Proceedings of the 32nd Conference
on Decision and Control
San Antonio, Texas • December 1993

Partial Least Squares Regression for Recursive System Identification

S. Joe Qin, *IEEE member*

Fisher-Rosemount Systems, Inc.
1712 Centre Creek Drive, Austin, TX 78754

Abstract

Industrial processes usually involve a large number of variables, many of which vary in a correlated manner. To identify a process model which has correlated variables, an ordinary least squares approach demonstrates ill-conditioned problem and the resulting model is sensitive to changes in sampled data. In this paper, a recursive *partial least squares* (PLS) regression is used for on-line system identification and circumventing the ill-conditioned problem. The partial least squares method is used to remove the correlation by projecting the original variable space to an orthogonal latent space. Application of the proposed algorithm to a chemical process modeling problem is discussed.

1 Introduction

The partial least squares (PLS) regression has been widely applied to chemometrics and chemical industries for static data analysis (Lindberg, et al., 1983; Wold, et al. 1984, Geladi & Kowalski, 1986, Fuller, et al., 1988; Haaland, et al., 1988; Martens & Naes, 1989; Piovoso, et al, 1992). Inspired from principal component analysis (PCA) and principal component regression (PCR), the PLS regression is able to give a robust solution in the case of collinear or correlated input variables, where the ordinary least squares regression gives rise to the ill-conditioned problem (Wold, 1966). The PLS regression removes correlation in the input variables by carrying out orthogonal projections from input space to a latent space. Linear regression is then done in the latent space, which makes the regression solution well-defined (Geladi & Kowalski, 1986).

The PLS regression has a number of orthogonal properties (Höskuldsson, 1988). Owing to these properties, the PLS regression has been used for dynamic chemical process modeling and control (Ricker, 1988; Wise et al., 1990; MacGregor, et al., 1991). Ricker (1988) applied the PLS regression for finite impulse response modeling and related a special case of PLS regression to singular value decomposition. Wise, et al. (1990) have demonstrated some properties of the PLS regression for dynamic modeling. Ljung (1991) addressed the point of using PLS and PCA for dynamic system identification. When a system identification method based on an ordinary least squares method is used to identify such a process with correlated inputs, one will have an ill-conditioned problem and the resulting model may not be reliable. The PLS regression is particularly useful in chemical process modeling because a chemical process usually has a large dimension of input variables, many of which undergo slow changes in a correlated manner.

In most of the PLS applications to date, the PLS regression is used as a batch type of identification approach. In other words, the data are collected and stored in a computer at first, then the PLS regression is carried out to model the data. This approach has a number of limitations: (i) the resulting model is hard to update to track possible changes of process dynamics; and (ii) when the data base is large in size, it may present memory problems for computers. In this paper, a recursive PLS regression proposed by Helland, et al. (1991) is modified to formulate a recursive PLS algorithm for dynamic system identification. Several benefits can be achieved using the recursive PLS method. First, the recursive PLS method updates the model when a new sample is available. Second, the recursive PLS method is able to adapt process correlation structure changes as well as process dynamics changes. If some variables were not correlated before but are correlated at present for some reasons, the recursive PLS method can pick up the changes. Third, the recursive PLS method can compress the data and get rid of potential memory problems.

The organization of the paper is as follows. Section one addresses the need for a recursive PLS identification method. Section two discusses the batch type PLS method and its use in dynamic modeling. Section three presents how to use the PLS method for recursive identification. Section four addresses application of the recursive PLS algorithm to a catalytic

0191-2216/93/$3.00 © 1993 IEEE

reformer in which correlated inputs exist. The final section gives conclusions.

2 PLS Method for Dynamic Modeling

2.1 Dynamic Process Modeling

A dynamic chemical process is usually governed by a set of nonlinear differential equations. Since the process data can be collected only at sampling instances, the process models are usually described in a discrete form. A simple stochastic model is the so-called auto-regressive model with exogenous inputs (ARX, Ljung, 1987),

$$\mathbf{y}(k)=\sum_{i=1}^{n_y}\mathbf{A}_i\mathbf{y}(k-i)+\sum_{j=1}^{n_u}\mathbf{B}_j\mathbf{u}(k-j)+\mathbf{v}(k) \tag{1}$$

where $\mathbf{y}(k)$, $\mathbf{u}(k)$ and $\mathbf{v}(k)$ are the process output, input, and noise vectors, respectively, with appropriate dimensions for multi-input-multi-output systems. $\mathbf{A}_i$ and $\mathbf{B}_j$ are matrices of model coefficients to be identified. n_y and n_u are time lags for the output and input, respectively. In order for the PLS method to build an ARX model, the following vector of variables is defined,

$$\mathbf{x}^T(k)=[\mathbf{y}^T(k-1),\mathbf{y}^T(k-2),\ldots, \mathbf{y}^T(k-n_y),\mathbf{u}^T(k-1),\mathbf{u}^T(k-2),\ldots,\mathbf{u}^T(k-n_u)] \tag{2}$$

whose dimension is denoted as m. Then two data matrices can be formulated as follows assuming the number of data records is n,

$$\mathbf{X}=[\mathbf{x}(1),\mathbf{x}(2),\ldots,\mathbf{x}(n)]^T\in\Re^{n\times m} \tag{3}$$

$$\mathbf{Y}=[\mathbf{y}(1),\mathbf{y}(2),\ldots,\mathbf{y}(n)]^T\in\Re^{n\times p} \tag{4}$$

where p is the dimension of output vector $\mathbf{y}(k)$. Defining all unknown parameters in the ARX model as,

$$\mathbf{C}=[\mathbf{A}_1,\mathbf{A}_2,\ldots,\mathbf{A}_{n_y},\mathbf{B}_1,\mathbf{B}_2,\ldots,\mathbf{B}_{n_u}]^T\in\Re^{m\times p} \tag{5}$$

Eqn.1 can be re-written as

$$\mathbf{y}(k)=\mathbf{C}^T\mathbf{x}(k)+\mathbf{v}(k) \tag{6}$$

and the two data matrices Y and X can be related as

$$\mathbf{Y}=\mathbf{XC}+\mathbf{V} \tag{7}$$

An ordinary least squares solution to C is given by

$$\mathbf{C}_{LS}=(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} \tag{8}$$

Assuming that the noise v(k) has zero mean and covariance as

$$Cov(\mathbf{v}(k),\mathbf{v}(k))=\sigma^2\mathbf{I}_p \tag{9}$$

where $\mathbf{I}_p$ is the identity matrix, the covariance of the least squares estimates is

$$Cov(\mathbf{C}_{LS},\mathbf{C}_{LS})=(\mathbf{X}^T\mathbf{X})^{-1}\sigma^2 \tag{10}$$

Therefore, if the process inputs or outputs are exactly collinear, the least squares solution is singular. Even if the process variables are not exactly collinear but the data covariance matrix $\mathbf{X}^T\mathbf{X}$ has a large condition number, the resulting least squares estimate will have large variance and thus is not reliable.

In many process control applications such as dynamic matrix control (Cutler, et al, 1979), a finite impulse response (FIR) model is often more convenient, which can be described as

$$\mathbf{y}(k)=\sum_{j=1}^{N}\mathbf{B}_j\mathbf{u}(k-j)+\mathbf{v}(k) \tag{11}$$

where N is the time lag which corresponds to the process settling time. Similar to the ARX model, two data matrices X and Y can be arranged according to the above equation. A least squares solution can be found for the FIR model if it is not ill-conditioned. However, if the process input variables are correlated, an ordinary least squares solution will also result in large variance in the estimate.

2.2 PLS Regression

Here the PLS regression method is briefly discussed so as to establish a background for presenting the neural net PLS method. If the process variables are collinear, the PLS method overcomes the ill-conditioned problem by decomposing matrices X and Y into bilinear relations plus residual matrices

$$\mathbf{X}=\mathbf{t}_1\mathbf{p}_1^T+\mathbf{E}_1 \tag{12}$$

$$\mathbf{Y}=\mathbf{u}_1\mathbf{q}_1^T+\mathbf{F}_1 \tag{13}$$

where $\mathbf{t}_1$ and $\mathbf{u}_1$ are latent score vectors of the first PLS factor, and $\mathbf{p}_1$ and $\mathbf{q}_1$ are corresponding loading vectors. All four vectors are determined such that the residuals $\mathbf{E}_1$ and $\mathbf{F}_1$ are minimized. The PLS decomposition is very similar to calculating the first principal components for X and Y, but PLS does the calculation such that the correlation between X and Y is emphasized (Geladi and Kowalski, 1986). The above two equations formulate a PLS outer model. After the outer calculation, the score vectors are related by a linear inner model:

$$\mathbf{u}_1=b_1\mathbf{t}_1+\mathbf{r}_1 \tag{14}$$

where b_1 is a coefficient which is determined by minimizing the residual $\mathbf{r}_1$. After going through the above calculation, the second factor is calculated by decomposing the residuals $\mathbf{E}_1$ and $\mathbf{F}_1$ using the same procedure as for the first factor. This procedure is repeated until the last factor a is calculated, which leaves almost no information in the residual matrices $\mathbf{E}_h$ and $\mathbf{F}_h$. The overall PLS algorithm is summarized below according to Geladi and Kowalski (1986). Note

that a minor difference of this algorithm from the one in Geladi and Kowalski (1986) is that the latent variables $\mathbf{t}_h$ is normalized instead of $\mathbf{w}_h$ and $\mathbf{p}_h$. This modification makes the presentation of a recursive PLS regression algorithm easier.

1. Scale $\mathbf{X}$ and $\mathbf{Y}$ to zero-mean and unit-variance, or otherwise specified with particular scaling factors. Initialize $\mathbf{E}_0 := \mathbf{X}$, $\mathbf{F}_0 := \mathbf{Y}$, and $h := 0$.
2. Let $h := h + 1$ and take $\mathbf{u}_h$ as some column of $\mathbf{F}_{h-1}$.
3. Calculate PLS outer model:

$$\mathbf{w}_h = \mathbf{E}_{h-1}^T \mathbf{u}_h \Big/ \left\| \mathbf{E}_{h-1}^T \mathbf{u}_h \right\| \quad (15)$$

$$\mathbf{t}_h = \mathbf{E}_{h-1} \mathbf{w}_h \quad (16)$$

$$\mathbf{q}_h = \mathbf{F}_{h-1}^T \mathbf{t}_h \Big/ \left\| \mathbf{F}_{h-1}^T \mathbf{t}_h \right\| \quad (17)$$

$$\mathbf{u}_h = \mathbf{F}_{h-1} \mathbf{q}_h \quad (18)$$

Iterate this step until it converges or until the maximum number of iterations is reached.

4. Calculate the X-loadings and normalize latent vector $\mathbf{t}_h$,

$$\mathbf{p}_h = \mathbf{E}_{h-1}^T \mathbf{t}_h / \mathbf{t}_h^T \mathbf{t}_h \quad (19)$$

$$\mathbf{w}_h := \mathbf{w}_h / \left\| \mathbf{t}_h \right\| \quad (20)$$

$$\mathbf{t}_h := \mathbf{t}_h / \left\| \mathbf{t}_h \right\| \quad (21)$$

5. Find the inner model:

$$b_h = \mathbf{u}_h^T \mathbf{t}_h / \mathbf{t}_h^T \mathbf{t}_h \quad (22)$$

6. Calculate the residuals:

$$\mathbf{E}_h = \mathbf{E}_{h-1} - \mathbf{t}_h \mathbf{p}_h^T \quad (23)$$

$$\mathbf{F}_h = \mathbf{F}_{h-1} - b_h \mathbf{t}_h \mathbf{q}_h^T \quad (24)$$

7. Return to Step 2 until all principal factors are calculated.

It should be noted that different versions of the PLS algorithm can be found and they are essentially equivalent (Manne, 1987). A useful feature of the PLS algorithm is that the loading vectors $\mathbf{w}_h, \mathbf{q}_h$ and the latent score vectors $\mathbf{t}_h, \mathbf{u}_h$ are eigenvectors of $\mathbf{E}_{h-1}^T \mathbf{F}_{h-1} \mathbf{F}_{h-1}^T \mathbf{E}_{h-1}$, $\mathbf{F}_{h-1}^T \mathbf{E}_{h-1} \mathbf{E}_{h-1}^T \mathbf{F}_{h-1}$, $\mathbf{E}_{h-1} \mathbf{E}_{h-1}^T \mathbf{F}_{h-1} \mathbf{F}_{h-1}^T$, and $\mathbf{F}_{h-1} \mathbf{F}_{h-1}^T \mathbf{E}_{h-1} \mathbf{E}_{h-1}^T$, respectively. More interestingly, the latent vectors, $\mathbf{t}_h (h = 1,2,\ldots,a)$, are orthogonal (Höskuldsson; 1988). The total number of factors needed is usually determined by cross-validation (Geladi and Kowalski, 1986), although elsewhere an F-test is suggested (Haaland, et al, 1988). The cross-validation method is used to avoid over-fitting the training data. A typical way of doing cross-validation is to leave one or several samples out at a time, and then train the model with the remaining data. After training, the model is tested on the samples which are not used in training. This procedure is repeated until every sample has been left out once. Summing up all the test errors over each factor, which is known as the *predicted sum of squares* (PRESS) error, the optimal number of factors is chosen as the location of the minimum of PRESS errors. One can see that the cross-validation method can be quite laborious, but it is useful in determining the number of factors. More details of the method can be found in Stone (1978) and Geladi and Kowalski (1986).

It can be shown that the PLS algorithm is a particular way of doing multiple linear regression which has robust prediction properties. If the number of factors is chosen as the number of input variables, the PLS method is equivalent to ordinary least squares. As a matter of fact, the PLS technique is widely applied, even when the data are not highly correlated. The recent work of Stone and Brooks (1990) shows that ordinary least squares, partial least squares and principal component regression are simply three particular cases of what is called *continuum regression.*

3 Recursive PLS Regression

In this section, the recursive PLS regression algorithm in Helland, et al. (1991) is modified for dynamic system identification. To illustrate the recursive updating of the PLS parameters, the number of samples n is explicitly used as a subscription in later derivation. By putting the PLS vectors into matrix forms, Eqns.23 and 24 can be rewritten as follows,

$$\mathbf{X}_n = \mathbf{T}_n \mathbf{P}_n^T \quad (25)$$

$$\mathbf{Y}_n = \mathbf{T}_n \mathbf{B}_n \mathbf{Q}_n^T + \mathbf{F}_n \quad (26)$$

where

$$\mathbf{T}_n = [\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_m]$$
$$\mathbf{P}_n = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_m]$$
$$\mathbf{B}_n = diag\{b_1, b_2, \ldots, b_m\}$$
$$\mathbf{Q}_n = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_m]$$

Since the number of factors is equal to the number of input dimensions, m, the residual matrix for $\mathbf{X}_n$ is vanished, as indicated in Eqn.25. However, the residual matrix $\mathbf{F}_n$ is generally not zero because of measurement noise. If the covariance matrix $\mathbf{X}_n^T \mathbf{X}_n$ is singular, a least squares solution is ill-conditioned. By minimizing the output residuals using least squares criterion, the following relation can be derived,

$$(\mathbf{X}_n^T \mathbf{X}_n) \mathbf{C}_n = \mathbf{X}_n^T \mathbf{Y}_n \quad (27)$$

Since columns of $\mathbf{T}_n$ are mutually orthonormal, the following relation can be derived through Eqns. 25 and 26,

$$\mathbf{X}_n^T \mathbf{X}_n = \mathbf{P}_n \mathbf{T}_n^T \mathbf{T}_n \mathbf{P}_n^T = \mathbf{P}_n \mathbf{P}_n^T \quad (28)$$

$$\mathbf{X}_n^T \mathbf{Y}_n = \mathbf{X}_n^T \mathbf{T}_n \mathbf{B}_n \mathbf{Q}_n^T + \mathbf{X}_n^T \mathbf{F}_n = \mathbf{P}_n \mathbf{B}_n \mathbf{Q}_n^T \quad (29)$$

The fact that the residual matrix $\mathbf{F}_n$ is orthogonal to $\mathbf{X}_n$ in least squares regression is used to derive Eqn.29. Substituting the above relations into Eqn.27, one gets,

$$(\mathbf{P}_n \mathbf{P}_n^T) \mathbf{C}_n = \mathbf{P}_n \mathbf{B}_n \mathbf{Q}_n^T \quad (30)$$

If the covariance matrix $\mathbf{X}^T\mathbf{X}$ is singular, $\mathbf{P}_n\mathbf{P}_n^T$ is also singular. Therefore, a generalized inverse is given in the PLS regression to find the coefficient matrix:

$$\mathbf{C}_n^{PLS} = (\mathbf{P}_n\mathbf{P}_n^T)^+\mathbf{P}_n\mathbf{B}_n\mathbf{Q}_n^T \tag{31}$$

where $(\bullet)^+$ denotes the generalized inverse defined through the PLS regression algorithm.

When a new sample at $n+1$ is available, an updated estimate needs to be found using the augmented data matrices:

$$\mathbf{X}_{n+1}^T = [\mathbf{X}_n^T \quad \mathbf{x}(n+1)] \tag{32}$$

$$\mathbf{Y}_{n+1}^T = [\mathbf{Y}_n^T \quad \mathbf{y}(n+1)] \tag{33}$$

Therefore, the PLS regression at $n+1$ can be updated as

$$\begin{aligned}\mathbf{C}_{n+1}^{PLS} &= (\mathbf{X}_{n+1}^T\mathbf{X}_{n+1})^+\mathbf{X}_{n+1}^T\mathbf{Y}_{n+1} \\ &= (\mathbf{P}_n\mathbf{P}_n^T + \mathbf{x}(n+1)\mathbf{x}^T(n+1))^+(\mathbf{P}_n\mathbf{B}_n\mathbf{Q}_n^T + \mathbf{x}(n+1)\mathbf{y}^T(n+1)) \\ &= ([\mathbf{P}_n \quad \mathbf{x}(n+1)][\mathbf{P}_n \quad \mathbf{x}(n+1)]^T)^+[\mathbf{P}_n \quad \mathbf{x}(n+1)] \\ &\quad [\mathbf{B}_n\mathbf{Q}_n \quad \mathbf{y}(n+1)]^T\end{aligned} \tag{34}$$

By comparing the above equation with

$$\mathbf{C}_{n+1}^{PLS} = (\mathbf{P}_{n+1}\mathbf{P}_{n+1}^T)^+\mathbf{P}_{n+1}\mathbf{B}_{n+1}\mathbf{Q}_{n+1}^T \tag{35}$$

one gets,

$$\mathbf{P}_{n+1}\mathbf{P}_{n+1}^T = \mathbf{P}_n\mathbf{P}_n^T + \mathbf{x}(n+1)\mathbf{x}^T(n+1) \tag{36}$$

and

$$\mathbf{P}_{n+1}\mathbf{B}_{n+1}\mathbf{Q}_{n+1}^T = \mathbf{P}_n\mathbf{B}_n\mathbf{Q}_n^T + \mathbf{x}(n+1)\mathbf{y}^T(n+1) \tag{37}$$

Hence, one can find matrix $\mathbf{P}_{n+1}$ by solving Eqn.36 and find $\mathbf{B}_{n+1}\mathbf{Q}_{n+1}^T$ through Eqn.37. By using the fact that columns of $\mathbf{Q}_{n+1}$ are normalized, $\mathbf{B}_{n+1}$ and $\mathbf{Q}_{n+1}$ can be found separately. Then all other vectors can be calculated through the relations in the PLS regression algorithm.

Although the aforementioned approach is a recursive algorithm, finding $\mathbf{P}_{n+1}$ and $\mathbf{B}_{n+1}\mathbf{Q}_{n+1}^T$ can be quite laborious. Instead of going through this algorithm, one can find the PLS update by simply carrying out PLS regression for matrices $[\mathbf{P}_n \quad \mathbf{x}(n+1)]^T$ and $[\mathbf{B}_n\mathbf{Q}_n \quad \mathbf{y}(n+1)]^T$ through the PLS algorithm given in Eqns.15-24. Therefore, assuming the PLS model at n is known, the PLS model at $n+1$ can be derived based on $\mathbf{P}_n, \mathbf{B}_n, \mathbf{Q}_n$, and new data $\mathbf{x}(n+1), \mathbf{y}(n+1)$. One does not have to deal with the data before $n+1$, because all the information is compressed in the PLS parameters.

Assuming the PLS algorithm is carried out through Eqns.15-24, the regression coefficient can be compounded as follows,

$$\mathbf{C}_{n+1}^{PLS} = \mathbf{W}_{n+1}^*\mathbf{B}_{n+1}\mathbf{Q}_{n+1}^T \tag{38}$$

where

$$\mathbf{W}_{n+1}^* = [\mathbf{w}_1^*, \mathbf{w}_2^*, \ldots, \mathbf{w}_m^*] \tag{39}$$

and

$$\mathbf{w}_i^* = \prod_{h=1}^{i-1}(\mathbf{I}_m - \mathbf{w}_h\mathbf{p}_h^T)\mathbf{w}_i \tag{40}$$

The above derivation can be found in Höskoldsson (1988) and Qin, et al. (1992).

It should be noted that in the previous derivation of the recursive PLS algorithm, the number of PLS factors is chosen as the dimension of the input vector. However, to make a prediction, the number of factors is usually less than the dimension of the input vector, $\mathbf{x}(k)$. Therefore, in using the recursive PLS algorithm, the PLS regression can be carried out through all factors. When used for prediction, the PLS model is truncated at the best number of factors, which can be determined by cross-validation or F-test validation. In practice, the number of factors can vary, as the process may change through time. For example, when some variables were correlated before, but are not correlated at present, an increase on the number of factors is expected.

4 Application to a Chemical Process

A process model is usually needed in advanced process control such as dynamic matrix control. The process model can be derived through first principles modeling, but system identification is often used because the first principles models are hard to find for many chemical processes. Figure 1 illustrates a catalytic reforming system in petroleum refineries. A typical catalytic reformer is composed of a number of reactors, heaters, and a fractioner. The catalytic performer usually performs four major reactions: (i) dehydrogenation of naphthenes to aromatics; (ii) dehydrocyclization of paraffin to aromatics; (iii) isomerization; and (iv) hydrocracking. It is usually hard to derive a mathematical model for such a chemical process. Further, many process variables are involved in this reforming system and many of these variables correlate to one another. Therefore, this process is a good candidate for applying PLS regression to derive a dynamic model.

The objective of the application is to derive a process model to predict important quality variables so as to implement better control. The process data can be collected from distributed control systems on-line. In this study, however, the data were collected before hand from five input variables and two output variables to test the PLS algorithm. About two thirds of the data are used for modeling, the other third are used for testing. An ARX model structure can be used to describe the process. A preliminary modeling result is shown in Figure 2, where the two process outputs are compared with the model prediction. The number of factors used

in this model is seven, which is determined from cross-validation.

After a dynamic process model is derived, it can be used in dynamic matrix control to predict the process outputs (Cutler, et al., 1979). Another way of using this model is for supervisory control. The predicted future outputs can be used as a guideline for the human operators to adjust set points for low level controllers. The supervisory control approach is useful when the predict process variables are directly related product quality. As new data are available along process operation, the process model can be updated using recursive PLS regression in a real-time manner. Slow process changes during operation will be captured in the model using this approach.

5 Conclusions

A recursive partial least squares regression approach is presented in this paper for dynamic system identification. If a system to be identified has correlated inputs, ordinary least squares methods are ill-defined and cannot give a robust process model. The PLS regression circumvents the correlation problem by projecting to a latent space in which the latent vectors are orthogonal. Application of the recursive PLS regression includes identification of chemical processes and refineries.

References

[1] Cutler, C. & B.L. Ramaker (1979). Dynamic matrix control---A computer control algorithm. *AIChE Annual Meeting*, Houston, TX.

[2] Fuller, M.P., Ritter, G.L. & Draper, C.S. (1988). Partial least-squares quantitative analysis of infrared spectroscopic data. Part I: algorithm implementation; Part II: application to detergent analysis. *Appl. Spectrosc.*, **42**, 217-236.

[3] Geladi, P. & Kowalski, B.R. (1986). Partial least-squares regression: a tutorial. *Analyt. Chim. Acta*, **185**, 1-17.

[4] Haaland, D.M. & Thomas, E.V. (1988). Partial least squares methods for spectral analysis: 1. Relation to other quantitative calibration methods and the extraction of qualitative information. 2. Application to simulated and glass spectral data. *Analyt. Chem.*, **60**, 1193-1208.

[5] Helland, K., H.E. Berntsen, O.S. Borgen & H. Martens (1991). Recursive algorithm for partial least squares regression. *Chemometrics and Intell. Lab. Syst.*, **14**, 129-137.

[6] Höskuldsson, A. (1988). PLS regression methods. *J. Chemometrics*, **2**, 211-228.

[7] Lindberg, W., Persson, J. & Wold, S. (1983). Partial least squares method for spectrofluorimetric analysis of mixtures of humic and ligninsulfonate. *Analyt. Chem.*, **55**, 643-648.

[8] Ljung, L. (1987). *System Identification: Theory for the User*, Prentice-Hall, Englewood Cliffs. New Jersey.

[9] Ljung, L. (1991). Issues in system identification. *IEEE Control System Magazine*. **11**(1), 25-29.

[10] MacGregor, J.F., T.E. Marlin, J. Kresta & B. Skagerberg (1991). Multivariate statistical methods in process analysis and control. *Proc. of the Chemical Process Control Conference, CPC-IV*, S. Padre Island, Feb. 18-22.

[11] Manne, R. (1987). Analysis of two partial least squares algorithms for multivariate calibration. *Chemom. & Int. Lab. Syst.*, **2**, 187-197.

[12] Martens, H. & Naes, T. (1989). *Multivariate Calibration*, Wiley, New York.

[13] Piovoso, M. & Owens, A.J. (1991). Sensor data analysis using artificial neural networks. *Proc. of Chemical Process Control (CPC-IV)*, pp101-118, Padre Island, TX.

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

[15] Ricker, N.L. (1988). Use of biased least-squares estimators for parameters in descrete-time pulse-response models. *Ind. Eng. Chem. Res.*, **27**(2), 343-350.

[16] Stone, M. (1978). Cross-validation: a review. *Math. Operationsforch. Statist., Ser. Statist.*, **9**(1).

[17] Stone, M. & Brooks, R.J. (1990). Continuum regression: cross-validated sequentially constructed prediction embracing ordinary least squares, partial least squares and principal component regression. *J. R. Statist. Soc. Ser. B*, **52**, 237-269.

[18] Wise, B.M. & N.L. Ricker (1990). The effect of biased regression on the identification of FIR and ARX models. *AIChE Annual Meeting*, Chicago, IL, November.

[19] Wold, H. (1966). Nonlinear estimation by iterative least squares procedures. In *Research Papers in Statistics*, (Edited by F. David), Wiley, New York.

[20] Wold, S., A. Ruke, H. Wold & W.J. Dunn III (1984). The collinearity problem in linear regression, the partial least squares (PLS) approach to generalized inverses. *SIAM J. Sci. Stat. Comput.*, **5**(3), 735-743.

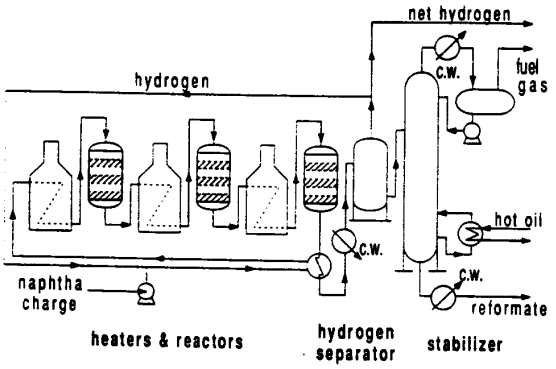

Figure 1. The process flow diagram of a catalytic reforming system.

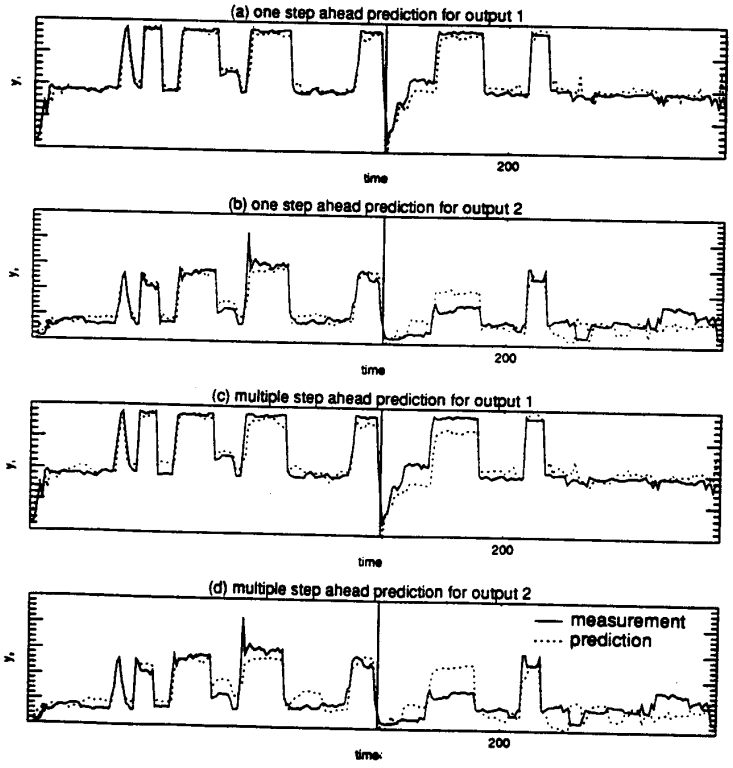

Figure 2. Prediction of the two process outputs of a catalytic reformer using partial least squares regression. Solid line: measurement; dotted line: prediction.