, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (9) (2000) 1042–1049.
[15] P. Hall, D. Marshall, R. Martin, Adding and subtracting eigenspaces with eigenvalue decomposition and singular value decomposition, Image and Vision Computing 20 (13-14) (2002) 1009–1016.
[16] Q. He, F. Kong, R. Yan, Subspace-based gearbox condition monitoring by kernel principal component analysis, Mechanical Systems and Signal Processing 21 (4) (2007) 1755–1772.
[17] L. Hoegaerts, L.D. Lathauwer, I. Goethals, J.A.K. Suykens, J. Vandewalle, B.D. Moor, Efficiently updating and tracking the dominant kernel principal components, Neural Networks 20 (2) (2007) 220–229.
[18] J.E. Jackson, A User's Guide to Principal Components. Wiley Series in Probability and Mathematical Statistics, John Wiley, New York, 1991.
[19] F. Jia, E.B. Martin, A.J. Morris, Nonlinear principal components analysis with application to process fault detection, International Journal of Systems Science 31 (11) (2000) 1473–1487.
[20] I.T. Joliffe, Principal component analysis, Springer, New York, 1986.
[21] B.J. Kim, I.K. Kim, Incremental nonlinear pca for classification, Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, Springer Verlag, 291–300, 2004.
[22] M.A. Kramer, Nonlinear principal component analysis using autoassociative neural networks, AIChE journal 37 (3) (1991) 233–243.

[23] U. Kruger, D. Antory, J. Hahn, G.W. Irwin, G. McCullough, Introduction of a nonlinearity measure for principal component models, Computers & Chemical Engineering 29 (11-12) (2005) 2355–2362.
[24] U. Kruger, Q. Chen, D.J. Sandoz, R.C. McFarlane, Extended pls approach for enhanced condition monitoring of industrial processes, AIChE Journal 47 (9) (2001) 2076–2091.
[25] U. Kruger, J. Zhang, L. Xie, 2007. Principal manifolds for data visualization and dimension reduction. Vol. 58 of Lecture notes in computational science and engineering. Springer Verlag, Berlin-Heidelberg-New York, Ch. Developments and applications of nonlinear principal component analysis — A review, pp. 1–44.
[26] D.S. Lee, M.W. Lee, S.H. Woo, Y.-J. Kim, J.M. Park, Multivariate online monitoring of a full-scale biological anaerobic filter process using kernel-based algorithms, Industrial & Engineering Chemistry Research 45 (12) (2006) 4335–4344.
[27] J.M. Lee, S.J. Qin, I.B. Lee, Fault detection and diagnosis based on modified independent component analysis, AIChE Journal 52 (10) (2006) 3501–3514.
[28] J.M. Lee, C. Yoo, S.W. Choi, P.A. Vanrolleghem, I.B. Lee, Nonlinear process monitoring using kernel principal component analysis, Chemical Engineering Science 59 (1) (2004) 223–234.
[29] W. Li, H. Yue, S. Valle-Cervantes, S.J. Qin, Recursive pca for adaptive process monitoring, Journal of Process Control 10 (5) (2000) 471–486.
[30] X. Liu, L. Xie, U. Kruger, T. Littler, S. Wang, Statistical-based monitoring of multivariate non-Gaussian systems, AIChE Journal 54 (9) (2008) 2379–2391.
[31] F.R. Malinowski, Factor analysis in chemistry, Wiley, New York, 1991.
[32] T.E. Morud, Multivariate statistical process control; example from the chemical process industry, Journal of Chemometrics 10 (5-6) (1996) 669–675.
[33] P. Nomikos, J.F. MacGregor, Multivariate spc charts for monitoring batch processes, Technometrics 37 (1) (1995) 41–59.
[34] C.C. Paige, Accuracy and effectiveness of the Lanczos algorithm, Linear Algebra and Its Applications 34 (1980) 235–258.
[35] B.N. Parlett, The symmetric eigenvalue problem. Prentice-Hall series in computational mathematics, Prentice Hall, Englewood Cliffs, NJ, 1980.
[36] S.J. Qin, R. Dunia, Determining the number of principal components for best reconstruction, Journal of Process Control 10 (2) (2000) 245–250.
[37] B. Scholkopf, S. Mika, C.J.C. Burges, P. Knirsch, K.R. Muller, G. Ratsch, Input space versus feature space in kernel-based methods, IEEE Transactions on Neural Networks 10 (5) (1999) 1000–1016.
[38] S. Tan, M.L. Mavrovouniotis, Reducing data dimensionality through optimizing neural network inputs, AIChE Journal 41 (6) (1995) 1471–1480.
[39] N.D. Tracy, J.C. Young, R.L. Mason, Multivariate control charts for individual observations, Journal of Quality Control 24 (2) (1992) 88–95.
[40] S. Valle, W.H. Li, S.J. Qin, Selection of the number of principal components: the variance of the reconstruction error criterion compared to other methods, Industrial & Engineering Chemistry Research 38 (11) (1999) 4389–4401.
[41] V.N. Vapnik, Statistical learning theory, Wiley, New York, 1998.
[42] X. Wang, U. Kruger, G.W. Irwin, Process monitoring approach using fast moving window pca, Industrial & Engineering Chemistry Research 44 (15) (2005) 5691–5702.
[43] X. Wang, U. Kruger, B. Lennox, Recursive partial least squares algorithms for monitoring complex industrial processes, Control Engineering Practice 11 (6) (2003) 613–632.
[44] L. Xie, S.Q. Wang, Recursive kernel pca and its application in adaptive monitoring of nonlinear processes, Journal of Chemical Industry and Engineering 58 (7) (2007) 1776–1782.
[45] C.K. Yoo, I.B. Lee, Nonlinear multivariate filtering and bioprocess monitoring for supervising nonlinear biological processes, Process Biochemistry 41 (8) (2006) 1854–1863.
[46] J. Zhang, S.J. Zhao, Y.M. Xu, Performance monitoring of processes with multiple operating modes through multiple pls models, Journal of Process Control 16 (7) (2006) 763–772.
[47] C. Zhao, F. Wang, N. Lu, M. Jia, Stage-based soft-transition multiple pca modeling and on-line monitoring strategy for batch processes, Journal of Process Control 17 (9) (2007) 728–741.