BALANCING PREDICTION QUALITY AND FAULT DETECTION PERFORMANCE IN BATCH PROCESS MONITORING

Geert Gins, Jef Vanlaer, Jan F.M. Van Impe
Chemical and Biochemical Process Technology and Control (BioTeC)
Department of Chemical Engineering, Katholieke Universiteit Leuven

Corresponding author: Jan Van Impe
Chemical and Biochemical Process Technology and Control (BioTeC)
Department of Chemical Engineering, Katholieke Universiteit Leuven
W. de Croylaan 46, B-3001 Heverlee (Belgium)
Email: jan.vanimpe@cit.kuleuven.be

Abstract. In this paper, the performance of four different *Multiway Partial Least Squares* (MPLS) model types is investigated with regards to true online batch-end quality prediction and fault detection. Hereto, a relevant case study of a simulated fed-batch penicillin fermentation process forms the basis of comparison. Based on the extensive simulation results, it is concluded that those model types displaying good batch-end quality prediction typically exhibit a poorer fault detection performance, and vice versa. However, a very well-performing online quality prediction and fault detection model can be obtained by adopting a multi-model approach, in which a phase-wise MPLS with unoptimized input variables is combined with either an optimized phase-wise or full MPLS model. This multi-model approach performs even better than the sum of its parts, as the impact of a process fault on the final product quality can be determined, something not possible with the traditional single-model fault detection methods. As a result, implementation of this multi-model scheme will result in only those specific batches where the disturbance impacts the batch-end quality significantly being corrected and/or terminated. In turn, this leads to a more cost-efficient, cleaner, and safer production process.

1 Introduction

In the chemical and biochemical process industry, batch processes are commonly used for the production of products with a high added value (e.g., medicines, enzymes, high-performance polymers). To achieve a constant and satisfactory product quality, a close online monitoring of these batch processes is necessary. For this reason, multivariate statistical methods have been extended from continuous to batch processes in the context of *Statistical Process Control* (SPC [6, 15, 17]).

Most research effort in this area has been directed towards fault detection and identification using *Principal Component Analysis* (PCA, e.g. [4, 6, 8, 17, 18, 20]) and derived techniques such as *AutoRegressive PCA* (ARPCA [5]) or *Batch Dynamic PCA* (BDPCA [3]). These PCA-based techniques deviations from the normal behavior of the batch process caused by process disturbances by comparing the measured process behavior with an in-control empirical model identified from periods of normal operation [17, 18, 19]. However, they are unable to provide estimates of batch-end quality parameters because they only focus on the measurement (input) space of the process, and disregard the quality (output) space.

(Multiway) Partial Least Squares models ((M)PLS [9]) are capable of making these batch-end quality predictions when required [8, 18, 19, 21]. While Nomikos and MacGregor combine batch-end quality prediction and fault detection in a single MPLS model [18, 19], they use a suboptimal method for obtaining the online estimates, as demonstrated by García-Munoz *et. al* [8]. However, García-Munoz *et. al* only focus on batch-end quality prediction, and ignore fault detection [8]. More recently, Ündey *et. al* have employed PLS-type models for online fault detection [21]. While they also provide estimates of the final quality of a batch fermentation, these estimates are available only at specific times during the operation and are, hence, not truly online.

Therefore, a true online batch-end quality prediction and fault detection scheme based on an MPLS model is investigated in this paper. Four different approaches are tested and compared: (*i*) an MPLS model using all available measurement variables from the complete batch operation as inputs, (*ii*) an (unoptimized) MPLS model with optimized model inputs based on selected measurement variables from the complete batch run, (*iii*) a phase-wise MPLS model with unoptimized inputs, composed of an unoptimized MPLS model for each relevant phase of the batch operation, and (*iv*) a phase-wise MPLS model with optimized inputs. Based on their performance with regards to batch-end quality prediction and fault detection, the optimal MPLS model type is selected.

The structure of this paper is as follows. In Section 2, *Multiway Partial Least Squares* is briefly described. Next, Sections 3 and4 discus the online prediction techniques and fault detection statistics, and the case study in which the MPLS model types are tested is presented in Section 5. Before the performance comparison is presented in Section 7, the investigated MPLS model types are discussed in Section 6. Finally, conclusions are drawn in Section 8.

2 Multiway partial least squares

When historic data from I different batches is available, each consisting of J variables sampled over K time points, the measurement (input) data matrix $\underline{\mathbf{X}}$ has a three-dimensional $I \times J \times K$ structure. The corresponding quality (output) matrix $\mathbf{Y}$, which contains the L corresponding quality measurements for each batch, is of size $I \times L$. In *Multiway Partial Least Squares* (MPLS [19]) this particular data structure is dealt with by reducing the dimensionality of the input matrix $\underline{\mathbf{X}}$ through batch-wise unfolding. This procedure is illustrated in Figure 1: the original data matrix $\underline{\mathbf{X}}$ is divided in K slices of size $I \times J$ which each contain the various measurements for all batches at a single point in time. Next, these time slices are placed side by side, and an unfolded data matrix $\mathbf{X}$ of size $I \times JK$ is obtained [17, 18]. Each row of $\mathbf{X}$ (i.e., each *complete* batch) is then related to its corresponding quality measurements contained in $\mathbf{Y}$ using a standard PLS model [9].

$$\begin{cases} \mathbf{X} = \mathbf{T}\mathbf{P}^T + \mathbf{E}_\mathbf{X} \\ \mathbf{Y} = \mathbf{U}\mathbf{Q}^T + \mathbf{E}_\mathbf{Y} \\ \mathbf{U} = \mathbf{T} + \mathbf{E}_\mathbf{U} \end{cases} \tag{1}$$

In PLS, the in- and output matrices are projected onto a lower-dimensional space, each dimension of which is defined by one of the R latent variables or components. The projection of $\mathbf{X}$ and $\mathbf{Y}$ onto this lower-dimensional space is defined by the model loading matrices $\mathbf{P}$ $(JK \times R)$ and $\mathbf{Q}$ $(L \times R)$ for in- and output space respectively. The scores matrices $\mathbf{T}$ $(I \times R)$ and $\mathbf{U}$ $(I \times R)$ are the resulting approximations of $\mathbf{X}$ and $\mathbf{Y}$ and contain the coordinates (scores) of the original points along each of the R new axes for each of the batches. The matrices $\mathbf{E}$ represent the modelling errors or residuals.

Because $\mathbf{P}$ is not necessarily invertible, a $JK \times R$ weight matrix $\mathbf{W}$ with orthonormal columns is introduced so that $\mathbf{P}^T\mathbf{W}$ is invertible and upper-triangular. The projection of the input space $\mathbf{X}$ onto the scores space $\mathbf{T}$, with corresponding regression matrix $\mathbf{B}$ $(JK \times R)$, is

$$\begin{aligned} &\mathbf{T} = \mathbf{X}\mathbf{B} \\ &\mathbf{B} \overset{\triangle}{=} \mathbf{W}\left(\mathbf{P}^T\mathbf{W}\right)^{-1}. \end{aligned} \tag{2}$$

This leads to the final relation between the measurement (input) variables $\mathbf{X}$ and the quality (output) variables $\mathbf{Y}$.

$$\mathbf{Y} = \mathbf{X}\mathbf{W}\left(\mathbf{P}^T\mathbf{W}\right)^{-1}\mathbf{Q}^T = \mathbf{X}\mathbf{B}\mathbf{Q}^T \tag{3}$$

3 Online scores & batch-end quality estimation

When monitoring one or more running new batches, the future measurements at a given time k are –evidently– unknown. As a result, the developed MPLS model cannot be employed directly because the input matrix $\mathbf{X}_{\text{new}}$ is only partially known (i.e., only the first Jk columns are available). García-Munoz *et. al* demonstrated the superiority of exploiting missing data techniques to compensate for these unknown future measurements without directly estimating the future behavior of the batch under investigation [8]. Of the missing data variants investigated by García-Munoz *et. al*, *Trimmed Scores Regression* (TSR [1, 8]) performs best. Additionally, previous results show that the performance of TSR for batch-end quality prediction is comparable to that obtained by training K new MPLS models, one for each time point at which such a prediction is requested [12]. Therefore, TSR is used in this study for obtaining online MPLS scores and quality estimates. Finally, one of the main advantages of TSR is that a single model can be used at all times of the batch operation.

TSR corrects the estimate of the batch-end quality variables of the N new (running) batches by means of a regression model which compensates for the missing data. At time k, the input matrix $\mathbf{X}_{\text{new}}$ for the N new batches is only partially known. Therefore, $\mathbf{X}_{\text{new}}$ and $\mathbf{B}$ are partitioned in a part corresponding with the known measurements (i.e., $\mathbf{X}_{\text{new},k}$ $(N \times Jk)$ and $\mathbf{B}_k$ $(Jk \times R)$ which contains the first Jk rows of $\mathbf{B}$) and a part corresponding with the unknown, future measurements (i.e., $\mathbf{X}_{\text{new},u}$ $(N \times J(K-k))$ and $\mathbf{B}_u$ $(J(K-k) \times R)$, the matrix containing the last $J(K-k)$

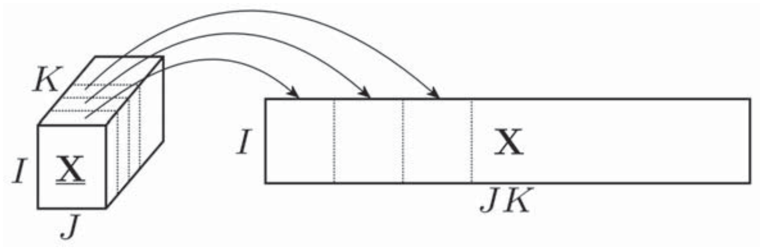

Figure 1: Illustration of data matrix unfolding.

rows of $\mathbf{B}$). Rewriting Equation (2) yields

$$\begin{aligned} \mathbf{T}_{\text{new},k} &= \mathbf{X}_{\text{new}}\mathbf{B} = \left[\mathbf{X}_{\text{new},k}\ \mathbf{X}_{\text{new},u}\right] \begin{bmatrix} \mathbf{B}_k \\ \mathbf{B}_u \end{bmatrix} = \mathbf{X}_{\text{new},k}\mathbf{B}_k + \mathbf{X}_{\text{new},u}\mathbf{B}_u \\ &\overset{\triangle}{=} \mathbf{T}^*_{\text{new},k} + \mathbf{T}^*_{\text{new},u}. \end{aligned} \tag{4}$$

It is evident from this expression that the scores of the new batches can be interpreted as an estimate based on the known samples, $\mathbf{T}^*_{\text{new},k}$ $(N \times R)$, with a correction as a result of the future measurements $\mathbf{T}^*_{\text{new},n}$ $(N \times R)$.

Because $\mathbf{T}^*_u$ is unknown for the running (and incomplete) batches, Equation (4) cannot be used to predict the scores online. Therefore, a *regression* matrix $\mathbf{A}$ $(R \times R)$ is identified to infer the final scores $\mathbf{T}_{\text{new},k}$ from the *trimmed scores* $\mathbf{T}^*_{\text{new},k}$.

$$\hat{\mathbf{T}}_{\text{new},k} = \mathbf{T}^*_{\text{new},k}\mathbf{A} \tag{5}$$

For the set of historic training batches, both $\mathbf{T}_{\text{tr},k}$ and $\mathbf{T}^*_{\text{tr},k}$ are known, and $\mathbf{A}$ is identified through a standard least squares estimation.

$$\mathbf{A} = \left(\mathbf{T}^{*T}_{\text{tr},k}\mathbf{T}^*_{\text{tr},k}\right)^{-1}\mathbf{T}^{*T}_{\text{tr},k}\mathbf{T}_{\text{tr}} \tag{6}$$

Substituting the expression of $\mathbf{A}$ into (5), the expression for the online estimate of new batches' scores is obtained.

$$\hat{\mathbf{T}}_{\text{new},k} = \mathbf{X}_{\text{new},k}\mathbf{B}_k\left(\mathbf{B}_k^T\mathbf{X}_{\text{tr},k}^T\mathbf{X}_{\text{tr},k}\mathbf{B}_k\right)^{-1}\mathbf{B}_k^T\mathbf{X}_{\text{tr},k}^T\mathbf{X}_{\text{tr}}\mathbf{B} \tag{7}$$

This scores estimate for the running batches is used to compute the end quality prediction.

$$\hat{\mathbf{Y}}_{\text{new},k} = \hat{\mathbf{T}}_{\text{new},k}\mathbf{Q}^T = \mathbf{X}_{\text{new},k}\mathbf{B}_k\left(\mathbf{B}_k^T\mathbf{X}_{\text{tr},k}^T\mathbf{X}_{\text{tr},k}\mathbf{B}_k\right)^{-1}\mathbf{B}_k^T\mathbf{X}_{\text{tr},k}^T\mathbf{X}_{\text{tr}}\mathbf{B}\mathbf{Q}^T \tag{8}$$

4 Fault detection statistics

Abnormal batch behavior is detected via *Statistical Process Control* (SPC). The process behavior is characterized by means of an empirical MPLS model, identified on process data from periods of normal behavior. For a new batch, abnormal behavior (i.e., the occurrence of process faults and disturbances) is detected by comparing the measured process behavior with this in-control MPLS model and its statistical properties [17, 18, 19]. Three scalar statistics are employed for this purpose: T^2, *SPE* (Squared Prediction Error), and Q^2.

The T^2-statistic is a measure of the difference between the scores of the historic training set and those of the new batch(es). For a new batch n[1], this statistic is computed as

$$T_n^2(k) = \hat{\mathbf{T}}_{n,k}\Sigma_{\mathbf{T}}^{-1}\hat{\mathbf{T}}_{n,k}^T, \tag{9}$$

where $\hat{\mathbf{T}}_{n,k}$ $(1 \times R)$ is the estimated scores vector for batch n at time k, and $\Sigma_{\mathbf{T}}$ $(R \times R)$ is the scores' covariance matrix, obtained from the training data. The upper control limit u_T, which is constant in time, at a specified tolerance level α is obtained as

$$u_T = \frac{R\left(I^2 - 1\right)}{I(I-R)}F_{(R,I-R;\alpha)}, \tag{10}$$

with $F_{(R,I-R;\alpha)}$ the upper critical value of the F-distribution with R numerator and $I - R$ denominator degrees of freedom and a tolerance α. A too large value for T^2 indicates the scores of the new batch are located outside the normal region observed in the training data. While the identified MPLS model is still valid in this case, the larger scores indicate the process is out of control.

The *SPE* measures the distance between the most recent measurements and the model space, and is equivalent to the instantaneous reconstruction error of the new batch $\mathbf{X}_{n,k}$ $(1 \times Jk)$. Q^2 is the total reconstruction error for all measurements up until time k. Both statistics are derived from the input space residuals vector of the new batch, $\mathbf{E}_{\mathbf{X},n,k}$ $(1 \times Jk)$.

$$\mathbf{E}_{\mathbf{X},n,k} = \mathbf{X}_{n,k} - \hat{\mathbf{T}}_{n,k}\mathbf{P}^T = \mathbf{X}_{n,k}\left(\mathbf{I}_{Jk} - \mathbf{B}_k\left(\mathbf{B}_k^T\mathbf{X}_{\text{tr},k}^T\mathbf{X}_{\text{tr},k}\mathbf{B}_k\right)^{-1}\mathbf{B}_k^T\mathbf{X}_{\text{tr},k}^T\mathbf{X}_{\text{tr}}\mathbf{B}\mathbf{P}^T\right) \tag{11}$$

$$SPE_n(k) = \mathbf{E}^c_{\mathbf{X},n,k}\mathbf{E}^{c\,T}_{\mathbf{X},n,k} \tag{12}$$

$$Q_n^2(k) = \mathbf{E}_{\mathbf{X},n,k}\mathbf{E}^T_{\mathbf{X},n,k} \tag{13}$$

[1] The online scores and quality estimation of Section 3 can be performed for multiple new batches at once. For computing the fault detection statistics, however, this is not possible, and each new batch must be treated individually.

Here, $\mathbf{X}_{n,k}$ contains the known data for running batch n and is equivalent to the n^{th} row of $\mathbf{X}_{\text{new},k}$. The vector $\mathbf{E}^{c}_{\mathbf{X},n,k}$ of size $1 \times J$ contains the most recent values of the J different measurement signals. These are the final J columns of $\mathbf{E}_{\mathbf{X},n,k}$. Both statistics are approximately $g\chi^2_k$-distributed [18]. The parameters g and k match the moments of the distribution to those identified from the training set. This yields the following expressions for the upper control limits:

$$u_{SPE}(k) = \frac{\sigma^2_{SPE}(k)}{2\mu_{SPE}(k)}\chi^2_{\left(2\mu_{SPE}(k)/\sigma^2_{SPE}(k);\alpha\right)} \tag{14}$$

$$u_Q(k) = \frac{\sigma^2_Q(k)}{2\mu_Q(k)}\chi^2_{\left(2\mu_Q(k)/\sigma^2_Q(k);\alpha\right)} \tag{15}$$

The parameters $\mu(k)$ and $\sigma^2(k)$ are the mean and variance of the *SPE* and Q^2 statistics of the training set, and $\chi^2_{(2\mu/\sigma^2;\alpha)}$ is the upper critical value of the χ^2-distribution with $2\mu/\sigma^2$ degrees of freedom and tolerance level α. Because $\mu(k)$ and $\sigma^2(k)$ are time-dependent, the upper control limits for the *SPE* and Q^2 are time-varying. Too large values for *SPE* and/or Q^2 indicate the new data are located far from the MPLS model space. Hence, the system displays a behavior different from that observed in the historic in-control data, and the MPLS model is no longer valid.

5 Case study

The performance of the online MPLS models for batch-end quality prediction and fault detection is tested on a relevant case study: a penicillin fermentation. The data of this biochemical (fed-)batch process is generated using the `Pensim` simulator [2]. A total of 200 batches are simulated. For each batch, the initial substrate, biomass concentration, and culture volume are subject to small variations. In a first phase, the fermenter is operated in batch mode. Once the substrate concentration drops below the threshold value of 0.3 g/L (after approximately 43 hours of operation), the feed flow is initiated. The fermentation is terminated after approximately 460 hours, once 25 L of substrate is added.

During the fermentation, 15 sensors record various concentrations and flows, and the fermenter temperature and pH. The measured signals are aligned and resampled to a length of 602 samples via *indicator variables*, identical to the procedure of Birol *et. al* [2]. Additionally, the transformed time signal is added as an extra (aligned) variable. This results in a data matrix $\underline{\mathbf{X}}$ of size $200 \times 16 \times 602$. After the completion of each batch, the batch-end penicillin concentration is measured and used as quality parameter. This leads to a quality matrix $\mathbf{Y}$ of size 200×1.

Based on the batch recipe, two main phases are identified: (*i*) a batch phase consisting of the first 101 samples, and (*ii*) a fed-batch phase with a length of 501 samples (samples 102 through 602).

To test the fault detection performance, extra –faulty– batches are simulated, with two types of process faults: (*i*) a 20% step decrease in agitator power, and (*ii*) a gradual decrease of 0.05% per hour in the substrate feed rate. Additionally, the faults are introduced at different times during the fermentation: (*i*) halfway throughout the batch phase, (*ii*) at the beginning of the fed-batch phase and after approximately (*iii*) 25%, (*iv*) 50%, and (*v*) 75% completion of the fed-batch phase. Because no feed is present in the batch phase, the feed rate decrease can only take place in the fed-batch phase. This results in a total of 9 combinations of fault and occurrence instant. For each of these combinations, five batches are simulated to obtain a representative data set. This leads to a grand total of 45 faulty batches, aligned and resampled identically to the normal batches of the training set.

6 Model identification and optimal input selection

For the penicillin fermentation process, an MPLS model is identified for predicting the batch-end penicillin concentration based on the 16 available measurements. To avoid overfitting, leave-one-out crossvalidation is used to select the number of latent variables. Instead of taking the number of latent variables corresponding to the observed minimum of the obtained *Sum of Squared Errors* (SSE) curve, an *adjusted Wold's R criterion* with a threshold of 0.9 is used, in accordance to the results of Li *et. al* [16]. This procedure leads to the selection of 9 latent variables for the MPLS model, which has a *rescaled* SSE of $5.27 \cdot 10^{-4}$ (i.e., on $\mathcal{N}(0,1)$ rescaled penicillin concentration values).

Because not all of the 16 available measurement variables are correlated to the final penicillin concentration, a significant amount of "noise" can still be present in the data. By eliminating these noisy measurement signals from the model input set, overfitting will be further reduced. Therefore, a second MPLS model is constructed. The optimal set of input variables is selected by means of a *bottom-up branch-and-bound* procedure. In a first step, each available measurement signal is used as the input for a different MPLS model. Each of these models is leave-one-out crossvalidated, and its performance at the optimal number of latent variables (again, selected with an *adjusted Wold's R criterion* with a threshold on 0.9) is compared with that of the other 15 models. The input

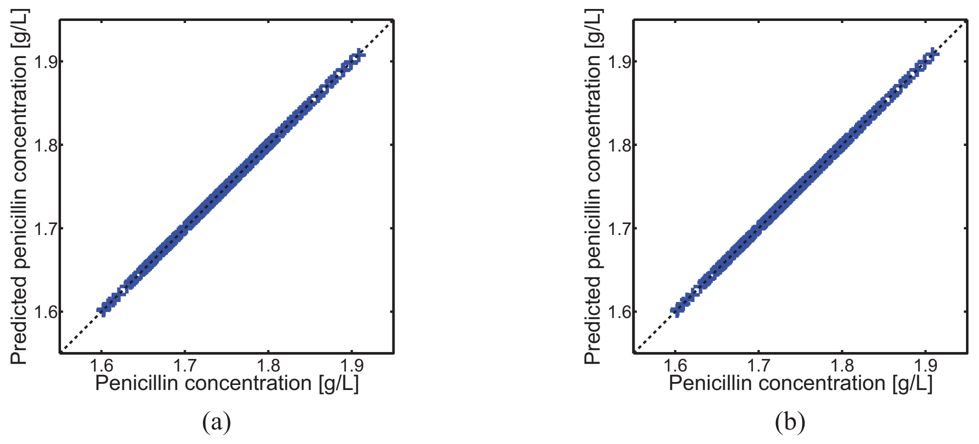

Figure 2: Comparison of measured and predicted penicillin concentrations for (a) the full MPLS model with 16 inputs, and (b) the optimized full MPLS model with 3 inputs. Both models use measurements from a complete fermentation as inputs.

variable of the best-performing single input MPLS model is then selected as the most relevant input. This variable is then combined with each of the remaining ones, resulting in 15 sets of 2 inputs. On each of these sets, a two-input MPLS model is crossvalidated, and the best input set is retained. This procedure is repeated until all 16 available measurements are ranked from most to least important. By comparing the performance of each of these sets as a function of the number of input variables, the optimal set of MPLS input variables is finally obtained. For the penicillin fermentation data in this study, a set of 3 variables yields the best results: (*i*) substrate concentration, (*ii*) biomass concentration, and (*iii*) culture volume. The MPLS model identified on this optimal input set has an optimal number of 9 latent variables; the corresponding rescaled SSE is $3.80 \cdot 10^{-5}$.

As shown in Figure 2, both the unoptimized and optimized model are more than capable of providing accurate estimates of the final penicillin concentration given the measurements for the entire fermentation. While both plots are visually not distinguishable from each other, the optimized input MPLS model nevertheless has an SSE which is approximately 14 times smaller than that of the MPLS model with 16 input variables.

In addition, two phase-wise MPLS models are constructed; one with all 16 available inputs, and one with an optimized input variable set. Each phase-wise models consists of two separate models, one for each phase of the fermentation process. The first submodel takes only measurements from the batch phase (samples 1 to 101), from which it infers the batch-end penicillin concentration. The second submodel provides an estimate of the final penicillin concentration based only on the measurements from the fed-batch phase (i.e., samples 102 to 502).

For the phase-wise MPLS model with 16 input variables, the optimal number of latent variables is determined at 4 for the batch phase and 9 for the fed-batch phase. The SSE of the penicillin concentration prediction is $9.57 \cdot 10^{-3}$ after completion of the (batch) first phase, and $4.31 \cdot 10^{-4}$ after completion of the (fed-batch) second phase. The optimal phase-wise input variable set consists of (*i*) the biomass concentration and (*ii*) the cold water flow rate for the first (batch) phase. For the second (fed-batch) phase of the fermentation, the optimal inputs are identical to those of the full model: (*i*) substrate concentration, (*ii*) biomass concentration, and (*iii*) culture volume. The final optimized phase-wise MPLS model has 3 and 11 latent variables for the batch and fed-batch phase respectively, and exhibits corresponding SSE values of $7.49 \cdot 10^{-3}$ and $2.47 \cdot 10^{-5}$.

7 Results & discussion

The online batch-end quality prediction and fault detection performance is investigated for the four developed MPLS models: (*i*) a full batch MPLS model with unoptimized inputs, (*ii*) a full batch MPLS model with optimized models, (*iii*) a phase-wise MPLS model with unoptimized inputs, and (*iv*) a phase-wise MPLS model with optimized inputs. Hereto, the online model scores and quality prediction is implemented employing TSR, as described in Section 3.

7.1 Batch-end quality prediction

To quantify the end quality prediction performance, each of the four identified MPLS models is used to make online TSR predictions of the final penicillin concentration for each of the available in-control batches. To obtain representative results, a leave-one-out procedure is again adopted: the MPLS models are trained on 199 of the in-control batches, and batch-end quality of the remaining, unused in-control batch is estimated online through TSR. The resulting evolution of the leave-one-out SSE as a function of time for each MPLS model type is presented in Table 1 for a selected number of time instances.

	All inputs		Optimized inputs	
Sample	Full	Phase-wise	Full	Phase-wise
51	$1.23 \cdot 10^{-2}$	$1.04 \cdot 10^{-2}$	$7.86 \cdot 10^{-3}$	$7.90 \cdot 10^{-3}$
101	$1.23 \cdot 10^{-2}$	$9.57 \cdot 10^{-3}$	$7.81 \cdot 10^{-3}$	$7.49 \cdot 10^{-3}$
152	$9.16 \cdot 10^{-3}$	$7.26 \cdot 10^{-3}$	$6.57 \cdot 10^{-3}$	$6.52 \cdot 10^{-3}$
202	$7.89 \cdot 10^{-3}$	$7.43 \cdot 10^{-3}$	$6.73 \cdot 10^{-3}$	$6.57 \cdot 10^{-3}$
252	$7.57 \cdot 10^{-3}$	$7.58 \cdot 10^{-3}$	$6.72 \cdot 10^{-3}$	$6.77 \cdot 10^{-3}$
302	$7.38 \cdot 10^{-3}$	$7.41 \cdot 10^{-3}$	$6.67 \cdot 10^{-3}$	$6.74 \cdot 10^{-3}$
352	$7.32 \cdot 10^{-3}$	$7.36 \cdot 10^{-3}$	$6.55 \cdot 10^{-3}$	$6.53 \cdot 10^{-3}$
402	$7.42 \cdot 10^{-3}$	$7.54 \cdot 10^{-3}$	$6.61 \cdot 10^{-3}$	$6.51 \cdot 10^{-3}$
452	$6.88 \cdot 10^{-3}$	$6.92 \cdot 10^{-3}$	$6.55 \cdot 10^{-3}$	$6.26 \cdot 10^{-3}$
502	$5.46 \cdot 10^{-3}$	$5.34 \cdot 10^{-3}$	$5.45 \cdot 10^{-3}$	$5.45 \cdot 10^{-3}$
552	$4.67 \cdot 10^{-3}$	$4.61 \cdot 10^{-3}$	$4.63 \cdot 10^{-3}$	$4.56 \cdot 10^{-3}$
602	$5.27 \cdot 10^{-4}$	$4.31 \cdot 10^{-4}$	$3.80 \cdot 10^{-5}$	$2.47 \cdot 10^{-5}$

Table 1: Quality prediction SSE of the four investigated MPLS model types as a function of time.

As can be seen from this table, the MPLS models with the optimized input variables consistently exhibit the lowest SSE values. In the initial stages of the fermentation (samples 51 through 152), the SSE of the optimized full MPLS model is approximately 66% of the unoptimized full model. For the phase-wise models, the ratio is 77%. Afterwards, the optimized models' SSE values fluctuate around 90% of the unoptimized models'. Near the end of the batch (samples 452 through 552), both model types perform almost identically. Only at the end of the batch does the major difference in performance become visible, with the optimized models' SSE being only 5-7% of the unoptimized models'.

The difference in prediction quality between the full and phase-wise MPLS models is minimal throughout the course of the batch, as they leapfrog for the lowest SSE. Near the end of the batch, however, the phase-wise models provide more accurate batch-end penicillin concentration predictions.

Based on these observations, it is concluded that the phase-wise MPLS model with optimized input variables yields the best online end quality predictions of the penicillin concentration. However, this model is the most labor-intensive to construct as it requires the identification of 2 separate MPLS submodels (one for each phase of the fermentation) and the selection of the optimal inputs for each submodel. The full MPLS model with optimized input variables necessitates only the identification of a single model. Since it performs nearly as well as the optimized phase-wise model, it is a valid alternative when the process under consideration consists of multiple phases and the identification and and optimization of a phase-wise model would be too time-consuming. Even less time consuming is the construction of the unoptimized models, with the full MPLS model being the fastest and easiest to train. This faster identification comes at the expense –albeit small– of prediction quality.

Hence, it is concluded that, in first instance, the full MPLS model with unoptimized input variables should be used for batch-end quality prediction. As more time is available for model construction, a switch should be made towards the unoptimized phase-wise MPLS model, the optimized full MPLS model, and finally the phase-wise MPLS model with optimized inputs.

7.2 Fault detection

For each of the 45 faulty batches, the three fault detection statistics (T^2, SPE, and Q^2) are computed online. Leave-one-out crossvalidation on the in-control batches yields the scores' covariance matrix $\Sigma_{\mathbf{T}}$, and $\mu_{SPE}(k)$, $\sigma^2_{SPE}(k)$, $\mu_Q(k)$, and $\sigma^2_Q(k)$, the mean and variance of the SPE and Q^2 statistics. These parameters are used to compute the 99.9% control limits. When one of the fault detection statistics exceeds its upper control limit on three consecutive samples, the batch is identified as faulty and an alarm is raised. As a consequence, a minimum delay of 2 samples is required to detect the occurrence of a fault. Table **??** summarizes the results for each of the four studied MPLS model types.

Both models with 16 input variables are able to detect all occurrences of the step-wise agitator power decrease almost instantly, as evidenced by the 100% detection rate and mean delay of 2.04 sample instances. The models with optimized input variables, on the other hand, detect not even 10% of the agitator power drops. Even when this fault is detected, an alarm is raised after only 211 samples for the full MPLS model and after 11 samples for the phase-wise model. This significant difference between the MPLS models with and without input variable optimization is due to the agitator power not being present in the optimal input set. However, the batch-end penicillin concentrations of these faulty batches are found to be located well within the range observed in the historic (in-control) training set. Hence, a 20% drop in agitator power does not seem to have a significant impact on yield of the fermentation process. Only the optimized phase-wise MPLS model displays three false alarms (i.e., an alarm before the true fault occurs, while the fermentation is still in control) in one of the test cases (4%). The

		Agitator power step decrease			Feed rate gradual decrease		
Type	Inputs	Detection rate	Mean delay	False alarms	Detection rate	Mean delay	False alarms
Full	16	100%	2.04	0%	100%	21.05	5%
Full	3	8%	211.00	0%	100%	19.65	5%
Phase-wise	16	100%	2.04	0%	100%	20.55	0%
Phase-wise	2,3	4%	11.00	0%	100%	17.35	15%

Table 2: Fault detection results of the four investigated MPLS model types.

incorrect alarms are once the result of an out-of-control *SPE*, once accompanied by a too high T^2.

The gradual decrease in feed rate is detected by all four model types (100% detection rate in all cases) with a delay of approximately 20 samples in all cases. The models with optimized input variables are slightly faster than those using all available measurements as inputs, and the phase-wise models are faster than the models which take the complete batch as input. This behavior is most likely due to the amount of samples each model takes as inputs: as more input samples are available (i.e., more input variables or longer sample periods are used), the process fault must be bigger to be picked up above the normal process variation. Both full MPLS models exhibit a false alarm on the basis of the *SPE* statistic in exactly one of the 20 test batches (5%) prior to the onset of the true process fault. The phase-wise MPLS model with 16 input variables never falsely identifies normal operation as faulty. Finally, in three of the test cases (15%), an alarm is raised at least once by the phase-wise MPLS model with optimized input variables while the process is still in control (i.e., before the onset of the feed rate decrease). For two batches, the alarms are the result of *SPE* exceeding its upper control limit; T^2 is out-of-control twice for the third batch.

From these results, it is concluded that the phase-wise MPLS model with unoptimized model inputs exhibits the best fault detection behavior, *provided the goal is the detection of all possible disturbances*, followed closely by the unoptimized full MPLS model. The unoptimized phase-wise and full MPLS models rank third and fourth because they fail to detect the sudden drop in agitator power, a fault on one of the measurements not present in their input variables. While reducing the number of input variables of the MPLS model increases the detection speed in some cases, it also prevents the detection of disturbances on the omitted measurement variables. Therefore, if the goal is the detection of all possible disturbances, all available measurements should always be employed. As a consequence, it is recommended that a phase-wise MPLS model with all available measurements as input variables be constructed for optimal fault detection performance. If insufficient time is available, a single MPLS model with all available inputs presents a valid alternative.

If only those disturbances which have a significant impact on the final product quality must be detected, the MPLS models with optimized input variables set yield better results. This is a direct result of their using only those measurement variables with a significant influence on the batch-end quality. In this case, both optimized model types perform nearly identically, with the full MPLS model combining a slightly slower detection speed with a lower false alarm rate.

7.3 Discussion

Based on the observations made in Sections 7.1 and 7.2, it is concluded models with good batch-end quality prediction performance typically display poorer fault detection properties, and vice versa. Hence, selecting the optimal MPLS model type is heavily dependent on the specific type of the SPC application.

For the case study investigated in this paper, the unoptimized phase-wise PLS model is selected as the optimal MPLS model type, balancing a good batch-end quality prediction with excellent fault detection performance. If sufficient time for model construction and identification is available, however, a phase-wise MPLS model with optimal input variables should also be identified. This multi-model approach combines the best of both worlds: the excellent batch-end quality prediction provided by the optimized model and the superior fault detection of the unoptimized model. As a middle ground, the unoptimized phase-wise MPLS model can be combined with an optimized full MPLS model.

Moreover, this multi-model approach displays additional synergy, and allows disturbances which do not have a direct influence on the final product quality to be distinguished from those which do. Thus, in the studied case, the sudden drop in aeration rate is detected but the fermentation is not terminated because this disturbance does not (yet) present a threat to the final penicillin concentration. Maintenance and repairs can, however, already be scheduled to be carried out during the next reactor down-time.

As a result, the general conclusion of this case study is that a multi-model approach for batch-end quality prediction and fault detection performs significantly better than the hitherto proposed single-model techniques. While, the implementation of the proposed multi-model methodology is slightly more time-consuming, the benefits more than outweigh the costs, as fewer, supposedly off-spec, batches will be unnecessarily corrected or terminated. In turn, this will lead to lower operational costs, lower raw material and energy consumption, higher process efficiency and safer operation.

8 Conclusions & future work

In this paper, the performance of four different *Multiway Partial Least Squares* (MPLS) model types has been investigated with regards to online batch-end quality prediction and fault detection on a relevant case study of a simulated fed-batch penicillin fermentation process. The first type of MPLS model takes all available measurements from a complete batch run as input variables, while the second type uses a reduced (optimized) set of input variables, in which only those measurement variables with a significant influence on the batch-end quality are retained. The third MPLS model type is of the phase-wise genre, and is composed of one submodel for each phase of the batch process under study. Finally, the fourth type of MPLS model is phase-wise with optimized input variables.

Based on the extensive simulation results, it has been concluded that the MPLS model types displaying good batch-end quality prediction typically exhibit a poorer fault detection performance, and vice versa. By combining a phase-wise MPLS model with unoptimized input variables with either an optimized phase-wise or full MPLS model in a multi-model methodology, however, the best of both worlds can be achieved. This multi-model approach even exhibits an additional synergy, as a distinction can be made between process faults which do not have a direct impact on product quality and those which do, something not possible with the traditional single-model fault detection methods. As a result, only those specific batches in which the disturbance impacts the batch-end quality significantly will be corrected and/or terminated. For non-critical process disturbances, maintenance can be scheduled for the next reactor down-time. This will eventually lead to a more efficient use of the available raw materials and energy, and reduce the downtime between subsequent batch runs. This increased production efficiency results in a more profitable and safer operation.

Future research will focus on comparing the studies MPLS model types for batch-end quality prediction and fault detection on other kinds of data, both simulated and industrial. Furthermore, additional fault detection methodologies, such as *AutoRegressive Principal Component Analysis* [5], *Batch Dynamic Principal Component Analysis* [3] or *Batch Dynamic Partial Least Squares* [3], will be investigated.

Acknowledgements

Work supported in part by Projects OT/03/30 and EF/05/006 (Center-of-Excellence Optimization in Engineering) of the Research Council of the Katholieke Universiteit Leuven, Leverage Project IOFHBKP/06/002 (SCORES4CHEM) of the Industrial Research Council of the Katholieke Universiteit Leuven, and the Belgian Program on Interuniversity Poles of Attraction, initiated by the Belgian Federal Science Policy Office. The scientific responsibility is assumed by its authors.

The authors would like to thank J. De Wilde of CYTEC (Drogenbos, Belgium) for the industrial polymerization data set provided for this study.

9 References

[1] F. Arteaga and A. Ferrer. Dealing with missing data in MSPC: several methods, different interpretations, some examples. *J. Chemom.*, 16:408-418, 2002.

[2] G. Birol, C. Ündey, and A. Çinar. A modular simulation package for fed-batch fermentation: penicillin production. *Comput. Chem. Eng.*, 26:1553-1565, 2002.

[3] J. Chen and K.-C. Liu. On-line batch process monitoring using dynamic PCA and dynamic PLS models. *Chem. Eng. Sci.*, 57:63-75, 2002.

[4] S.W. Choi, E.B. Martin, A.J. Morris, and I.-B. Lee. Fault detection based on a maximum likelihood PCA mixture. *Ind. Eng. Chem. Res.*, 55(7):2316-2327, 2005.

[5] S.W. Choi, E.B. Martin, A.J. Morris, and I.-B. Lee. Dynamic model-based batch process monitoring. *Chem. Eng. Sci.*, 63(3):622-636, 2008.

[6] L. Eriksson, E. Johansson, N. Kettaneh, and S. Wold. *Multi- and Megavariate Data Analysis: Principles and Applications* Umetrics Academy, 91-973730-1-X, 2002.

[7] S. García-Munoz, T. Kourti, J.F. MacGregor, A.G. Mateos, and G. Murphy. Troubleshooting of an industrial batch process using multivariate methods. *Ind. Eng. Chem. Res.*, 42:3592-3601, 2003.

[8] S. García-Munoz, T. Kourti, and J.F. MacGregor. Model predictive monitoring for batch processes. *Ind. Eng. Chem. Res.*, 43:5929-5941, 2004.

[9] P. Geladi and B.R. Kowalski. Partial least-squares regression: a tutorial. *Anal. Chim. Acta*, 185:1-17, 1986.

[10] G. Gins, J. Espinosa, I.Y. Smets, W. Van Brempt, and J.F. Van Impe. Data alignment via dynamic time warping as a prerequisite for batch-end quality prediction. P. Perner (Ed.), *Lect. Not. Artif. Intell.*, 4065:506-510, 2006.

[11] G. Gins. *Modelling of (bio)chemical processes using data-driven techniques*. PhD thesis, Faculteit Ingenieurswetenschappen, Katholieke Universiteit Leuven, 2007.

[12] G. Gins, J. Vanlaer, and J.F. Van Impe. Online batch-end quality estimation: does laziness pay off? *(submitted)*, 2008.

[13] A. Kassidas, J.F. MacGregor, and P.A. Taylor. Derivative dynamic time warping. *AIChE J.*, 44(4):864-875, 1998.

[14] E.J. Keogh and M.J. Pazzani. Synchronization of batch trajectories using dynamic time warping. *SIAM Int. Conf. Data Mining, (Chicago, IL)*, 2001.

[15] J.H. Lee and A.W. Dorsey. Monitoring of batch processes through state-space models. *AIChE J.*, 50(6):1198-1210, 2004.

[16] B. Li, J. Morris, and E.B. Martin. Model selection for partial least squares regression. textitChemom. Intell. Lab. Syst., 64:79-89, 2002.

[17] P. Nomikos and J.F MacGregor. Monitoring batch processes using multiway principal component analysis. *AIChE J.*, 40(8):1361-1375, 1994.

[18] P. Nomikos and J.F. MacGregor. Multivariate SPC charts for monitoring batch processes. *Technometrics*, 37(1):41-59, 1995.

[19] P. Nomikos and J.F. MacGregor. Multiway partial least squares in monitoring batch processes. *Chemom. Intell. Lab. Syst.*, 30:97-108, 1995.

[20] A. Simoglou, P. Georgieva, E.B. Martin, A.J. Morris, and S.F. de Azevedo. On-line monitoring of a sugar crystallization process. *Comput. Chem. Eng.*, 29(6):1411-1422, 2005.

[21] C. Ündey, S. Ertunç, and A. Çinar. Online batch/fed-batch process performance monitoring, quality prediction, and variable-contribution analysis for diagnosis. *Ind. Eng. Chem. Res.*, 42:4645-4658, 2003.

[22] S. Wold, P. Geladi, K. Ebensen, and J. Öhman. Multi-way principal components- and PLS-analysis. *J. Chemom.*, 1(1):41-56, 1987.