Nearest-Neighbor Method for the Automatic Maintenance of Multivariate Statistical Soft Sensors in Batch Processing

ARTICLE *in* INDUSTRIAL & ENGINEERING CHEMISTRY RESEARCH · MARCH 2010

Impact Factor: 2.59 · DOI: 10.1021/ie9013919

3 AUTHORS, INCLUDING:

Pierantonio Facco
University of Padova
39 PUBLICATIONS **313** CITATIONS

SEE PROFILE

Fabrizio Bezzo
University of Padova
104 PUBLICATIONS **1,331** CITATIONS

SEE PROFILE

Nearest-Neighbor Method for the Automatic Maintenance of Multivariate Statistical Soft Sensors in Batch Processing

Pierantonio Facco,* Fabrizio Bezzo, and Massimiliano Barolo*

Computer-Aided Process Engineering Laboratory (CAPE-Lab), Dipartimento di Principi e Impianti di Ingegneria Chimica, Università di Padova, via Marzolo 9, 35131 Padova PD, Italy

Soft sensors based on multivariate statistical models are used very frequently for the monitoring of batch processes. From the moment of model calibration onward, the model is usually assumed to be time-invariant. Unfortunately, batch process conditions are subject to several events that make the correlation structure between batches change with respect to that of the original model. This can determine a decay of the soft sensor performance, unless periodic maintenance (i.e., updating) of the model is carried out. This article proposes a methodology for the automatic maintenance of PLS soft sensors in batch processing. Whereas the adaptation scheme usually follows chronological order in classical recursive updating, the proposed strategy defines the reference data set for model recalibration as the set of batches (nearest neighbors) that are most similar to the currently running batch. The nearest neighbors to a running batch are identified during the initial evolution of the batch following a concept of proximity in the latent space of principal components. In this way, for any new batch to be run, a model can be tailored on the running batch itself. The effectiveness of the proposed updating methodology is evaluated in two case studies related to the development of adaptive soft sensors for real-time product quality monitoring: a simulated fed-batch process for the production of penicillin and an industrial batch polymerization process for the production of a resin.

1. Introduction

Multivariate statistical methods based on principal component analysis[1] (PCA) and partial least-squares (PLS) regression[2] have long been used for process monitoring.[3,4] Both methods are correlative representations of massive volumes of highly collinear and noisy (process) variables that are examined by projecting them onto a subspace made of a few fictitious variables, called principal components (PCs) or latent variables (LVs). Originally proposed for continuous process monitoring, these methods have been shown to be extremely powerful for batch process monitoring as well.[5,6]

One of the most popular uses of multivariate statistical methods is soft sensing,[7−9] and also in this application, a model is designed using a set of available reference process and quality data in order to estimate the quality variables in real time from online process measurements. From the moment of model calibration onward, a multivariate statistical model is assumed to be time-invariant. From a batch processing perspective, this amounts to assuming that each new incoming batch belongs to the same distribution of the reference data set and that the batch-to-batch variability is within the range of the variability already retained in the model.[10] Unfortunately, batch process conditions are subject to several events that make the correlation structure between batches move away from the conditions that characterized the structure of the original model. For instance, abrupt process variations might be noticed when raw materials suppliers are changed or when the processing equipment undergoes maintenance. Changes are also experienced when slow phenomena take place, such as seasonal drifts, catalyst deactivation, equipment fouling, and contamination. Furthermore, a flexible production strategy might dictate slight but frequent variations in the recipe and in the production grades to satisfy market requirements, and this impacts the trends of the process variables. Therefore, because of process/plant changes, the original reference data set might progressively lack representativeness. This can determine a decay in the soft sensor performance, unless periodic maintenance of the multivariate statistical model is carried out.

The key idea of an adaptive methodology for model maintenance is to periodically recalibrate the model, that is, to update its parameters in response to the new process/plant conditions. An adaptive strategy should be robust to outliers, effective in dealing with both sudden and slow variations, and computationally efficient. The basic adaptation strategy is to include within the model structure those available new data that are representative of the incoming changes. Typically, a recursive approach is used, wherein the mean, variance, and covariance of the changing data and the confidence limits of the model are updated.[11,12] Several enhancements to standard recursive updating have been proposed. For example, moving windows can be used to avoid the accumulation of obsolete data,[11] whereas the use of forgetting factors allows more importance to be given to the most recent data compared to the oldest data.[13] Variable forgetting factors[13,14] or variable sizes of the moving window[15] can speed the adaptability of the model in case of sudden changes and limit the model adaptation when slow variations occur. The adaptation algorithm can be improved to reduce the computational load and relieve the memory of the process computers from the storage of data and improve the computational efficiency.[16,17] The robustness to outliers can be improved, in such a way as to avoid that data in abnormal conditions distort the model.[18−20] Finally, recent issues in terms of model maintenance for batch processes relate to the selection of the optimal number of reference batches, the selection of an optimal structure for the monitoring models, and the decision of when the model should be updated for a rapid but parsimonious adaptability.[21−23]

Recursive adaptation approaches assume that the sequential order in which the batches are run also determines a sort of similarity between batches; that is, the batches that are closest in time to the current batch are assumed to be the most similar

* To whom correspondence should be addressed. E-mail: pierantonio.facco@unipd.it (P.F.), max.barolo@unipd.it (M.B.).

10.1021/ie9013919
Published on Web 02/02/2010

to the current batch. As a consequence, these batches represent the most appropriate references to be used for model calibration. This assumption is likely to hold when batch-to-batch variations are slow. However, whenever process changes are not gradual (e.g., because of changing operating policies), the conjecture that temporal proximity is also a measurement of similarity is often invalid.

This article proposes a methodology for the automatic maintenance of PLS soft sensors in batch processing. The proposed approach updates the estimation model following a nearest-neighbor strategy, which determines the most appropriate reference data set for model recalibration by inspecting the similarity between batches, regardless of the chronological order in which the data have been collected. The performance of the proposed adaptation strategy is compared to a state-of-the-art moving-window updating method. The methodology is tested for two case studies related to the development of adaptive soft sensors for real-time product quality monitoring: a simulated fed-batch process for the production of penicillin and an industrial batch polymerization process for the production of a resin.

2. The Processes

Two processes are used in this work as case studies to illustrate the proposed procedure for soft sensor updating. Both processes have already been presented in the literature, and only some basic features will be summarized herein.

2.1. Industrial Batch Polymerization Process. The first process is described in refs 24 and 25. It consists of a batch catalytic polycondensation between carboxylic polyacids and polyalcohols in a 12-m^3 reactor for the production of a polyester resin for coating applications. Product quality measurements are performed offline at a low and uneven frequency (about one measurement every 2 h). Therefore, the objective is to design a soft sensor that is able to estimate in real time two product quality indices (namely, the resin viscosity μ and the resin acidity number N_A) using available process measurements. The measurements of 34 process variables are collected by online sensors and recorded by the process computer every 30 s.

The process recipe evolves through a complex and mostly manually driven sequence of operating steps, and the batches exhibit almost unpredictable batch-to-batch variations, also due to the fact that the same pieces of equipment are used to produce several different products. For an idea of the batch variability, consider that, within the database of 72 batches used in this work (which correspond to ~2.5 years of operation), the batch length ranges between 30 and 60 h.

For the purpose of product quality estimation, the complex series of operating steps can be simplified to a sequence of three estimation phases. Within each estimation phase, a moving-average PLS soft sensor is used to provide a real-time estimation of the quality indices.[24] Variable-wise unfolding is used for model development.

2.2. Simulated Fed-Batch Fermentation Process. The second benchmark is a fed-batch fermentation process for the production of penicillin. The process is described in ref 26 and has been the object of several studies in the field of multivariate statistical techniques.[27−30] Basically, the operation goes through two operating stages. The first one is a batch phase where the cell mass grows, consuming the initial substrate. In the second stage, the substrate is fed continuously to allow penicillin production. Penicillin concentration measurements are performed once every 2 h. In this work, the objective is to estimate the penicillin concentration in real time using 15 process measurements (Table 1), which are measured every 0.02 h.

Table 1. List of Measured Process Variables in the Fermentation Process for the Production of the Penicillin

process variable	units
aeration rate	L/h
agitator power	W
substrate feed rate	L/h
substrate feed temperature	K
substrate concentration	g/L
dissolved oxygen concentration	%
culture volume	L
CO_2 concentration	mmol/L
pH	
temperature	K
generated heat	kcal/h
acid flow rate	mL/h
base flow rate	L/h
cooling water flow rate	L/h
heating water flow rate	L/h

A database consisting of 79 equal-length (400-h) batches was created. To include different sources of variability, both random variability and special-cause variability were introduced through several inputs. Random input variability was simulated by adding zero-mean, uniform random numbers to the nominal inputs, with maximum variations with respect to the reference input values as indicated in the last column of Table 2. Special-cause variations (accounting for changes in the operating conditions, sensor drifts, and maintenance interventions) were introduced by changing the value of one parameter (first column of Table 2) from its reference value (second column of Table 2) to a modified value (third column of Table 2). This change was carried out at the beginning of an assigned batch (indicated in the fourth column of Table 2) and then maintained throughout all of the remaining batches.

An evolving PLS soft sensor was used for the real-time estimation of the penicillin concentration. The soft sensor model has a multimodel structure,[31] in which one model is designed at every time instant: Every time the process variables are measured, the soft sensor uses all of the past history of the batch as an input to the soft sensor. Batch-wise unfolding is used for model development.

3. Recursive Model Adaptation

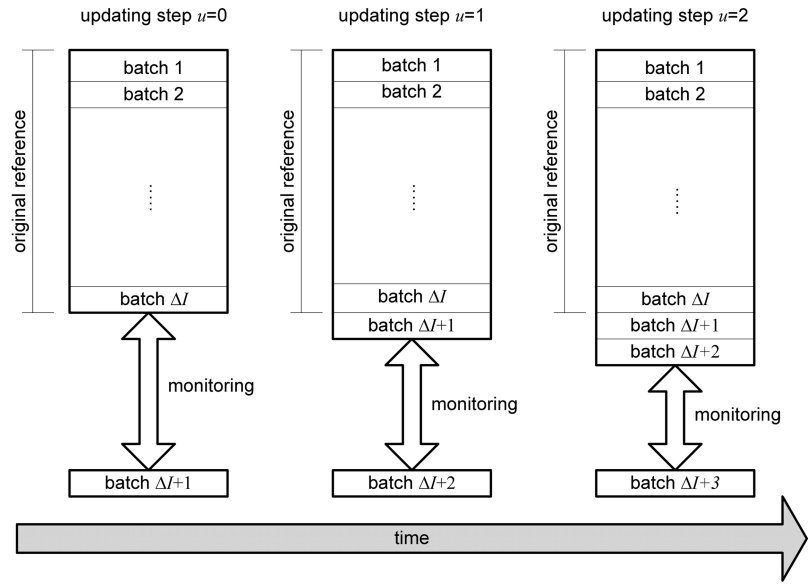
In this section, a short description of standard recursive techniques for model updating is presented. Furthermore, a method is proposed to determine the effective dimension, ΔI, of the reference database used for model recalibration, where ΔI is the number of completed batches that are used to calibrate the model.

Batch process data can be collected in a three-dimensional array $\underline{\mathbf{X}} = \{x_{i,j,k}\}$ $[I \times J \times K_i]$ of J variables collected on K_i sampling instants for I batches whose sequence follows a time order (with batch 1 being the oldest one). This array might be of irregular shape because of different lengths of the batches. However, if the batches are synchronized, then $K_i = K$ is found for any batch. Data on M quality indices are collected in a three-dimensional array $\underline{\mathbf{Y}}$ $[I \times M \times H_i]$, in which the sampling rate of the quality is usually much lower than the sampling rate of the process variables ($H_i \ll K_i$) and the sampling frequency might be uneven. Multiway statistical methods can be applied to three-dimensional arrays provided that $\underline{\mathbf{X}}$ and $\underline{\mathbf{Y}}$ are unfolded to a proper bidimensional shape. Variable-wise unfolding[32] (VWU) and batch-wise unfolding[5] (BWU) perform a deployment of the original matrix that allows different types of variability to be captured.[6]

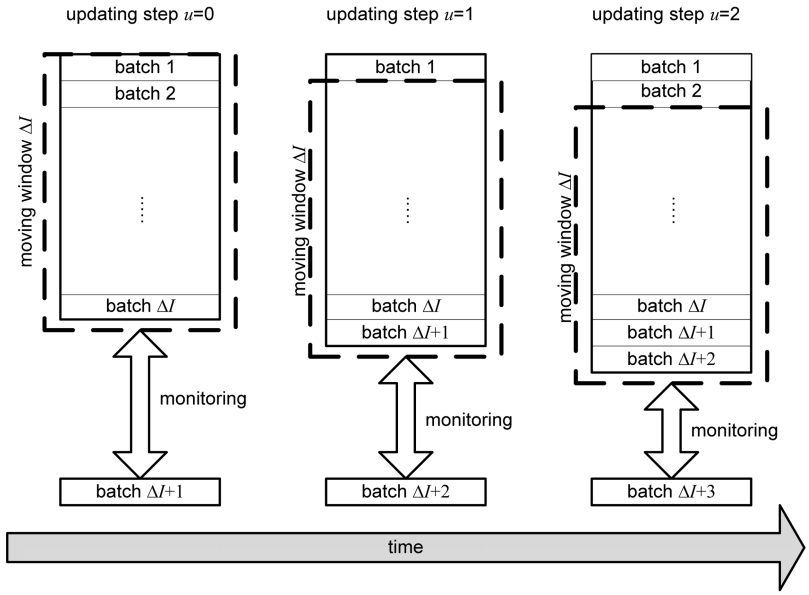
Table 2. Simulated Fed-Batch Process: Summary of the Variability Introduced through the Process Parameters[a]

input variable	reference value	modified value	first batch of occurrence of modification	maximum variation due to noise
substrate concentration (g/L)	25	35	35	±5
pH	5	$5 + (i - 50)/60$	50	±0.05
aeration rate (L/h)	$8.6 \sin[40 \times (1 - i)/54 + 90]$	$8.6 \sin[2 \times (55 - i)/5 + 90]$	55	±0.05
agitator power (W)				
for $i < 40$	30	35	40	−
for $i \geq 40$	35	40	60	−
biomass concentration (g/L)	0.1	−	−	±0.099
culture volume (L)	100.5	−	−	±0.5
CO_2 concentration (mmol/L)	0.75	−	−	±0.2
substrate feed rate (L/h)	0.04	−	−	±0.05

[a] i indicates the batch number, in chronological order.



(a)



(b)

Figure 1. Basic adaptive algorithms for the updating of multivariate statistical models: (a) adaptive procedure, (b) adaptive procedure on a moving window.

The basic scheme of a recursive procedure for model adaptation is based on the inclusion of the last completed batch within the set of reference batches (Figure 1a). This strategy uses all of the batches from the very initial one to the last completed one as the available reference database: at each updating step u, the dimension of the reference data set

increases, and the computational burden becomes heavier. Moving-window (MW) updating (Figure 1b) couples the updating of the new available data to the discounting of the oldest ones: every time a batch is completed, the relevant set of data is included in the reference data set, and the set of data related to the oldest batch included in the reference set is discarded at the same time. Therefore, in MW adaptation, the soft sensor calibration is carried out on the basis of ΔI consecutive previously completed batches (how this number can be determined is discussed later on). Note that, for ease of discussion, in the remainder of the article, it will be assumed that the model adaptation is carried out at the beginning of any new batch, although, in principle, this large updating frequency might not always be necessary.

With reference to the MW adaptation scheme, the uth updating step is carried out by calibrating the model on the last ΔI completed batches in order to provide product quality estimates for batch $i = u + 1 + \Delta I$. The reference data set for the monitoring of batch i at the updating step u is therefore

$$\mathbf{X}_u^{\mathrm{VWU}} = \begin{bmatrix} \mathbf{X}_{i-\Delta I} \\ \mathbf{X}_{i-\Delta I+1} \\ \vdots \\ \mathbf{X}_{i-1} \end{bmatrix} \tag{1}$$

in the case of VWU and

$$\mathbf{X}_u^{\mathrm{BWU}} = \begin{bmatrix} \mathbf{x}_{i-\Delta I}^{\mathrm{BWU}} \\ \mathbf{x}_{i-\Delta I+1}^{\mathrm{BWU}} \\ \vdots \\ \mathbf{x}_{i-1}^{\mathrm{BWU}} \end{bmatrix} \tag{2}$$

in the case of BWU, where ΔI is the size of the MW in terms of number of batches, $\mathbf{X}_i$ is the ith horizontal slice of $\underline{\mathbf{X}}$, and $\mathbf{x}_i^{\mathrm{BWU}}$ is a row vector obtained by transforming the matrix $\mathbf{X}_i$ by BWU, namely

$$\mathbf{X}_i = \begin{bmatrix} x_{i,1,1} & x_{i,2,1} & \dots & x_{i,J,1} \\ x_{i,1,2} & x_{i,2,2} & \dots & x_{i,J,2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{i,1,K} & x_{i,2,K} & \dots & x_{i,J,K} \end{bmatrix} \xrightarrow{\mathrm{BWU}} \mathbf{x}_i^{\mathrm{BWU}} \tag{3}$$

where

$$\mathbf{x}_i^{\mathrm{BWU}} = [x_{i,1,1} \;\; x_{i,2,1} \;\; \dots \;\; x_{i,J,1} \;\; \dots \;\; x_{i,1,K} \;\; x_{i,2,K} \;\; \dots \;\; x_{i,J,K}] \tag{4}$$

These matrices are the reference data set for model updating at step u. Note that the mean, variance, and covariance matrix and the confidence limits need to be updated as well.[16,33]

3.1. Effective Dimension of the Reference Database for Model Recalibration. In MW adaptation schemes, the ΔI previously completed batches that are closest in time to the current batch are regarded as the most similar to the current batch, and soft sensor adaptation is therefore carried on the basis of these batches. Hence, ΔI represents the effective dimension of the reference database for model recalibration. Few systematic procedures for the selection of the optimal number of reference batches are available in the literature.[15] In this subsection, a heuristic approach is proposed to help in the selection of the most appropriate size of the MW with the purpose of model recalibration.

The window size is a parameter that tunes the speed of adaptation of the soft sensor model to process changes.[17] Large window sizes mean that the model relies heavily on past batches and therefore provides the model with high robustness; however, this also implies scarce sensitivity to changes (high inertia). With short window sizes, the model adapts quickly to the new conditions, but small values of ΔI might not guarantee that the information retained in the model is adequate. Generally speaking, large windows are more useful when slow process variations are experienced, whereas small windows fit the case of sudden changes. However, excessively small windows are never recommended, as the model becomes oversensitive to new data and can suffer from reliability and robustness issues.

3.1.1. Procedure for the Determination of the Window Size. Whenever a change in the variability of the data occurs, this impacts the parameters that characterize a model. Therefore, the appropriateness of a given window size can be assessed by observing how the model parameters change in response to the updating for different lengths, ΔI, of the moving window. The most significant information in a PLS model is related to the displacement of the director cosines of the directions of maximum variability of the input data (i.e., the latent variables). Therefore, the loadings $\mathbf{P}_u$ and the weights $\mathbf{W}_u$ of the recursively updated model at the updating step u (i.e., corresponding to batch $i = u + 1 + \Delta I$) can be analyzed to understand how the subspace of the latent variables changes. Additionally, further information on the model updated at step u is provided by the confidence limit, $T_{u,\mathrm{lim}}^2$, on the Hotelling statistics and the confidence limit, $q_{u,\mathrm{lim}}$, on the squared predicting error.

For an assigned value of ΔI, the procedure assessing the model movement in the score space in response to the updating goes through the following steps:

1. Compute the PLS model at each updating step u [note that the first (original) model is set up at the updating step $u = 0$].
2. Extract the loading matrix $\mathbf{P}_u$ [$A \times J$] and the weight matrix $\mathbf{W}_u$ [$A \times J$] at each step u.
3. Place the loading matrices of all of the $I - \Delta I + 1$ models obtained for each updating step u in horizontal position, one underneath the other, to build a three-dimensional array [$(I - \Delta I + 1) \times A \times J$] of [models × loadings × variables] (Figure 2a).
4. Place the weight matrices of all of the $I - \Delta I + 1$ models obtained for each updating step u in horizontal position, one underneath the other to build a three-dimensional array [$(I - \Delta I + 1) \times A \times J$] of [models × weights × variables] (Figure 2b).
5. Unfold both arrays model-wise (i.e., the same as BWU) in such a way as to describe each model with a row vector of loadings and a row vector of weights (Figure 2c,d).
6. For each model, place the row vector of the unfolded loadings and that of the unfolded weights side by side, together with the respective values of the confidence limits $T_{u,\mathrm{lim}}^2$ and $q_{u,\mathrm{lim}}$; repeat this for all models and stack each row one beneath the other to form a matrix of models (Figure 2e).
7. Perform PCA on this matrix and observe the movements of the models in the score plot.

In summary, every model is identified by a sequence of loadings, weights, and confidence limits, and analysis of the behavior of the models becomes a typical multivariate problem. PCA checks the behavior of the models by projecting the model parameters onto a subspace of two PCs. The score point movements identify changes in the models due to the updating, which reflect changing process/plant conditions. If the score points are spread out all over the score plot, then the distribution

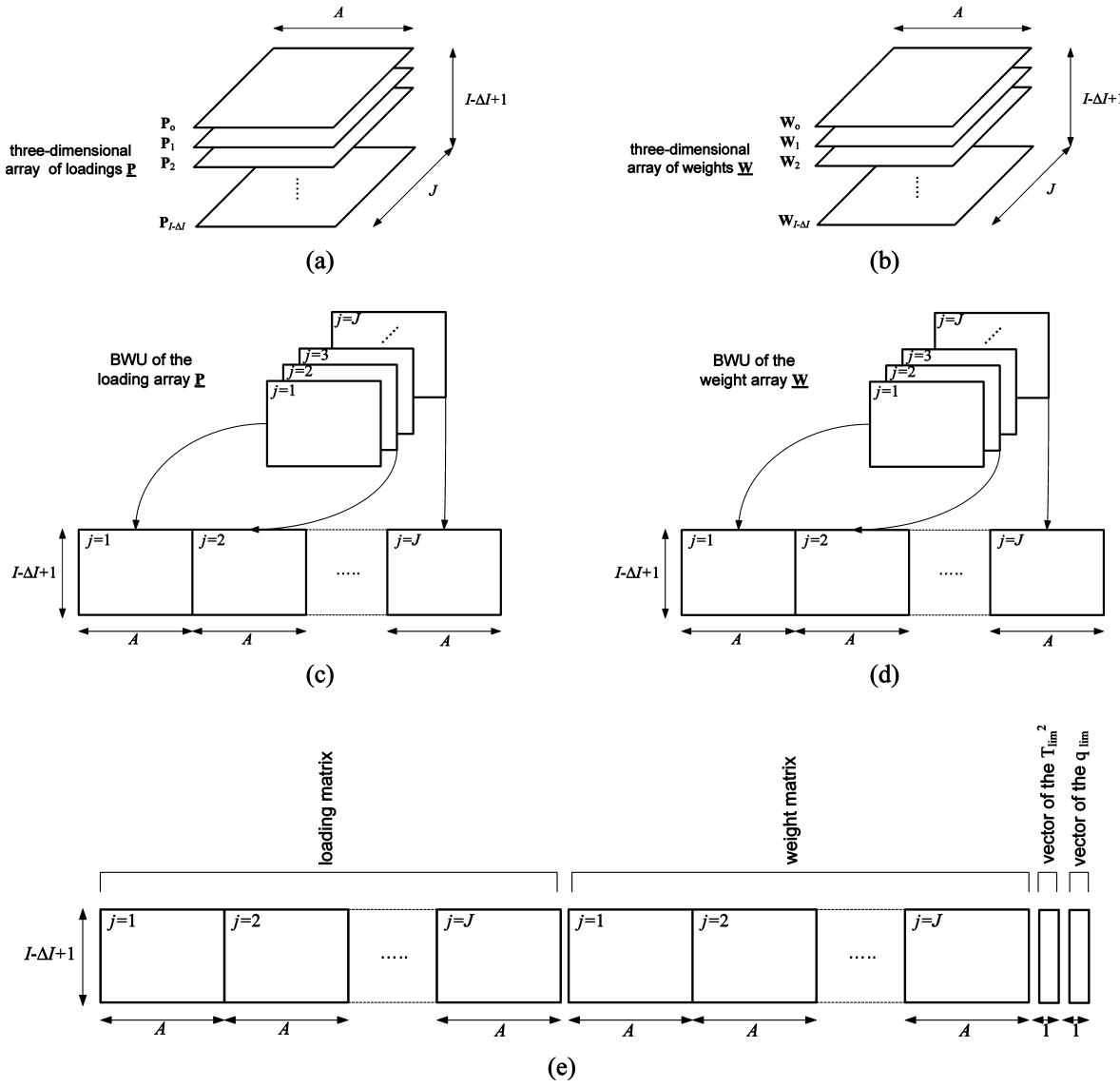


Figure 2. Schematic of the procedure for the determination of the window size in moving-window recursive updating: (a,b) three-dimensional arrays of loadings and weights are built for the models related to each updating step; (c,d) the arrays are unfolded model-wise into two-dimensional matrices; and (e) these matrices are placed side by side, together with the two column vectors of limits on model statistics.

of the models is likely to be normal. This means that even one single model update (i.e., a difference of one single batch within the MW set of reference batches) can change the model completely, indicating that the window size is probably too small and the model is too sensitive to process variations. Conversely, if the distribution of the score points shows an evident autocorrelation in the score plot, the MW size is probably too large for the model to adapt to the new process conditions in a sufficiently rapid manner; that is, the model is not sufficiently responsive to process changes.

3.1.2. Example. The industrial batch polymerization process is taken as an example to show how the window size can be determined for an MW adaptation scheme. The PLS model (e.g., for the estimation of acidity number during estimation phase 1) was updated with three different sizes of the MW ($\Delta I = 10$, 30, and 50 batches), and the behavior of the model throughout the updating was studied using the score plots of Figure 3. In parts a−c of Figure 3, the relationship between the updated models is represented when the MW size is $\Delta I = 10$, 30, and 50 batches, respectively. Here, each model is identified by the number of the batch i that is being monitored at the current updating step for the selected width of the MW. For example, in the models with a reference set of 10 batches (Figure 3a), model 11 is the one that is assigned to monitor batch 11 and is built on reference batches 1−10; model 12 monitors batch 12 and is built on batches 2−11; and so on.

In Figure 3a, the models built on 10 batches are spread out all over the score plane, with a more Gaussian distribution and less autocorrelated behavior than the models based on 30 batches (Figure 3b) and on 50 batches (Figure 3c). In models with $\Delta I = 50$ batches, the effect of autocorrelation between models is evident, particularly for $i > 56$. In many occasions, the models recursively recalibrated on too few reference batches show great sensitivity to a change in the reference set even for consecutive updatings (Figure 3a), which means that the model is too sensitive to process variations or to anomalies in a set of data. On the other hand, when too many batches are used to build the models, replacing the oldest batch with the most recently completed one usually does not change the model very much (Figure 3c), meaning that the model is nearly insensitive to process changes. Therefore, by analyzing the score plots of Figure 3, a sensible number of batches to include as a reference in the model can be selected as $\Delta I = 30$, which seems to provide a good compromise between sensitivity to changes and memory

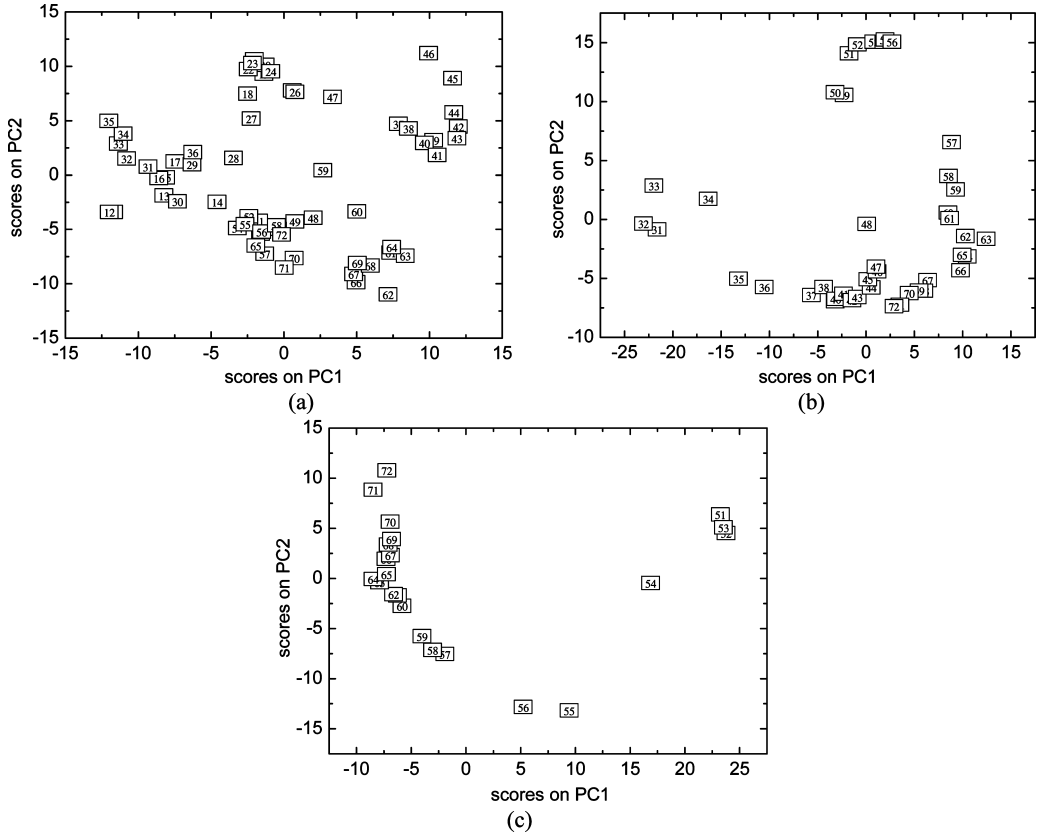


Figure 3. Batch polymerization process. Determination of the size of the moving window for a recursively updated PLS soft sensor that estimates the acidity number. Score plots for the models updated using (a) 10, (b) 30, and (c) 50 reference batches. Each square represents a model, whose number is indicated inside the square.

of past completed batches. It should be noted that the relative location of the models in the score plane changes only slightly if the confidence limits $T_{u,\mathrm{lim}}^2$ and $q_{u,\mathrm{lim}}$ are omitted from the matrix of Figure 2e, which, in turn, means that the most appropriate width of the MW does not depend markedly on $T_{u,\mathrm{lim}}^2$ and $q_{u,\mathrm{lim}}$ (at least for the examples considered in this work).

4. Nearest-Neighbor Model Adaptation

The underlying idea in MW adaptation schemes is that the chronological order in which the batches come one after another also serves as an indication of the degree of similarity between batches. Stated differently, if no adaptation of the model is carried out, the model performance is expected to decrease with time in a gradual fashion. This occurrence is indeed verified in many batch processes, especially when events that change the data correlation structure occur slowly.[17,34] However, there are several batch process systems for which the model performance degradation does not follow a temporal pattern, mostly because the process or the plant is subject to sudden changes.[19] As an example, Figure 4 shows the mean relative error (MRE) of estimation (see eq 10 in section 5) for a product quality property (acidity number) in the industrial batch polymerization process when a PLS soft sensor calibrated on $\Delta I = 30$ batches with no model adaptation is used.

The soft sensor maintains a good estimation accuracy for about three batches after it is put into service (i.e., from batch 31 to batch 33). Then, the MRE jumps to exceedingly high and totally unacceptable values until batch 54. However, starting from batch 55 and for five subsequent batches, the model performance is again adequate. Finally, from batch 60 onward,

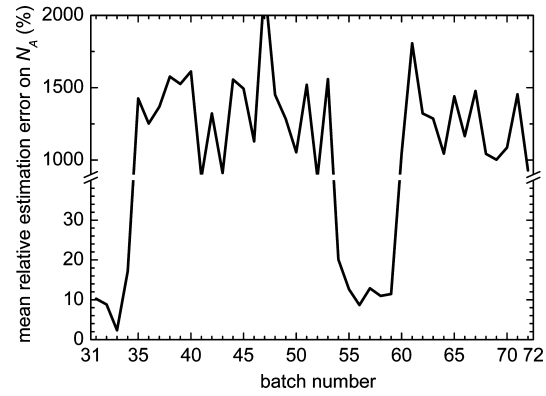


Figure 4. Batch polymerization process. Mean relative error for the estimation of the acidity number (during estimation phase 2) using a PLS soft sensor calibrated on 30 batches with no model updating.

the soft sensor again provides unsatisfactory estimations. Therefore, it is clear that the loss of performance of the model is far from being gradual, and adaptation of the soft sensor on the basis of temporal proximity concepts might not be the most convenient option for this process. This peculiar behavior suggests that one should look for a different adaptation logic, where temporal proximity between batches is replaced by proximity by similarity and where similarity between batches is then used to determine the most appropriate reference data set on which the model can be recursively recalibrated. The concept of similarity between time-series data sets is not new. In different contexts, it was considered in refs 35 and 36 to provide a strategy for pattern matching in batch processing, and

in ref 37, it was used to evaluate the dissimilarity of process data in order to improve the performance of continuous process monitoring.

The basic idea of the proposed updating scheme is that, in many instances, the monitoring model can be tailored on the batch currently being run. The method does not assume that the most recent batches are also the most representative ones for model updating, but by inspection of the database consisting of all of the batches that have been completed so far, it selects the ΔI batches that are most similar to the batch being run and recalibrates the soft sensor model on these batches. To apply this strategy, the following issues must be resolved: (i) how to define a similarity concept and (ii) how to classify the running batch according to this similarity concept. These issues are analyzed in the following subsections.

4.1. Extended Distance in the Score Space as a Measure of Similarity. Let us assume that all batches have the same length K or (if not) that they have all been aligned to length K; furthermore, let us assume that a total of $I > \Delta I$ batches have already been completed. The batch-wise-unfolded process data matrix of this set of batches is indicated with $\mathbf{X}^{\mathrm{BWU}}$ $[I \times JK]$. By application of PCA, at each updating step u, the following information can be extracted from $\mathbf{X}^{\mathrm{BWU}}$

$$\mathbf{X}^{\mathrm{BWU}} \xrightarrow{\mathrm{PCA}} \mathbf{T}, \mathbf{q} \tag{5}$$

where $\mathbf{T}$ is an $[I \times A]$ score matrix, $A \ll JK$ is the effective dimension of the latent space (i.e., the number of retained principal components), and $\mathbf{q}$ is the $[I \times 1]$ vector of the sum of squared predicting error

$$\begin{aligned} &\mathbf{T} = \{t_{i,a}\} \\ &\mathbf{q} = \{q_i\} \end{aligned} \qquad i = 1, 2, ..., I;\ a = 1, 2, ..., A \tag{6}$$

Therefore, after this operation, each batch can be associated with a certain location on the score space and with an error that accounts for the distance of the projected batch from the original batch. The "extended" location $\mathbf{l}_i$ of any batch i in the score space can therefore be summarized by the location matrix $\mathbf{L}$, obtained by concatenating horizontally the score matrix to the vector of the square root of the residuals

$$\begin{aligned} \mathbf{L} &= [\mathbf{T} \vdots \sqrt{\mathbf{q}}] \\ &= \begin{bmatrix} t_{1,1} & t_{1,2} & ... & t_{1,A} & \sqrt{q_1} \\ t_{2,1} & t_{2,2} & ... & t_{2,A} & \sqrt{q_2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ t_{I,i} & t_{I,21} & \cdots & t_{I,A} & \sqrt{q_I} \end{bmatrix} \\ &= \begin{bmatrix} l_{1,1} & l_{1,2} & ... & l_{1,A} & l_{1,A+1} \\ l_{2,1} & l_{2,2} & ... & l_{2,A} & l_{2,A+1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ l_{I,1} & l_{I,2} & ... & l_{I,A} & l_{I,A+1} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{l}_1 \\ \mathbf{l}_2 \\ \vdots \\ \mathbf{l}_I \end{bmatrix} \end{aligned} \tag{7}$$

The similarity between batch i and any other batch of the available database can be evaluated as the Euclidean distance between the locations $\mathbf{l}_i$ and $\mathbf{l}_{r \neq i}$. Note that, in order for any two batches to be similar, they must have both close values of the scores and close values of the residual errors. A similar concept of distance between models was used in ref 38.

The vector of the distances between a batch i and all the other batches is

$$\mathbf{d}_i = \{d_{i,r}\} \qquad r = 1, 2, ..., i-1, i+1, ..., I \tag{8}$$

where

$$d_{i,r} = \sqrt{\sum_{a=1}^{A+1} (l_{i,a} - l_{r,a})^2} \qquad r = 1, 2, ..., i-1, i+1, ..., I \tag{9}$$

By sorting the elements of vector $\mathbf{d}_i$ in ascending order, it is possible to identify which batches, among all those available in the database, are closer (i.e., most similar) to batch i. For example, by considering the first ΔI elements of the sorted vector $\mathbf{d}_i$, the ΔI nearest neighbors (NNs) to batch i are identified.

4.2. Detection of the Nearest Neighbors to a Running Batch. The NNs to the running batch i are the ideal candidates for use as a reference data set for calibrating a soft sensor model able to monitor batch i. However, so far, it has been assumed that any of the available I batches has already been completed. Therefore, if, in order to monitor batch i, a model is calibrated on the NNs to i, then only a post mortem (i.e., retrospective) analysis of batch i would be possible. This is unacceptable if a soft sensor has to be designed, as quality estimations are usually needed in real time. The question therefore arises of how the NNs to a running batch can be identified.

Engineering judgment and field experience suggest that, in most cases, the entire history of a batch is markedly influenced by the initial evolution of the batch itself. In fact, the greatest part of the batch variability is often gathered in the early stages of the batch, because of marked process dynamics, or the initial conditions of the pieces of equipment, or the state of the utilities needed to heat/cool the raw materials, or unpredictable variations in the raw materials themselves. Although these issues reveal themselves during the early stages of a batch, in many cases, they shape the remaining part of the batch as well.

Whether or not this conjecture occurs for the process being investigated can be assessed by preliminarily checking which batches are the NNs to a new running batch i (not included in the set of I already completed batches) when only a fraction $K' \ll K$ of measurements have been collected from i. If the set of NNs to i after K' time instants is roughly the same as would be found at the end of the batch, then only a small fraction of i needs to be completed in order to be able to determine the batches that are expected to be most similar to i itself. Using these neighbors as the reference data set, a soft sensor can then be recalibrated online while batch i is running and used for all time instants larger than K'.

The detection of the ΔI nearest neighbors to batch i can be carried out using the procedure outlined in the previous subsection, with the only difference being that the distances between batch i and all of the other I batches are computed using K' (rather than K) measurement sets. Note that batch i is running, and all of the other batches included in the available data set have already been completed. Therefore, the number of batches included in the available data set increases by one unit every time a batch has been completed, and the procedure to determine the NNs to i can be applied recursively at the beginning of each new batch run.

As for the determination of K' (i.e., the number of sampling instants to be considered before the model can be recalibrat-

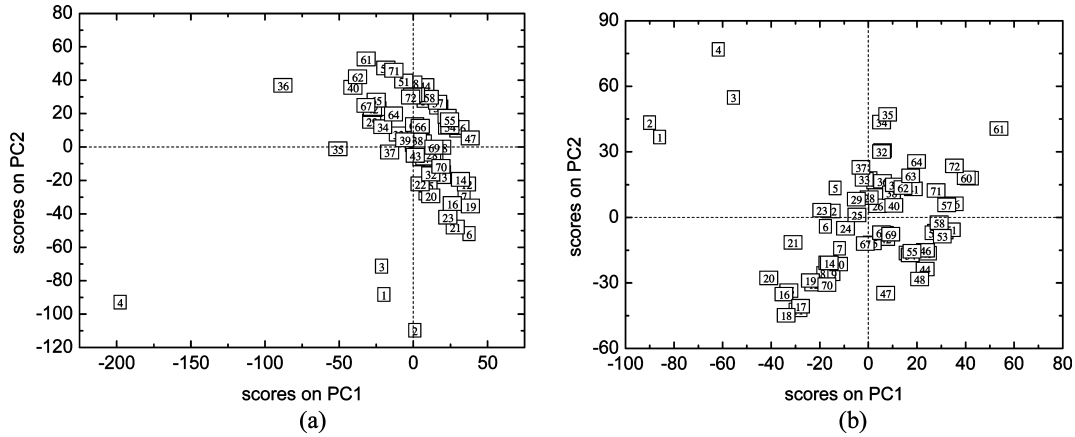


Figure 5. Batch polymerization process. PCA on data from 72 batches with (a) the first 800 sampling instants only for each batch and (b) the completed and synchronized batches. Each square represents a batch, whose number is indicated inside the square.

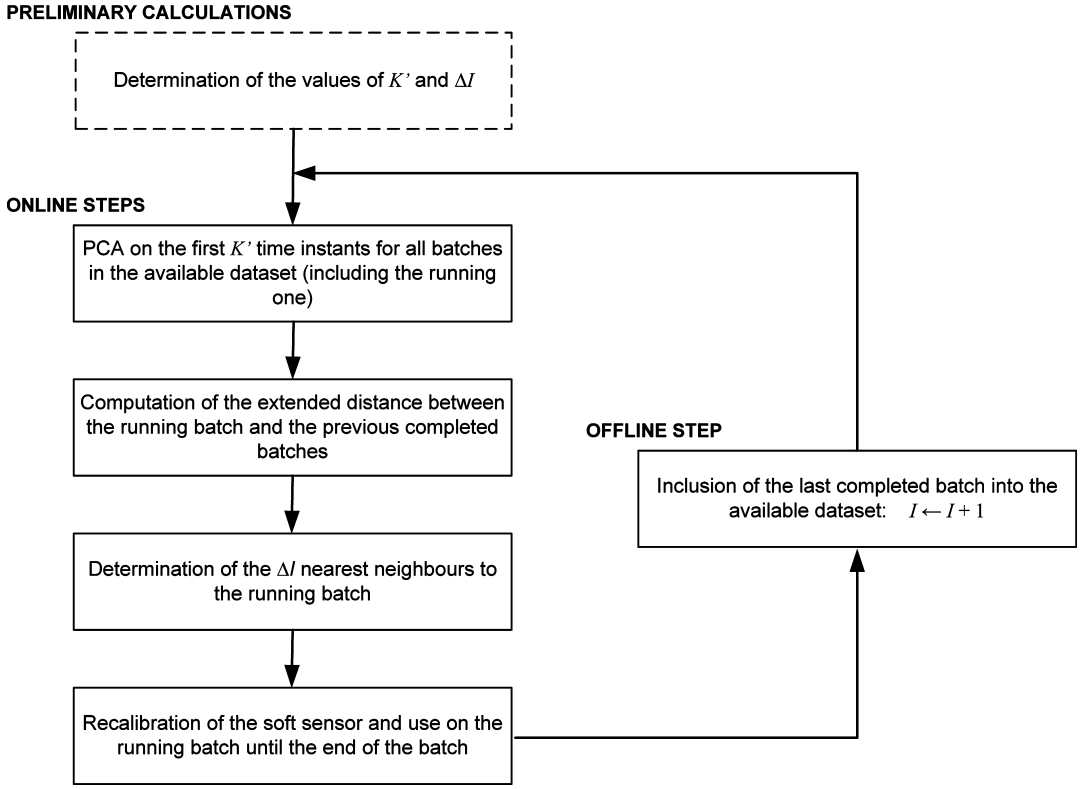


Figure 6. Flowchart of the procedure for soft sensor nearest-neighbor online updating. The online steps are carried out once per batch, at time instant K'.

ed), it should be noted that multiway methods usually work well after at least ~10% of the batch is completed.[39] We found that setting K' to a value between 10% and 30% of the expected total batch length usually provides good results. Again, we take the industrial batch polymerization process as an example to clarify this issue. Figure 5a (which refers to a total of 72 batches of the batch polymerization process) shows the projection of each batch onto the score plane of the first two PCs after $K' = 800$ time instants. Figure 5b shows the projections of the same batches after batch completion (note that the batches were aligned through downsampling in order to be able to apply BWU after batch completion.).

It can be noticed that the mutual location of each batch within the score plot is very similar in the two diagrams. In fact, the two figures look quite similar if Figure 5a is rotated by 90° clockwise (the axis scales are obviously different). To quantify this correspondence more precisely, we consider the most recent batch in the reference set (batch 72) as an example. If the $\Delta I = 30$ nearest neighbors to this batch are determined after K' time instants and after the end of the batch, a 70% overlap is found between the two sets of NNs. The overlapping for "older" batches ($i < 72$) is generally even better than this one. This result can be interpreted by saying that K' represents a sufficiently long processing time for the correlation between batches to stabilize approximately at the same value as the correlation between completed batches. Therefore, batch similarity can be evaluated much earlier than the completion of a batch for this system, and this allows online updating of the soft sensor after K' time instants by using the set of the NNs to the running batch as the most convenient reference data set. A schematic illustration of how the algorithm works for a system like this is shown in Figure 6.

5. Performance of the Nearest-Neighbor Updating Scheme

One way to evaluate the performance of a soft sensor is to check the estimated value of the quality property in which one is interested against the true value for all available measurement points. Thus, by indicating the value of a quality property estimated at sampling instant h of batch i as $\hat{y}_{i,h}$ and the measured value of the same property at the same instant of the same batch as $y_{i,h}$, the mean relative error of estimation during the batch is defined as

$$\mathrm{MRE}_i = 100 \times \frac{\sum_{h=1}^{H_i} \left[\frac{\sqrt{(y_{i,h} - \hat{y}_{i,h})^2}}{y_{i,h}} \right]}{H_i} \tag{10}$$

where H_i is the number of measurements available for the quality property in batch i.

However, MRE_i can provide only a retrospective evaluation of the soft sensor performance. Quality measurements are usually not available while the soft sensor is running, and to evaluate the soft sensor performance, the reliability of the estimation should also be assessed. The estimation reliability can be evaluated in terms of the number of alarms in the Hotelling statistic and in the squared prediction error statistic. Specifically, at batch i, the alarm rate on the Hotelling statistic is

$$\mathrm{AR}_i^{T^2} = 100 \times \frac{n_i^{T_i^2 > T_{u,\mathrm{lim}}^2}}{H_i} \tag{11}$$

and the alarm rate of the squared prediction error statistic is

$$\mathrm{AR}_i^{q} = 100 \times \frac{n_i^{q_i > q_{u,\mathrm{lim}}}}{H_i} \tag{12}$$

where $n_i^{T_i^2 > T_{u,\mathrm{lim}}^2}$ and $n_i^{q_i > q_{u,\mathrm{lim}}}$ are the number of times the relevant statistic (T_i^2 or q_i) for batch i overcomes the relevant confidence limit ($T_{u,\mathrm{lim}}^2$ or $q_{u,\mathrm{lim}}$) of the model at updating step u. More generally, the alarm rate will be taken as the greater between $\mathrm{AR}_i^{T^2}$ and AR_i^{q}; that is, $\mathrm{AR}_i = \max(\mathrm{AR}_i^{T^2}, \mathrm{AR}_i^{q})$.

In the following subsections, a comparison between MW updating and NN updating is presented for the two selected case studies. It should be noted that, although the length ΔI of the MW adaptation scheme was optimized as described in section 3.1, no attempt was carried out to optimize the effective dimension of the reference database for the NN updating scheme, and the same value of ΔI determined for the MW approach was used. Although this might not guarantee the best performance for the NN approach, in order to compare the two methods, we preferred to maintain the same model "complexity" in terms of the number of data values used to build the model.

5.1. Industrial Batch Polymerization Process. This is a particularly interesting case study for the proposed methodology. In fact, the process has been observed to go through frequent variations due to seasonal variability, multiple materials suppliers, several manual operations, sensor drifts, periodic production restarts due to temporary equipment usage for other products. In such a situation (unfortunately, quite common in the batch production of fine chemicals in small- to medium-sized enterprises), an NN adaptive scheme seems to be an appropriate choice, as change is often (although not always) related to known or unknown events rather than to the progress of time.

The results of the two updating methods were compared in terms of average mean relative estimation error

$$\mathrm{AMRE} = \frac{1}{I - \Delta I - 1} \sum_{i=\Delta I+1}^{I} \mathrm{MRE}_i \tag{13}$$

and the average alarm rate

$$\mathrm{AAR} = \frac{1}{I - \Delta I - 1} \sum_{i=\Delta I+1}^{I} \mathrm{AR}_i \tag{14}$$

obtained by averaging MRE_i and AR_i, respectively, over all batches for which model updating was carried out.

It was found that the AAR index decreases from 93.3% in the case without model updating to 31.9% with MW updating to 28.9% with NN updating. Therefore, the NN adaptation scheme does not extend very significantly the reliability of the soft sensor with respect to the MW adaptation scheme. However, NN updating allows for a better estimation performance (Figure 7), especially during the critical estimation phase 3, where the AMRE values for N_A and μ are decreased by 4 and 10 percentage points, respectively.

The results of the NN updating method can be further explored by analyzing the estimation accuracy throughout all batches where NN adaptation is applied (only the case of acidity number estimation is reported for conciseness).

Figure 8 shows that the mean relative error is almost always sufficiently small in any estimation phase and for all batches. Occasionally, the MRE grows to very large values (e.g., for batches 35 and 43), but this is indicated by the alarm rate that increases to 100% in these batches (Figure 9). A retrospective analysis can then be carried out to determine whether a cause can be assigned to the high alarm rate.

For example, by retrospective analysis, it was found that the unsatisfactory estimation results in batch 43 were due to a change in the supplier of one of the raw materials (a fermentation product). The impurity level in the new raw material was probably different from that of previous batches, and this partially changed the correlation structure in the data. It took two to three batches before the model adapted to the new process conditions and could capture the new correlation structure.

5.2. Simulated Semibatch Fermentation Process. The second case study is the simulated process for penicillin production discussed previously. By using the procedure described in section 3.1, the optimal width of the reference database was again found to be $\Delta I = 30$ batches. With reference to the detection of the NNs to a running batch, it was found that ~30% of the batch needed to be completed in order to be able to capture most of the end-of-batch correlation. Therefore, K' was set to 60 sampling instants (where the total length of the batches was $K = 200$ sampling instants).

Also in this case, the average alarm rates were very similar (and, in most batches, close to zero) for both the MW and NN adaptation procedures. The average mean relative errors of the penicillin concentration estimation in all batches subject to model adaptation were AMRE = 10.6% for the NN updating scheme and AMRE = 12.1% for the MW updating scheme.

The fact that, in this process, all batches have the same length also allows for the comparison of the time trajectories of the relative estimation errors, on average, for all of the batches subject to model adaptation. Figure 10 shows that the NN adaptation is superior to the MW adaptation from the very beginning of the estimation, that is, for every instantaneous model comprising the overall evolving PLS model. The

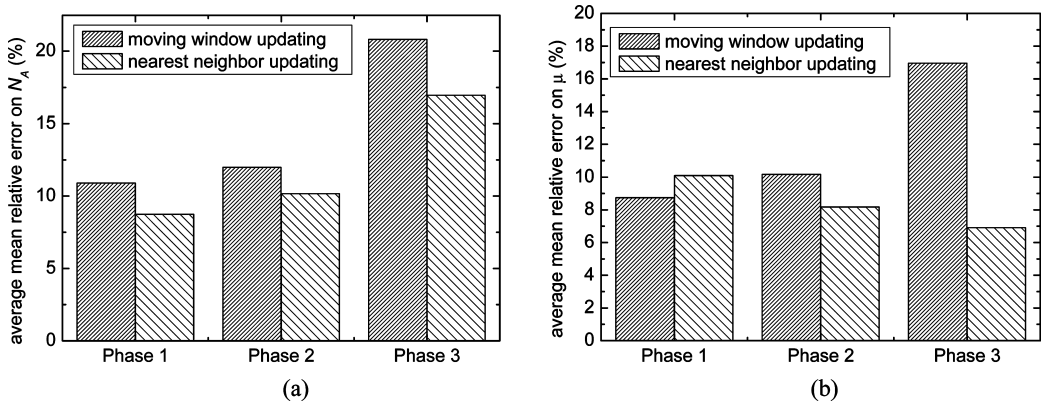


Figure 7. Batch polymerization process. Comparison between the performance of MW adaptation and NN adaptation in terms of average mean relative error in the estimation of (a) acidity number and (b) viscosity.

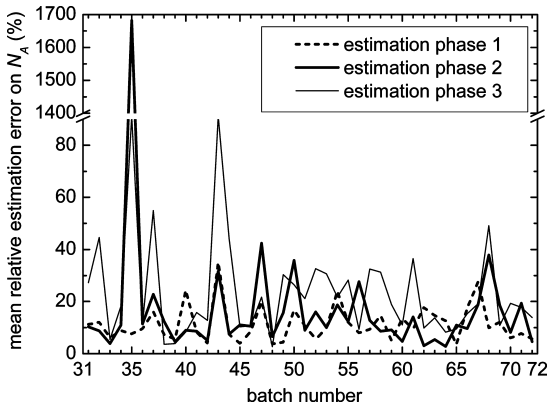


Figure 8. Batch polymerization process. Mean relative error in the estimation of the acidity number using NN updating.

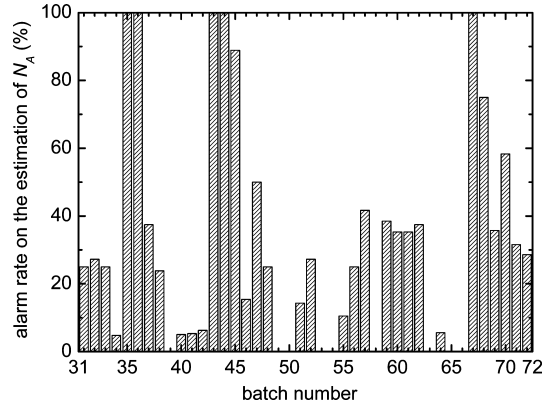


Figure 9. Batch polymerization process. Alarm rate in the estimation of the acidity number using NN updating.

consistent (although not as marked) outperformance of the NN approach with respect to the MW approach suggests that, even in a rather controlled and fully known process, a more effective sensor can be designed by adapting the model according to a similarity rationale.

6. Conclusions

In this article, a methodology for the automatic maintenance (updating) of PLS soft sensors has been proposed. This issue is particularly important in batch processing, where slow or sudden changes are typical and can affect the reliability and accuracy of data-driven soft sensors for product quality estimation.

Whereas the adaptation scheme in classical recursive updating usually follows chronological order, the strategy proposed herein

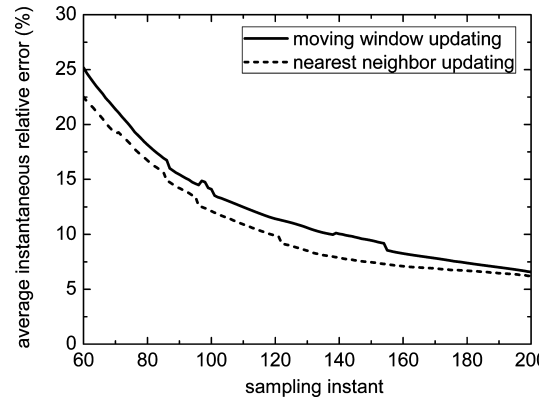


Figure 10. Semibatch fermentation process. Average instantaneous mean relative error for MW updating and NN updating of a soft sensor estimating the penicillin concentration.

defines the reference data set for model recalibration as the set of batches (nearest neighbors) that are most similar to the running batch. It was shown that the set of nearest neighbors can be identified during the initial evolution of a batch. In this way, for any new batch to be run, a model can be tailored on the running batch itself. With reference to classical moving-window adaptation schemes, a heuristic method was also proposed to determine the most appropriate window width. An industrial case study and a simulated case study were presented to demonstrate that the proposed nearest-neighbor updating approach improves the soft sensor performance with respect to a moving-window updating scheme.

The proposed approach relies on the fact that only a fraction of a batch needs to be completed before the correlation structure among the available process data approaches the structure that would be found at the end of the batch. Although this turned out to be true for the processes considered in this research, it cannot be guaranteed that it holds true for any batch process.

The effective dimension of the reference database used for model recalibration in the nearest-neighbor adaptation scheme was set equal to the most appropriate window width of a standard moving-window recursive approach. This proved to be a simple and effective way to determine a reasonable database dimension. Although not shown herein for the sake of conciseness, it was found that a slightly larger dimension usually does not impact the estimation results significantly; however, the soft sensor performance degrades quite quickly as the dimension becomes smaller.

The overall procedure for soft sensor maintenance is computationally undemanding. Even for the relatively large

databases considered in this research, the time needed to perform the nearest-neighbor search plus soft sensor recalibration was just a few seconds using a standard personal computer.

Acknowledgment

The authors gratefully acknowledge the financial support granted to this work by the University of Padova under Project CPDR088921-2008 on "Innovative techniques for multivariate and multiscale monitoring of quality in the industrial production of high added-value products".

List of Symbols

a = generic latent variable number
A = total number of latent variables
AAR = average alarm rate
AMRE = average mean relative error
AR = alarm rate
AR_i = alarm rate for batch i (%)
AR_i^q = alarm rate on the q statistic for batch i (%)
$AR_i^{T^2}$ = alarm rate on the T^2 statistic for batch i (%)
$d_{i,r}$ = Euclidean distance between the locations $\mathbf{l}_i$ of batch i and $\mathbf{l}_r$ of batch r
ΔI = size of the moving window (number of batches)
h = generic sample for the laboratory measurement of the quality variables
H_i = total number of quality laboratory measurements for batch i
i = index denoting a generic batch
I = total number of batches
j = generic process variable
J = total number of process variables
K = total number of samples for the process variables in synchronized batches
K' = fraction of the process measurements K that is sufficient to evaluate the nearest neighbors in the first part of the batch
K_i = total number of samples for the process variables for batch i
$l_{i,a}$ = element of the location matrix $\mathbf{L}$
M = total number of quality variables
μ = viscosity (P)
MRE = mean relative error
MRE_i = mean relative error for batch i
N_A = acidity number (g_{KOH}/g_{resin}))
$n_i^{q_i>q_{u,lim}}$ = number of times the q_i statistic of batch i overcomes the confidence limit $q_{u,lim}$ of the model at updating step u
$n_i^{T_i^2>T_{u,lim}^2}$ = number of times the T_i^2 statistic of batch i overcomes the confidence limit $T_{u,lim}^2$ of the model at updating step u
q_i = square predicting error of batch i
$q_{u,lim}$ = 99% confidence limit of the square predicting error for the model at updating step u
T_i^2 = Hotelling statistic for batch i
$T_{u,lim}^2$ = 99% confidence limit of the Hotelling statistic for the model at updating step u
u = index denoting the updating step
$x_{i,j,k}$= kth sample of process variable j during batch i (element of the three-dimensional matrix $\underline{\mathbf{X}}$)
$y_{i,h}$ = hth laboratory measurement of the quality variable y during batch i
$\hat{y}_{i,h}$= soft sensor estimation of $y_{i,h}$

Matrices and Vectors

$\mathbf{d}_i$ = distances between the location $\mathbf{l}_i$ of batch i and the locations of all the other available batches [$(I - 1) \times 1$]
$\mathbf{L}$ = location matrix [$I \times (A + 1)$]
$\mathbf{l}_i$ = extended location of the batch i [$1 \times (A + 1)$]
$\mathbf{P}_u$ = matrix of the loadings for the model at the updating step u [$\Delta I \times A$]
$\mathbf{q}$ = vector of the squared predicting errors [$I \times 1$]
$\mathbf{T}$ = vector of Hotelling statistics [$I \times A$]
$\underline{\mathbf{X}}$ = three-dimensional matrix of the process variables [$I \times J \times K$]
$\mathbf{X}_i$ = matrix of K_i samples of J process variables for batch i (horizontal slice of $\underline{\mathbf{X}}$) [$J \times K_i$]
$\mathbf{x}_i^{BWU}$ = vector of the batch-wise-unfolded $\mathbf{X}_i$ [$1 \times JK$]
$\mathbf{X}^{BWU}$ = generic batch-wise-unfolded matrix [$I \times JK$]
$\mathbf{X}_u^{BWU}$ = matrix of the batch-wise-unfolded data at updating step u [$\Delta I \times JK$]
$\mathbf{X}_u^{VWU}$ = matrix of the variable-wise-unfolded data at updating step u [$\Delta I \cdot K \times J$]
$\underline{\mathbf{Y}}$ = three-dimensional matrix of the quality variables [$I \times M \times H_i$]
$\mathbf{W}_u$ = matrix of the weights for the model at updating step u [$\Delta I \times A$]

Acronyms

BWU = batch-wise unfolding
LV = latent variable
MW = moving window
PC = principal component
PCA = principal component analysis
PLS = partial least-squares (also called projection on latent stuctures)
NN = nearest neighbor
VWU = variable-wise unfolding

Literature Cited

(1) Jackson, J. E. *A User's Guide to Principal Components*; John Wiley & Sons Inc.: New York, 1991.

(2) Geladi, P.; Kowalski, R. Partial least squares regression: A tutorial. *Anal. Chim. Acta* **1986**, *185*, 1–17.

(3) Kourti, T.; MacGregor, J. F. Process analysis, monitoring and diagnosis, using multivariate projection methods. *Chemom. Intell. Lab. Syst.* **1995**, *28*, 3–21.

(4) Wise, B. M.; Gallagher, N. B. The process chemometrics approach to process monitoring and fault detection. *J. Process Control* **1996**, *6*, 329–348.

(5) Nomikos, P.; MacGregor, J. F. Monitoring batch processes using multiway principal component analysis. *AIChE J.* **1994**, *40*, 1361–1375.

(6) Kourti, T. Mulivariate dynamic data modeling for analysis and statistical process control of batch processes, start-ups and grade transitions. *J. Chemom.* **2003**, *17*, 93–109.

(7) Kresta, J. V.; Marlin, T. E.; MacGregor, J. F. Development of inferential process models using PLS. *Comput. Chem. Eng.* **1994**, *18*, 597–611.

(8) Qin, S. J.; Yue, H.; Dunia, R. Self-validating inferential sensors with application to air emission monitoring. *Ind. Eng. Chem. Res.* **1997**, *36*, 1675–1685.

(9) Kadlec, P.; Gabrys, B.; Strandt, S. Data-driven soft sensors in process industry. *Comput. Chem. Eng.* **2009**, *33*, 795–814.

(10) Yoo, C. K.; Lee, I. B.; Vanrolleghem, P. A. On-line adaptive and nonlinear process monitoring of a pilot-scale sequencing batch reactor. *Environ. Monit. Assess.* **2006**, *119*, 349–366.

(11) Qin, S. J. Recursive PLS algorithms for adaptive data modeling. *Comput. Chem. Eng.* **1998**, *22*, 503–514.

(12) Rännar, S.; MacGregor, J. F.; Wold, S. Adaptive batch monitoring using hierarchical PCA. *Chemom. Intell. Lab. Syst.* **1998**, *41*, 73–81.

(13) Dayal, B. S.; MacGregor, J. F. Recursive exponentially weighted PLS and its applications to adaptive control and prediction. *J. Process Control* **1997**, *7*, 169–179.

(14) Choi, S. W.; Martin, E. B.; Morris, A. J.; Lee, I. B. Adaptive multivariate statistical process control for monitoring time-varying processes. *Ind. Eng. Chem. Res.* **2006**, *45*, 3108–3118.

(15) He, X. B.; Yang, Y. P. Variable MWPCA for adaptive process monitoring. *Ind. Eng. Chem. Res.* **2008**, *47*, 419–427.

(16) Li, W.; Yue, H. H.; Valle-Cervantes, S.; Qin, S. J. Recursive PCA for adaptive process monitoring. *J. Process Control* **2000**, *10*, 471–486.

(17) Wang, X.; Kruger, U.; Irwin, G. W. Process monitoring approach using fast moving window PCA. *Ind. Eng. Chem. Res.* **2005**, *44*, 5691–5702.

(18) Wold, S. Exponentially weighted moving principal components analysis and projection on latent structures. *Chemom. Intell. Lab. Syst.* **1994**, *23*, 149–161.

(19) Lee, Y. H.; Jin, H. D.; Han, C. On-line process state classification for adaptive monitoring. *Ind. Eng. Chem. Res.* **2006**, *45*, 3095–3107.

(20) Lee, H. W.; Lee, M. W.; Park, J. M. Robust adaptive partial least squares modelling of a full-scale industrial wastewater treatment process. *Ind. Eng. Chem. Res.* **2007**, *46*, 955–964.

(21) Vijaysai, P.; Gudi, R. D.; Lakshminarayanan, S. Identification on demand using blockwise recursive partial least-squares techenique. *Ind. Eng. Chem. Res.* **2003**, *42*, 540–554.

(22) Capron, X.; Walczak, B.; de Noord, O. E.; Massart, D. L. Selection and weighting of samples in multivariate regression model updating. *Chemom. Intell. Lab. Syst.* **2005**, *76*, 205–214.

(23) Lee, Y. H.; Kim, M.; Chu, Y. H.; Han, C. Adaptive multivariate regression modeling based on model performance assessment. *Chemom. Intell. Lab. Syst.* **2005**, *78*, 63–73.

(24) Facco, P.; Doplicher, F.; Bezzo, F.; Barolo, M. Moving-average PLS soft sensor for online product quality estimation in an industrial batch polymerization process. *J. Process Control* **2009**, *19*, 520–529.

(25) Faggian, A.; Facco, P.; Doplicher, F.; Bezzo, F.; Barolo, M. Multivariate statistical real-time monitoring of an industrial fed-batch process for the production of specialty chemicals. *Chem. Eng. Res. Des.* **2009**, *87*, 325–334.

(26) Birol, G.; Ündey, C.; Çinar, A. A modular simulation package for fed-batch fermentation: Penicillin production. *Comput. Chem. Eng.* **2002**, *26*, 1553–1565.

(27) Ündey, C.; Ertunç, S.; Çinar, A. Online batch/fed-batch process performance monitoring, quality prediction, and variable-contribution analysis for diagnosis. *Ind. Eng. Chem. Res.* **2003**, *42*, 4645–4658.

(28) Ündey, C.; Tatara, E.; Çinar, A. Real-time batch process supervision by integrated knowledge-based systems and multivariate statistical methods. *Eng. Applic. Artif. Intell.* **2003**, *16*, 555–566.

(29) Ündey, C.; Tatara, E.; Çinar, A. Intelligent real-time performance monitoring and quality prediction for batch/fed-batch cultivations. *J. Biotechnol.* **2004**, *108*, 61–77.

(30) Zhao, C.; Wang, F.; Gao, F.; Lu, N.; Jia, M. Adaptive monitoring method for batch processes based on phase dissimilarity updating with limited modeling data. *Ind. Eng. Chem. Res.* **2007**, *46*, 4943–4953.

(31) Ramaker, H. J.; van Sprang, E. N. M.; Westerhuis, J. A.; Smilde, A. K. Fault detection properties of global, local and time evolving models for batch process monitoring. *J. Process Control* **2005**, *15*, 799–805.

(32) Wold, S.; Geladi, P.; Esbensen, K.; öhman. J. Multi-way principal components and PLS analysis. *J. Chemom.* **1987**, *1*, 47–56.

(33) Wang, X.; Kruger, U.; Lennox, B. Recursive partial least squares algorithms for monitoring complex industrial processes. *Control Eng. Practice* **2003**, *11*, 613–632.

(34) Zhao, C.; Wang, F.; Gao, F.; Zhang, Y. Enhanced process comprehension and statistical analysis for slow-varying batch processes. *Ind. Eng. Chem. Res.* **2008**, *47*, 9996–10008.

(35) Singhal, A.; Seborg, D. E. Pattern matching in historical batch data using PCA. *IEEE Control Syst. Mag.* **2005**, *22*, 53–63.

(36) Gunther, J. C.; Baclaski, J.; Seborg, D. E.; Conner, J. S. Pattern matching in batch bioprocesses—Comparison across multiple products and operating conditions. *Comput. Chem. Eng.* **2009**, *33*, 88–96.

(37) Kano, M.; Hasebe, S.; Hashimoto, I.; Ohno, H. Statistical process monitoring based on dissimilarity of process data. *AIChE J.* **2002**, *48*, 1231–1240.

(38) Cheng, C.; Chiu, M. S. A new data-based methodology for nonlinear process modelling. *Chem. Eng. Sci.* **2004**, *59*, 2801–2810.

(39) Nomikos, P.; MacGregor, J. F. Multi-way partial least squares in monitoring batch processes. *Chemom. Intell. Lab. Syst.* **1995**, *30*, 97–108.

Received for review September 4, 2009
Revised manuscript received November 4, 2009
Accepted January 13, 2010

IE9013919