MPE — Volume 2, pp. 63-93
Reprints available directly from the publisher
Photocopying permitted by license only

PARTIAL LEAST-SQUARES: THEORETICAL ISSUES AND ENGINEERING APPLICATIONS IN SIGNAL PROCESSING

FREDRIC M. HAM and IVICA KOSTANIC

Florida Institute of Technology, 150 West University Boulevard, Melbourne, Florida 32901-6988

Accepted for publication by Dr. V. Lakshmikantham

(Received March 20, 1995)

In this paper we present partial least-squares (PLS), which is a statistical modeling method used extensively in analytical chemistry for quantitatively analyzing spectroscopic data. Comparisons are made between classical least-squares (CLS) and PLS to show how PLS can be used in certain engineering signal processing applications. Moreover, it is shown that in certain situations when there exists a linear relationship between the independent and dependent variables, PLS can yield better predictive performance than CLS when it is not desirable to use all of the empirical data to develop a calibration model used for prediction. Specifically, because PLS is a *factor analysis* method, optimal selection of the number of PLS factors can result in a calibration model whose predictive performance is considerably better than CLS. That is, factor analysis (*rank reduction*) allows only those features of the data that are associated with information of interest to be retained for development of the calibration model, and the remaining data associated with noise are discarded. It is shown that PLS can yield physical insight into the system from which empirical data has been collected. Also, when there exists a non-linear cause-and-effect relationship between the independent and dependent variables, the PLS calibration model can yield prediction errors that are much less than those for CLS. Three PLS application examples are given and the results are compared to CLS. In one example, a method is presented using PLS for parametric system identification. Using PLS for system identification allows simultaneous estimation of the system dimension and the system parameter vector associated with a minimal realization of the system.

KEYWORDS: *Partial least-squares*; *factor analysis*; *overfitting*; *noise reduction*; *estimation*; *aystem identification*; *minimal realization*

Notation. *Note*: All matrices and vectors have real entries.

$\mathfrak{R}$, $\mathfrak{R}^{n\times m}$, $\mathfrak{R}^{n\times n}$	real numbers, $n \times m$ matrices, $n \times n$ matrices
$\mathfrak{R}^{m\times m}$, $\mathfrak{R}^{n}$, $\mathfrak{R}^{m}$	$m \times m$ matrices, $\mathfrak{R}^{n \times 1}$, $\mathfrak{R}^{m \times 1}$
$\boldsymbol{I}_m$, $(\ )^T$, $(\ )^{-1}$	$m \times m$ identity matrix, matrix or vector transpose, matrix inverse
$\boldsymbol{b}_f$	PLS calibration model (final calibration coefficients)
$\boldsymbol{e}_y$	PLS residual vector of $\boldsymbol{y}$
$\boldsymbol{E}_z$	PLS residual matrix of $\boldsymbol{Z}$
f	subscript to denote *final*
f_s	sampling frequency, Hz
$f_s/2$	Nyquist sampling frequency, Hz
h	PLS factors
h^O	optimal number of PLS factors
k	discrete time index

$\hat{t}_h$	PLS score (latent variable) vector
T_s	sampling period, sec
$\hat{v}_h$	scalar PLS regression coefficient (inner relationship)
$\hat{w}_h$	PLS weight loading vector
y	dependent variable block
Z	independent variable block

1. INTRODUCTION

We present in this paper the partial least-squares (PLS) regression method [1–5]. Partial least-squares is considered as one of several stătistical modeling methods used in analytical chemistry. This group of methods used in analytical chemistry for quantitatively analyzing spectroscopic data is referred to as *chemometrics* [6] and [4]. Theoretical issues are addressed pertaining to PLS applied to engineering problems encountered in signal processing. Examples are presented which illustrate the applicability, and in some cases the superior performance, of PLS to selected engineering problems in signal processing, such as model development from empirical data used for predicting certain key system parameters, e.g., parametric system identification. Applying PLS regression to the parametric system identification problem allows simultaneous estimation of the system dimension, as well as yielding an estimate of the system parameter vector associated with a minimal realization of the system. It will be shown that under certain conditions PLS is identical to classical least-squares (CLS) [3], however, in certain cases PLS can out perform CLS for situations when it is not advantageous to use all of the available data. That is, in the case of CLS, there does not exist a systematic method to assess the entire data set to select pertinent features from the data for calibration model development that will yield optimal performance. However, in the case of PLS, a systematic method does exist, i.e., factor analysis, which can result in an optimized calibration model.

PLS was originally introduced by Wold [7] and [8], and arose as a practical solution to data-analytic problems in econometrics and social sciences. The basic problem is to *fit* a calibration model to empirical data, and use this model to predict certain quantities given a set of test data as input to the calibration model after the training phase. PLS is sometimes referred to in the context of abstract *factor analysis*, because physically significant quantities within the physical system can not always be identified directly with the mathematical modeling process [9]. However, if properly interpreted, the mathematical basis which constitutes the factor analysis method of PLS can yield very powerful information relating to certain key features within the physical setting from which the data were extracted and used in the PLS modeling process. Moreover PLS, which is considered as a *full-spectrum* technique, is one of several factor analysis methods that are available and used extensively in analytical chemistry to quantitatively analyze spectroscopic data. When the data contains noise, principal component regression (PCR) [2] and [5], another factor analysis technique based on principal component analysis (PCA), can reduce errors optimally (in a least-squares sense), however, the resulting data compression does not yield information specifically related to physical quantities [3]. However, with PLS, overall reduction in noise is not necessarily optimal, but the PLS basis vectors (data compression) that are generated can be more easily related to physical quantities [3].

In the physical setting, when the cause-and-effect relationship is non-linear, i.e., a non-linear relationship between the independent and dependent variables, CLS in many cases fails to properly predict certain quantities. However, PLS can *fit* a piece-wise linear calibration model to the non-linear data by judiciously selecting the *optimal* number of factors [5] and [10], thus, achieving improved prediction performance as compared with CLS. Moreover, even if a linear relationship does exist between the independent and dependent variables, there are certain cases when it is not beneficial to use all of the available data. For this situation, again the PLS method can yield an optimized model developed from the empirical data, using only the essential features of the data by use of factor analysis (rank reduction). However, the CLS result using all of the available data will give sub-optimal performance as compared with PLS.

It is our intention to first present the CLS method (Section 2) which is developed in a mathematical setting that allows a comparison to PLS, and then present a detailed explanation of the PLS algorithm (Section 3). Two examples are presented in Section 4 using both CLS and PLS, and their performance results are compared. The third example in Section 4 uses only PLS applied to a parametric system identification problem. Finally, conclusions are drawn in Section 5 along with suggestions for future research.

2. CLASSICAL LEAST-SQUARES

The CLS result presented here is formulated in a mathematical setting which allows a comparison to PLS given in the next section. Assume that there exists a linear relationship, i.e., a vector $\boldsymbol{h} \in \Re^n$, between a set of independent variables given in matrix form as

$$\mathbf{X}(\mathrm{k}) = \begin{bmatrix} \mathrm{x}_{11}(\mathrm{k})\ \mathrm{x}_{12}(\mathrm{k})\cdots\mathrm{x}_{1\mathrm{m}}(\mathrm{k}) \\ \mathrm{x}_{21}(\mathrm{k})\ \mathrm{x}_{22}(\mathrm{k})\cdots\mathrm{x}_{2\mathrm{m}}(\mathrm{k}) \\ \vdots \\ \mathrm{x}_{\mathrm{n}1}(\mathrm{k})\ \mathrm{x}_{\mathrm{n}2}(\mathrm{k})\cdots\mathrm{x}_{\mathrm{nm}}(\mathrm{k}) \end{bmatrix} \tag{1}$$

where $X(k) \in \Re^{n \times m}$, k is the time index, and the dependent variable is a vector $\boldsymbol{y}(k) \in \Re^m$, such that we can write

$$\mathbf{y}(\mathrm{k}) = \mathbf{X}^{\mathrm{T}}(\mathrm{k})\mathbf{h} \tag{2}$$

Each column vector in the matrix $X(k)$ consists of a time sequence of a wide-sense stationary process, and the columns of $X(k)$ are not necessarily linearly independent. However, the columns of $X(k)$ in (1) are assumed to be corrupted by Gaussian white noise $N(k) \in \Re^{n \times m}$, i.e., each column of $N(k)$ is a white sequence with zero mean and the covariance matrix of $N(k)$ is given as $\boldsymbol{R}_N = E[N(k)N^T(k)]$, where $\boldsymbol{R}_N \in \Re^{n \times n}$. Therefore, a *measurement* of $X(k)$ in (1) is given as

$$\mathbf{Z}(\mathrm{k}) = \mathbf{X}(\mathrm{k}) + \mathbf{N}(\mathrm{k}) \tag{3}$$

where $Z(k) \in \Re^{n \times m}$. Because the information in the columns of $X(k)$ is not known, dependent variable $y(k)$ using $Z(k)$ as

$$\hat{\mathbf{y}}(\mathrm{k}) = \mathbf{Z}^{\mathrm{T}}(\mathrm{k})\mathbf{g} \tag{4}$$

where $g \in \Re^n$ and $\hat{y}(k) \in \Re^m$. Therefore, the objective is to find the linear relationship g in (4) which relates the measured variables $Z(k)$ to an estimate of the dependent variable, $\hat{y}(k)$. To accomplish this we use the classical least squares (CLS) approach by first defining an error vector given as

$$\mathbf{e}(\mathrm{k}) = \mathbf{y}(\mathrm{k}) - \hat{\mathbf{y}}(\mathrm{k}) \tag{5}$$

where $e(k) \in \Re^m$. Using (5), a quadratic performance measure is defined as

$$J_g = E[\mathbf{e}^{\mathrm{T}}(\mathrm{k})\mathbf{W}\mathbf{e}(\mathrm{k})] \tag{6}$$

which is to be minimized with respect to g. However, the positive-definite symmetric weight matrix in (6), $W \in \Re^{m \times m}$, is equal to the $m \times m$ identity matrix, i.e., $W = I_m$, because all errors are considered to be equally weighted. Therefore, (6) can be written as

$$J_g = E[\mathbf{e}^{\mathrm{T}}(\mathrm{k})\mathbf{e}(\mathrm{k})] = \mathrm{E}[(\mathbf{y}(\mathrm{k}) - \hat{\mathbf{y}}(\mathrm{k}))^{\mathrm{T}}(\mathbf{y}(\mathrm{k}) - \hat{\mathbf{y}}(\mathrm{k}))] \tag{7}$$

Substituting (2), (3) and (4) into (7) gives

$$\begin{gathered} J_g = E[(\mathbf{y}(\mathrm{k}) - \hat{\mathbf{y}}(\mathrm{k}))^{\mathrm{T}}(\mathbf{y}(\mathrm{k}) - \hat{\mathbf{y}}(\mathrm{k}))] = \\ \mathbf{h}^{\mathrm{T}}\mathrm{E}[\mathbf{X}(\mathrm{k})\mathbf{X}^{\mathrm{T}}(\mathrm{k})]\mathbf{h} - 2\mathbf{g}^{\mathrm{T}}\mathrm{E}[\mathbf{X}(\mathrm{k})\mathbf{X}^{\mathrm{T}}(\mathrm{k})]\mathbf{h} + \\ \mathbf{g}^{\mathrm{T}}\mathrm{E}[\mathbf{X}(\mathrm{k})\mathbf{X}^{\mathrm{T}}(\mathrm{k})]\mathbf{g} + \mathbf{g}^{\mathrm{T}}\mathrm{E}[\mathbf{N}(\mathrm{k})\mathbf{N}(\mathrm{k})^{\mathrm{T}}]\mathbf{g} = \\ \mathbf{h}^{\mathrm{T}}\mathbf{R}_{\mathrm{X}}\mathbf{h} - 2\mathbf{g}^{\mathrm{T}}\mathbf{R}_{\mathrm{X}}\mathbf{h} + \mathbf{g}^{\mathrm{T}}\mathbf{R}_{\mathrm{X}}\mathbf{g} + \mathbf{g}^{\mathrm{T}}\mathbf{R}_{\mathrm{N}}\mathbf{g} \end{gathered} \tag{8}$$

where $E[X(k)X^T(k)] = R_X$ and $R_X \in \Re^{n \times n}$, is the covariance matrix of $X(k)$, and it is assumed that $N(k)$ and $X(k)$ are uncorrelated, i.e.,

$$E[\mathbf{N}(\mathrm{k})\mathbf{X}^{\mathrm{T}}(\mathrm{k})] = \mathrm{E}[\mathbf{X}(\mathrm{k})\mathbf{N}^{\mathrm{T}}(\mathrm{k})] = 0 \tag{9}$$

Taking the partial derivative of (8) with respect to g and setting the result equal to zero gives

$$\frac{\partial J_g}{\partial \mathbf{g}} = \frac{\partial}{\partial \mathbf{g}}\{\mathbf{h}^{\mathrm{T}}\mathbf{R}_{\mathrm{X}}\mathbf{h} - 2\mathbf{g}^{\mathrm{T}}\mathbf{R}_{\mathrm{X}}\mathbf{h} + \mathbf{g}^{\mathrm{T}}\mathbf{R}_{\mathrm{X}}\mathbf{g} + \mathbf{g}^{\mathrm{T}}\mathbf{R}_{\mathrm{N}}\mathbf{g}\} = -2\mathbf{R}_{\mathrm{x}}\mathbf{h} + 2\mathbf{R}_{\mathrm{x}}\mathbf{g} + 2\mathbf{R}_{\mathrm{N}}\mathbf{g} = 0 \tag{10}$$

Solving for g from (10) yields

$$\mathbf{g} = (\mathbf{R}_{\mathrm{X}} + \mathbf{R}_{\mathrm{N}})^{-1}\mathbf{R}_{\mathrm{X}}\mathbf{h} \tag{11}$$

To determine g in (11) we need the known information associated with the measurements, $Z(k)$, and the dependent variable, $y(k)$. However, the classical least-squares result [5] is given as

$$\mathbf{g} = (\mathbf{Z}(\mathrm{k})\mathbf{Z}^{\mathrm{T}}(\mathrm{k}))^{-1}\,\mathbf{Z}(\mathrm{k})\mathbf{y}(\mathrm{k}) \tag{12}$$

which is directly related to (11). This can be shown by first using the measurement data, $Z(k)$, and forming

$$E[\mathbf{Z}(\mathrm{k})\mathbf{Z}^{\mathrm{T}}(\mathrm{k})] = \mathrm{E}[\mathbf{X}(\mathrm{k})\mathbf{X}^{\mathrm{T}}(\mathrm{k})] + \mathrm{E}[\mathbf{N}(\mathrm{k})\mathbf{N}^{\mathrm{T}}(\mathrm{k})] = \mathbf{R}_{\mathrm{X}} + \mathbf{R}_{\mathrm{N}} \tag{13}$$

using (3) and the assumption that $N(k)$ and $X(k)$ are uncorrelated. The inverse of (13) gives the first part of the expression in (11). Next, using the information in the dependent variable, $y(k)$, along with the measurement data, $Z(k)$, we can write

$$E[\mathbf{Z}(\mathrm{k})\mathbf{y}(\mathrm{k})] = \mathrm{E}[\mathbf{X}(\mathrm{k})\mathbf{X}^{\mathrm{T}}(\mathrm{k})]\mathbf{h} = \mathbf{R}_{\mathrm{X}}\mathbf{h} \tag{14}$$

using (2) and (3), and the assumption that $N(k)$ and $X(k)$ are uncorrelated. The expression in (14) gives the second part of (11). Therefore, from the results in (13) and (14), the relationship between (11) and (12) can be seen.

The CLS result in (12) uses all of the available information from the measurements, $Z(k)$, and the dependent variable, $y(k)$. However, there are many situations when this is not desirable, and can result in *overfitting* of the data [5] and [11]. That is, in the process of developing the model g in (12), the model parameters could be based on not only the essential features associated with the empirical data, $Z(k)$ and $y(k)$, but also unwanted effects in the data. This can result in a model with poor predictive capability. These unwanted effects could be not only be associated with noise, but other features of the data associated with additional cause-and-effect phenomenan that is not of any interest to the analyst. Therefore, in this sense these effects could also be considered as noise because the nature of the phenomenon may not be known *a priori*. It will be shown that when taking the PLS approach to develop a calibration model, and if all of the PLS factors are retained in the development of the model, the result given in (12) for CLS is identical to the PLS result.

3. PARTIAL LEAST-SQUARES REGRESSION

When PLS is used in analytical chemistry [5], the objective is to develop a calibration model from spectroscopic data, $A \in \Re^{m\times n}$, (typically infrared absorption spectroscopic data [12]) and the associated *target* data, $c \in \Re^{m}$, which is the reference concentration information of a particular analyte that is resident in each of the spectroscopically analyzed measurements in varying concentration. Typically in PLS, the independent variable block is formulated in terms of a matrix whose rows are associated with the individual measurements, e.g., in the case of analyzing spectroscopic data each row is a sequence of spectral frequencies (or wave numbers [12]). That is, the absorption matrix A discussed above consists of m measurements (or m individual spectra) each with n spectral frequencies. Now it is possible to simultaneously account for more than one analyte of interest in the spectroscopic data, however, we will limit our discussions to the *single component* case. The dependent variable block, e.g., $\boldsymbol{c}$ given above, is a column vector, for the *single component* case, where each element in the vector is the associated dependent data for each of the corresponding spectra in the rows of $\boldsymbol{A}$. Therefore, comparing this structure to the previous structure of the independent and dependent variable blocks for CLS, and assuming that the data are now measured spectroscopic data, the time index k is replaced with frequency (or wave number), and

$$\mathbf{A} = \mathbf{Z}^{\mathrm{T}} \tag{15}$$

and

$$\mathbf{c} = \mathbf{y} \tag{16}$$

Therefore, from (16), the dependent variable block for CLS and PLS directly corresponds to one another. With these two data blocks, $\{\boldsymbol{A},\boldsymbol{c}\}$, a PLS calibration model (vector) can be developed that can in turn be used to *predict* concentrations of the analyte of interest from spectroscopic data that was not used in the model development phase, i.e., *training*. Thus, for spectroscopic data that are collected and the concentrations of the analyte of interest are not known, the PLS calibration model can be used to predict these unknown concentrations.

Presented below is a detailed explanation of the PLS algorithm for the single or one component case as explained above. The corresponding variable blocks are general, in the sense that the data can be considered in the time-domain or frequency-domain. Therefore, the independent variable block is a matrix $\boldsymbol{Z}^T \in \Re^{m \times n}$ consisting of *measured* data, and the dependent variable block is a column vector, $\boldsymbol{y} \in \Re^m$, consisting of the *target* data.

The basic PLS algorithm, referred to as PLS1 [3] (also known as PLS regression), discussed here is for the case where only one *component* in the data is of interest. This is equivalent to the single-input/single-output (SISO) case in system theory. The one *component* (or SISO) case is addressed because the PLS algorithm is the easiest to understand, however, an extension to the multi-component case is straightforward and can be found in Martens and Naes [5]. Unlike the one-component PLS, the multi-component PLS requires the use of singular value composition (SVD) [5]. An alternate method that has been used extensively in lieu of SVD is termed NIPALS (Non-linear Iterative Partial Least Squares) [2]. Typically, PLS is discussed in two parts, PLS1 calibration and PLS1 prediction. The calibration algorithm will be explained first (PLS1 calibration), followed by an explanation of the development of the calibration model that can be used for prediction (PLS1 prediction). The PLS1 calibration algorithm will be explained according to a seven-step process and follows the explanation given by Haaland and Thomas [3].

For this general explanation of the calibration algorithm, the data $\{\boldsymbol{Z}^T,\boldsymbol{y}\}$ are used as *training* data, i.e., $\{\mathbf{Z}^{\mathrm{T}}_{\text{train}},\boldsymbol{y}_{\text{train}}\}$ for ultimately developing the calibration model, however, the *train* and *test* subscripts are omitted from the PLS1 calibration algorithm explanation to avoid confusion. After the explanation of the PLS1 calibration algorithm, a method is given for development of the calibration model.

PLS1 Calibration Algorithm

Step 1. Mean-centering and variance scaling of the data. The first step typically taken is pretreatment of the data, i.e., mean centering and variance scaling [2] and [5]. An exhaustive explanation of the underlying reasons for this type of pretreatment of the data will not be given, however, typical situations that require pretreatment of the data will be explained. For example, if the collected data (the measurements), i.e., the rows of $\boldsymbol{Z}^T$, have a *bias* associated with the data, then mean-centering of the data would be advisable. This process would also be carried out for the dependent variables, i.e., $\boldsymbol{y}$, as well. What this basically accomplishes is the elimination of the need to fit a non-zero intercept to the data which often results in a decrease of the complexity of the calibration model. That is, a reduction in the number of PLS factors by one required to model the data [3]. If the collected data are measured with different units, variance scaling is then advisable. Mean-centering involves calculating the mean value of each column of $\boldsymbol{Z}^T$ and subtracting the mean value for the particular column from each of the elements in the respective columns. If $\boldsymbol{Z}^T$ is mean centered, $\boldsymbol{y}$ should also be mean-centered. Variance scaling involves calculating the standard deviation of each column in $\boldsymbol{Z}^T$, and then dividing each element in the respective column by the associated standard deviation value. This same process should be carried out for the dependent variables in $\boldsymbol{y}$. However, this is usually only necessary for the multi-component case. There is a continuing debate among statisticians relating to mean-centering and variance scaling. Many individuals insist that the data should always be preconditioned, and others maintain that the data should never be mean-centered and variance scaled [13]. We feel that pretreatment of the data is necessary for the reasons given above, however, if there is no compelling reason to mean-center or variance scale the data these processes should not be performed arbitrarily.

First, an index $\boldsymbol{h}$ (number count for the PLS factors) is initially set to 1.

Step 2. Forming the weight loading vector, $\hat{\boldsymbol{w}}_h \in \Re^n$. This step is actually a CLS calibration, and the model used is given as *model*:

$$\mathbf{Z}^{\mathrm{T}} = \mathbf{y}\mathbf{w}_{\mathrm{h}}^{\mathrm{T}} + \mathbf{E}_{\mathrm{Z}^{\mathrm{T}}}^{\mathrm{T}} \tag{17}$$

where the least-squares solution is given as *least-squares solution*:

$$\hat{\mathbf{w}}_{\mathrm{h}} = \mathbf{Z}\mathbf{y}/\mathbf{y}^{\mathrm{T}}\mathbf{y} \tag{18}$$

In (18) each vector $\hat{\boldsymbol{w}}_h$, for each $\boldsymbol{h}$ increment, is the weight vector which is proportional to a weighted average of the row elements in the matrix $\boldsymbol{Z}^T$, where the weights in the average are proportional to the elements in $\boldsymbol{y}$. Each of the weight vectors $\hat{\boldsymbol{w}}_h$ are normalized and constructed to be mutually orthogonal, therefore, the $\hat{\boldsymbol{w}}_h$ vectors are orthonormal. This step is quite different than with the PCR method [14, 2, 15] and [5] because the information in the dependent variable, $\boldsymbol{y}$, is used in addition to $\boldsymbol{Z}^T$ to form the weight vectors in PLS, in PCR only the information in the matrix $\boldsymbol{Z}^T$ is used in this step. The matrix $\mathbf{E}_{\mathrm{Z}^{\mathrm{T}}}^{\mathrm{T}} \in \Re^{\mathrm{m}\times\mathrm{n}}$ in (17) contains the residuals associated with $\boldsymbol{Z}^T$.

Step 3. Generation of the score (latent variable) vector, $\boldsymbol{t}_{\mathrm{h}} \in \Re^m$. In this step, $\boldsymbol{Z}^T$ is now written with respect to the latent variables or the scores as

$$\textit{model}: \quad \mathbf{Z}^{\mathrm{T}} = \mathbf{t}_{\mathrm{h}}\hat{\mathbf{w}}_{\mathrm{h}}^{\mathrm{T}} + \mathbf{E}_{\mathrm{Z}^{\mathrm{T}}}^{\mathrm{T}} \tag{19}$$

where the least-squares solution is given as

$$\textit{least-squares solution}: \quad \hat{\boldsymbol{t}}_h = \mathbf{Z}^{\mathrm{T}}\hat{\mathbf{w}}_{\mathrm{h}}/\hat{\mathbf{w}}_{\mathrm{h}}^{\mathrm{T}}\hat{\mathbf{w}}_{\mathrm{h}} = \mathbf{Z}^{\mathrm{T}}\hat{\mathbf{w}}_{\mathrm{h}} \tag{20}$$

This step is also CLS, where the least-squares estimate of $\boldsymbol{t}_h$, $\hat{\boldsymbol{t}}_{\mathrm{h}}$, is obtained by regressing $\boldsymbol{Z}^T$ on $\hat{\boldsymbol{w}}_h$ as shown in (20). The individual elements of $\hat{\boldsymbol{t}}_{\mathrm{h}}$ indicate how much of $\hat{\boldsymbol{w}}_h$ is contained in each row of the matrix $\boldsymbol{Z}^T$. The vector $\hat{\boldsymbol{t}}_1$ represents the intensities (or amounts) of the first weight loading vector in the row data of $\boldsymbol{Z}^T$ for the new PLS coordinate system. Since $\hat{\boldsymbol{w}}_1$ is a first-order attempt to represent the uncorrupted data, $\boldsymbol{X}^T$, from the corrupted data contained in $\boldsymbol{Z}^T$, $\hat{\boldsymbol{t}}_1$ represents a first-order attempt to determine the amount of the pure component of interest (i.e., the information contained in $\boldsymbol{y}$) in each of the associated rows of $\boldsymbol{Z}^T$. Thus, in the PLS method, each $\hat{\boldsymbol{t}}_{\mathrm{h}}$ vector is related to both $\boldsymbol{Z}^T$ and $\boldsymbol{y}$ rather than solely to $\boldsymbol{Z}^T$ as in the case of PCR.

Step 4. Relating the score vector, $\hat{\boldsymbol{t}}_{\mathrm{h}}$, to the elements of $\boldsymbol{y}$. In this step the score vector, $\hat{\boldsymbol{t}}_{\mathrm{h}}$, (or the latent variable associated with key features relating to the pure component of interest contained in each row of $\boldsymbol{Z}^T$), representing the intensities in the new PLS coordinate system is related to the elements of the vector $\boldsymbol{y}$ using a linear least-squares regression. In PLS, as opposed to inverse least-squares (ILS) [16] and [3] and PCR, a separate relation between the scores, $\hat{\boldsymbol{t}}_{\mathrm{h}}$, and the elements of the $\boldsymbol{y}$ vector (or the $\boldsymbol{y}$ residuals) is found after each weight vector is estimated. The relation between $\hat{\boldsymbol{t}}_{\mathrm{h}}$ and $\boldsymbol{y}$ is modeled as

$$\textit{model}: \quad \mathbf{y} = \nu_{\mathrm{h}}\hat{\mathbf{t}}_{\mathrm{h}} + \mathbf{e}_{\mathrm{y}} \tag{21}$$

and the least-squares solution is given as

$$\textit{least-squares solution}: \quad \hat{\nu}_{\mathrm{h}} = \hat{\mathbf{t}}_{\mathrm{h}}^{\mathrm{T}}\mathbf{y}/\hat{\boldsymbol{t}}_h^T\hat{\boldsymbol{t}}_h \tag{22}$$

where for each h increment (22) gives an estimate of ν_h which is the scalar regression coefficient (inner relationship) relating $\hat{\boldsymbol{t}}_{\mathrm{h}}$ to the elements in $\boldsymbol{y}$. The vector $\boldsymbol{e}_y \in \Re^m$ contains the PLS residuals associated with $\boldsymbol{y}$. The relation in (22) is similar to an ILS solution in that the sum of the squared $\boldsymbol{y}$ errors is minimized.

Step 5. Generation of $\hat{\boldsymbol{b}}_{\mathrm{h}} \in \Re^n$, the PLS loading vector for $\boldsymbol{Z}^T$. Orthogonal $\hat{\boldsymbol{t}}_{\mathrm{h}}$ vectors are desirable in order to remove collinearities (i.e., linear dependence). Orthogonal $\hat{\boldsymbol{t}}_{\mathrm{h}}$ vectors can be obtained by forming a new model for $\boldsymbol{Z}^T$ based on the latent variable $\hat{\boldsymbol{t}}_{\mathrm{h}}$. The new model is given as

$$\textit{model}: \quad \mathbf{Z}^{\mathrm{T}} = \hat{\boldsymbol{t}}_h\boldsymbol{b}_h^T + \mathbf{E}_{\mathrm{Z}^{\mathrm{T}}}^{\mathrm{T}} \tag{23}$$

where the least-squares solution is given as

$$\textit{least-squares solution}: \quad \hat{b}_h = \boldsymbol{Z}\hat{\boldsymbol{t}}_h/\hat{\boldsymbol{t}}_h^T\hat{\boldsymbol{t}}_h \tag{24}$$

The vectors $\boldsymbol{b}_h$, for $h = 1,2,\ldots$, are the PLS loading vectors. This step along with the next one in the algorithm assures that the $\hat{\boldsymbol{t}}_{\mathrm{h}}$ vectors will be mutually orthogonal. The least-squares regression is simultaneously performed for all samples in each row of $\boldsymbol{Z}^T$ as indicated in (24). Unlike the first PCA loading vector in PCR, the first PLS loading vector, $\hat{\boldsymbol{b}}_1$ determined from (24) does not account for the maximum variance in the rows of $\boldsymbol{Z}^T$. However, it does represent an attempt to account for as much variation in $\boldsymbol{Z}^T$ while simultaneously correlating with $\boldsymbol{t}_{\mathrm{h}}$ which approximates $\boldsymbol{y}$. Also unlike PCA, the $\hat{\boldsymbol{b}}$k vectors are not mutually orthogonal. Moreover, since $\boldsymbol{t}_1$ is the first-order approximation to the $\boldsymbol{y}$ vector, column elements in $\boldsymbol{Z}^T$ associated with the largest positive elements in $\hat{\boldsymbol{b}}$1 tend to indicate those column elements in $\boldsymbol{Z}^T$ which exhibit the greatest dependence on the elements in $\boldsymbol{y}$ for that particular loading vector. However, the $\hat{\boldsymbol{w}}_1$ vector which is directly related to $\boldsymbol{y}$ will exhibit this tendency better than $\hat{\boldsymbol{b}}$1, and therefore $\hat{\boldsymbol{w}}_1$ will be more useful than $\hat{\boldsymbol{b}}_1$ for extracting information from the PLS1 analysis.

Step 6. Calculation of the residuals in $\boldsymbol{Z}^T$ and $\boldsymbol{y}$. The product of the scores ($\hat{\boldsymbol{t}}_{\mathrm{h}}$) and the loading vectors ($\hat{\boldsymbol{b}}_{\mathrm{h}}$) is the PLS approximation to $\boldsymbol{Z}^T$. The residuals, $\mathbf{E}^{\mathrm{T}}_{\mathrm{Z^T}}$, in the matrix $\boldsymbol{Z}^T$ are computed by subtracting the PLS approximation to the rows of the matrix $\boldsymbol{Z}^T$ from the measurements in the rows of $\boldsymbol{Z}^T$ as given in

$$\mathbf{Z}^{\mathrm{T}}\text{ residuals:}\quad \mathbf{E}^{\mathrm{T}}_{\mathrm{Z^T}} = \mathbf{Z}^{\mathrm{T}} - \hat{\mathbf{t}}_{\mathrm{h}}\hat{\mathbf{b}}^{\mathrm{T}}_{\mathrm{h}} \tag{25}$$

$$\text{y residuals:}\quad \mathbf{e}_{\mathrm{y}} = \mathbf{y} - \hat{\mathrm{v}}_{\mathrm{h}}\hat{\mathbf{t}}_{\mathrm{h}} \tag{26}$$

Similarly, the portion of the information in the vector $\boldsymbol{y}$ that has been modeled by PLS can be removed to obtain the residuals in $\boldsymbol{y}$, i.e., $\boldsymbol{e}_y$, as given in (26). The product $\hat{v}_h\hat{\boldsymbol{t}}_h$ in (26) represents the PLS estimate of $\boldsymbol{y}$, $\hat{\boldsymbol{y}}$, based on the information in the matrix $\boldsymbol{Z}^T$.

Step 7. Increment $\boldsymbol{h}$, substitute h, $\mathbf{E}^{\mathrm{T}}_{\mathrm{Z^T}}$ for $\boldsymbol{Z}^T$ and $\boldsymbol{e}_y$ for $\boldsymbol{y}$ in Step 2 and continue for the desired number of loading vectors (or the optimal number of PLS factors, h^o).

PLS1 Prediction Algorithm

The prediction algorithm presented here follows the *Method 2* explanation given in Haaland and Thomas [3]. An alternate method for prediction is presented as *Method 1* in [3], however, the following procedure is much easier to understand, and is given in two parts: (i) calibration model development, and (ii) prediction using the calibration model. One drawback to this method is it does not allow a determination of the residuals of $\boldsymbol{Z}^T$, therefore, no diagnostic information about the quality of the calibration model fit to the data is available when predictions are obtained by using the calibration model.

From the above *PLS1 Calibration Algorithm*, a calibration model, $\boldsymbol{b}_f \in \Re^n$, can be formed from the weight loading vectors, $\hat{\boldsymbol{w}}_h$ for $h = 1,2,\ldots,q$, the loading vectors for $\boldsymbol{Z}^T$, $\hat{\boldsymbol{b}}_{\mathrm{h}}$ for $h = 1,2,\ldots,q$, and the inner relationships, $\hat{v}_h$ for $h = 1,2,\ldots,q$. If all of the PLS factors are generated, then $q = m$, for $m < n$, and $q = n$, for $m \geq n$. Ideally, we want to only use the optimal number of PLS factors, i.e., $h = h^o$. A procedure to determine h^o will be presented next, however, this procedure requires the use of the calibration model, $\boldsymbol{b}_f$. To generate the calibration model we first form the matrices

$$\hat{\mathbf{W}}^T = [\hat{\mathbf{w}}_1, \hat{\mathbf{w}}_2 \cdots \hat{\mathbf{w}}_q] \tag{27}$$

and

$$\hat{\mathbf{B}}^T = [\hat{\mathrm{b}}_1\, \hat{\mathrm{b}}_2 \cdots \hat{\mathrm{b}}_q] \tag{28}$$

and

$$\hat{v}^T = [\hat{\mathrm{v}}_1\, \hat{\mathrm{v}}_2 \cdots \hat{\mathrm{v}}_q] \tag{29}$$

where $\hat{\boldsymbol{W}} \in \Re^{q \times n}$, $\hat{\boldsymbol{B}} \in \Re^{q \times n}$, and $\hat{v} \in \Re^q$. From (27), (28) and (29), and $q = h^o$, the optimal calibration model (final calibration coefficients), $\boldsymbol{b}_f$, can be formed as

$$\mathbf{b}_f = \hat{\mathbf{W}}^T(\hat{\mathbf{B}}\hat{\mathbf{W}}^T)^{-1}\hat{\mathrm{v}} \tag{30}$$

The calibration model (vector) given in (30), developed from the training data, $\{\mathbf{Z}_{\mathrm{train}}^T, \mathbf{y}_{\mathrm{train}}\}$, can now be used for prediction. That is, given another set of measurements, $\mathbf{Z}_{\mathrm{test}}^T$ that was not used in the development of the calibration model and projecting the matrix $\mathbf{Z}_{\mathrm{test}}^T$ onto $\boldsymbol{b}_f$, an estimate of $\boldsymbol{y}$ can be obtained, i.e.,

$$\hat{\mathbf{y}}_{\mathrm{test}} = \mathbf{Z}_{\mathrm{test}}^T \mathbf{b}_f \tag{31}$$

and if the data were mean-centered, where the mean of the dependent reference data is given as , $\hat{y}_{\mathrm{train}}$ then

$$\hat{\mathbf{y}}_{\mathrm{test}} = \mathbf{Z}_{\mathrm{test}}^T \mathbf{b}_f + \bar{\mathbf{y}}_{\mathrm{train}} \tag{32}$$

To take advantage of the full capabilities of PLS, we need to select the optimal number of factors, h^o.

Selection of the Optimal Number of PLS Factors (Rank Reduction)

The optimal number of PLS factors (h^o) can be determined by use of several different methods [14, 2–4, 17, 13, 18, 5] and [19]. It is rarely desirable to retain all of the PLS factors, especially when the measurements contain noise. Therefore, the objective is to retain only those PLS factors that contain the desired information relating to the pure component of interest, and discard all others that will be associated with noise. The method that is presented here for selecting h^o is probably the most straightforward, and thus, the easiest to understand. This method is used in the examples shown in the next section to select h^o.

The method presented here to select h^o requires a measure of performance relative to predictions yielded by the PLS calibration model given a set of test data. Using an independent set of data, i.e., test data $\{\mathbf{Z}_{\mathrm{test}}^T, \mathbf{y}_{\mathrm{test}}\}$, not used in the development of the PLS calibration model, the standard error of prediction (SEP) [5] is defined as

$$SEP = \left\{ \sum_{i=1}^{m_{test}} (y_{i\,test} - \hat{y}_{i\,test})^2 / m_{test} \right\}^{1/2} \tag{33}$$

where $y_{i\,test}$ is the reference (actual) pure component of interest for the test data, $\hat{y}_{itest}$ is the PLS prediction (estimate) of $y_{i\,test}$, and m_{test} is the total number of test *measurements*. For the sake of completeness, we also define a similar measure of performance that is computed for the training data, $\{\mathbf{Z}^{\mathrm{T}}_{\mathrm{train}}, \mathbf{y}_{\mathrm{train}}\}$ which will be used later. This is referred to as the standard error of calibration (SEC), and is defined for the training data $\{\mathbf{Z}^{\mathrm{T}}_{\mathrm{train}}, \mathbf{y}_{\mathrm{train}}\}$ as

$$SEC = \left\{ \sum_{i=1}^{m_{train}} (y_{i\,train} - \hat{y}_{i\,train})^2 / (m_{train} - h - 1) \right\}^{1/2} \tag{34}$$

where $y_{i\ train}$ is the reference (actual) pure component of interest for the training data, $\hat{y}_{i\ train}$ is the PLS prediction (estimate) of $y_{i\ train}$, m_{train} is the total number of training *measurements*, and h is the number of PLS factors. The number of PLS factors (h) is included in the denominator of (34) because a penalty must be placed on the prediction performance of the calibration model as additional factors are included for the training data. Therefore, (34) actually represents a weighted performance measure for the training data. In other words, by including h in the denominator of (34), as more PLS factors are added to the model, the predictive performance of the model must account for this increase in performing a fit to the training data, therefore, minimizing overfitting of the training data [5] and [11].

The SEP performance measure in (33) is used to determine the *optimal* number of PLS factors (h^o) to retain for the development of the PLS calibration model, i.e., $\boldsymbol{b}_f$ given in (30). That is, first a set of calibration models, $\boldsymbol{b}_f$ for $h = 1,2,\ldots,q$, are generated for a range of PLS factors using the training data set $\{\mathbf{Z}^{\mathrm{T}}_{\mathrm{train}}, \mathbf{y}_{\mathrm{train}}\}$. Using the set of q calibration models, $\boldsymbol{b}_f$ for $h = 1,2,\ldots,q$, the test data set $\{\mathbf{Z}^{\mathrm{T}}_{\mathrm{test}}, \mathbf{y}_{\mathrm{test}}\}$ is used to assess the predictive performance of each calibration model for different numbers of PLS factors. Using the relationship in (33), the SEP can be calculated for each prediction of $\boldsymbol{y}_{test}$ that the calibration models yield, i.e., $\hat{\boldsymbol{y}}_{test}$ for $h = 1,2,\ldots,q$, using $\mathbf{Z}^{\mathrm{T}}_{\mathrm{test}}$. The selection of h^o is determined by observing the SEP values as a function of the number of PLS factors q. Typically, the number of factors, h, associated with the *minimum* SEP that is observed will indicate the optimal number of PLS factors, i.e., h^o, given as

$$h^o = \{h\colon SEP_{min} = min\{SEP(h)\}\ \forall\, h = 1,2,\cdots,h^o,\cdots,q\} \tag{35}$$

This method for selecting the optimal number of PLS factors, h^o, will be referred to as *independent validation*. However, care must be taken when using (35) because the absolute minimum may give a result that would allow additional factors to be retained that are potentially associated with noise [5]. Therefore, each individual case must be assessed carefully to reconcile what is actually the *optimal* number of PLS factors to retain. In some cases, the absolute minimum from (35) is not the best choice, and many times one factor less than this absolute minimum will give the best overall performance. Additional information can be obtained by observing the PLS loading vectors, $\hat{\boldsymbol{w}}_h$, to facilitate this selection process for the number of PLS factors to retain. In one of the examples which follows, it will be shown that by observing the PLS loading vectors, $\hat{\boldsymbol{w}}_h$, the proper choice of the number of PLS factors to retain for the calibration model development using the independent validation method is confirmed.

Another method that can be used for selecting the number of PLS factors to retain is called *cross-validation* [4] and [5]. Cross-validation is sometimes referred to as a *leave-one-out-at-a-time* analysis approach because one measurement is left out of the data set $\{\mathbf{Z}_{\text{train}}^{\text{T}}, \mathbf{y}_{\text{train}}\}$ and the PLS model is developed on the remaining measurements, then the measurement that was left out is tested to yield a prediction of y_{train}, i.e., $\hat{y}_{train}$. This process is repeated until all measurements are left out and used for prediction and the SEP is computed for the entire m_{train} measurements according to (33). Cross-validation is often used when the number of measurements is sparse, i.e., not many measurements are available. In this case there is not enough data to form a training set and a test set. The results obtained using this method are not typically as good as the method given above that uses a training set and a test set. Ideally, it is desirable to have a statistically representative set of measurements such that separate training and test sets can be formed [14, 17] [15] and [5]. In one of the examples which follows, the cross-validation method gives the same result as independent-validation.

Summary

The PLS presentation given above is only one of many approaches that can be taken to present the underlying principles of the algorithms [1–5,19,12,7] and [8]. As discussed above, the *single-component* case was presented in order to allow tractability of the algorithms, although the *single-component* case is applicable in many situations [3]. Comparing the PLS and CLS approaches for prediction (or estimation), it is obvious that PLS has one key feature that CLS does not possess, i.e., the ability to select only certain key features relative to the empirical data to retain for calibration model development, i.e., *factor analysis*. More specifically, shown in (11), Section 2, if the covariance matrix, $\boldsymbol{R}_X$, the covariance matrix, $\boldsymbol{R}_N$, and the linear relationship, $\boldsymbol{h}$, were all known quantities, then the linear relationship, $\boldsymbol{g}$, could be computed. Where $\boldsymbol{g}$ is the linear relationship between the measurement data, $\boldsymbol{Z}$ and the estimate of the dependent variable, $\hat{\boldsymbol{y}}$, see (4). It was shown, however, that the CLS solution of $\boldsymbol{g}$ given in (12) is directly related to (11), and the CLS result in (12) is typically used because the covariance matrices, $\boldsymbol{R}_X$, $\boldsymbol{R}_N$, and the linear relationship, $\boldsymbol{h}$, are usually not known. However, when $\boldsymbol{g}$ is determined from (12) all of the data are used, and as previously mentioned, this is not always desirable because of the problem of *overfitting*, i.e., fitting the model to not only the desirable data but also the noise and potentially other unwanted effects in the data.

If, for example, there was *a priori* knowledge of the noise corrupting the data, i.e., $\boldsymbol{N}$, then it is possible to develop methods to determine $\boldsymbol{g}$, that would give better results than the direct CLS approach in (12). With this prior knowledge of the corrupting noise, a filter could be designed to suppress this disturbance. However, one method which does not require *a priori* knowledge of the statistical properties of the corrupting noise for example, but can essentially perform noise reduction by not allowing the noisy data to be included in the development of the calibration model by properly performing the *factor analysis* (or *rank reduction*) and can typically yield better predictive performance results than CLS is PLS. This is shown in *Example 2* in the next section. This example also illustrates that the *noise* can also be a component feature of the data that is of no interest to the analyst and must be removed to increase the predictive performance of the PLS calibration model. Therefore, when the number of factors is properly chosen, PLS can simultaneously optimize the calibration model, $\boldsymbol{b}_f$, and reduce the potential for overfitting by not allowing the model to be fit to the noise.

4. SIMULATION EXAMPLES

Presented in this section are three examples that illustrate how PLS regression can be used in selected applications. Applications of PLS are not limited to the classes of problems addressed in this section, and we believe that this method has very wide applicability to many other areas in engineering, specifically signal processing problems. All simulations were run on a Pentium®-90 PC using MATLAB®. The PLS results in Examples 2 and 3 were obtained using the MATLAB® chemometrics toolbox [20]. The results in Example 1 were obtained by implementation of the algorithms given above written as *m-files* in MATLAB®. The MATLAB® chemometrics toolbox uses a slightly different algorithm for both *calibration* and *prediction* previously given. The prediction phase in the toolbox more closely resembles *Method 1* given in Haaland and Thomas [3], where the residuals of the independent variable block are used to predict the dependent variables. Another capability within the MATLAB® chemometrics toolbox is an option to compute the PLS calibration model. The method utilized is also slightly different than the algorithm given above, however, to avoid confusion the same notation used above for the PLS calibration model will also be used for the MATLAB® PLS calibration model, i.e., $\boldsymbol{b}_f$.

Example 1

In this example, the objective is to determine a calibration model that is best suited for prediction. Both the CLS and PLS methods are used and give the same results when all of the factors are used in PLS. The independent data block consists of two straight lines, i.e., $r = u$ and $r = 2u$. For $-1 \leq u \leq 1$ in Δu increments of 10^{-5}, the data matrix $\boldsymbol{X}$ in (1) has two rows of length 200001. The assumed linear relationship is given as $\boldsymbol{h}^T = [1\ 2]$, and the data are corrupted with Gaussian white noise with zero mean and variance $(0.15)^2$ for $r = u$ and $(0.1)^2$ for $r = 2u$. Therefore in Figure 1, the two straight lines are shown with the noisy samples which constituent the measurement matrix $\boldsymbol{Z}$ given in (3). The data were not mean-centered because a bias does not exist in the data, this can be seen in Figure 1. The covariance matrix for $\boldsymbol{X}$ was computed as

$$\mathbf{R}_X = \begin{bmatrix} 0.33334 & 0.66668 \\ 0.66668 & 1.33335 \end{bmatrix} \tag{36}$$

and the covariance matrix for the noise was

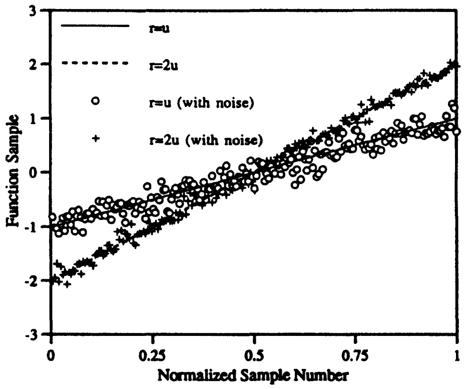

Figure 1. Data for Example 1, the matrix *Z* contains the noisy samples, and *X* contains the samples without noise. (Note: only 1/1000 of the samples are shown).

$$\mathbf{R}_\mathrm{N} = \begin{bmatrix} 0.02245 & 0.00002 \\ 0.00002 & 0.01001 \end{bmatrix} \tag{37}$$

The non-zero off-diagonal elements in (37) are a result of the noise not being truly *white.* The actual (or reference) dependent variable, *y*, is given as in (2), i.e.,

$$\mathbf{y} = \mathbf{X}^\mathrm{T}\mathbf{h} \tag{38}$$

and is shown in Figure 2 as the solid line. The CLS and PLS models were computed according the previously derived expressions in (11) and (30), respectively. The number of factors retained to develop the PLS calibration model was two. The factor analysis process described above to select the optimal number of PLS factors was not carried out for this trivial case. It is obvious that two factors are necessary to retain because there are

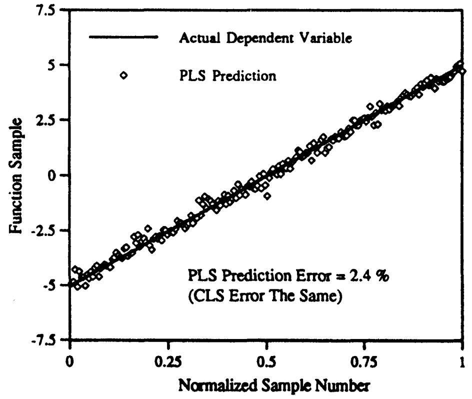

Figure 2. Partial least-squares prediction of the dependent variable, $\hat{y}_{PLS}$. (*Note*: only 1/1000 of the samples are shown).

two components of interest, i.e., $r = u$ and $r = 2u$. The computed CLS calibration model is given as

$$\mathbf{g}^{\mathrm{T}} = [0.49654\ 2.23494] \tag{39}$$

and the computed PLS calibration model is

$$\mathbf{b}_{\mathrm{f}}^{\mathrm{T}} = [0.49683\ 2.23511] \tag{40}$$

From (39) and (40), it can be seen that the two calibration models are essentially the same. The difference is due to the covariance matrix, $\boldsymbol{R}_N$, not being exact. In fact, if the CLS calibration model is computed according to (12) the result is identical to the PLS result in (40). Therefore, when all of the factors are retained to develop the PLS calibration model, the results are the same as CLS. In Figure 2 the PLS prediction of $\boldsymbol{y}$, i.e.,

$$\hat{\mathbf{y}}_{\mathrm{PLS}} = \mathbf{Z}^{\mathrm{T}}\mathbf{b}_{\mathrm{f}} \tag{41}$$

using the calibration model given in (40), is shown along with the actual (or reference) dependent variable, $\boldsymbol{y}$. The PLS prediction error was computed by using (33), where $m_{test} = 200001$, and was 2.4% (the SEP_{PLS} was multiplied by 10 to obtain the percent error because the range of $\boldsymbol{y}$ is from -5 to $+5$). The CLS prediction error was the same as the PLS result when the calibration model was developed according to (12) and the estimate of $\boldsymbol{y}$ is computed using (4).

It is worthy of note that even though the CLS and PLS (retaining all of the factors) results were the same, there are major differences in the manner that the calibration models are computed. First, in the case of CLS, a matrix inverse must be computed to develop the calibration model, and for very large m in $\boldsymbol{X}$ or $\boldsymbol{Z}$ (i.e., when many measurements must be processed) the computational burden increases, see (12). In the case of PLS (see the PLS1 Calibration and Prediction Algorithms in Section 3), a matrix inverse is never computed. Therefore, when many measurements must be processed, PLS is more computationally efficient than CLS. Second, when columns of $\boldsymbol{Z}^T$ are linearly dependent (collinearites [5]), the matrix inverse required to develop the CLS calibration model can not be computed. In the previous example, this would have been the case if there was no noise corrupting the data. In the case of PLS, this is not a problem. Therefore, as noise (which corrupts the data matrix $\boldsymbol{X}$) levels diminish, $\boldsymbol{ZZ}^T$ becomes ill-conditioned [21].

Example 2

In this example simulated near infrared (NIR) absorption spectroscopic data [12] were generated, although this data could represent any type of signals that are to be processed. In Figure 3, a *component of interest* is shown as the solid line, and an *obscuring component* is shown in the figure as the dashed line. Both of the components were generated as Gaussian functions with two absorption bands each, as shown in Figure 3. In the simulated NIR spectroscopic data, the component of interest could be the NIR absorption spectrum of a particular analyte that is to be quantitatively analyzed. The

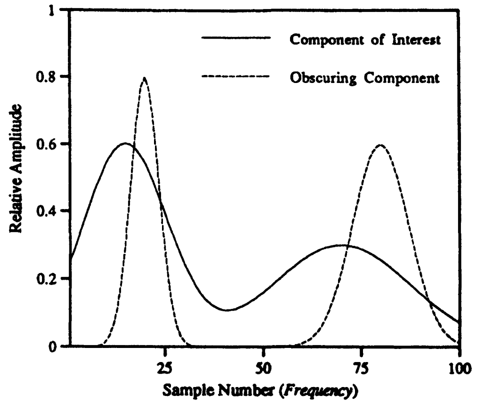

Figure 3. Component (signal) of interest and an obscuring component for Example 2. These are used to simulate 200 NIR spectra.

obscuring component could be the NIR absorption spectrum of water which is typically orders of magnitude larger in amplitude than the component of interest [12, 22], and [23]. Therefore, the water (obscuring component) is often referred to as an intrinsically high background absorption component for aqueous solutions that are analyzed by infrared (IR) spectrophotometric methods.

The obscuring component is one noise contribution to the data (the component of interest) that are to be quantitatively identified. That is, 200 simulated NIR spectroscopic *measurements* were generated by adding the obscuring component to the component of interest, where the obscuring component amplitudes were three orders of magnitude larger. In addition, Gaussian white noise, with zero mean and a relative variance of $\sigma^2 = 9$ was added to the component of interest before adding the obscuring component. In order to simulate the typical nature of spectroscopic data of a particular analyte in an aqueous solution in varying concentration (in milligrams per deci-liter, mg/dl), the *spectral* amplitudes of the component of interest were multiplied by a linear sequence given in vector from as $\boldsymbol{c}$, which simulates normalized concentration values of the analyte of interest to form 200 spectra. The linear sequence elements in $\boldsymbol{c}$ are given as $\{c_i\colon c_i = c_{i-1} + \Delta c, \forall\ i = 1,2,\ldots,200 \text{ and } \Delta c = 0.005,\ c_0 = 0\}$. Therefore, the areas under the curves of the 200 spectra associated with the component of interest are directly proportional to the elements in $\boldsymbol{c}$. That is, under ideal conditions, there exists a linear relationship between the amount of absorption and the concentration of the analyte in the aqueous solution according to the well known Lambert-Beer's law [12], i.e.,

$$A = log_{10}(I_0/I) = \Xi\, cl \tag{42}$$

where A ≡ absorption, I ≡ absorption light intensity, I_0 ≡ incident light intensity, Ξ ≡ molar absorption coefficient, c ≡ concentration of the analyzed substance(s), and l ≡ path length of the sample cell. As indicated in (42), the amount of absorption of a particular substance being analyzed is directly proportional to the concentration of the substance. The *normalized* concentration values will be referred to as the *reference values*. The obscuring component was then added to the 200 spectra with a random amplitude to simulate the non-ideal nature of a spectrophotometer, i.e., baseline variations [24, 25] and [5]. Therefore, the 200 composite simulated spectra contains 200 spectra of interest, that are to be identified and correlated to its associated reference value (*normalized concentration*), a highly dominant obscuring component, i.e., *water absorption*, random noise, and baseline variations. Figure 4 shows 5 of the 200 composite simulated spectra (*measurements*). It is apparent from Figure 4 that the component of interest in each of the five spectra is essentially indistinguishable, mostly due to the obscuring component shown in Figure 3. In fact, all five of the spectra in Figure 4 have a spectral shape which resembles only the obscuring component.

The objective in this example is to predict the reference values (concentrations) given only the spectral measurements. We will use both CLS and PLS methods and compare their predictive performances. For both CLS and PLS, a calibration model must be developed. To accomplish this, and also compare their predictive performances, the 200 spectra were divided into a training set and a test set, along with the associated reference values (normalized concentrations). Beginning with the 200 composite spectra, and associated reference values in ascending order, every other spectrum (measurement) was selected, along with the associated reference value for the test set. The remaining 100 spectra were used for the training set, therefore, $m_{train} = m_{test} = 100$. The *absorption* spectra (measurements) are considered as the *independent variable block*, see (15), and the reference values (concentrations) are considered as the *dependent variable block*, see (16), [2]. Therefore, $\{A_{train}, A_{test}\}$ are the training and test *absorption* matrices, both with dimension *100* $\times$ *100* (*measurements* $\times$ *spectral components*), and $\{c_{train}, c_{test}\}$ contains the associated training and test reference normalized concentration values, both of dimension *100* $\times$ *1*.

Using the training data, $\{A_{train}, c_{train}\}$ the CLS calibration model (vector) was computed as

$$\mathbf{g} = (\mathbf{A}_{\text{train}}^{\mathrm{T}} \mathbf{A}_{\text{train}})^{-1} \mathbf{A}_{\text{train}}^{\mathrm{T}} \mathbf{c}_{\text{train}} \tag{43}$$

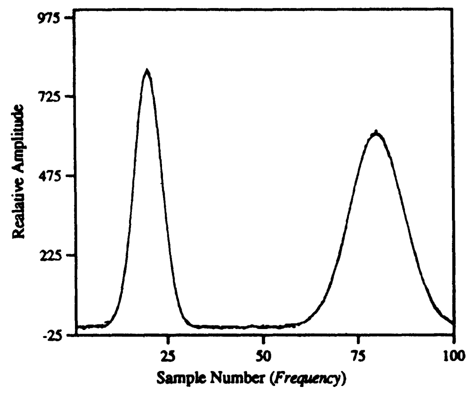

Figure 4. Five representative simulated NIR spectra from the set of 200.

according to the relationship in (12). Figure 5 shows the CLS calibration vector components as a function of the sample number (these are the spectral components or simulated *frequency*). As seen in Figure 5, the calibration model is very erratic, therefore, it is expected that using this model to predict concentrations of unknown spectra would result in relatively poor prediction performance. If the test data absorption matrix A_{test}, is projected onto the CLS calibration model g, the associated test concentrations, c_{test}, can be predicted, see (4), as

$$\hat{\mathbf{c}}_{\text{test-CLS}} = \mathbf{A}_{\text{test}}\mathbf{g} \tag{44}$$

Now using (33), the SEP can be computed which will give a quantitative measure of the predictive capability of the CLS calibration model. It is interesting to also compute the SEC from (34), where in the denominator of (34) the "$m_{train} - h - 1$" term is replaced by $m_{train} = 100$, because CLS is not a factor analysis method. Therefore, the CLS prediction of c_{train}, i.e., $\hat{c}_{train\text{-}CLS}$, was determined from

$$\hat{\mathbf{c}}_{\text{train-CLS}} = \mathbf{A}_{\text{train}}\mathbf{g} \tag{45}$$

Because the reference concentration values are normalized in (0,1], the SEP and SEC are computed in terms of percent error by multiplying (33) and (34) by 100. Therefore, for CLS the $\%SEC_{CLS} = 0.0012$, which indicates that CLS is able to fit the calibration model extremely well to the training data. However, conclusions should not be drawn from this result about the overall predictive capability of the CLS approach, e.g., that CLS is thus able to perform exceptionally well when predicting concentrations from *unknown* spectra. In fact, CLS is actually *overfitting* the data, that is, it is fitting the calibration model to the *noise* which is not desirable, however, with CLS this is unavoidable. This can be seen when the SEP is computed, i.e., $\%SEP_{CLS} = 296.5$. However, this should not be too surprising when observing the CLS calibration model in Figure 5. The prediction results

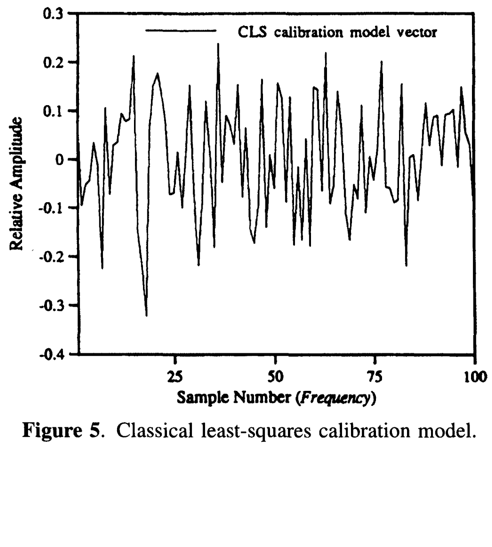

Figure 5. Classical least-squares calibration model.

for the training and test sets are shown in Figure 6. As seen in Figure 6, the CLS predictions on the *unknown* spectra are very poor, even though the predictions on the training data are very good.

Using the same training data set $\{A_{train}, c_{train}\}$ used for CLS, a PLS model was developed. In this example there are the same number of measurements as spectral samples ($m = n$), therefore, as previously stated, if all of the PLS factors were computed a total of $q = 100$ exist. However, because of the manner in which the simulated data were constructed, it is expected that a small number of factors will be necessary to retain for development of the PLS calibration model. Specifically, there is one pure component of interest which is corrupted by noise, and one obscuring component. Therefore, intuitively, it seems plausible that 2 PLS factors would be sufficient to characterize the calibration model in an optimal sense. Using the independent-validation method described above to select the optimal number of PLS factors, h^0, using the training, $\{A_{train}, c_{train}\}$, to develop the calibration models and the test data, $\{A_{test}, c_{test}\}$, to assess their predictive performances, Figure 7 shows the SEP values as a function of the number of PLS factors (only carried out to 10). The main graph in Figure 7 indicates that 2 PLS factors would be optimal, and the insert in the figure shows that 2 factors is the absolute minimum because after 2 the SEP continues to increase monotonically. As previously stated, many times it is worthwhile observing the PLS weight loading vectors, $\hat{w}_h$, see (18). In Figure 8 the first 3 PLS weight loading vectors are shown. It is obvious that the third PLS weight loading vector is associated with noise and should not be retained for model development which confirms the independent-validation selection method result. This is intuitively pleasing based on the speculative discussion previously given. Even though it is not necessary in this example to observe the PLS weight loading vectors, it does illustrate that in some cases it would be necessary if the independent-validation selection method alone was not definitive. Therefore, when the number of factors is properly chosen, PLS can simultaneously optimize the calibration model, b_f, and reduce the potential for overfitting. An interesting result is obtained when the cross-validation method is used to select the optimal number of PLS factors. Using only the training data, $\{A_{train}, c_{train}\}$, for

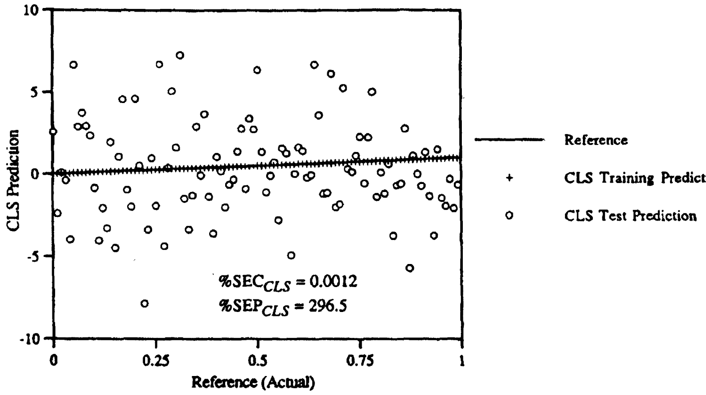

Figure 6. Classical least-squares predictions for the training and test data using the calibration model in (43) and shown in Figure 5.

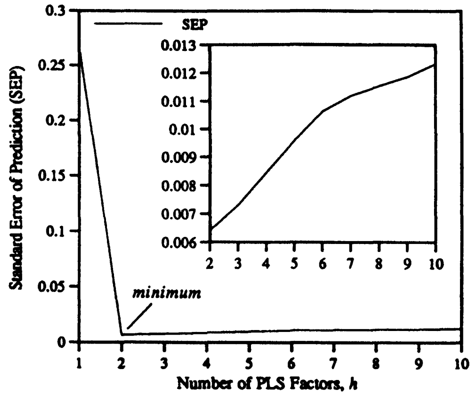

Figure 7. Selection of the optimal number of PLS factors.

cross-validation, the results also showed that 2 PLS factors were optimal. That is, when the SEP was observed as a function of the first 10 PLS factors, i.e., $h = 1, 2, \ldots 10$, the absolute minimum occurred at 2.

To illustrate that very powerful physical insight can be obtained from PLS, in Figure 9 only the first 2 PLS weight loading vectors are shown that were used to develop the calibration model (i.e., $h = h^0 = 2$), along with the original pure component of interest, the obscuring component, and the PLS calibration model vector, $\boldsymbol{b}_f$. All five of the components shown in Figure 9 have been normalized with respect to their maximum value in order to compare their relative shapes. This was done because the PLS calibration model vector has absolute amplitudes which are three orders of magnitude less than the four other components shown. From Figure 9 it is obvious that the first PLS weight loading vector, $\hat{\boldsymbol{w}}_1$, is almost identical to the obscuring component. This is not coincidental, in fact it is typical of PLS to extract this information from the data because $\hat{\boldsymbol{w}}_1$ is the first-order approximation to the obscuring component. After the first iteration through the *PLS1 Calibration Algorithm* discussed above, the score (or latent variable) vector, $\hat{\boldsymbol{t}}_1$, and the loading vector, $\hat{\boldsymbol{b}}_1$, that are associated with the first weight loading

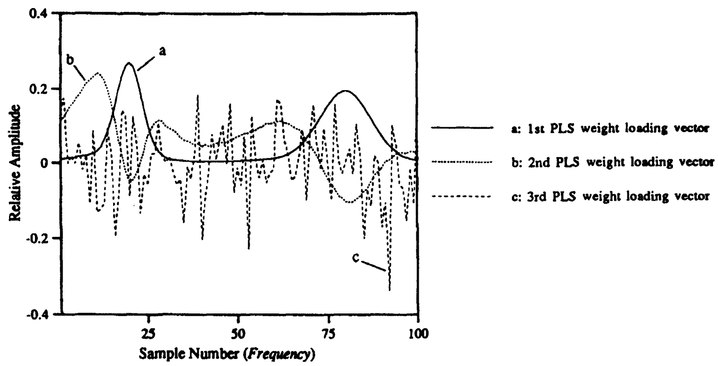

Figure 8. First three PLS weight loading vectors.

vector, $\hat{w}_1$, are used in Step 6 of the algorithm to generate the first *spectral* residue matrix, see (25). Thus, an approximation to the obscuring component is removed from the *spectral* data and the process repeats. The second, and final PLS weight loading vector retained, $\hat{w}_2$, shown in Figure 9, is seen to be very similar to the PLS calibration model vector, b_f. Also, comparing both $\hat{w}_2$ and b_f to the component of interest in the Figure 9, it can be seen that there exists spectral features similar to the component of interest. Therefore, this is also not coincidental that $\hat{w}_2$ and b_f appear to be similar because, as stated above, the calibration model vector contains the final calibration coefficients which are used for prediction of the dependent reference variable from *unknown spectra*. However, $\hat{w}_2$ and b_f are not identical in shape to the component of interest because the obscuring component (which is highly dominant, see Figure 4) must be suppressed in order for PLS to achieve relatively high predictive performance relative to the component of interest. Therefore, the dominating effects of the obscuring component must be quelled, which is the case in this example, and can be seen by again observing Figure 9. Comparing $\hat{w}_1$ and $\hat{w}_2$, which are orthonormal, where the obscuring component has dominant *spectral* bands, the second PLS weight loading vector, $\hat{w}_2$, has a mitigating effect to suppress this dominance.

Using the PLS calibration model, b_f, developed from the training data, $\{A_{train}, c_{train}\}$, for two PLS factors, a prediction of c_{test}, i.e., $\hat{c}_{test\text{-}PLS}$, was obtained using the test data, A_{test}, as

$$\hat{\mathbf{c}}_{\text{test-PLS}} = \mathbf{A}_{\text{test}}\mathbf{b}_{\text{f}} \tag{46}$$

From (46) the SEP can be computed using (33). As with the CLS results shown above, it is also interesting to compute the SEC using (34), therefore, the PLS prediction of c_{train}, i.e., $\hat{c}_{train\text{-}PLS}$, was determined from

$$\hat{\mathbf{c}}_{\text{train-PLS}} = \mathbf{A}_{\text{train}}\mathbf{b}_{\text{f}} \tag{47}$$

Therefore, for PLS, the $\%SEC_{PLS} = 0.7$ and $\%SEP_{PLS} = 0.64$. Comparing the respective SECs for CLS and PLS, it can be seen that CLS shows much better performance than PLS,

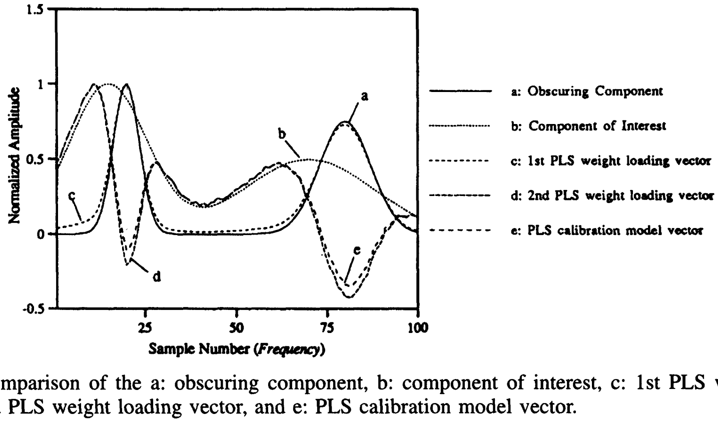

Figure 9. Comparison of the a: obscuring component, b: component of interest, c: 1st PLS weight loading vector, d: 2nd PLS weight loading vector, and e: PLS calibration model vector.

however, as explained above this is due to overfitting of the training data (i.e., CLS is using all 100 *factors* for the development of the calibration model, $\boldsymbol{g}$). In the case of PLS, overfitting of the training data does not occur because only 2 PLS factors were retained to develop the calibration model, $\boldsymbol{b}_f$, thus not allowing the ***noisy*** data to participate in the development of the model. Comparing the respective SEPs for CLS and PLS, it is seen that PLS has three orders of magnitude better predictive performance than CLS. Again, this is a result of not allowing the noisy data to be used for the development of the PLS calibration model as was the case with CLS, because all of the data was used to develop $\boldsymbol{g}$. Figure 10 shows the PLS predictions of $\boldsymbol{c}_{train}$ and $\boldsymbol{c}_{test}$. Comparing the PLS prediction of $\boldsymbol{c}_{test}$ in Figure 10 to the CLS result in Figure 6, the difference is startling.

Example 3

In this final example, PLS is applied to a parametric system identification problem using the auto-regressive moving average (ARMA) approach [26] and [27]. The objective is to estimate (or predict) the parameters of a system given only the collected input/output data of the system, without prior knowledge of the system characteristics. There is no intention to give an exhaustive explanation of system identification using the ARMA approach, but to show how the system parameter vector and the system dimension can be simultaneously estimated. Although, a brief introduction to system identification will be given in order to facilitate an understanding of how PLS can be applied.

Applying PLS to the identification of the parameters of a single input/single output (SISO) discrete-time noise-free system, i.e., the coefficients of the numerator and denominator polynomials of a strictly proper rational system z-transfer function or the matrix elements of a canonical state-space model of the system, we use the ARMA matrix approach. Specifically, given an appropriate input $\mathbf{u}(k)$ along with the system output (response) vector $\mathbf{y}(k)$ for $k = 0, 1, 2, \ldots$, the ARMA matrix can be formed for a selected number of data samples, N, and an assumed system dimension, n. The *appropriate* input to the system is always taken to be a stimulus that is "*persistently exciting of order n*" [26]

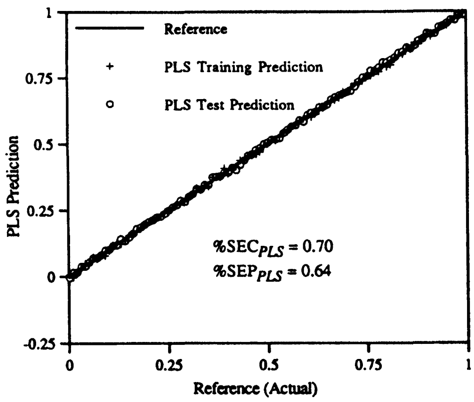

Figure 10. Partial least-squares predictions for the training and test data using the calibration model $\boldsymbol{b}_f$ shown in Figure 9.

and [27], i.e., one that will excite the system in a manner so enough information will reside in the output data to properly estimate the system parameters.

The strictly proper [28] rational z-transfer function has the general form

$$G(z) = \frac{Y(z)}{U(z)} = \frac{b_1 z^{n-1} + b_2 z^{n-2} + b_3 z^{n-3} + \cdots + b_n}{z^n + a_1 z^{n-1} + a_2 z^{n-2} + \cdots + a_n} \tag{48}$$

A time-domain difference equation can formed from (48) which can be written as

$$\begin{aligned} & y(k) + a_1 y(k-1) + a_2 y(k-2) + \cdots a_n y(k-n) \\ & - b_1 u(k-1) - b_2 u(k-2) - b_3 u(k-3) \cdots - b_n u(k-n) \\ & = \epsilon(\mathrm{k}; \mathrm{a}_1, \mathrm{a}_2, \cdots, \mathrm{a}_n, \mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \cdots, \mathrm{b}_n) \end{aligned} \tag{49}$$

where $\epsilon(k; a_1, a_2, \ldots, a_n, b_1, b_2, b_3, \ldots, b_n)$ is an error term which is zero when the parameter vector $\boldsymbol{\theta}^T = [a_1, a_2, \ldots, a_n, b_1, b_2, b_3, \ldots, b_n]$ contains the actual or true plant parameters [26]. Therefore, (49) can be written as

$$y(k) = \boldsymbol{\phi}^{\mathrm{T}}(\mathrm{k})\theta + \epsilon(\mathrm{k}; \boldsymbol{\theta}) \tag{50}$$

where $\boldsymbol{\phi}^T(k) = [-y(k-1) - y(k-2) \ldots - y(k-n)\, u(k-1)\, u(k-2)\, u(k-3) \ldots u(k-n)]$. For the time index taken as $k = n, n+1, n+2, \ldots, N$, where n is the assumed system dimension and N is the total number of data samples used from the data set $\{\mathbf{u}(\mathrm{k}), \mathbf{y}(\mathrm{k})\}$, the collective set of equations resulting from (50) can be written as

$$\mathbf{y}(\mathrm{N}) = \boldsymbol{\Phi}(\mathrm{N})\boldsymbol{\theta} + \boldsymbol{\epsilon}(\mathrm{N}; \boldsymbol{\theta}) \tag{51}$$

where $\mathbf{y}^T(N) = [y(n)\; y(n+1)\; y(n+2) \ldots y(N)]$, $\boldsymbol{\Phi}^T(N) = [\boldsymbol{\phi}(n)\; \boldsymbol{\phi}(n+1)\; \boldsymbol{\phi}(n+2) \ldots \boldsymbol{\phi}(N)]$, $\boldsymbol{\epsilon}^T(N; \boldsymbol{\theta}) = [\epsilon(n; \boldsymbol{\theta})\; \epsilon(n+1; \boldsymbol{\theta})\; \epsilon(n+2; \boldsymbol{\theta}) \ldots \epsilon(N; \boldsymbol{\theta})]$, and $\boldsymbol{\Phi}(N) \in \Re^{N-n+1 \times 2n}$, $y(k) \in \Re^{N-n+1}$, $\boldsymbol{\epsilon}(N; \boldsymbol{\theta}) \in \Re^{N-n+1}$, and $\boldsymbol{\theta} \in \Re^{2n}$.

For a specified number of data samples N and as assumed system dimension n the ARMA matrix shown in (51) has the form

$$\boldsymbol{\Phi}(\mathrm{N}) = \begin{bmatrix} -y(n-1) & -y(n-2) & \cdots & -y(0) & u(n-1) & u(n-2) & \cdots & u(0) \\ -y(n) & -y(n-1) & \cdots & -y(1) & u(n) & u(n-1) & \cdots & u(1) \\ -y(n+1) & -y(n) & \cdots & -y(2) & u(n+1) & u(n) & \cdots & u(2) \\ -y(n+2) & -y(n+1) & \cdots & -y(3) & u(n+2) & u(n+1) & \cdots & u(3) \\ & & & & \vdots & & & \\ -y(N-1) & -y(N-2) & \cdots & -y(N-n) & u(N-1) & u(N-2) & \cdots & u(N-n) \end{bmatrix} \tag{52}$$

CLS can yield an estimate of the parameter vector, $\hat{\boldsymbol{\theta}}_{CLS} \in \Re^{2n}$, in (51) using (12) [26] and [27], i.e.,

$$\hat{\boldsymbol{\theta}}_{\mathrm{CLS}} = (\boldsymbol{\Phi}^{\mathrm{T}}(\mathrm{N})\boldsymbol{\Phi}(\mathrm{N}))^{-1}\, \boldsymbol{\Phi}^{\mathrm{T}}(\mathrm{N})\mathbf{y}(\mathrm{N}) \tag{53}$$

The basic problem with the CLS approach, or any other method that requires prior knowledge of the system dimension, is that the actual system dimension $\bar{n}$ is usually not known *a priori*. There exist methods which have been developed to determine the dimension of the system (or order estimation), however, we have found these to be unreliable in some situations. For example, methods presented in Ljung [27] include: (i) examining the spectral analysis estimate of the transfer function, (ii) testing ranks in sample covariance matrices, (iii) correlating variables, and (iv) examining the information matrix. Therefore, if a parameter estimate is obtained according to (53) with no prior knowledge of the actual system dimension, then $\hat{\boldsymbol{\theta}}_{CLS}$ would contain $2n$ parameter estimates to yield a system realization that would be either *over-specified*, if $n > \bar{n}$, or possibly *under-specified* if $n < \bar{n}$. In either case, the resulting parameter vector could give erroneous results when used as a model for the actual system.

Therefore, it is desirable to obtain a simultaneous estimate of the dimension of the system, $\bar{n}$, and the parameter vector, $\hat{\boldsymbol{\theta}} \epsilon\, \Re^{2\bar{n}}$. When using PLS for this purpose, the ARMA matrix, $\boldsymbol{\Phi}(N)$, takes the place of the absorption matrix, $\boldsymbol{A}$, or $\boldsymbol{Z}^T$, in (15), and the concentration matrix, $\boldsymbol{c}$, or $\boldsymbol{y}$, is replaced with the vector $\boldsymbol{y}(N)$ in (16). The resulting estimate of the system dimension using PLS, $\bar{n}$, will yield the system order for a minimal realization [28]. Therefore, the parameter estimate is $\hat{\boldsymbol{\theta}}_{PLS} \epsilon\, \Re^{2\bar{n}}$, where $\bar{n} < n$, because in the development of the ARMA matrix $\boldsymbol{\Phi}(N)$ in (52) the system is intentionally *over-specified*, i.e., it is assumed that $n > \bar{n}$.

An estimate of the system dimension, $\bar{n}$ for the minimal realization is obtained using a variation of the independent-validation method given above referred to as PRESS (prediction residual error sum of squares) [14, 2, 4]. This is one of many functions in the MATLAB® chemometrics toolbox [20]. As with the independent-validation method, a training set and a test set are required, i.e.,

$$\textit{training data set} = \{\boldsymbol{\Phi}_{\text{train}}(\text{N}), \mathbf{y}_{\text{train}}(\text{N})\} \tag{54}$$

and

$$\textit{test data set} = \{\boldsymbol{\Phi}_{\text{test}}(\text{N}), \mathbf{y}_{\text{test}}(\text{N})\} \tag{55}$$

where the test data set in (55) is taken from a different portion of the collected data $\{\boldsymbol{u}(k), \boldsymbol{y}(k)\}$ than that used for the training data set in (54). It is assumed in generating the two data sets in (54) and (55) that the system is over-specified, i.e., $n > \bar{n}$. The PRESS values are computed as a function of the number of PLS factors. In the system identification problem, typically there will be many more *measurements* than the number of parameters to be estimated for an assumed system dimension, therefore, in PLS the maximum number of factors will be $q = 2n$. Therefore, the optimal number of factors will directly indicate the system dimension, i.e., $h^o = 2\bar{n}$, for a minimal realization. The optimal number of PLS factors will be twice the estimated system dimension because "$2n$" parameters must be determined, see (48).

After the optimal number of factors is determined, which will give an estimate of the system dimension, $\bar{n}$, the parameter vector can be estimated by forming a new set of data

$$\textit{final data set} = \{\boldsymbol{\Phi}_{\text{f}}(\text{N}; \hat{\text{n}}), \mathbf{y}_{\text{f}}(\text{N}; \hat{\text{n}})\} \tag{56}$$

i.e., the final data set used to estimate $\boldsymbol{\theta}$ from the collected data $\{\boldsymbol{u}(k), \boldsymbol{y}(k)\}$. The estimated system dimension is used to generate the ARMA matrix $\boldsymbol{\Phi}_f(N;\hat{n})$, and the output vector $\boldsymbol{y}_f(N;\hat{n})$ which contains the appropriate output samples from $\boldsymbol{y}(N)$. The number of samples used to generate $\boldsymbol{\Phi}_f(N;_{\hat{n}})$ and $\boldsymbol{y}_f(N;_{\hat{n}})$ does not necessarily have to be the same as used to develop $\{\boldsymbol{\Phi}_{train}(N), \boldsymbol{y}_{train}(N)\}$ and $\{\boldsymbol{\Phi}_{test}(N), \boldsymbol{y}_{test}(N)\}$, but must be $N > 2_{\hat{n}}$ [26]. However, using more samples will typically result in better estimates of the system parameters. An estimate of the parameter vector can be determined by computing the PLS calibration model vector. The calibration model vector will contain the $2_{\hat{n}}$ estimates of the system parameters, i.e.,

$$\hat{\boldsymbol{\theta}}_{\mathrm{PLS}} = \mathbf{b}_{\mathrm{f}} \tag{57}$$

where $\hat{\boldsymbol{\theta}}_{PLS} \in \Re^2_{\hat{n}}$ or $\hat{\boldsymbol{\theta}}_{PLS} \in \Re^{ho}$.

Before presenting the system identification example, a brief explanation of the relationship between the PLS calibration step and the estimate of the system dimension will be given. When the system dimension is intentionally chosen as $n > \bar{n}$ (over-specification) to generate (54) and (55), superfluous information exists in each row of the ARMA matrix, $\boldsymbol{\Phi}_{train}(N)$. That is, in each row of this matrix there will exist input/output samples from $\{\boldsymbol{u}(k), \boldsymbol{y}(k)\}$ that are associated with the system difference equation for the *actual* system of dimension $\bar{n} < n$, or a system of dimension less than $\bar{n}$ which is a minimal realization of the actual system. However, $\boldsymbol{\Phi}_{train}(N)$ for $(n > \bar{n})$ also contains redundant data. In *Step 5* of the *PLS1 Calibration Algorithm*, Section 3, the loading vectors, $\hat{\boldsymbol{b}}_h$, are generated for the data matrix, i.e., $\boldsymbol{\Phi}_{train}(N)$ in this example. Therefore, because the $\hat{\boldsymbol{t}}_h$ vectors (for $h = 1, 2,\ldots, 2n$) from *Step 3* are repeated approximations to the elements in $\boldsymbol{y}_{train}(N)$, the column elements in $\boldsymbol{\Phi}_{train}(N)$ associated with the largest positive elements in the $\hat{\boldsymbol{w}}_h$ vectors (for $h = 1, 2,\ldots, 2n$) from *Step 2* tend to indicate those column elements in $\boldsymbol{\Phi}_{train}(N)$ which exhibit the largest correlation with the elements in $\boldsymbol{y}_{train}(N)$. Therefore, when $\boldsymbol{\Phi}_{train}(N)$ is formed for an *over-specified* system $(n > \bar{n})$, this process will select the appropriate elements in the rows of $\boldsymbol{\Phi}_{train}(N)$ which are more closely associated with the system of minimal dimension. In the factor analysis process, when the *optimal* number of factors (h^o) is selected (as explained above), the resultant number of *optimal* factors will be $2_{\hat{n}}$. This must be the case because the dimension of the parameter vector, $\boldsymbol{\theta}$, is always twice the system dimension, therefore,

$$h^o = 2_{\hat{n}} \tag{58}$$

Thus, for a *minimal realization*, we seek the *first* minimum, see (35), to select the optimal number of PLS factors, h^o.

The example which follows illustrates how the selection of the system order can be carried out, and using this estimate of the system dimension the parameter vector can be estimated. The system analyzed is third order with parameters for the actual discrete system given as

$$\boldsymbol{\theta}^{\mathrm{T}} = [-1.005\ 0.505\ -0.0025\ 0.316\ 0.158632\ 0.000316] \tag{59}$$

with the sampling period $T_s = 2\pi/10$ sec and $\bar{n} = 3$. The parameters in (59) are related to the strictly proper rational z-transfer function given in its general form in (48). The simulated empirical data $\{u(k), y(k)\}$ were generated in MATLAB® using (59) and two different inputs, i.e., a chirp signal and Gaussian white noise with zero mean and unity variance. Both signals were generated with sequence lengths of 1024, and the chirp signal start frequency was 0 Hz and the end frequency was $f_s/4$ Hz (where f_s is the sampling frequency). Figure 11 shows the chirp signal used in the first simulation, and Figure 12 shows the fast Fourier transform (FFT) magnitude of the chirp signal. First, given the input $\mathbf{u}(k)$ as the chirp signal of length 1024, an equal length output vector $\mathbf{y}(k)$ was generated using a digital simulation process with the parameter vector in (59). Using the first 300 samples ($N = 300$) from the input/output data $\{\mathbf{u}(k), \mathbf{y}(k)\}$, and assuming the system dimension to be $n = 6$, the training ARMA matrix $\mathbf{\Phi}_{train}$ was generated along with the associated vector $\mathbf{y}_{train}$. Using the next N + 2 samples from the input/output data, the test ARMA matrix $\mathbf{\Phi}_{test}$ was generated along with the associated vector $\mathbf{y}_{test}$, for $N = 300$ and $n = 6$. The PRESS was generated using $\{\mathbf{\Phi}_{train}, \mathbf{y}_{train}, \mathbf{\Phi}_{test}, \mathbf{y}_{test}\}$, and the results are shown in Figure 13. That is, first using $\{\mathbf{\Phi}_{train}, \mathbf{y}_{train}\}$, the PLS weight loading vectors, $\hat{\mathbf{w}}_h$, the PLS score (latent variable) vectors, $\hat{\mathbf{t}}_h$, the PLS regression coefficients (inner relationships), $\hat{v}_h$, and the PLS loading vectors, $\hat{\mathbf{b}}_n$, for $h = 1, 2, \ldots, 12$, were generated. Then the test set, $\{\mathbf{\Phi}_{test}, \mathbf{y}_{test}\}$, was then used to compute the PRESS for $h = 1,2,\ldots, 12$, and the results are shown in Figure 13. Note that the first *minimum* in Figure 13 occurs at 4 and not at 6, which indicates the optimal number of PLS factors, i.e., $h^o = 4$. Because $h^o = 2_{\hat{n}}$, from (58), the PLS estimate of the system dimension is $_{\hat{n}}= 2$, which is not the actual system order, $\hat{n} = 3$. Therefore, the *minimal* PLS realization of the system is a second-order system.

Carrying out the same procedure, with Gaussian white noise now used as the system input $\boldsymbol{u}(k)$, results in the PRESS shown in Figure 14. Again the PLS estimate of the system dimension is shown to be $_{\hat{n}}= 2$, because $h^o = 4$ from Figure 14, i.e., from the first *minimum*. Using the first 300 samples from the $\{u(k), y(k)\}$ simulation data, with the system input $\boldsymbol{u}(k)$ as the chirp signal and $_{\hat{n}}= 2$, the final ARMA matrix, $\mathbf{\Phi}_f$, and the associated vector $\mathbf{y}_f$ were generated. From the $\{\mathbf{\Phi}_f, \mathbf{y}_f\}$ data, the PLS calibration vector was

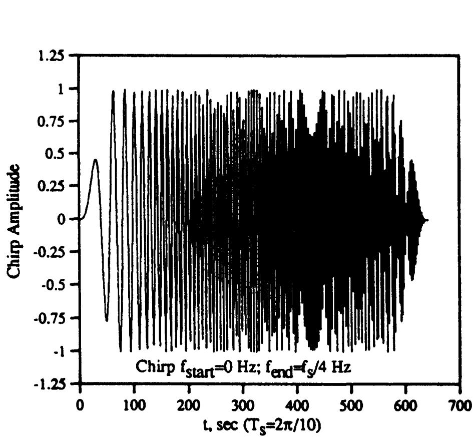

Figure 11. Chirp signal used as the system input, $u(k)$, in Example 3.

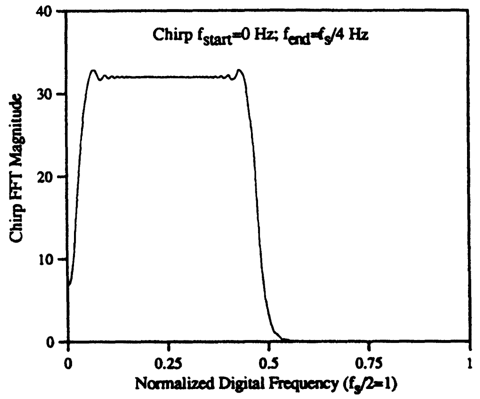

Figure 12. FFT of the chirp signal in Figure 11.

generated, i.e., $b_f = \hat{\theta}_{PLS}$. The PLS estimate of the parameter vector is given as

$$\hat{\theta}^T_{PLS} = [-0.999233\ 0.499353\ 0.315462\ 0.161980] \tag{60}$$

which is the PLS minimal realization of the system in (59). For the purpose of comparison, when 6 PLS factors were retained to develop the PLS calibration model vector, i.e., $\hat{\theta}_{PLS}$, the results were identical to (59), the actual system used to simulate the data.

It can be shown in a number of ways that (60) is a minimal realization of the actual system given in (59). First, if the poles and zeros are plotted in the z-plane for the actual system given in (59), it can be seen that a pole and zero close to the origin will almost cancel each other, see Figure 15, which if they were canceled would result in a second-order system. The poles and zeros for the minimal realization given in (60) are plotted in the z-plane shown in Figure 16. The poles and zeros for the actual third-order

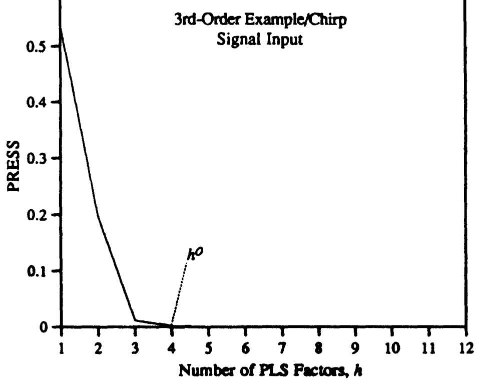

Figure 13. Selection of the optimal number of PLS factors from the PRESS for a third-order system with **u**(k) a chirp signal, h^0 is selected as 4 ($_{\hat{n}}$ = 2), for the minimal realization of the system.

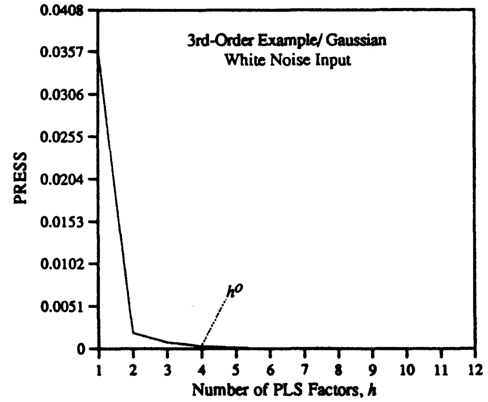

Figure 14. Selection of the optimal number of PLS factors from the PRESS for a third-order system with **u**(k) as Gaussian white noise (zero mean and unity variance), h^0 is selected as 4 ($_{\hat{n}}$ = 2), for the minimal realization of the system.

system given in (59) are

$$poles_{3rd\text{-}order\ system} = [0.5 + j0.5\ 0.5 - j0.5\ 0.005] \tag{61}$$

$$zeros_{3rd\text{-}order\ system} = [-0.5 - 0.002] \tag{62}$$

From (61) and (62), and Figure 15, it can be seen that the pole at 0.005 essentially cancels the zero at −0.002. The poles and zeros for the PLS minimal realization given in (60) are

$$poles_{2nd\text{-}order\ system} = [0.499616 + j0.499736\ 0.499616 - j0.499736] \tag{63}$$

$$zeros_{2nd\text{-}order\ system} = [-0.513468] \tag{64}$$

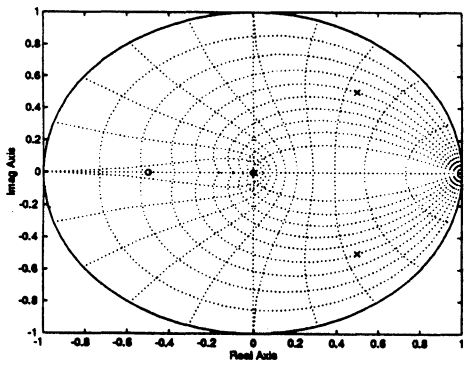

Figure 15. Plot of the poles and zeros for the 3rd-order system given in (59) in the z-plane.

Now, in (61) and (62), if the pole at 0.005 cancels the zero at -0.002, the resulting system is essentially the minimal realization given in (60), whose poles and zeros are shown in (63) and (64). This can also be seen by comparing the pole/zero locations for the two systems in Figures 15 and 16. The fact that the system in (60) is a minimal realization of the actual system in (59) can also be seen by comparing their unit-step responses. Figure 17 shows the difference between the unit step response for (59) and (60), and indicates they are essentially the same in both the transient and steady-state regions.

5. CONCLUSIONS AND FUTURE DIRECTIONS

Partial least-squares (PLS) regression has been presented as an alternate method to classical least-squares (CLS), and direct comparisons of the two methods were made. It has been shown that in many applications, PLS can out perform CLS, and thus, results in a calibration model that can yield better predictive performance than CLS. Because PLS is a *factor analysis* (*rank reduction*) method, noise reduction is an inherent feature in the selection process to determine the optimal number of factors to retain for the development of the PLS calibration model. The CLS method does not have this capability, therefore, CLS essentially retains all of the factors, or uses all of the available data to develop the calibration model. By comparison, PLS can simultaneously optimize the calibration model, $\boldsymbol{b}_f$, and reduce the potential for overfitting by not allowing the model to be fit to the noise, when the number of PLS factors is properly chosen.

When all of the factors in the PLS method are retained and the calibration model is developed, the results are identical to CLS. This was illustrated in the first simulation example that was presented. The second simulation example illustrated the exact nature of what the factor analysis in PLS can accomplish. Synthetic NIR *spectral data* were simulated, i.e., a component of interest was corrupted by Gaussian white noise and a highly dominant obscuring component was then added to these data. The first two PLS weight loading vectors, which are orthonormal, were related directly to the various components in the simulated data, and the resulting calibration model that resulted from

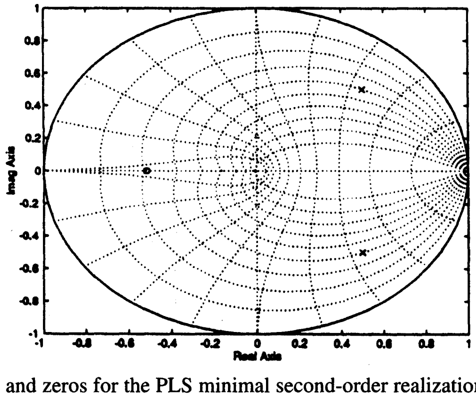

Figure 16. Plot of the poles and zeros for the PLS minimal second-order realization of (59) given in (60) in the z-plane.

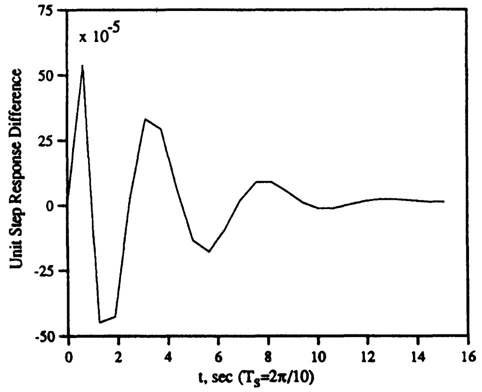

Figure 17. Difference of the unit-step responses for the actual third-order system in (59) and the PLS second-order minimal realization of (59) given in (60).

retaining 2 PLS factors. The first PLS weight loading vector was shown to be a first-order approximation to a highly dominant obscuring component in the synthetic data. This is subsequently removed in the recursive PLS calibration algorithm. Therefore, without any prior knowledge of the nature of the unwanted component, PLS can approximate it and remove its effect from the data, and thus, enhance the component of interest. In addition, it was shown that the third PLS weight loading vector was associated with noise, and thus, was not retained to develop the calibration model. The PLS calibration model (for 2 factors) resulted in three orders magnitude better predictive performance than CLS (where all of the data are used to develop the CLS calibration model), when tested on an independent, i.e., *unknown*, test data set. In the third example, it was shown how PLS can be applied to parametric system identification. It was shown that the system dimension and the system parameter vector can simultaneously be estimated using PLS when using the auto-regressive moving average (ARMA) method. When applying PLS for this purpose, the ARMA data matrix is intentionally *over-specified*. That is, in forming the ARMA data matrix the assumed system dimension is $n > \bar{n}$, where $\bar{n}$ is the actual system dimension. It was also shown in this example that the PLS estimate of the system dimension, $_{\hat{n}}$, can be less than the actual system dimension, i.e., $_{\hat{n}}< \bar{n}$. Therefore, PLS can yield an estimate of the parameter vector, θ, i.e., $\hat{\theta}_{PLS} \in \Re^{\hat{n}}$, which is associated with the *minimal realization* of the system.

We believe that PLS is an important and valuable analytical method that has a wide range of applications in signal processing. Our current research efforts are focused on developing algorithms for real-time implementation of PLS, development of PLS-based power spectrum estimation methods, and applications in controls to perform state estimation, to name a few. Another very important issue that was not discussed above pertains to the detection of *outliers* [3], [5], and [11]. An outlier, or abnormal observation, is a particular measurement than shows some type of departure from the global set of data. Many times is very important to be able to detect outliers so they can be discarded and not used for the development of the PLS calibration model. If retained and used in the development of the calibration model, the predictive performance of the model could be diminished. Therefore, there have been numerous methods developed for the detection of outliers [3], [5], and [11]. However, we feel that improved methods for detecting outliers is an important area that needs to be addressed, and the results from these efforts could lead to highly robust PLS calibration models.

References

1. K. R. Beebe and B. R. Kowalski, An Introduction to Multivariate Calibration and Analysis, *Analytical Chem.* **59** (1987), 1007A–1017A.
2. P. Geladi and B. R. Kowalski, Partial-Least-Squares Regression: A Tutorial, *Analytica Chimica Acta* **185** (1986), 1–17.
3. D. M. Haaland and E. V. Thomas, Partial Least-Squares Methods for Spectral Analyses. 1. Relation to Other Quantitative Calibration Methods and the Extraction of Qualitative Information, *Analytical Chem.* **60** (1988), 1193–1202.
4. B. R. Kowalski (Ed.), *Chemometrics-Mathematics in Statistics and Chemistry*, (Series C: Mathematical and Physical Sciences, vol. 138) D. Reidel, Dordrecht, Holland, 1984.
5. H. Martens and T. Naes, *Multivariate Calibration*, Wiley, New York, 1989.
6. S. D. Brown, Chemometrics, *Analytical Chem.* **62** (1990), 84R–101R.
7. H. Wold, Soft Modelling: The Basic Design and Some Extensions, *Systems Under Direct Observation, Causality-Structure-Prediction*, (Eds. K. G. Joreskog and H. Wold), Elsevier North Holland, Amsterdam, 1981.
8. H. Wold, In *Food Research and Drug Analysis*, (Eds. H. Martens and H. Russwurm), Applied Science Publ., London, 1983.
9. E. R. Malinowski, Theory of Error in Factor Analysis, *Analytical Chem.* **49** (1977), 606–612.
10. T. Naes, T. Isaksson, and B. Kowalski, Locally Weighted Regression and Scatter Correction for Near-Infrared Reflectance Data, *Analytical Chem.* **62** (1990), 664–673.

12. M. Avram and G. H. Mateescu, *Infrared Spectroscopy*, Wiley-Interscience, New York, 1972.

11. P. Williams and K. Norris (Eds.), *Near-Infrared Technology*, Amer. Assoc. of Cereal Chemists, St. Paul, 1987.

13. E. R. Malinowski et al., *Factor Analysis in Chemistry*, (2nd. ed.), Wiley, New York, 1991.
14. N. Draper and H. Smith, *Applied Regression Analysis, Second Edition*, Wiley-Interscience, New York, 1981.
15. K. V. Mardia, J. T. Kent, and J. M. Bibby, *Multivariate Analysis*, Academic, London, 1992.
16. D. M. Haaland, Classical versus Inverse Least-Squares Methods in Quantitative Spectral Analyses, *Spectroscopy* **2** (1987), 56–57.
17. E. R. Malinowski, Determination of the Number of Factors and the Experimental Error in a Data Matrix, *Analytical Chem.* **49** (1977), 612–617.
18. R. Marbach and H. M. Heise, Calibration Modeling by Partial-Least Squares and Principal Component Regression and Its Optimization Using an Improved Leverage Correction for Prediction Testing, *Chemometrics and Intelligent Lab. Syst.* **9** (1990), 45–63.
19. E. V. Thomas and D. M. Haaland, Comparison of Multivariate Calibration Methods for Quantitative Spectral Analysis, *Analytical Chem.* **62** (1990), 1091–1099.
20. R. Kramer, *Chemometrics Toolbox—For Use with MATLAB®*, The Mathworks, Natick, MA, 1993.
21. G. H. Golub and C. F. Van Loan, *Matrix Computations*, (2nd. ed.), Johns Hopkins, Baltimore, 1989.
22. C. N. Banwell, *Fundamentals of Molecular Spectroscopy*, McGraw-Hill, New York, 1966.
23. B. P. Straughan and S. Walker (Eds.), *Spectroscopy*, vol. 2, Wiley, New York, 1976.
24. F. M. Ham, G. M. Cohen, K. Patel, and B. R. Gooch, Multivariate Determination of Glucose Using NIR Spectra of Human Blood Serum, *Proc. IEEE Engr. in Med. and Bio. Soc., 16th Annual Inter. Conf.* **16 (II)** (1994), 818–819.
25. F. M. Ham, I. Kostanic, G. M. Cohen, and B. R. Gooch, Determination of Glucose Concentrations in an Aqueous Matrix from NIR Spectra Using Optimal Time-Domain Filtering and Partial Least-Squares Regression, Submitted to IEEE Transactions on Biomedical Engineering for review (1994).
26. G. F. Franklin, J. D. Powell, and M. L. Workman, *Digital Control of Dynamical Systems*, (2nd ed.) Addison Wesley, Reading, MA, 1990.
27. L. Ljung, *System Identification Theory for the User*, PTR Prentice-Hall, Englewood Cliffs, NJ, 1987.
28. C. T. Chen, *Linear System Theory and Design*, Saunders College Publ. (HRW), New York, 1984.