This document contains the **post-print pdf version** of the refereed paper:

“Dynamic model-based fault diagnosis for (bio)chemical batch processes”

by *Pieter Van den Kerkhof, Jef Vanlaer, Geert Gins, and Jan F.M. Van Impe*

which has been archived on the university repository Lirias (<https://lirias.kuleuven.be/>) of the KU Leuven.

The content is identical to the content of the published paper, but without the final typesetting by the publisher.

When referring to this work, please cite the full bibliographic info:

P. Van den Kerkhof, J. Vanlaer, G. Gins and J.F.M. Van Impe (2012). Dynamic model-based fault diagnosis for (bio)chemical batch processes, Computers and Chemical Engineering, 40, 12-21.

The journal and the original published paper can be found at:

<http://www.journals.elsevier.com/computers-and-chemical-engineering>

<http://dx.doi.org/10.1016/j.compchemeng.2012.01.013>

The corresponding author can be contacted for additional info.

Conditions for open access are available at:

<http://www.sherpa.ac.uk/romeo/>

Dynamic model-based fault diagnosis for (bio)chemical batch processes

Pieter Van den Kerkhof^a, Geert Gins^a, Jef Vanlaer^a, Jan F.M. Van Impe^{a,*}

^a*Chemical and Biochemical Process Technology and Control (BioTeC), Department of
Chemical Engineering, Katholieke Universiteit Leuven, W. de Croylaan 46, B-3001
Heverlee (Leuven), Belgium*

Abstract

To ensure constant and satisfactory product quality, close monitoring of batch processes is an absolute requirement in the (bio)chemical industry. *Principal Component Analysis* (PCA)-based techniques exploit historical databases for fault detection and diagnosis. In this paper, the fault detection and diagnosis performance of *Batch Dynamic* PCA (BDPCA) and *Auto-Regressive* PCA (ARPCA) is compared with *Multi-way* PCA (MPCA). Although these methods have been studied before, the performance is often compared based on few validation batches. Additionally, the focus is on fast fault detection, while correct fault identification is often considered of lesser importance. In this paper, MPCA, BDPCA, and ARPCA are benchmarked on an extensive dataset of a simulated penicillin fermentation. Both the detection speed, false alarm rate and correctness of the fault diagnosis are taken into account. The results indicate increased detection speed when using ARPCA as opposed to MPCA and BDPCA at the cost of fault classification accuracy.

Keywords: Biochemical batch processes, Fault detection/diagnosis, Principal Component Analysis (PCA), Process monitoring, Statistical Process Control (SPC)

*Corresponding author

Email addresses: pieter.vandenkerkhof@cit.kuleuven.be (Pieter Van den Kerkhof), geert.gins@cit.kuleuven.be (Geert Gins), jef.vanlaer@cit.kuleuven.be (Jef Vanlaer), jan.vanimpe@cit.kuleuven.be (Jan F.M. Van Impe)

1. Introduction

Due to their high flexibility and low capital cost, batch processes are commonly used in the chemical and biochemical industry for the production of products with high added value (e.g., pharmaceuticals, polymers, food, semiconductors). Because of their dynamic nature, close monitoring and control of these processes is an absolute requirement to ensure a constant and satisfactory product quality. Process plants nowadays routinely dispose of historical databases containing frequent measurements of online sensors on hundreds of variables. *Principal Component Analysis* (PCA)-based techniques exploit these historical data for fault detection and diagnosis.

Most recent research has been devoted to fault detection and identification using *Multi-way Principal Component Analysis* (MPCA; [1]). While better-performing monitoring schemes have been developed (e.g., *Auto-Regressive PCA* (ARPCA; [2]), *Batch Dynamic PCA* (BDPCA; [3])), the performance comparison is often based on few test batches (e.g., only 4 faulty batches are considered in [2] and 3 in [3]). In this paper, an extensive dataset of 840 faulty batches is numerically generated in which faults on different variables with varying magnitude, fault sign and fault type are introduced at distinct onset times. This validation set is subsequently monitored. To provide a thorough performance comparison, the speed of detection, the false alarm rate and correctness of the fault diagnosis are taken into account.

The basic method of *Principal Component Analysis* (PCA) is summarized in Section 2. Next, PCA-based fault detection techniques are described in Section 3. Fault detection and fault diagnosis are detailed in Sections 4 and 5, followed by a description of the data on which the performance of the methods is tested in Section 6. Section 7 details the model identification procedures. The results of the fault detection and fault diagnosis performance comparison are presented and discussed in Section 8. Finally, conclusions are drawn in Section 9.

2. Principal Component Analysis (PCA)

Industrial data is typically heavily correlated as a consequence of redundant sensors, mass balances, physical laws, etc. PCA is a popular technique to reduce the dimension of a dataset by exploiting these correlations [5, 4]. Multivariate data, consisting of K measurements on J variables, is conveniently organized in a $K \times J$ data matrix $\mathbf{X}$. Each column of $\mathbf{X}$ is first

normalized to zero mean and unit variance. Next, $\mathbf{X}$ is projected onto a lower dimensional space of dimension R ($R \leq J$) by a $J \times R$ loading matrix $\mathbf{P}$. $\mathbf{X}$ is approximated by the product of a $K \times R$ scores matrix $\mathbf{T}$ and the transpose of the loading matrix $\mathbf{P}$. $\mathbf{E}_\mathbf{X}$ represents the residuals.

$$\mathbf{X} = \mathbf{T}\mathbf{P}^T(+\mathbf{E}_\mathbf{X}) \quad (1)$$

The columns of $\mathbf{P}$ correspond to the principal components of the input data. The first principal component is the direction of maximum variance in the data; subsequent components explain gradually less variance. The loading matrix $\mathbf{P}$ maximizes the information content in the scores matrix $\mathbf{T}$ while minimizing its dimension (i.e., number of columns R).

When at time point k a new $1 \times J$ measurement vector $\mathbf{x}_k$ comes in, the corresponding $1 \times R$ scores vector $\mathbf{t}_k$ and residual vector $\mathbf{e}_k$ are calculated by

$$\mathbf{t}_k = \mathbf{x}_k \mathbf{P} \quad (2)$$

$$\mathbf{e}_k = \mathbf{x}_k - \mathbf{t}_k \mathbf{P}^T \quad (3)$$

Sections 4 and 5 detail how the scores and residuals can be used for fault detection and fault diagnosis. Different techniques exist to employ PCA for batch process monitoring. The computation of scores and residuals for the PCA-based techniques in this study is discussed in Section 3.

3. PCA-based fault detection techniques

For each batch a $K \times J$ multivariate time series $\mathbf{X}$ arises, making a batch dataset inherently three-dimensional. The data of a historical database containing I batches can be conveniently organized in an array of I batches $\times J$ variables $\times K$ time points. PCA, however, can only handle two-dimensional data, hence the dimensionality has to be reduced first.

Multi-way PCA (MPCA), originally developed by Nomikos and MacGregor [1, 6] based on the pioneering work of Wold *et al.* [7], relies on unfolding the $I \times J \times K$ batch data array to a matrix prior to modeling. The two most widely used unfolding methods are batch-wise and variable-wise unfolding which are detailed in Section 3.1 and 3.2, respectively. To deal with the serial dependence within each variable series at past time instances (autocorrelation) and the dependence between different measured variables (crosscorrelation), dedicated techniques have been developed.

Variable-wise unfolded MPCA (MPCAV) only incorporates the instantaneous cross-correlations between the measured variables [8, 9]. Leaving serial correlation in the residuals violates the assumptions of the fault detection statistics and may lead to decreased monitoring performance [10]. For a detailed study on the impact of cross- and autocorrelation on the significance level for hypothesis testing in monitoring statistics, the reader is referred to the work of Xie *et al.* [11].

Batch-wise unfolded MPCA (MPCAB) incorporates the full linear dynamics around the average trajectory, but entails the need for estimating future observations [8, 9].

Batch Dynamic PCA (BDPCA) incorporates dynamic behavior by adding time-lagged variables [3]. BDPCA is a generalization of the previous unfolding techniques. If no lagged variables are added, the resulting matrix is unfolded variable-wise. If all possible lag variables are added, a batch-wise unfolded matrix emerges [9, 12]. BDPCA is discussed in Section 3.3.

Auto-Regressive PCA (ARPCA) filters out auto- and crosscorrelations by applying PCA on the residuals of a times series modeling approach [2]. In this way, the deviations from the linear dynamics around the average trajectory are monitored. ARPCA is discussed in Section 3.4.

3.1. MPCAB–Batch-wise unfolded multi-way PCA

Nomikos and MacGregor proposed a batch-wise unfolding technique to transform the $I \times J \times K$ batch data array $\mathbf{\bar{X}}$ to a matrix of size $I \times JK$ as depicted in Fig. 1A [1, 6, 13]. Each row contains the data of a single batch. In fact, batch-wise unfolding treats each measurement at a different time point as a different variable which allows for modeling the full dynamic behavior of the process.

Before applying PCA, the unfolded data is normalized. First, the mean trajectory of each variable is subtracted to remove the main non-linearities in the data [1, 6, 8]. For batch-wise unfolded data, subtracting the average trajectory corresponds to mean-centering each column. Next, the columns are scaled to unit variance to give equal weight to each variable [6, 8].

To obtain the scores and residuals in Eq. 2 and 3 the measurements of the entire batch are necessary. Hence, the downside of batch-wise unfolding is the need to estimate future measurements when monitoring a newly evolving batch online. Several adequate techniques for this purpose exist [6, 14]. In this work the *Trimmed Scores Regression* [14] method is adopted.

Figure 1: (A) Batch-wise unfolding of a three-way array of I batches $\times$ J variables $\times$ K time points to a $I \times JK$ matrix. (B) Variable-wise unfolding to a $IK \times J$ matrix.

3.2. MPCAV–Variable-wise unfolded multi-way PCA

Wold *et al.* proposed a different approach by first applying variable-wise unfolding which maintains the variable direction and merges the batch and time directions followed by normalizing each column through mean-centering and scaling to unit variance [15]. As illustrated in Fig. 1B, variable-wise unfolding results in a matrix $\mathbf{X}$ of dimension $IK \times J$. When applying PCA to variable-wise unfolded data, the process dynamics are ignored as measurements on different time instances are regarded as the same variables. As in MPCAB, the data is normalized before applying PCA. Mean-centering variable-wise unfolded data corresponds to subtracting the grand mean of each variable and leaves non-linear behavior in the data [8].

Using variable-wise unfolding avoids the need to estimate future measurements. Nevertheless, it may lead to a decrease in performance as compared to batch-wise unfolding because the deviation of the process variables from their mean values is monitored instead of the deviations from their average trajectories [8]. Therefore in this paper the data is first normalized around the average trajectory as discussed in the previous paragraph. Next, the data is unfolded variable-wise. No normalization step is performed on the variable-wise unfolded data.

3.3. BDPCA–Batch dynamic PCA

Batch dynamic PCA (BDPCA) was proposed by Chen and Liu [3] as an extension of dynamic PCA from continuous processes to batch processes [16]. BDPCA models dynamic behavior by applying a window of time-lagged measurements. Each current observation vector is stacked with the L previous

observation vectors. Hence for each batch i this results in a $(K - L) \times J(L + 1)$ data matrix of the following form

$$\mathbf{X}_L^i = \begin{bmatrix} \mathbf{x}_{L+1}^i & \mathbf{x}_L^i & \cdots & \mathbf{x}_1^i \\ \mathbf{x}_{L+2}^i & \mathbf{x}_{L+1}^i & \cdots & \mathbf{x}_2^i \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_K^i & \mathbf{x}_{K-1}^i & \cdots & \mathbf{x}_{K-L}^i \end{bmatrix} \quad (4)$$

where L is the number of time lags and $\mathbf{x}_k^i = [x_{1,k} \ x_{2,k} \ \dots \ x_{J,k}]$ the $1 \times J$ observation vector at time point k of batch i .

Chen and Liu proposed the following modeling procedure [3]. First, for each batch i a time-lagged data matrix according to Eq. 4 is constructed and the covariance matrix is calculated as

$$\mathbf{S}_{\mathbf{X}_L}^i = \frac{(\mathbf{X}_L^i)^T (\mathbf{X}_L^i)}{K - L - 1} \quad (5)$$

The information contained in the covariance matrix is then pooled over all batches by computing the average covariance matrix:

$$\mathbf{S}_{\mathbf{X}_L}^{avg} = \frac{(K - L - 1) \sum_{i=1}^I \mathbf{S}_{\mathbf{X}_L}^i}{I(K - L)}. \quad (6)$$

The eigenvectors of the average covariance matrix correspond to the principal components in standard PCA. When monitoring a new batch, the scores and residuals at each time point k can be calculated as

$$\mathbf{t}_k = \mathbf{x}_{k:k-L} \mathbf{P} \quad (7)$$

$$\mathbf{e}_k = \mathbf{x}_{k:k-L} - \mathbf{t}_k \mathbf{P}^T \quad (8)$$

where $\mathbf{x}_{k:k-L}$ is a $1 \times J(L + 1)$ row vector containing the measurements of the current and L previous time points. Note that monitoring can only start when the first $L + 1$ time points have passed.

Chen and Liu do not mention any normalization of the measured data [3]. In this paper, the data are first normalized batch-wise as detailed in Section 3.1 before constructing the time-lagged data matrices. No normalization is performed on the time-lagged data matrices.

The number of time lags L is an important parameter of the BDPCA-method. BDPCA assumes a constant correlation structure during the batch that can

be captured by considering the L previous measurement vectors in addition to the current one. In this respect, BDPCA is very similar to auto-regression models. The addition of lag variables also allows for incorporating dynamic behavior without raising the need for future measurements estimation, as opposed to batch-wise MPCA. The optimal number of lags can be determined by the method of Ku *et al.* [16].

3.4. ARPCA–Auto-regressive PCA

Auto-Regressive PCA (ARPCA) was proposed by Choi *et al.* [2] as a combination of a multivariate auto-regressive (AR) model and multi-way PCA. After the standard data-pretreatment step of normalizing around the average trajectory, the remaining cross- and autocorrelations are estimated via a multivariate auto-regressive model which attempts to predict the $1 \times J$ measurement vector $\mathbf{x}_k$ as a linear combination of the L previous measurement vectors:

$$\mathbf{x}_k = \sum_{l=1}^L \mathbf{x}_{k-l} \mathbf{C}_l + \mathbf{e}_k. \quad (9)$$

L represents the number of lags, i.e., the number of previous time instances used to predict the current measurements, $\mathbf{C}_l$ is a $J \times J$ system matrix containing the AR coefficients corresponding to the l -th lag which model the cross- and autocorrelations and $\mathbf{e}_k$ represents the residuals. By stacking each of the L system matrices vertically to obtain the $JL \times J$ augmented system matrix $\mathbf{C}$, Eq. 9 can be transformed to

$$\mathbf{x}_k = \mathbf{x}_{k-1:k-L} \mathbf{C} + \mathbf{e}_k \quad (10)$$

where $\mathbf{x}_{k-1:k-L}^i$ is a row vector of length JL containing the J measurements of the L previous time points. In auto-regressive modeling the same assumption of constant correlation structure during the batch is made as in BDPCA. Therefore, Eq. 10 can be expanded to include each time point k of a batch i :

$$\begin{bmatrix} \mathbf{x}_{L+1}^i \\ \mathbf{x}_{L+2}^i \\ \vdots \\ \mathbf{x}_K^i \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{L:1}^i \\ \mathbf{x}_{L+1:2}^i \\ \vdots \\ \mathbf{x}_{K-1:K-L}^i \end{bmatrix} \mathbf{C} + \mathbf{E}$$

or using the notation $\mathbf{Y}^{i*}$ and $\mathbf{X}^{i*}$:

$$\mathbf{Y}^{i*} = \mathbf{X}^{i*} \mathbf{C} + \mathbf{E}. \quad (11)$$

In order to use all the reference data to determine the auto-regression coefficients in $\mathbf{C}$, Eq. 11 is expanded over all I batches by stacking $\mathbf{Y}^{i*}$ and $\mathbf{X}^{i*}$ for each batch, resulting in a $(K - L)I \times J$ matrix $\mathbf{Y}^*$ and a $(K - L)I \times JL$ matrix $\mathbf{X}^*$.

$$\mathbf{Y}^* = \mathbf{X}^* \mathbf{C} + \mathbf{E} \quad (12)$$

The augmented system matrix $\mathbf{C}$ is found by regression between the $\mathbf{X}^*$ and $\mathbf{Y}^*$ matrices. To obtain a stable solution and avoid any rank-deficiency issues, the Partial Least Squares-method (PLS; [17]) is used for regression instead of normal least squares [2]. In PLS the $\mathbf{Y}^*$ -block is related to the $\mathbf{X}^*$ -block as follows:

$$\begin{aligned} \mathbf{X}^* &= \mathbf{T}_{\text{PLS}} \mathbf{P}_{\text{PLS}}^T + \mathbf{E}_{\mathbf{X}^*} \\ \mathbf{Y}^* &= \mathbf{T}_{\text{PLS}} \mathbf{Q}_{\text{PLS}}^T + \mathbf{E}_{\mathbf{Y}^*} \end{aligned} \quad (13)$$

where $\mathbf{P}_{\text{PLS}}^T$ and $\mathbf{Q}_{\text{PLS}}^T$ are the loading matrices, $\mathbf{T}_{\text{PLS}}$ is the score matrix and $\mathbf{E}_{\mathbf{X}^*}$ and $\mathbf{E}_{\mathbf{Y}^*}$ represent the residuals on $\mathbf{X}^*$ and $\mathbf{Y}^*$, respectively. The loading matrices are computed using the NIPALS algorithm [17]. The augmented system matrix $\mathbf{C}$ is then given by

$$\mathbf{C} = \mathbf{W}(\mathbf{P}_{\text{PLS}}^T \mathbf{W})^{-1} \mathbf{Q}_{\text{PLS}}^T = \mathbf{J} \mathbf{Q}_{\text{PLS}}^T \quad (14)$$

with $\mathbf{W}$ a weight matrix following from the NIPALS algorithm and

$$\mathbf{J} \triangleq \mathbf{W}(\mathbf{P}_{\text{PLS}}^T \mathbf{W})^{-1}. \quad (15)$$

In the approach of Choi *et al.* the scores of the PLS model are also used for monitoring a new batch [2].

$$\mathbf{t}_{\text{PLS},k} = \mathbf{J} \mathbf{x}_{k-1:k-L} \quad (16)$$

After calibrating the AR model, the AR residuals $\mathbf{E}$ are organized in a variable-wise unfolded fashion and analyzed by standard PCA.

4. Fault detection

In Statistical Process Monitoring, the normal behavior of the process is characterized by an empirical PCA-based model identified on a reference dataset obtained under Normal Operating Conditions (NOC). For a new process run, abnormal behavior is detected by referencing the measured process

data against this in-control model [1].

By projecting the new observations onto the model hyperplane defined by the loading matrix $\mathbf{P}$, the scores and residuals at each time point k of the current process run can be calculated (Eq. 2 and 3). If the scores or residuals of the new observations differ significantly from those of the reference set, a process fault is detected. This difference is measured by two (scalar) fault detection statistics: the Hotelling's T^2 and the Squared Prediction Error (SPE). A graphical interpretation of the T^2 and SPE statistics is shown in Fig. 2.

The T^2 statistic checks if a new observation projects onto the model plane defined by the loading matrix $\mathbf{P}$ within the limits determined by the NOC:

$$T_k^2 = (\mathbf{t}_k - \boldsymbol{\mu}_{t,k}) \mathbf{S}_k^{-1} (\mathbf{t}_k - \boldsymbol{\mu}_{t,k})^T \quad (17)$$

where $\mathbf{t}_k$ is the scores vector of the new run at time k , $\mathbf{S}_k^{-1}$ is the inverse covariance matrix of the reference set scores at time point k and $\boldsymbol{\mu}_{t,k}$ is the mean scores vector at time point k over all I batches of the reference set.

The SPE detects the occurrence of any abnormal events that cause the new observations to move away from the model plane.

$$\text{SPE}_k = \mathbf{e}_k \mathbf{e}_k^T = \sum_{j=1}^J e_{jk}^2 \quad (18)$$

with e_{jk} the residual for the j -th variable at time point k .

The process is assumed in-control at time k if the calculated values for both statistics are below their corresponding Upper Control Limits (UCL) with tolerance level α . The UCL of the T^2 statistic with significance level α is given by

$$\text{UCL}_{T^2} = \frac{R(I^2 - 1)}{I(I - R)} F(R, I - R; \alpha) \quad (19)$$

where R is the model order (i.e., the number of principal components included in the PCA model) and $F(R, I - R; \alpha)$ is the upper critical value of an F -distribution with R and $I - R$ degrees of freedom [18]. The control limit of the SPE statistic is

$$\text{UCL}_{\text{SPE}}(k) = \frac{\sigma_k^2}{2\mu_k} \chi^2(2\mu_k^2/\sigma_k^2; \alpha) \quad (20)$$

where the mean and variance of the SPE statistic over all training batches at each time point k are denoted by μ_k and σ_k^2 and $\chi^2(2\mu_k^2/\sigma_k^2; \alpha)$ is the upper

Figure 2: Graphical representation of the Hotelling's T^2 and SPE fault detection statistics. The SPE statistic checks the model validity by measuring the distance between the data in the original measurement space $\mathbf{X}$ and its projection onto the model plain $\mathbf{T}$. The T^2 statistic equals the Mahalanobis distance of $\mathbf{T}$ to the center of the projected reference data and checks if new observations project within the limits of normal operation.

critical value of a χ^2 -distribution with $2\mu_k^2/\sigma_k^2$ degrees of freedom at specified tolerance level α [6]. The UCL of the T^2 statistic is independent of k whereas the SPE control limit varies with time. To mitigate the occurrence of false alarms, an alarm signal is raised only when a statistic is above its control limit at three or more consecutive time points.

5. Fault diagnosis

If an abnormal situation is detected by either the T^2 or SPE statistic, information about the cause of the disturbance is obtained by examining the contribution of each variable to the out-of-control statistic on contribution plots [19, 20]. In this paper, contributions are calculated using the approach proposed by Choi *et al.* [2]. The contribution to the T^2 statistic can be found by substituting Eq. 2 in the expression for the T^2 statistic.

$$\begin{aligned} T_k^2 &= (\mathbf{t}_k - \boldsymbol{\mu}_{t,k}) \mathbf{S}_k^{-1} (\mathbf{t}_k - \boldsymbol{\mu}_{t,k})^T \\ &= (\mathbf{x}_k \mathbf{P} - \boldsymbol{\mu}_{t,k}) \mathbf{S}_k^{-1} (\mathbf{t}_k - \boldsymbol{\mu}_{t,k})^T \end{aligned}$$

Decomposing the leading factor yields:

$$\begin{aligned}
T_k^2 &= \left(\begin{bmatrix} x_{k,1} & x_{k,2} & \cdots & x_{k,J} \end{bmatrix} \begin{bmatrix} \mathbf{P}(1,:) \\ \mathbf{P}(2,:) \\ \vdots \\ \mathbf{P}(J,:) \end{bmatrix} - \boldsymbol{\mu}_{t,k} \right) \mathbf{S}_k^{-1} (\mathbf{t}_k - \boldsymbol{\mu}_{t,k})^T \\
&= \left(\sum_{j=1}^J x_{k,j} \mathbf{P}(j,:) - \sum_{j=1}^J \boldsymbol{\mu}_{t,k,j} \right) \mathbf{S}_k^{-1} (\mathbf{t}_k - \boldsymbol{\mu}_{t,k})^T \\
&= \sum_{j=1}^J (x_{k,j} \mathbf{P}(j,:) - \boldsymbol{\mu}_{t,k,j}) \mathbf{S}_k^{-1} (\mathbf{t}_k - \boldsymbol{\mu}_{t,k})^T
\end{aligned}$$

with $\boldsymbol{\mu}_{t,k,j}$ the contribution of variable j to the mean score vector $\boldsymbol{\mu}_{t,k}$ and $\mathbf{P}(j,:)$ the j -th row of $\mathbf{P}$. Hence, the contribution of variable j to the T^2 statistic at time point k can be defined as

$$c_{j,k}^{T^2} \triangleq (x_{k,j} \mathbf{P}(j,:) - \boldsymbol{\mu}_{t,k,j}) \mathbf{S}_k^{-1} (\mathbf{t}_k - \boldsymbol{\mu}_{t,k})^T. \quad (21)$$

The variable contributions to the SPE statistic are derived in the same manner by first substituting Eq. 3 and then Eq. 2 in the formula of the SPE statistic.

$$\begin{aligned}
\text{SPE}_k &= \mathbf{e}_k \mathbf{e}_k^T \\
&= (\mathbf{x}_k - \mathbf{t}_k \mathbf{P}^T) (\mathbf{x}_k - \mathbf{t}_k \mathbf{P}^T)^T \\
&= \mathbf{x}_k (\mathbf{I} - \mathbf{P} \mathbf{P}^T) (\mathbf{I} - \mathbf{P} \mathbf{P}^T)^T \mathbf{x}_k^T \\
&= \mathbf{x}_k \mathbf{B} \mathbf{B}^T \mathbf{x}_k^T
\end{aligned}$$

where $\mathbf{I}$ is an identity matrix of appropriate size and $\mathbf{B}$ is equal to $\mathbf{I} - \mathbf{P} \mathbf{P}^T$. Decomposing the leading factor yields:

$$\begin{aligned}
\text{SPE}_k &= \begin{bmatrix} x_{k,1} & x_{k,2} & \cdots & x_{k,J} \end{bmatrix} \begin{bmatrix} \mathbf{B}(1,:) \\ \mathbf{B}(2,:) \\ \vdots \\ \mathbf{B}(J,:) \end{bmatrix} \mathbf{B}^T \mathbf{x}_k^T \\
&= \sum_{j=1}^J x_{k,j} \mathbf{B}(j,:) \mathbf{B}^T \mathbf{x}_k^T.
\end{aligned}$$

Hence, the expression for the contribution of a variable $x_{k,j}$ to the SPE_k can be defined as

$$c_{j,k}^{\text{SPE}} \triangleq x_{k,j} \mathbf{B}(j, :) \mathbf{B}^T \mathbf{x}_k^T \quad (22)$$

with $\mathbf{B}(j, :)$ the j -th row of $\mathbf{B}$.

The preceding equations are valid only for the MPCAB and MPCAV models. When including lag variables, as in the BDPCA approach, the contributions of the variable owing to the lagged measurements must also be considered. Therefore in BDPCA the expressions for the contributions to the T^2 and SPE statistic become:

$$c_{j,k}^{T^2} = \sum_{l=0}^L [(x_{k-l,j} \mathbf{P}(j(l+1), :) - \mu_{t,k,j}) \mathbf{S}_k^{-1} (\mathbf{t}_k - \boldsymbol{\mu}_{t,k})^T] \quad (23)$$

$$c_{j,k}^{\text{SPE}} = \sum_{l=0}^L [x_{k-l,j} \mathbf{B}(j(l+1), :) \mathbf{B}^T \mathbf{x}_k^T] \quad (24)$$

where the contribution of one variable is summed over the current time instance and all previous lags. The expressions of the contributions for the ARPCA approach are very similar to those of BDPCA and can be found in Choi *et al.* [2].

When a fault is detected the contributions of the different variables to the out-of-control statistic are plotted on a bar chart. Limits for each contribution are set based on the reference set as $m_c \pm t_{\text{inv}}(I-1; \alpha) s_c$. Where m_c and s_c are the mean and standard deviation of the contributions for each time point and variable over all training batches and $t_{\text{inv}}(I-1; \alpha)$ is the inverse of the Student's T cumulative distribution with $I-1$ degrees of freedom and significance level α .

A contribution plot indicates the variables involved in the abnormal situation whether they have caused the fault or are influenced by it. Therefore it does not allow direct fault diagnosis. However, contribution plots greatly narrow the search for the cause of the disturbance and have been proved useful in many applications (e.g., [21, 22, 23]).

6. Data

For a reliable fault detection and diagnosis performance comparison, large amounts of both reference and faulty data are mandatory. For practical, economic and safety reasons, it is simply impossible to generate a sufficient

amount of faulty batches in a real industrial installation. Furthermore, even if companies were to provide a set of faulty batches, the exact fault onset time is most often unknown and the cause unclear. Therefore, a numerical simulator which mimics an industrial process is used for data generation.

6.1. Data generation

An industrial-scale biochemical process for penicillin fermentation is simulated using an expanded version of the **Pensim** simulator [24]. During the fermentation, 15 sensors record various concentrations, flows, the bioreactor temperature, and pH. The the same settings and noise levels as in the original **Pensim** simulator were used. The measured signals are aligned and resampled to a length of 602 samples via indicator variables, following the procedure of Birol *et al.* [24]. The time signal is added as an extra (aligned) variable, resulting in a training data array $\mathbf{X}$ of size $200 \times 16 \times 602$. Two main phases are identified in the dataset: (i) a batch phase of 101 samples, and (ii) a fed-batch phase of 501 samples.

As a reference dataset, 200 normal batches are simulated with varying initial conditions. Initially, the bioreactor is operated in batch mode. Once the substrate concentration reaches 0.3 g/L (after about 43 h), the fed batch phase is started, and additional substrate is fed into the reactor. After adding 25 L of substrate (after approximately 550 h), the fermentation is stopped.

Additionally, a grand total of 840 faulty batches are used for testing the fault detection and diagnosis. Step faults and gradual faults are introduced on the aeration rate, stirrer power and feed rate, with magnitudes of $\pm 2\%$ and $\pm 5\%$ for step faults and $\pm 0.05\%/h$ and $\pm 0.10\%/h$ for gradual faults. To simulate sensor drift, gradual faults are also introduced on the dissolved oxygen (DO) measurement. Each combination of fault type, magnitude and variable is simulated five times at three different onset times in the batch phase (after 5, 15 and 25 h of operation) and three onset times in the fed-batch phase (after 180, 315, and 450 h of operation).

6.2. Data properties

An autocorrelation plot of the penicillin concentration variable during the batch phase is depicted in Fig. 3. The autocorrelation plot indicates that a high degree of autocorrelation is present in the data. autocorrelation plots of other variables show similar behavior. Hence, the generated process data form a relevant case study to compare models that take autocorrelation into account.

Figure 3: Autocorrelation plot of the penicillin concentration variable during the batch phase.

Figure 4: Normal probability plot of the penicillin concentration variable half-way through the batch phase ($k = 50$).

A normal probability plot of the variable penicillin concentration half-way through the batch phase ($k = 50$) is shown in Fig. 4. The points on this plot form a nearly linear pattern, which indicates that the data is normally distributed to a good approximation. Computing the Anderson-Darling (AD) statistic yields a value of 0.427 thus the assumption of normality is not rejected ($\alpha = 5\%$). Normal probability plots of other variables and at different time points show that non-Gaussian distributions are not an issue for this dataset and therefore the standard PCA-based approach is valid.

6.3. Impact of normalization on autocorrelation

The autocorrelation plot of the raw penicillin concentration measurement in Fig. 3 indicated a significant amount of autocorrelation. The PCA-based models do not work directly on the data as the unfolded data is first normalized. Therefore, it is necessary to investigate the effect of normalization on the autocorrelation. Fig. 5A depicts an autocorrelation plot of the penicillin concentration measurement after normalizing the variable-wise unfolded data. As evidenced from this plot, no autocorrelation is filtered out. Variable-wise normalization corresponds to normalizing around the grand mean of each variable over all batches and time points. Therefore, the time dependency remains unchanged.

Fig. 3B depicts the autocorrelation plot of the same variable after normalizing the batch-wise unfolded data. The amount of autocorrelation is reduced significantly compared to the raw data. Normalizing around average profile partly removes the autocorrelation, which corroborates the results of Choi *et al.* [2]. This result underlines the importance of normalizing around the mean trajectory, not only for removal of non-linearity, but also for the reduction of autocorrelation present in the data. The need of an equal amount of autocorrelation present in the data for an honest performance comparison justifies why each PCA-based method in Section 3 begins with a batch-wise normalization.

7. Model identification

To identify each of the PCA-based models on the reference dataset, some parameters have to be determined first. This section discusses the parameter selection methods and results for each approach. Each phase of the fermentation process is modeled with a separate model. The benefits of building multiple phase models instead of a single batch model are shown by many examples. An overview is provided by Yao and Gao [25].

7.1. Batch-wise unfolded multi-way PCA

The data of each phase is unfolded batch-wise and normalized as discussed in Section 3.1. For each model, the only parameter is the PCA model order (i.e., the number of retained principal components). Principal components are added until the adjusted Wold R criterion [26] with a threshold of 0.90

(A)

(B)

Figure 5: Autocorrelation plots of the penicillin concentration variable during the batch phase of (A) the normalized variable-wise unfolded data and (B) the normalized batch-wise unfolded data.

on the fraction of explained variance is reached.

$$\text{If } \frac{\%var_{R+1}}{\%var_R} > 0.9 \quad \text{then} \quad \text{Model order} = R \quad (25)$$

This results in a model order of 5 components for the batch phase and 4 for the fed-batch phase.

The control limit of the T^2 can be readily calculated using Eq. 17. To obtain the remaining control limits a leave-one-out (LOO) crossvalidation procedure is executed [27]. One batch is left out and the model is trained on the remaining 199 batches. Next, the scores, residuals and contributions of the left out batch are calculated as if it were monitored online. These two steps are repeated for each batch in the training set. The SPE statistic is computed for each left out batch to determine the mean μ_k and standard deviation σ_k

of the SPE statistic needed to set the SPE control limits. Based on the contributions to the SPE-static, the mean contributions and their standard deviations are computed to set the contribution control limits. The mean score vector $\boldsymbol{\mu}_{t,k}$, the contributions to the mean score vector $\mu_{t,k,j}$ and the scores covariance matrix $\mathbf{S}_k$ are computed based on the LOO scores. Using the mean and standard deviation of the contributions to the LOO T^2 , the control limits of the T^2 statistic's contributions are determined.

7.2. Variable-wise unfolded multi-way PCA

As in the batch-wise unfolded approach, the data is first normalized by subtracting the average trajectory and scaling to unit variance. Next, the data of each phase is unfolded variable-wise as detailed in Section 3.2 and the corresponding PCA model is identified. Via the adjusted Wold R criterion, 4 principal components are selected for each phase. The control limits are calculated using the LOO procedure described in Section 7.1.

7.3. Batch dynamic PCA

Two parameters are needed for BDPCA: the number of time lags L and the number of principal components R . First, the optimal number of lags L is determined via the method described by Ku et al. [16] which results in 2 lags for the batch phase and 3 lags for the fed-batch phase. Next, the number of retained principal components is selected, again based on the adjusted Wold R criterion. For each phase, 4 components are selected. The control limits are determined using the LOO procedure described in Section 7.1.

7.4. Auto-regressive PCA

In ARPCA three parameters are determined: (i) the model order of the PLS model R_{PLS} , (ii) the number of time lags L and (iii) the model order of the PCA model R_{PCA} . For each combination of L and R_{PLS} , the sum of squared errors (SSE) is calculated using LOO crossvalidation. One batch of the training set is left out, an AR model is trained on the 199 remaining batches and using Eq. 11 the residual SSE of the left out batch is computed. This procedure is repeated for each training batch and the SSE is averaged over the 200 batches.

For each Mag L the model order R is determined using the adjusted Wold criterion on the SSE:

$$\text{If } \frac{\text{SSE}_{R+1}}{\text{SSE}_R} > 0.9 \quad \text{then} \quad \text{Model order} = R \quad (26)$$

Next, the lag L with the lowest SSE is selected. For the batch phase, L and R_{PLS} are 5 and 8, respectively and for the fed-batch phase L is 2 and R_{PLS} is 10.

The order of the PCA model of the AR residuals is selected based on the adjusted Wold criterion resulting in R_{PCA} equal to 3 for the first phase and 4 for the second phase. Control limits are computed using the same LOO procedure detailed in the Section 7.1.

8. Results and discussion

To study the fault detection and diagnosis performance, the 840 faulty batches are monitored with MPCAB, MPCAV, BDPCA and ARPCA after identifying the models on the 200 training batches. As soon as a fault is detected, contribution plots of the out-of-control statistic are generated to assess whether the fault can be diagnosed correctly. The results of the fault detection and fault diagnosis performance comparison are presented and discussed in Sections 8.1 and 8.2, respectively. The clearly different detection and diagnosis behavior shown by the ARPCA approach is further explained in Section 8.3.

8.1. Fault detection

Fault detection performance is characterized by the detection time, i.e., the number of samples between onset and detection of a fault. A fault is detected if either the UCL_{T^2} or UCL_{SPE} is exceeded on three consecutive samples. Thus, the minimum detection time is two samples. The significance level α is set to 1%. The measured false alarm rates, based on the fault-free part of the validation batches, are 0.68%, 0.79%, 0.87% and 0.84% for MPCAB, MPCAV, BDPCA and ARPCA, respectively. The measured false alarm rates are lower than the significance level α . Requiring three consecutive out-of-control samples for an alarm signal and setting α equal to 1% leads to a false alarm rate of $10^{-3}\%$ under the assumption that subsequent samples are independent. However, this assumption does not hold for the investigated approaches as some autocorrelation is still present [2]. Therefore, the measured false alarm rates are closer to (but lower than) 1%. The four methods detect all simulated step faults after two samples. The difference in performance between MPCAB, MPCAV, BDPCA and ARPCA becomes clear when monitoring gradual process disturbances. Mean detection times for gradual faults in the batch and fed-batch phase are presented

in Fig. 6 and Fig. 7, respectively. The length of the error bars equals two times the standard deviation over the 15 faulty batches that were simulated for each combination of faulty variable and fault magnitude.

The detection times shown in Fig. 6 and 7 correspond to the fastest fault detection statistic. The performance of a specific statistic depends on the relative position of the model plane to the fault direction. If the fault direction has a large component along the model hyperplane, the T^2 statistic has a shorter detection time than the SPE statistic. If the fault direction has a large component orthogonal to the model plane, the SPE statistic will be faster. For the MPCAV and BDPCA models, the SPE statistic detects faults on the aeration rate, agitator power and DO sensor on average 17 samples earlier and faults on the feed rate on average 21 samples later than the T^2 statistic. For MPCAB, the SPE is the fastest statistic for all faults (on average 49 samples faster). For ARPCA, the T^2 statistic of the PLS model (T_{AR}^2) has the lowest detection time for drifts on the feed rate, on average 13 and 7 samples earlier than the T^2 and SPE statistics, respectively. The remaining three faults are detected faster by the SPE statistic, on average 11 and 13 samples faster than the T^2 and T_{AR}^2 , respectively.

Fig. 6 shows that ARPCA has lower detection times than MPCAB, MPCAV and BDPCA for the detection of drifts on the aeration rate and agitator power. ARPCA also detects drifts on the DO sensor faster. The logarithmic scale shows that the relative difference in mean detection times between ARPCA and the other methods is smaller as compared to the detection performance of aeration rate and agitator power faults. However, in absolute values the ARPCA approach is on average 35 samples faster than MPCAB, MPCAV and BDPCA in detecting DO sensor drifts. The detection times of MPCAB, MPCAV and BDPCA are comparable for all cases.

The mean detection times in phase 2 (Fig. 7) show a similar trend. ARPCA has lower detection times than the other three methods in detecting gradual faults on aeration rate and agitator power. For the detection of DO sensor faults there is little difference in detection performance, ARPCA is only 2-4 samples faster than the other three methods. This also applies for feed rate faults where the mean detection time for each approach is between 20 and 22 samples and 8 and 11 samples for fault magnitudes of $\pm 0.05\%/h$ and $\pm 0.10\%/h$, respectively. The fault detection performance of BDPCA, MPCAB and MPCAV is also comparable in phase 2.

When examining the fault detection times in Figs. 6 and 7, it is clear that fault detection performance depends on the specific fault. In this case study,

Figure 6: Mean detection times of MPCAB, MPCAV, BDPCA and ARPCA for gradual faults occurring during the batch phase on a logarithmic scale. The length of the error bars equals two times the standard deviation.

ARPCA is superior to MPCAB, MPCAV and BDPCA. ARPCA outperforms the other approaches in detecting step faults and drifts on the aeration rate and agitator power and performs equally well in detecting faults on the DO sensor and feed rate. The performance of the MPCAB, MPCAV, BDPCA approaches are comparable. Preprocessing with an AR model is a better way to incorporate dynamic behavior than adding lag variables to the unfolded matrix as in BDPCA.

8.2. Fault diagnosis

When a fault is detected, contribution plots of the current and two previous time points are studied. Contribution plots show which variables are involved in the detected fault. When a faulty variable is correlated with other variables, a contribution pattern arises.

The simulated faults on the DO are sensor faults and have no influence on other process variables. The aeration rate and agitator power are weakly correlated to other variables so when faulty, the immediate effect on other process variables is small. Thus, for these three disturbance types, no contribution patterns are observed. Faults on the aeration rate, agitator power and DO are correctly diagnosed if the contribution of the faulty variable is the only contribution exceeding its upper control limit.

Figure 7: Mean detection times of MPCAB, MPCAV, BDPCA and ARPCA for gradual faults occurring during the fed-batch phase on a logarithmic scale. The length of the error bars equals two times the standard deviation.

On the other hand, changes in the feed rate have a large impact on the fermentation process. Because substrate is added via the feed, the substrate concentration is heavily influenced by faults on the feed rate. Moreover, an increase or decrease of the substrate concentration in the reactor has a direct impact on the growth rate. The microbial growth is associated with acid production thus the base flow increases or decreases to keep the pH at a constant level. Therefore, the substrate concentration and base flow often exhibit a significant contribution when a disturbance on the feed rate occurs. When diagnosing feed rate faults, a contribution of the substrate concentration or base flow exceeding its upper control limit apart from the feed rate was allowed. An example of a contribution plot diagnosing a gradual feed rate fault is shown in Fig. 8.

Contribution plots are analyzed as soon as a fault is detected to mitigate smearing out of the contributions as the fault propagates through the process. Examining the contributions of previous time points helps in fault diagnosis. This especially rings true when detecting step faults with ARPCA. When the step occurs, there is a clear break in correlation structure. Hence the fault detection statistics based on the AR residuals exhibit high values and clear contribution patterns were observed. However at later time points these patterns become less clear and correct fault diagnosis proves more difficult. Comparing the contributions of the current and two previous time

Figure 8: A typical example of a contribution plot during a feed rate drift. The contributions of the following variables are shown: time, substrate concentration (S), dissolved oxygen concentration (DO), biomass concentration (X), fermentation volume (V), dissolved CO₂ concentration (CO₂), pH, reactor temperature (Tr), generated heat (Q_{rxn}), feed (F), aeration rate (Aer), agitator power (Pw), feed temperature (Tf), cooling water flow (Fw), base flow (Fb) and acid flow (Fa). The solid lines depict the control limits.

points also helps in identifying trends which might provide more certainty about the fault cause. For example an increasing contribution of the substrate concentration is observed in case of a feed rate fault.

The percentage of correctly diagnosed faults by visual examination of the contribution plots is shown in Table 1. For step faults, a high correct diagnosis rate is achieved in all methods. The percentage of correctly diagnosed gradual faults is lower in ARPCA than MPCAB, MPCAV and BDPCA.

The results in Table 1 evidence the practical usefulness of contribution plots for fault diagnosis. Diagnosis of gradual faults is more difficult than diagnosing step faults. Still, more than 90% of the gradual faults is assigned the correct cause in MPCAB, MPCAV and BDPCA. Diagnosis of drifts within the ARPCA approach was somewhat less successful (84% success rate) as contribution plots are more often clouded with contributions of differing variables as compared to MPCAB, MPCAV and BDPCA. Therefore, a multi-model approach in which an ARPCA model is applied for fault detection and a separate MPCAB model for fault diagnosis could combine the fault detection strength of ARPCA with the diagnosis capabilities of MPCAB.

8.3. ARPCA versus BDPCA

The performance comparison results indicate that ARPCA has faster fault detection yet poorer diagnosis properties as compared to the other methods. The difference between BDPCA and ARPCA performance is particularly interesting as both methods assume a constant correlation structure throughout the batch. BDPCA adds lag variables to the variable-wise unfolded data, which means that autocorrelation (correlation between the rows, i.e., different time points) is modeled as a crosscorrelation (correlations between the columns at a fixed time point, i.e., different sensors) between the current and lagged variables. If the crosscorrelations are stronger than the correlation between the current and lag variables, the autocorrelation structure is reflected insufficiently in the PCA loadings. Similarly, the loadings will not reflect the crosscorrelation structure if a very dominant autocorrelation is present. This makes the BDPCA monitoring scheme less sensitive to faults impacting the autocorrelation structure in the former case or to faults impacting the crosscorrelation structure in the latter. If both correlation types are suitably reflected in the PCA loadings, a trade-off is made between the detection speed of both fault types, resulting in a slightly slower detection for both. ARPCA on the other hand, applies a two-step procedure. The focus of the first step is on estimating the autocorrelation structure via an AR model. The second step performs the actual PCA-based data reduction on the residuals. This two-step procedure allows for a more accurate estimation of the correlation structure, resulting in lower detection times when new samples violate the expected autocorrelations.

As is evident from Table 1, ARPCA has a lower fault diagnosis success rate compared to the other methods in detecting ramp faults. When detecting ramp faults with ARPCA, the fastest statistic in almost all cases is the SPE of the PCA model on the AR residuals. As part of the crosscorrelations are inevitably filtered out by the AR model, contributions are more often clouded with contributions of variables unrelated to the detected fault, leading to a decreased fault diagnosis success rate.

Table 1: Fault diagnosis success rates for step and gradual faults.

	MPCAB	MPCAV	BDPCA	ARPCA
Step faults	98%	99%	99%	99%
Gradual faults	96%	92%	94%	84%

9. Conclusions

Batch processes play an important role in the (bio)chemical industry. Online monitoring of the process variables with statistical process control allows for a quick detection of disturbances. This permits timely interventions to bring the final quality within specification bounds or save valuable production time by aborting the process when the batch is beyond recovery. PCA-based techniques reduce process monitoring from monitoring all measurements simultaneously to monitoring a small number of latent variables with fault detection statistics. When a disturbance is detected, the contributions of each process variable to these statistics is computed for fault diagnosis.

Different PCA-based monitoring schemes exist. In this paper the fault detection and fault diagnosis performance of batch-wise unfolded multi-way PCA (MPCAB), variable-wise unfolded multi-way PCA (MPCAV), Auto-Regressive PCA (ARPCA), and Batch Dynamic PCA (BDPCA) have been benchmarked on a simulated fed-batch fermentation process. Fault detection performance was found to depend on the specific fault: ARPCA outperformed MPCAB, MPCAV and BDPCA in detecting gradual faults on the aeration rate and agitator power and performed equally well in detecting drifts on the DO sensor and feed rate.

Examining contribution plots allows for a correct fault diagnosis in virtually all cases ($> 98\%$). When diagnosing gradual faults, diagnosis success rates are still high for MPCAB (96%), MPCAV (92%) and BDPCA (94%), except for ARPCA where only (84%) of the gradual faults were diagnosed correctly. A multi-model approach which employs a separate ARPCA model for fault detection and a separate MPCAB model for fault diagnosis could combine fast fault detection with accurate fault diagnosis.

Acknowledgements

Work supported in part by Projects OT/09/25/TBA and PVF/10/002 (OPTEC Optimization in Engineering Center) of the Research Council of the KU Leuven, Project KP/09/005 (SCORES4CHEM) of the Industrial Research Council of the KU Leuven, and the Belgian Program on Interuniversity Poles of Attraction initiated by the Belgian Federal Science Policy Office. J. Vanlaer and P. Van den Kerkhof are funded by a Ph.D grant of the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen). J. Van Impe holds the chair Safety Engineer-

ing sponsored by the Belgian chemistry and life sciences federation essenscia. The authors assume scientific responsibility.

References

1. Nomikos, P., MacGregor, J.F.. Monitoring batch processes using multiway principal component analysis. *AIChE Journal* 1994;**40**(8):1361–1375.
2. Choi, S.W., Morris, J., Lee, I.B.. Dynamic model-based batch process monitoring. *Chemical Engineering Science* 2008;**63**(3):622–636.
3. Chen, J.H., Liu, K.C.. On-line batch process monitoring using dynamic PCA and dynamic PLS models. *Chemical Engineering Science* 2002;**57**(1):63–75.
5. Jackson, J.E., editor. *A user's guide to principal component analysis*. New York: John Wiley & Sons; 1991.
4. Jolliffe, I.T., editor. *Principal component analysis*. New York: Springer Verlag; 1986.
6. Nomikos, P., MacGregor, J.F.. Multivariate SPC charts for monitoring batch processes. *Technometrics* 1995;**37**(1):41–59.
7. Wold, S., Geladi, P., Esbensen, K., Ohman, J.. Multi-way principal components and PLS analysis. *Journal of Chemometrics* 1987;**1**(1):41–56.
8. Kourti, T.. Multivariate dynamic data modeling for analysis and statistical process control of batch processes, start-ups and grade transitions. *Journal of Chemometrics* 2003;**17**(1):93–109.
9. Camacho, J., Picó, J., Ferrer, A.. Bilinear modelling of batch processes. Part I: theoretical discussion. *Journal of Chemometrics* 2008;**22**(5-6):299–308.
10. Jackson, J., Mudholkar, G.. Control procedures for residuals associated with principal component analysis. *Technometrics* 1979;**21**(3):341–349.

11. Xie, L., Kruger, U., Lieftucht, D., Littler, T., Chen, Q., Wang, S.Q.. Statistical monitoring of dynamic multivariate processes–Part 1. Modeling autocorrelation and cross-correlation. *Industrial & Engineering Chemistry Research* 2006;**45**(5):1659–1676.
12. Camacho, J., Picó, J., Ferrer, A.. The best approaches in the on-line monitoring of batch processes based on PCA: Does the modelling structure matter? *Analytica Chimica Acta* 2009;**642**(1-2):59–68.
13. Nomikos, P., MacGregor, J.F.. Multi-way partial least squares in monitoring batch processes. *Chemometrics and Intelligent Laboratory Systems* 1995;**30**(1):97–108.
14. Arteaga, F., Ferrer, A.. Dealing with missing data in MSPC: several methods, different interpretations, some examples. *Journal of Chemometrics* 2002;**16**(8-10):408–418.
15. Wold, S., Kettaneh, N., Fridén, H., Holmberg, A.. Modelling and diagnostics of batch processes and analogous kinetic experiments. *Chemometrics and Intelligent Laboratory Systems* 1998;**44**(1-2):331–340.
16. Ku, W.F., Storer, R.H., Georgakis, C.. Disturbance detection and isolation by dynamic principal component analysis. *Chemometrics and Intelligent Laboratory Systems* 1995;**30**(1):179–196.
17. Geladi, P., Kowalski, B.R.. Partial least-squares regression: a tutorial. *Analytica Chimica Acta* 1986;**185**:1–17.
18. Tracey, N.D., Young, J.C., Mason, R.L.. Multivariate control charts for individual observations. *Journal of Quality Technology* 1992;**24**(2):88–95.
19. Miller, P., Swanson, R.E., Heckler, C.E.. Contribution plots: a missing link in multivariate quality control. *Applied Mathematics and Computer Science* 1998;**8**(4):775–792.
20. Westerhuis, J.A., Gurden, S.P., Smilde, A.K.. Generalized contribution plots in multivariate statistical process monitoring. *Chemometrics and Intelligent Laboratory Systems* 2000;**51**(1):95–114.

21. Bissessur, Y., Martin, E.B., Morris, J.A.. Monitoring the performance of the paper making process. *Control Engineering Practice* 1999; **7**(11):1357–1368.
22. Zhang, Y., Dudzic, M.S.. Industrial application of multivariate spc to continuous caster start-up operations for breakout prevention. *Control Engineering Practice* 2006;**14**(11):1357–1375.
23. AlGhazzawi, A., Lennox, B.. Monitoring a complex refining process using multivariate statistics. *Control Engineering Practice* 2008; **16**(3):294–307.
24. Birol, G., Ündey, C., Çinar, A.. A modular simulation package for fed-batch fermentation: penicillin production. *Computers & Chemical Engineering* 2002;**26**(11):1553–1565.
25. Yao, Y., Gao, F.. A survey on multistage/multiphase statistical modeling methods for batch processes. *Annual Reviews in Control* 2009; **33**(2):172–183.
26. Li, B.B., Morris, J., Martin, E.B.. Model selection for partial least squares regression. *Chemometrics and Intelligent Laboratory Systems* 2002;**64**(1):79–89.
27. Ramaker, H.J., van Sprang, E.N.M., Westerhuis, J.A., Smilde, A.K.. The effect of the size of the training set and number of principal components on the false alarm rate in statistical process monitoring. *Chemometrics and Intelligent Laboratory Systems* 2004;**73**(2):181–187.