Intelligent Software Sensors for Fed-Batch Fermentation Processes

Hongwei Zhang

Faculty of Arts, Computing, Engineering and Science, Sheffield Hallam University, Howard Street, Sheffield, S1 1WB, UK (Tel: +44-114-225 3432; e-mail: H.Zhang@shu.ac.uk).

Abstract: Software sensors have attracted great research interests due to the problem of lacking suitable and robust online sensors for key fermentation variables in fed-batch fermentation processes. In this paper, intelligent software sensors have been developed based on multivariate statistical process control methods. The software sensors not only provide real time estimation of key variables but also have the facilities of self-diagnosis, self-validation and self-calibration using available lab assay data. An application of the software sensors to a fed-batch penicillin fermentation process is presented, and significant improvements over ordinary methods have been shown in the simulation results.

1. INTRODUCTION

Industrial bioprocesses present a very difficult challenge to scientists and engineers. Problems associated with the nature of the organisms in the process and difficulties related to obtaining accurate information regarding the progression of the process make controlling and monitoring particularly challenging. The lack of suitable and robust on-line sensors for key variables such as biomass or product concentration has been considered as a serious obstruction for the implementation of control and optimisation of bioprocesses (Aynsley, *et al.*, 1993; James et al., 2002). In addition to the high costs associated with these measuring devices, their reliability can be poor when applied to large-scale systems (Montague, 1997). It is still the case that most industrial fermentation control policies are based upon the use of infrequent off-line assay information for process operator supervision (Dacosta *et al*., 1997). The low sampling frequency associated with such measurements and the inevitable delays in taking samples and performing laboratory tests inevitably compromises the quality of control that is possible using such measurements. Therefore the method of indirect measurement has attracted a great deal of attention over the last 20 years or so. Indirect measurements of biomass are mathematical algorithms that can produce estimates of unmeasured biomass concentration using the continuously measured variables such as temperature, dissolved oxygen, pH and off-gas concentration. The method of estimating the quality related variables from measurements of secondary variables is referred to as 'Inferential Estimation' and these mathematical estimators are usually referred to as 'Software Sensors'. Improved control of the process can be achieved by measuring and setting up a feedback control system using these secondary variables. Such control strategies are referred to as 'Inferential Controllers' (Joseph, 1999). Software sensors usually rely on a model to describe the process, thus different techniques have been proposed for on-line inferential estimation in bioprocesses just as different models exist. Among these applications, the majority have been based upon mechanistic, neural network or other empirical models.

Standard non-linear estimation techniques such as the Extended Kalman Filter (EKF) have been successfully used to construct software sensors in both simulation and experimental investigations (Gudi et al., 1995; Aubrun et al., 2001). They can, however, suffer from numerical problems and convergence difficulties (de Assis and Filho, 2000).

Artificial Neural Networks (ANNs) have been proven to be successful in modelling fermentation processes, thus they have become an obvious approach for the development of software sensors. There has been extensive interest in the use of ANNs and many applications, employing various ANN model architectures have been reported (Dacosta *et al.*, 1997; Tham *et al.*, 2002). The main advantage of ANN using techniques for software sensors is that they do not need any prior knowledge about the kinetic growth rate, but the disadvantage is the large amount of training and test data that is required for this development.

Multivariate statistical methods based on linear projection, such as Principal Components Analysis (PCA) and Partial Least Squares (PLS), have also attracted considerable interest as a method for producing robust empirical models, particularly when there are high dimensionality and collinearities in the data. PCA and PLS, in particular, and their variations, such as neural network partial least squares (NNPLS) (Qin and McAvoy, 1992) and nonlinear principal components analysis (NLPCA) (Dong and McAvoy, 1996), have been applied to many practical regression problems to estimate quality related variables in chemical engineering processes such as distillation columns, combustion processes, the paper and pulp making process and polymerization processes. Multiway Partial Least Squares (MPLS) has been utilized for fed-batch fermentation processes and compared with EKF in Zhang et al. (2005).

The applications of the above-mentioned software sensors on fermentation processes have been proved successful in providing estimates of the unmeasured variables (e.g., biomass concentration, product concentration, specific growth rate, etc). However, the performance of the software sensors could be significantly improved by employing the concept of 'intelligent sensors'. The term 'intelligent sensors' represents a new generation of sensors which are capable of performing self-diagnosis, self-validation and self-calibration during operation. The ability to carry out real-time fault and drift detection, fault isolation is considered as one of the key features of intelligent sensors. To provide such a capability, software-based solutions have obvious advantages. Boltryk *et al* (2005) proposed a generic software approach for intelligent sensors where a software framework used to implement tasks such as condition monitoring were described.

Apart from providing software sensing facilities, PCA and PLS are very promising approaches for fault detection and isolation in application to industrial fermentation processes because they are capable of handling high dimensional and correlated variables and they are powerful tools for revealing the presence of process abnormalities. Therefore, PLS is used in this paper to incorporate its software sensing capability with advanced data modelling techniques for condition monitoring.

This paper aims to develop intelligent software sensors using MPLS and demonstrate its capability on the application of fed-batch fermentation processes.

2. STATISTICAL MODELLING AND SOFT-SENSING USING MULTIWAY PARTIAL LEAST SQUARES

2.1 Partial Least Squares

PLS is a system identification tool that is capable of identifying the relationships between cause (X) and effect (Y) variables. The advantage that this approach offers over more traditional identification techniques, such as ordinary least squares, is that it is able to extract robust models even in applications involving large numbers of highly correlated and noisy process variable measurements.

The approach works by selecting factors of cause variables in a sequence that successively maximises the explained covariance between the cause and effect variables. Given a matrix of cause data, $\mathbf{X}$, and effect data, $\mathbf{Y}$, a factor of the cause data, $\mathbf{t}_k$, and effect data, $\mathbf{u}_k$, is evaluated, such that:

$$\mathbf{X} = \sum_{k=1}^{np<nx} \mathbf{t}_k \mathbf{p}_k^T + \mathbf{E} \tag{1}$$

and

$$\mathbf{Y} = \sum_{k=1}^{np<nx} \mathbf{u}_k \mathbf{q}_k^T + \mathbf{F} \tag{2}$$

where $\mathbf{E}$ and $\mathbf{F}$ are residual matrices, *np* is the number of inner components that are used in the model and *nx* is the number of causal variables. $\mathbf{p}_k$ and $\mathbf{q}_k$ are referred to as loading vectors.

These equations are referred to as the *outer relationships*. The vectors $\mathbf{t}_k$ are mutually orthogonal. These vectors and $\mathbf{u}_k$ are selected so as to maximise the covariance between each pair, ($\mathbf{t}_k$, $\mathbf{u}_k$). Linear regression is performed between the $\mathbf{t}_k$ and the $\mathbf{u}_k$ vectors to produce the inner relationship, such that:

$$\mathbf{u}_k = b_k \mathbf{t}_k + \varepsilon_k \tag{3}$$

where b_k is a regression coefficient, and ε_k refers to the prediction error. The PLS method provides the potential for a regularised model through selecting an appropriate number of latent variables, $\mathbf{u}_k$ in the model (*np*). The number of latent variables is typically generated through the use of cross validation.

2.2 Multiway Partial Least Squares

PLS is a linear tool, which unfortunately limits its effectiveness when applied to non-linear fed-batch processes. Two options exist for improving the capabilities of PLS when applied to fed-batch systems. The first is to develop non-linear counterparts to PLS and the second is to transform the fed-batch data in such a way as to remove the non-linear characteristics (Nomikos and MacGregor, 1994). Although non-linear PLS techniques exist (Qin and McAvoy, 1992), the transformation of batch data has proved to be a more effective option and has been adopted in this investigation. The most common form of data transformation, termed multiway PLS (MPLS), was initially proposed by Nomikos and MacGregor (1994). Since then other researchers have adopted the approach and applied it to a variety of processes. For example, Gallagher *et al*. (1996) applied the technique to monitor nuclear waste storage vessels and Lennox *et al*. (2001) and Lakshminarayanan *et al.* (1996) investigated the detection of faults in fed-batch fermentation processes.

In a fed-batch process, the cause and effect data can be thought of as being in two 3-dimensional arrays of size nb × nx × mx and nb × ny × my, where nb is the number of batches for which data is available, nx and ny are the number of cause and effect variables respectively and mx and my are the number of observations of the cause and effect variables respectively that are made during a batch. Unfortunately, PLS requires that the cause and effect arrays be two-dimensional. To address this problem the three-dimensional arrays are recast into two-dimensional arrays in a process referred to as unfolding. The concept of unfolding is illustrated in figure 1. The original data arrays are unfolded into a cause variable array, X, of size nb × (nx*mx) and an effect variable array of size nb × (ny*my). It should be noted that the number of observations made of the cause variables need not be equal to the number of observations made for the effect variables. In fact, it is relatively common to have a single effect measurement made during a batch. This measurement is the final product quality taken at the end of the batch. Following

the unfolding of the data, it is then possible to apply PLS to the data in the conventional manner.

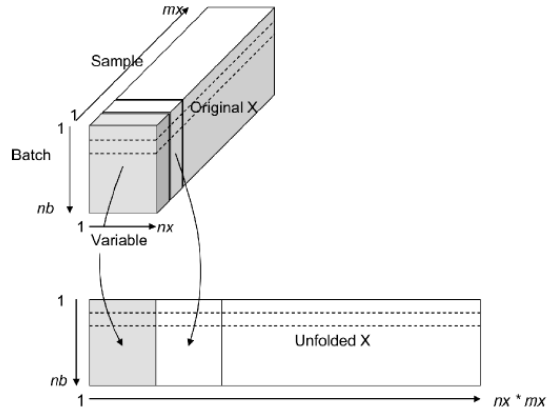

Fig. 1 Unfolding

The subsequent use of this model on-line poses the problem that it is necessary to know the values of all process measurements through to the end of the batch, since the unfolded array contains the measurements of each of the variables throughout the duration of the batch. This means that with the exception of the end point of the batch, it is necessary to estimate the future values of all the measured variables. The prediction of future process values is referred to as filling up the array. Of the three methods that were suggested by Nomikos and MacGregor (1994) for filling up the array, Lennox *et al.* (2001) found that the most appropriate method for an industrial fed-batch process was to assume that the values of all process measurements remain at their current offset from the mean trajectory through to the end of the batch. Whilst the most suitable filling up method is likely to be process dependent, this method was also found to be the most appropriate in this work.

3. A BENCHMARK FED-BATCH PENICILLIN FERMENTATION PROCESS

Secondary metabolites such as antibiotics, and in particular penicillin, have important added value, and therefore improvements in their production are of great interest to industry. For this reason there has been a great deal of research conducted during the last decade on all aspects of penicillin production (Gomez Sanchez et al. 1999; Undey et al. 2000). The work described in this paper is concerned with providing intelligent software sensing capabilities in the production of penicillin. To demonstrate the benefits of intelligent software sensors proposed in this paper, the simulation of a penicillin fermentation process developed by the Process Modelling, Monitoring and Control Research Group at the Illinois Institute of Technology (Birol *et al*, 2002) has been used.

This simulator is based on the unstructured mechanistic model of Bajpai and Reuss (1980) and is capable of simulating a controlled fed-batch fermentation system. The *load variables* are: aeration rate, agitator power, substrate feed rate and substrate feed temperature; the *manipulated variables* are: acid/base and heating/cooling water flow rates; the *internal state variables* are: culture volume, generated heat, carbon dioxide, dissolved oxygen, biomass, penicillin and substrate feed concentrations; and the *controlled variables* are: pH and bioreactor temperature.

4. CONSTRUCTION OF THE INTELLIGENT SOFTWARE SENSORS

4.1 PLS Model Development

The first stage in the development of the PLS model is to generate suitable training data. In this application data from 30 batches was collected. For each batch a Pseudo-Random Binary Signal (PRBS) was applied to the substrate feedrate to ensure the data was sufficiently rich. 20 of these batches were used to train the PLS model (training batches) with the remainder used to validate the model (validating batches).

A PLS model, containing 3 latent variables, was then developed using the data. In this model the following measurements were used as input, or cause, variables: substrate feed rate, aeration rate, agitator power, substrate feed temperature, substrate concentration, dissolved oxygen concentration, co2 concentration, culture volume, pH, fermenter temperature and generated heat.

4.2 Implementation of the intelligent software sensors

Based on this model, a software sensor network containing seven software sensors has been constructed to estimate the substrate concentration, biomass concentration, penicillin concentration, dissolved oxygen concentration, co_2 concentration, pH and fermenter temperature. For all the software sensors, biomass and penicillin concentration are not used as the cause variables as they are assumed to be only available from off-line assay. For each of the other sensors, the variable concerned does not use itself as a cause variable in the PLS model for the obvious reason. The accuracy of the estimates provided by these software sensors are illustrated in figures 2 to 5 which compare the actual values with those predicted on-line by the MPLS model for a typical batch. From the figures, all software sensors provide stable and satisfactory estimations.

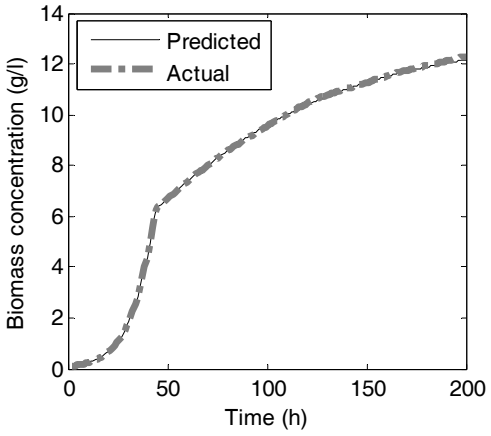

Fig. 2 Software sensor for biomass concentration

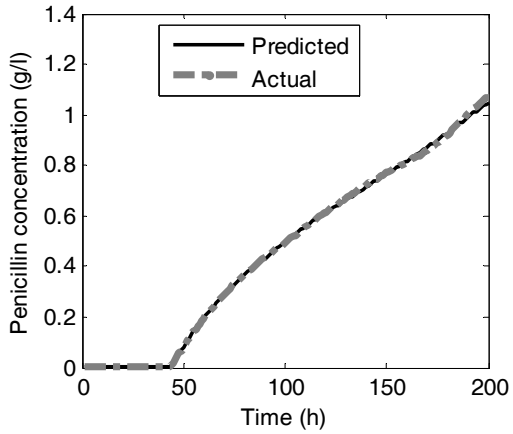

Fig. 3 Software sensor for penicillin concentration

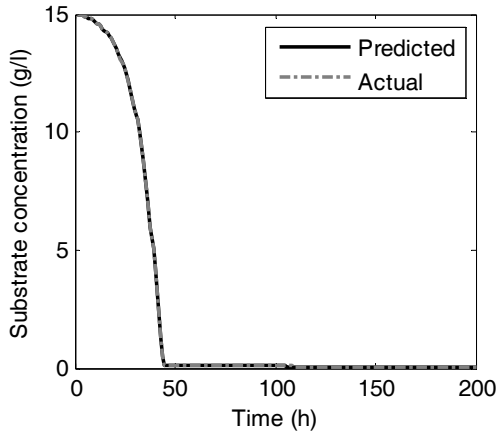

Fig. 4 Software sensor for substrate concentration

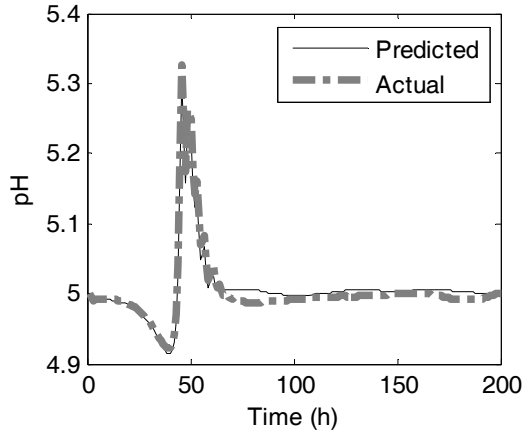

Fig. 5 Software sensor for pH

4.3 Self-diagnosis of the software sensors

An intelligent sensor should be able to perform self-diagnosis. For software sensors, sensor failures are unlikely unless the algorithms suffer convergence problems. But for data driven software sensors, their performances can deteriorate if there are faults with the variables which the software sensor's models are relied upon. This can be demonstrated by the following example. When there is a step fault occurred in the substrate temperature at 80 hours, the prediction from the software sensor for biomass concentration at 81 hours is displayed in Fig 6. As it can be seen from the figure, the software sensor no longer gives accurate prediction after the fault. This is because substrate temperature is one of the cause variables for the PLS model and the filling up method being used for online software sensor. To solve this problem, the intelligent software sensor is designed to work together with a fault detection and diagnosis regime. The online multivariate statistics for monitoring the batch in real time consist of the *SPE* and T^2 charts as well as the individual variable contributions charts. When the substrate temperature fault occurs, the *SPE* chart violates the 95% confidence limit immediately to detect the fault and the SPE contribution chart, that is produced immediately after the *SPE* limit is violated, indicates that variable 5, which is the substrate temperature, is the likely cause of the fault condition. This information is then passed to the software sensor for reconstruction of the PLS model in which substrate temperature is no longer used as a cause variable. The resulting software sensor performance is shown in Fig 7 where the prediction of the biomass concentration is back to normal after a blip. Therefore, by integrating the condition monitoring facility into the software sensors, self-diagnosis can be realised and it can improve the performances of software sensors.

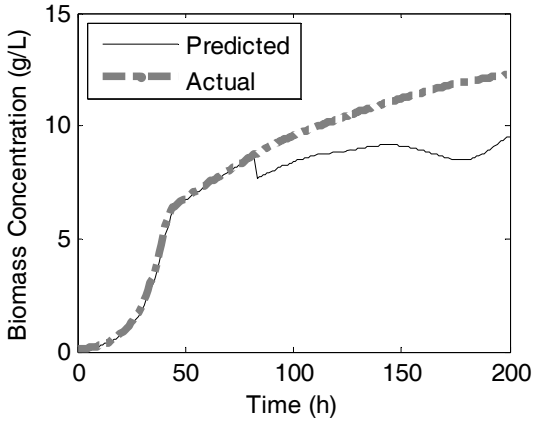

Fig. 6 Biomass concentration prediction when a fault occurred at 80h.

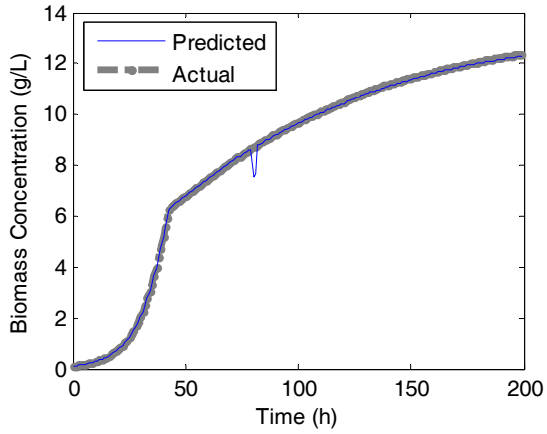

Fig. 7 Biomass concentration prediction when self-diagnosis is integrated in the software sensor.

4.4 Self-validating of the software sensors

In last section, it was demonstrated that software sensors integrated with condition monitoring functions can perform self-diagnosis for unmeasured variables. For those easily available and continuously measured variables such as the temperature, dissolved oxygen concentration and pH, the software sensors can be used to validate the outputs of the hardware sensors and provide drift compensation. In the following example, the pH software sensor's predictions are continuously compared with the hardware pH sensor's outputs. Any discrepancy between the two sensors will invoke fault detection and diagnosis for sensor drift or failure. In the example, when the drift of 0.5 is introduced into the pH sensor measurement at a time of 100 hours, the contradiction is picked up by the software sensor immediately. Following the confirmation that the pH sensor is at fault from the condition monitoring system then the software sensor is switched on to supply inferred pH sensor measurement to the controllers. The inference capabilities of the PLS model are found to be very good in this application, this is illustrated in figure 8 which compares the actual pH measurement with that predicted, or inferred, using the MPLS model and that measured using the pH sensor. This figure shows that the pH value inferred by the PLS model is very consistent with the actual pH within the reactor. As a result of the accurate inference of the pH measurement the performance of the controller, and subsequent productivity of the batch, is found to be unaffected by the pH sensor fault.

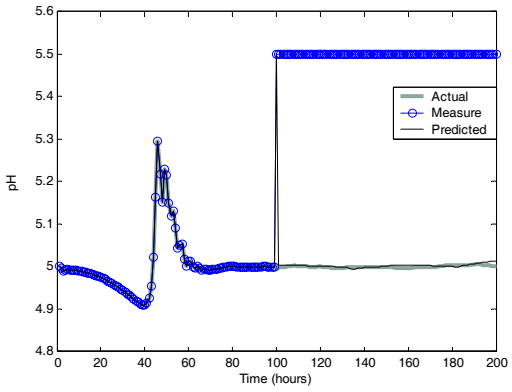

Fig. 8 Software sensor for pH

4.5 Self-calibration of the software sensors

Although the MPLS based software sensors were shown capable of providing stable and good estimates, the performance may be deteriorate as time elapses due to high level noises and disturbances or the time-variant characteristics of industrial fermentation processes caused by equipment aging, sensor and process drifting.

To deal with the time-variant processes, dynamic PLS was proposed. Li *et al* (2000) proposed an efficient recursive PCA algorithm for adaptive process monitoring by updating the mean, variance and correlation matrix of the samples. For software sensors' self-calibration, the idea can be employed to update the PLS model online using samples from off-line assays. The mean and variance of the training samples are recursively updated using methods described in Mu *et al* (2006). Fig. 9 shows the prediction of the biomass concentration from the software sensor for a particular batch and the relative error is displayed in the same figure. With the output offset updating, predictions for the same batch are improved and the relative error is significantly reduced, as can be seen from Fig. 10. The Mean Square Error (MSE) chart in Fig. 11 shows the error is halved with the online updating. This is due to that the earliest available off-line assay sample provided self-calibration facility to the software sensor.

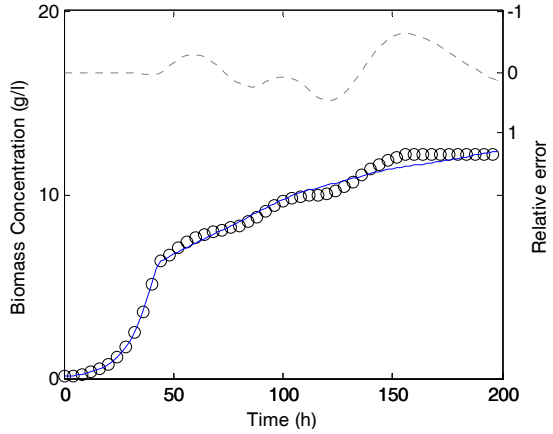

Fig. 9 Software sensor for biomass concentration without model updating

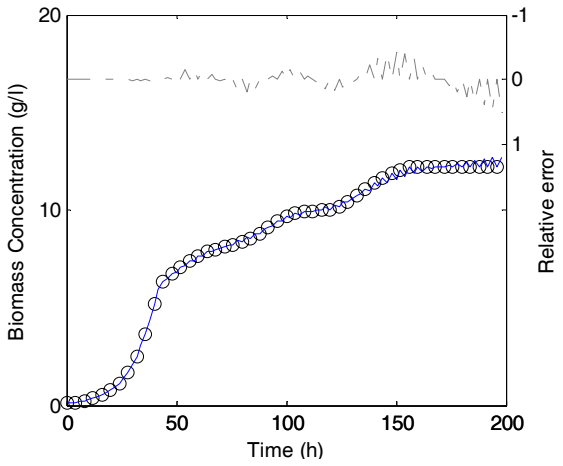

Fig. 10 Software sensor for biomass concentration with model offset updating

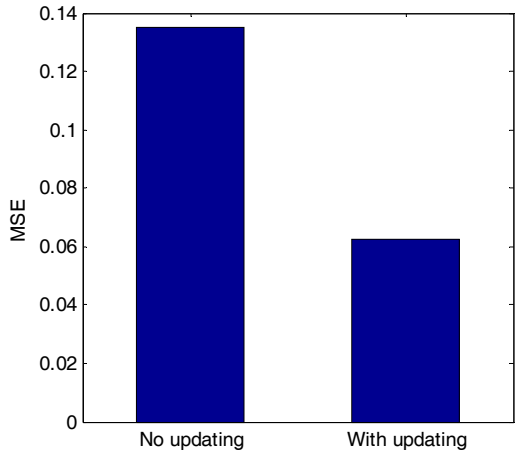

Fig. 11 MSE comparison

5. CONCLUSIONS

In this paper, intelligent software sensors have been developed based on MPLS. It is demonstrated that the software sensors not only provide real time estimation of key variables but also have the facilities of self-diagnosis, self-validation and self-calibration using available lab assay data. An application of the software sensors to a fed-batch penicillin fermentation process is presented, and significant improvements over ordinary methods have been shown in the simulation results. These intelligent software sensors can be further integrated into to a sensor management system to be used with hardware sensors and for tasks such as multi-sensor fusion.

REFERENCES

Aynsley M., Holfland A. *et al.* (1993). Artificial Intelligence and the supervison of bioprocesses. *Advances in Biotechnology*. **48**(1): 1-28.

Aubrun C., Theilliol D., Harmand J. and Steyer J. P. (2001). Software sensor design for COD estimation in an anaerobic fluidized bed reactor. *Water Science and Technology*. **43**(7): 115-122.

Bajpai R. K. and Reuss M. (1980). A Mechanistic Model for Penicillin Production. *Journal of Chemical Technology and Biotechnology*. **30**: 332-344.

Birol G., Undey C. and Cinar A. (2002). A modular simulation package for fed-batch fermentation: penicillin production. *Computers & Chemical Engineering*. 26(11): 1553-1565.

Boltryk, P. J., Harris, C. J. and White, N. M. (2005). Intelligent sensors-a generic software approach. *Journal of physics: Conference series*, 15 (1): 155-60.

Dacosta P., Kordich C., Williams D. and Gomm J. B. (1997). Estimation of inaccessible fermentation states with variable inoculum sizes. *Artificial Intelligence in Engineering*. **11**(4): 383-392.

de Assis, Adilson Jose and FILHO, Rubens M. (2000). Soft sensors development for on-line bioreactor state estimation. *Computers and chemical engineering*, **24** (2): 1099-1103.

Dong D. and McAvoy T. J. (1996). Nonlinear principal component analysis-based on principal curves and neural networks. *Computers & Chemical Engineering*. **20**(1): 65-78.

Gallagher N. B., Wise B. M. and Stewart C.W. (1996). Application of multiway principal components analysis to nuclear waste storage tank monitoring. *Computers and Chemical Engineering*. **20S**: 739-744

Golobic I., Gjerkes H., Bajsic I. and Malensek J. (2000). Software sensor for biomass concentration monitoring during industrial fermentation. *Instrumentation Science and Technology*. **28**(4): 323-334.

Gudi R. D., Shah S. L. and Gray M. R. (1995). Adaptive multirate state and parameter estimation strategies with application to a bioreactor. *AIChE Journal*. **41**(11): 2451-2464.

James S., Legge R. and Budman H. (2002). Comparative study of black-box and hybrid estimation methods in fed-batch fermentation. *Journal of Process Control*. **12**(1): 113-121.

Joseph, B. (1999). Tutorial on inferential control and its applications. *Proceedings of the American Control Conference*, **5**: 3106-3118

Lakshminarayanan S., Gudi R. D., Shah S. L. and Nandakumar K. (1996). Monitoring batch proceses using multivariate statistical tools: extensions and practical issues. *Proceeding of IFAC World Congress. San Francisco*: 241-246

Lennox B., Montague G. A., Hiden H. G., Kornfeld G. and Goulding P. R. (2001). Process monitoring of an industrial fed-batch fermentation. *Biotechnology and Bioengineering*. **74**(2): 125-135.

Li, W., Yue, H., Sergio, V. and Qin S. J. (2000). Recursive PCA for adaptive process monitoring. *Journal of process control*. **10** (5): 471-486.

Montague G. A. (1997). Monitoring and control of fermenters. *Institution of Chemical Engineers*.

Mu, S., Zeng, Y., Liu, R., Wu, P., Su, H. and Chu, J. (2006) Online dual updating with recursive PLS model and its application in predicting crystal size of purified terephthalic acid (PTA) process. *Journal of Process Control*. 16(6): 557-566.

Nomikos P. and MacGregor J. F. (1994). Monitoring batch processes using multiway principal component analysis. *AIChE Journal*. **40**(8): 1361-1373.

Qin S. J. and McAvoy T. J. (1992). Nonlinear PLS modeling using neural networks. *Computers & Chemical Engineering*. **16**(4): 379-391.

Tham M. T., Montague G. A., Glassey J. and Willis M. J. (2002). Practical inferential estimation using artificial neural networks. *Measurement and Control*. **35**(1): 5-9.

Undey C., Tatara E., Williams B. A., Birol G. and Cinar A. (2000). Hybrid supervisory knowledge-based system for monitoring penicillin fermentation. *American Control Conference*, Chicago, IL, USA, **6**: 3944-3948.

Zhang, H., Z. Zouaoui and B. Lennox (2005). A comparative study of soft-Sensing methods for fed-batch fermentation processes. *IFAC World Congress*, Prague.