Partial Least Squares Regression (PLSR) – What Is It? How To Use It.

Fredric M. Ham, Ph.D., FIEEE, FSPIE, FINNS
Dean, College of Engineering
Harris Professor of Electrical Engineering

Florida Institute of Technology
High Tech with a Human Touch™

COLLEGE OF ENGINEERING
We Engineer the Future™

Introduction

- **Partial Least Squares Regression (PLSR) is a factor analysis-based method.**
- **It is used to construct predictive models when the number of factors is very large and highly collinear (factors are significantly redundant)**
- **What are factors? Answer: Measured variables**
- **These factors are used many times to explain, regulate, or predict the behavior of other variables or the responses.**

Introduction (cont')

- **Multiple Linear Regression (MLR) methods work well when (a) the number of factors is small, (b) they are not significantly collinear (redundant) and (c) there exists a well-defined relationship to the system responses. DATA ➔ INFORMATION**
- **When any or all of the above conditions don't exist MLR will fail.**
- **That is when PLSR becomes a very valuable modeling tool.**

Introduction (cont')

- **Therefore, when one is faced with many variables, the relationship between the inputs (independent variables) and the responses (dependent variables) are not well-known, the PLSR statistical calibration modeling approach can be employed.**

- **This calibration model when properly constructed can produce highly robust predictive results.**

- **PLSR methods are known as bilinear factor models.**

Introduction (cont')

- **It is important to note that PLSR is used to "predict" a target value (that is, predict a response) given a measured quantity. It is not used to discover the underlying relationship between the system variables.**

- **PLSR is an excellent method used for prediction when the physical system characteristics, or the details of the process, is not well understood.**

- **Through the factor analysis approach, unneeded factors can be removed that can improve the predictive performance of the calibration model.**

Introduction (cont')

- **The removal of the unwanted factors can be carried out without detailed knowledge of their characteristics.**

- **Thus, unwanted, obscuring components that can exist in the data can be removed in a systematic manner, thus improving the predictive performance of the calibration model.**

- **Some applications include: statistical calibration model development for parameter estimation, estimation of sinusoids in additive noise, parametric system identification, robust event classification …**

The goal here is to have a useful predictive tool when we finish this tutorial that can be used for your future work in many cases …

Let's get into the details …

Florida Institute of Technology
High Tech with a Human Touch™

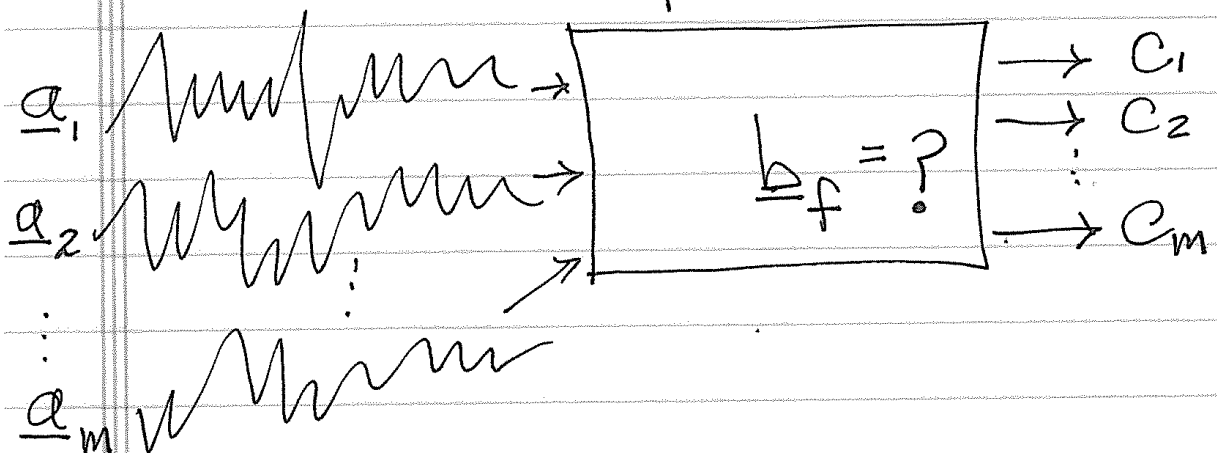
Part I

Florida Institute of Technology
High Tech with a Human Touch™

COLLEGE OF ENGINEERING
We Engineer the Future™

* We will now begin the study of Partial Least-Squares Regression ⟹ PLS regression, or just PLSR.

* The basic problem (revisited) is:



* What is the relationship between each of the M measurements (each of length N) and the M associated target values, $c_1, c_2, \ldots, c_M$.

* In other words what is the calibration model?

* We saw before that least-squares (classical) solution

leads to a very basic problem ⟹

$$\underline{A} = \begin{bmatrix} \underline{a}_1 \\ \underline{a}_2 \\ \vdots \\ \underline{a}_m \end{bmatrix} \quad (23)$$

→ each row is a measurement (there are n columns).

and $\underline{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \end{bmatrix}$ (24) associated target values.

And what links together the independent variables blocks of data $\underline{A}$, and the dependent variable blocks, $\underline{c}$ is the calibration model, $\underline{b}_f$, with a defined error of calibration:

$$\underline{e} = \underline{c} - \underline{A}\underline{b}_f \quad (25)$$

Florida Institute of Technology
High Tech with a Human Touch™

COLLEGE OF ENGINEERING
We Engineer the Future™

Defining the error cost function as:

$$\mathcal{E}(b_f) = \frac{1}{2}\|e\|_2^2 \quad (26)$$

and minimizing this leads to the classical least-squares solution:

$$\mathcal{E}(\underline{b}_f) = \frac{1}{2}\|\underline{e}\|_2^2 = \frac{1}{2}\underline{e}^T\underline{e} = \left[(\underline{c} - \underline{A}\underline{b}_f)^T(\underline{c} - \underline{A}\underline{b}_f)\right] =$$

$$\left[(\underline{c}^T - \underline{b}_f^T\underline{A}^T)(\underline{c} - \underline{A}\underline{b}_f)\right] = \underline{c}^T\underline{c} - \underline{c}^T\underline{A}\underline{b}_f - \underline{b}_f^T\underline{A}^T\underline{c} + \underline{b}_f^T\underline{A}^T\underline{A}\underline{b}_f$$

Want to compute the gradient of $\mathcal{E}(\underline{b}_f) = 0$

$$\Downarrow$$

$$\nabla_{\underline{b}_f}\mathcal{E}(\underline{b}_f) = \frac{\partial}{\partial \underline{b}_f}\left(\underline{c}^T\underline{c} - \underline{c}^T\underline{A}\underline{b}_f - \underbrace{\underline{b}_f^T\underline{A}^T\underline{c}}_{\underline{c}^T\underline{A}\underline{b}_f} + \underline{b}_f^T\underline{A}^T\underline{A}\underline{b}_f\right)$$

$$= \frac{\partial}{\partial \underline{b}_f}\left(\underline{c}^T\underline{c} - 2\underline{c}^T\underline{A}\underline{b}_f + \underline{b}_f^T\underline{A}^T\underline{A}\underline{b}_f\right)$$

Recall:

$$\frac{\partial}{\partial \underline{x}}(\underline{x}^T\underline{y}) = \frac{\partial}{\partial \underline{x}}(\underline{y}^T\underline{x}) = \underline{y}$$

$$\frac{\partial}{\partial \underline{x}}(\underline{x}^T\underline{A}\underline{x}) = 2\underline{A}\underline{x}$$

$$\nabla_{\underline{b}_f}\mathcal{E}(\underline{b}_f) = \frac{\partial}{\partial \underline{b}_f}\left(\underline{c}^T\underline{c} - 2(\underline{A}^T\underline{c})^T\underline{b}_f\right) + \frac{\partial}{\partial \underline{b}_f}\left(\underline{b}_f^T\underline{A}^T\underline{A}\underline{b}_f\right) =$$

$$-2\underline{A}^T\underline{c} + 2\underline{A}^T\underline{A}\underline{b}_f = 0 \Rightarrow$$

$$\underline{A}^T\underline{A}\underline{b}_f = \underline{A}^T\underline{c} \Rightarrow$$

$$\underline{b}_f = (\underline{A}^T\underline{A})^{-1}\underline{A}^T\underline{c} \Rightarrow \text{Classical least Squares Result.}$$

Florida Institute of Technology
High Tech with a Human Touch™

COLLEGE OF ENGINEERING
We Engineer the Future™

$$\underline{b}_f = (\underline{A}^T\underline{A})^{-1}\underline{A}^T\underline{c} \quad (27)$$

* The problem is, if the measurements are heavily corrupted with measurement noise, then the calibration model, $\underline{b}_f$, will also be

⇓

* Obviously this is highly undesirable, because when using the calibration model to predict output values, the predictions will be unreliable.

∴ It is desired to have a method that can create a model with minimal corruptive influence from the noisy measurements, without having to know the source of noise or even other components that may reside in the collected signals that are of no interest to us, but none-the-less still exist.

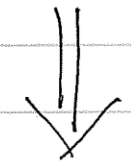
⇓

PLS Regression is the answer.

* There are two fundamental parts of the PLS regression method:
(1) Calibration, & (2) Prediction.

* Let's start with the calibration step first then we'll move on to the prediction part!

* Remember what we are trying to do ⇒ Build a calibration model that will have minimal effects in the model itself due to noise corrupting the measurements and other unwanted effects in the data.



* PLS Regression is a factor Analysis method.

* Assume we have available $\{\underline{A}, \underline{c}\}$ ($\underline{A}$: data matrix, $\underline{c}$: target values) $\underline{A} \in \mathbb{R}^{m \times n}$, $\underline{c} \in \mathbb{R}^{m \times 1}$

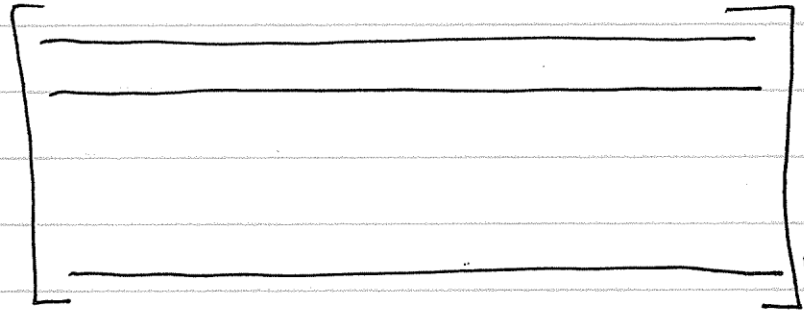
Here are the calibration steps:

→ only one target value for each measurement. (Univariate Calibration)

PLS1 Calibration Algorithm

Step 1. scaling: Mean-centering and variance. Given the data matrix

$$\underline{A} \in \mathbb{R}^{m \times n} \quad (28)$$

$\underline{A} =$ 

Mean center and variance scale the data if necessary.

And the debate rages on whether this is a good thing to do!

$\Rightarrow$ First an index h (number count for the PLS factors) is initially set to one

$h \leftarrow 1$

Step 2. Form the weight loading vector $\hat{w}_h \in \mathbb{R}^{n \times 1}$

This is actually a CLS (classical least-squares) calibration step, the model is given by:

$$\underline{A} = \underline{C}\,\underline{w}_h^T + \underline{E}_A \quad (29)$$

where $\underline{E}_A \in \mathbb{R}^{m \times n}$ of residuals

and the least-squares solution is:

$$\hat{\underline{w}}_h = \underline{A}^T \underline{c} / \underline{c}^T \underline{c} \qquad (30)$$

Then normalize, $\hat{\underline{w}}_h$ by dividing by the h_2-norm, i.e.,

$$\hat{\underline{w}}_h = \frac{\hat{\underline{w}}_h}{\|\hat{\underline{w}}_h\|_2} \qquad (31)$$

Details: Again given $\{\underline{A}, \underline{c}\}$, $\underline{A} \in \mathbb{R}^{m \times n}$; $\underline{c} \in \mathbb{R}^{m \times 1}$

Solve from the model: $\underline{A} = \underline{c}\,\underline{w}_h^T + \underline{E}_A \qquad (32)$

the weight loading vector: $\underline{w}_h^T \in \mathbb{R}^{n \times 1}$

Define the error cost functionals:

$$\underset{\sim}{\overset{w_h}{\mathcal{L}}}(\underset{\sim}{w_h}) = \frac{1}{2}\,tr\left(\underset{\sim}{E_A}^T \underset{\sim}{E_A}\right) \quad (33)$$

where: $\underset{\sim}{E_A} = \underset{\sim}{A} - \underset{\sim}{a}\,\underset{\sim}{w_h}^T \quad (34)$

↖ from Eq. 24.

$$\therefore \underset{\sim}{\overset{w_h}{\mathcal{L}}}(\underset{\sim}{w_h}) = \frac{1}{2}\,tr\left\{\underbrace{\left(\underset{\sim}{A} - \underset{\sim}{a}\,\underset{\sim}{w_h}^T\right)^T}_{\underset{\sim}{A}^T - \underset{\sim}{w_h}\underset{\sim}{a}^T}\left(\underset{\sim}{A} - \underset{\sim}{a}\,\underset{\sim}{w_h}^T\right)\right\} =$$

$$= \frac{1}{2}\,tr\left\{\underset{\sim}{A}^T\underset{\sim}{A} - \underset{\sim}{A}^T\underset{\sim}{a}\,\underset{\sim}{w_h}^T - \underset{\sim}{w_h}\underset{\sim}{a}^T\underset{\sim}{A} + \underset{\sim}{w_h}\underset{\sim}{a}^T\underset{\sim}{a}\,\underset{\sim}{w_h}^T\right\} \quad (35)$$

① ② ③

XIII 13

Computing the gradient:

Recall: $\frac{\partial}{\partial A} tr(BAC) = B^T C^T$

$\frac{\partial}{\partial A} tr(BA^T C) = CB$

$\therefore$ (1) $-\frac{1}{2}\frac{\partial}{\partial w_h} tr\left(A^T \underline{e}\, w_h^T\right) = -\frac{1}{2} A^T \underline{e}$ (36)

(2) $-\frac{1}{2}\frac{\partial}{\partial w_h} tr\left(w_h \underline{e}^T A\right) = -\frac{1}{2} A^T \underline{e}$ (37)

(3) $\frac{1}{2}\frac{\partial}{\partial w_h} tr\left(w_h \underline{e}^T \underline{e}\, w_h^T\right) = \frac{1}{2}\left(w_h \underline{e}^T \underline{e} + w_h \underline{e}^T \underline{e}\right)$ (38)

$\therefore \nabla_{w_h} h(w_h) = -A^T \underline{e} + w_h \underline{e}^T \underline{e}$ (39)

and setting this equal to zero

gives:

$$-\underline{A}^T\underline{a} + w_h\underline{a}^T\underline{a} = 0 \Rightarrow$$

$$w_h\underline{a}^T\underline{a} = \underline{A}^T\underline{a}$$

$$\Downarrow$$

$$\hat{w}_h = \underline{A}^T\underline{a} / \underline{a}^T\underline{a} \quad (40)$$

↳ Which is what was stated in Eq. 30.

Notes:

Each vector $\hat{w}_h$ (for $h = 1, 2, \ldots$) is the weight vector which is proportional to

a weighted average of the row elements in the data matrix $\underline{A}$, where the weights on the average are proportional to the elements in $\underline{c}$. There $\hat{n}$ vectors are constructed to be orthonormal.

Step 3. Generation of the score (latent variable) vector, $\underline{\hat{t}}_h \in \mathbb{R}^{m \times 1}$

* In this step, $\underline{A}$ is now written w.r.t. the latent variables (or scores), i.e., the model is:

$$\underline{A} = \underline{t}_h \hat{w}_h^T + \underline{E}_A \quad (41)$$

and the least-squares solution is:

$$\hat{\underline{t}}_h = \underline{A}\,\hat{w}_h \Big/ \underbrace{\hat{w}_h^T \hat{w}_h}_{=1} = \underline{A}\,\hat{w}_h \quad (42)$$

$\Rightarrow$ because the $\hat{w}_h$ are orthonormal

Details: Given the model:

$$\underline{A} = \underline{t}_h w_h^T + \underline{E}_A$$

we define the error cost function as:

$$h^{t_h}(\underline{t}_h) = \frac{1}{2} tr(\underline{E}_A^T \underline{E}_A)$$

where: $\underline{E}_A = \underline{A} - \underline{t}_h \hat{\underline{w}}_h^T$

$$\therefore h^{t_h}(\underline{t}_h) = \{ \underbrace{\underline{A} - \underline{t}_h \hat{\underline{w}}_h^T}_{\underline{A}^T - \hat{\underline{w}}_h \underline{t}_h^T} \}^T (\underline{A} - \underline{t}_h \hat{\underline{w}}_h^T) \}$$

$$= \frac{1}{2} tr \{ \underline{A}^T \underline{A} - \underline{A}^T \underline{t}_h \hat{\underline{w}}_h^T - \hat{\underline{w}}_h \underline{t}_h^T \underline{A} + \hat{\underline{w}}_h \underline{t}_h^T \underline{t}_h \hat{\underline{w}}_h^T \}$$

① ② ③

Computing the gradient:

① $-\frac{1}{2} \frac{\partial}{\partial \underline{t}_h} tr(\underline{A}^T \underline{t}_h \hat{\underline{w}}_h^T) = -\frac{1}{2} \frac{\partial}{\partial \underline{t}_h} tr(\underline{t}_h \hat{\underline{w}}_h^T \underline{A}^T)$

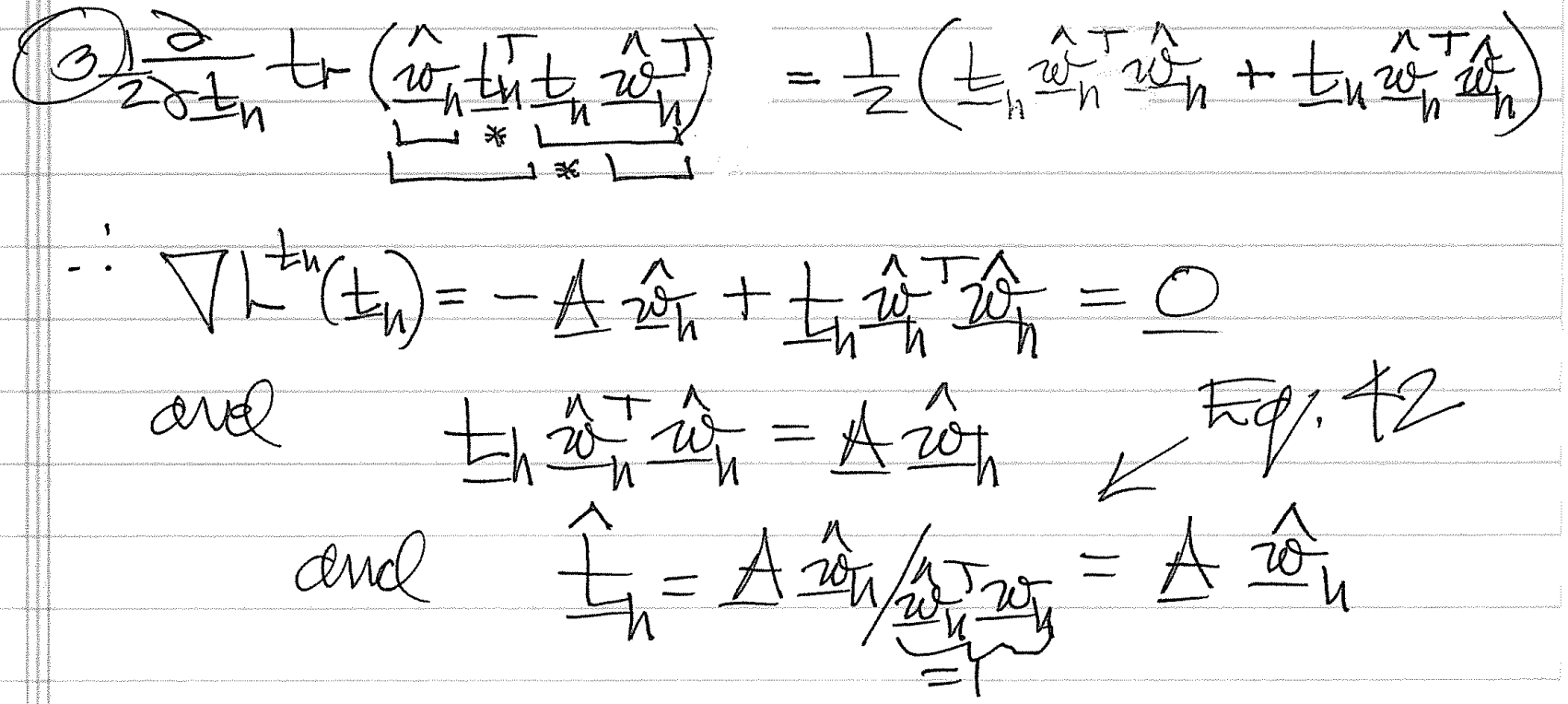
$n \times m \quad m \times 1 \quad 1 \times n$

$$= -\frac{1}{2} \underline{A} \hat{\underline{w}}_h$$

② $-\frac{1}{2} \frac{\partial}{\partial \underline{t}_h} tr(\hat{\underline{w}}_h \underline{t}_h^T \underline{A}) = -\frac{1}{2} \frac{\partial}{\partial \underline{t}_h} tr(\underline{t}_h^T \underline{A} \hat{\underline{w}}_h)$

$n \times 1 \quad 1 \times m \quad m \times n$

$$= -\frac{1}{2} \underline{A} \hat{\underline{w}}_h$$


(3) $\frac{\partial}{2\partial \underline{t}_h} tr\left(\hat{\underline{w}}_h \underline{t}_h^T \underline{t}_h \hat{\underline{w}}_h^T\right) = \frac{1}{2}\left(\underline{t}_h \hat{\underline{w}}_h^T \hat{\underline{w}}_h + \underline{t}_h \hat{\underline{w}}_h^T \hat{\underline{w}}_h\right)$
$\therefore \nabla L^{t_h}(\underline{t}_h) = -\underline{A}\hat{\underline{w}}_h + \underline{t}_h \hat{\underline{w}}_h^T \hat{\underline{w}}_h = \underline{0}$
and $\underline{t}_h \hat{\underline{w}}_h^T \hat{\underline{w}}_h = \underline{A}\hat{\underline{w}}_h$
Eq. #2
and $\hat{\underline{t}}_h = \underline{A}\hat{\underline{w}}_h / \hat{\underline{w}}_h^T \hat{\underline{w}}_h = \underline{A}\hat{\underline{w}}_h$
$= 1$

Notes: The score vectors are obtained as a least-squares estimate (i.e., $\hat{E}_h$) by regressing $\underline{A}$ on the weight loading vectors, i.e., $\hat{W}_h$.

$\hat{W}_1$ → is the first-order attempt to represent the uncorrupted data, from the rows of the data matrix $\underline{A}$, that are corrupted, and

$\hat{E}_1$ → represents a first-order attempt to determine the amount of the pure component of interest (i.e., information contained in the target vector $\underline{c}$) in each of the associated rows of $\underline{A}$.

∴ In PLS, the vector $\hat{E}$ is related to both $\underline{A}$ & $\underline{c}$, as well as $\hat{W}_h$.

* We will continue on from here next time.

Part II

Florida Institute of Technology
High Tech with a Human Touch™

COLLEGE OF ENGINEERING
We Engineer the Future™

Partial Least Squares (PLS1) Regression (Cont')

Given data: $\{\underline{A}, \underline{c}\}$; Data Matrix: $\underline{A} \in \mathbb{R}^{m \times n}$; Target vector: $\underline{c} \in \mathbb{R}^{m \times 1}$

* We have thus far derived expressions to compute the:

Weight loading vectors, $\hat{\underline{w}}_h \in \mathbb{R}^{n \times 1}$

$$\hat{\underline{w}}_h = \underline{A}^T \underline{c} / \underline{c}^T \underline{c} \quad (1)$$

$$\hat{\underline{w}}_h \leftarrow \frac{\hat{\underline{w}}_h}{\|\hat{\underline{w}}_h\|_2} \quad (2)$$

and the score vector (latent variable vector), $\hat{\underline{t}}_h \in \mathbb{R}^{m \times 1}$

$$\hat{\underline{t}}_h = \underline{A} \hat{\underline{w}}_h \quad (3)$$

This is as far as we get.

Step 4: Relating the score vector, $\hat{t}_h$, to the elements of $\underline{c}$.

* Here the score vector $\hat{t}_h$ (these are sometimes even referred to as the intensities in the new coordinate system) is related to the elements of the $\underline{c}$ vector using a linear least-squares regression $\Rightarrow$

where again $\hat{\underline{t}}_h$ (latent variable) is associated with the key feature relating to the pure component of interest contained in each row of the data matrix, $\underline{A}$.

* The relationship between $\hat{\underline{t}}_h$ & $\underline{e}$ is modeled as:

$$\underline{e} = V_h \hat{\underline{t}}_h + \underline{e}_a \quad (4)$$

and the least-squares solution is:

$$\hat{V}_h = \hat{\underline{t}}_h^T \underline{e} \,/\, \hat{\underline{t}}_h^T \hat{\underline{t}}_h \quad (5)$$

where $\hat{V}_h \in \mathbb{R}$ (6)

and the scalar regression coefficients are the inner relationships, for $h = 1, 2, \ldots, h^*$ (i.e., the optimal number of factors).

* $\underline{e} \in \mathbb{R}^{m \times 1}$ contains the PLS residuals associated with $\underline{e}$.

Details:

Starting with the model:

$$\underline{c} = V_h \hat{\underline{t}}_h + \underline{e}_c \quad (7)$$

* We define an error cost function as:

$$L^{V_h}(V_h) = \frac{1}{2}\|\underline{e}_c\|_2^2 = \frac{1}{2}\underline{e}_c^T \underline{e}_c \quad (8)$$

and from Eq. 7 we can write

$$\underline{e}_c = \underline{c} - V_h \hat{\underline{t}}_h \quad (8)$$

$$\therefore \quad L^{V_h}(V_h) = \frac{1}{2}\underline{e}_c^T \underline{e}_c = \frac{1}{2}\underbrace{\left(\underline{c} - V_h \hat{\underline{t}}_h\right)^T}_{\underline{c}^T - V_h \hat{\underline{t}}_h^T} * \left(\underline{c} - V_h \hat{\underline{t}}_h\right)$$

$$= \frac{1}{2}\left\{ \underline{e}^T\underline{e} - \underline{e}^T V_h \hat{\underline{E}}_h - V_h \hat{\underline{E}}_h^T \underline{e} + V_h \hat{\underline{E}}_h^T V_h \hat{\underline{E}}_h \right\}$$

$$\nabla L^{V_h}(V_h) = \frac{\partial}{\partial V_h} L^{V_h}(V_h) = \qquad V_h^2 \hat{\underline{E}}_h^T \hat{\underline{E}}_h$$

$$\frac{1}{2}\left\{ -\underline{e}^T \hat{\underline{E}}_h - \hat{\underline{E}}_h^T \underline{e} + 2 V_h \hat{\underline{E}}_h^T \hat{\underline{E}}_h \right\}$$

$$-\hat{\underline{E}}_h^T \underline{e}$$

$$= \frac{1}{2}\left\{ -2\hat{\underline{E}}_h^T \underline{e} + 2 V_h \hat{\underline{E}}_h^T \hat{\underline{E}}_h \right\} \Rightarrow$$

Setting this equal to zero $\Rightarrow$

$$V_h \hat{\underline{E}}_h^T \hat{\underline{E}}_h = \hat{\underline{E}}_h^T \underline{e}$$

$$\therefore \quad \frac{\hat{\underline{E}}_h^T \underline{e}}{\hat{\underline{E}}_h^T \hat{\underline{E}}_h}$$

$$\hat{v}_h = \hat{t}_h^T e / \hat{t}_h^T \hat{t}_h \quad (9)$$

Which is what was reported in Eq. 5.

Step 5. Generation of the PLS loading vector, $\underline{b}_h \in \mathbb{R}^{n \times 1}$, for the data matrix, $\underline{A}$.

* We now form a new model for $\underline{A}$ as:

$$\underline{A} = \hat{t}_h \underline{b}_h^T + \underline{E}_A \quad (10)$$

and the least-squares solution is:

$$\hat{\underline{b}}_h = \underline{\underline{A}}^T \hat{\underline{t}}_h \Big/ \hat{\underline{t}}_h^T \hat{\underline{t}}_h \qquad (11)$$

Details:

Starting with the model in Eq. 10 and a defined error cost function:

$$h^{b_h}(b_h) = \frac{1}{2} tr\left(\underline{E}_A^T \underline{E}_A\right) \quad (12)$$

where $\underline{E}_A = \underline{A} - \hat{\underline{t}}_h \underline{b}_h^T$ (13)

$$\therefore h^{b_h}(b_h) = \frac{1}{2} tr\left(\underline{E}_A^T \underline{E}_A\right) =$$

$$\frac{1}{2} tr\left\{ \underbrace{\left(\underline{A} - \hat{\underline{t}}_h \underline{b}_h^T\right)^T}_{\underline{A}^T - \underline{b}_h \hat{\underline{t}}_h^T} \left(\underline{A} - \hat{\underline{t}}_h \underline{b}_h^T\right) \right\} =$$

$$\frac{1}{2} tr\left\{ \underline{A}^T\underline{A} - \underline{A}^T \hat{\underline{t}}_h \underline{b}_h^T - \underline{b}_h \hat{\underline{t}}_h^T \underline{A} + \underline{b}_h \hat{\underline{t}}_h^T \hat{\underline{t}}_h \underline{b}_h^T \right)$$

① ② ③

Now compute the gradient of $L^{b_h}(b_h)$: Against use:

$$\frac{\partial}{\partial A} tr(BAC) = B^T C^T$$

$$\frac{\partial}{\partial A} tr(BA^T C) = CB$$

$$① \quad -\frac{1}{2}\frac{\partial}{\partial b_h} tr(A^T \hat{t}_h b_h^T) = -\frac{1}{2} A^T \hat{t}_h \quad (14)$$

$$② \quad -\frac{1}{2}\frac{\partial}{\partial b_h} tr(b_h \hat{t}_h^T A) = -\frac{1}{2} A^T \hat{t}_h \quad (15)$$

$$③ \quad \frac{1}{2}\frac{\partial}{\partial b_h} tr(b_h \hat{t}_h^T \hat{t}_h b_h^T) = \frac{1}{2}(b_h \hat{t}_h^T \hat{t}_h + b_h \hat{t}_h^T \hat{t}_h) \quad (16)$$

(* ... *)

$$\therefore \nabla L^{b_h}(b_h) = -A^T \hat{t}_h + b_h \hat{t}_h^T \hat{t}_h$$

and $-\hat{A}^T \hat{\xi}_h + \hat{b}_h \hat{\xi}_h^T \hat{\xi}_h = 0 \Rightarrow$

$\hat{b}_h \hat{\xi}_h^T \hat{\xi}_h = \hat{A}^T \hat{\xi}_h \Rightarrow$

$$\hat{b}_h = \hat{A}^T \hat{\xi}_h / \hat{\xi}_h^T \hat{\xi}_h \quad (17)$$

↳ Which is what was reported in Eq. 11.

Notes:

$\underline{b}_1$ represents an attempt to account for as much of the variation in $\underline{A}$ while simultaneously correlating with the score vector $\underline{t}_h$ which approximates $\underline{c}$. The $\underline{b}_h$ are not mutually orthogonal.

Step 6. Calculation of the Residuals in $\underline{A}$ and $\underline{c}$.

* From Eq. 10, which is repeated here,

$$\underline{A} = \underline{t}_h \underline{b}_h^T + \underline{E}_A \quad (10)$$

we can see that the outer product of the score vector, $\underline{t}_h$ with the loading

vector, $\hat{b}_h$, will give a PLS approximation to the data matrix, $\underline{A}$. Therefore, for each successive h, an approximation to the rows of $\underline{A}$ can be obtained as:

$$\hat{t}_h \hat{b}_h^T$$

and the $\underline{A}$ residuals are written from Eq. 10 (or 18) as:

$\underline{A}$ residuals:

$$\underline{E}_A = \underline{A} - \hat{t}_h \hat{b}_h^T \quad (19)$$

* Also, from Eq. 7, repeated here,

$$\underline{c} = V_h \hat{\underline{t}}_n + \underline{e}_c \quad (20)$$

we can write the residuals for the target vector $\underline{c}$, as:

$\underline{c}$ residuals:

$$\boxed{\underline{e}_c = \underline{c} - \hat{V}_h \hat{\underline{t}}_n \quad (21)}$$

where for each successive h, an approximation to $\underline{c}$ is given as

$\hat{v}_h$ $\hat{\underline{t}}_h$ based on the information in the data matrix $\underline{A}$.

Step 7. Increment h, Substitute $\underline{E}_A$ for $\underline{A}$ and $\underline{e}_c$ for $\underline{c}$ in Step 2 and continue for the desired number of loading vectors (or the optimal number of PLS factors → h°).

---

* All 7 steps are summarized in the following Table:

PLS1 Calibration Algorithm

step 1. Pretreatment of the data:

mean center $A \in \Re^{mxn}$ and

$c \in \Re^{mx1}$, and variance scale A
if necessary. Set the index h to 1
(where h is the number of PLS factors)

step 2. Forming the *weight loading vector*, $\hat{w}_h \in \Re^{nx1}$

model: $A = cw_h^T + E_A$ ($E_A \in \Re^{mxn}$ contains the A residuals)

least-squares solution: $\hat{w}_h = A^T c / c^T c$

normalize $\hat{w}_h$, i.e., $\hat{w}_h \leftarrow \dfrac{\hat{w}_h}{\|\hat{w}_h\|_2}$

step 3. Generation of the *score (latent variable) vector*, $\hat{t}_h \in \Re^{mx1}$

model: $A = t_h \hat{w}_h^T + E_A$

least-squares solution: $\hat{t}_h = A\hat{w}_h / \hat{w}_h^T \hat{w}_h = A\hat{w}_h$

step 4. Relating the score vector, $\hat{t}_h$, to the elements of c

model: $c = v_h \hat{t}_h + e_c$ ($e_c \in \Re^{mx1}$ contains the c residuals)

least-squares solution: $\hat{v}_h = \hat{t}_h^T c / \hat{t}_h^T \hat{t}_h$ ($v_h \in \Re$ is the *scalar regression coefficient* (inner relationship) relating $\hat{t}_h$ to the elements in c)

step 5. Generation of $\hat{b}_h \in \Re^{nx1}$, the *loading vector* for A

model: $A = \hat{t}_h b_h^T + E_A$

least-squares solution: $\hat{b}_h = A^T \hat{t}_h / \hat{t}_h^T \hat{t}_h$

step 6. Calculation of the residuals in A and c

A residuals: $E_A = A - \hat{t}_h \hat{b}_h^T$

c residuals: $e_c = c - \hat{v}_h \hat{t}_h$

step 7. Increment h, substitute E_A for A and e_c for c in step 2 and continue for the desired number of loading vectors (or the optimal number of PLS factors, h^o †)

† If all of the factors were retained, $h^o = m$ (for $m < n$) or $h^o = n$ (for $m \geq n$)

PLS1 Calibration Algorithm

step 1. Pretreatment of the data:
mean center $A \in \Re^{mxn}$ and
$c \in \Re^{mx1}$, and variance scale A
if necessary. Set the index h to 1
(where h is the number of PLS factors)

step 2. Forming the *weight loading vector*, $\hat{w}_h \in \Re^{nx1}$

model: $A = c w_h^T + E_A$ ($E_A \in \Re^{mxn}$ contains the A residuals)

least-squares solution: $\hat{w}_h = A^T c / c^T c$

normalize $\hat{w}_h$, i.e., $\hat{w}_h \leftarrow \frac{\hat{w}_h}{\|\hat{w}_h\|_2}$

step 3. Generation of the *score (latent variable) vector*, $\hat{t}_h \in \Re^{mx1}$

model: $A = t_h \hat{w}_h^T + E_A$

least-squares solution: $\hat{t}_h = A\hat{w}_h / \hat{w}_h^T \hat{w}_h = A\hat{w}_h$

step 4. Relating the score vector, $\hat{t}_h$, to the elements of c

model: $c = v_h \hat{t}_h + e_c$ ($e_c \in \Re^{mx1}$ contains the c residuals)

least-squares solution: $\hat{v}_h = \hat{t}_h^T c / \hat{t}_h^T \hat{t}_h$ ($v_h \in \Re$ is the *scalar regression coefficient* (inner relationship) relating $\hat{t}_h$ to the elements in c)

step 5. Generation of $\hat{b}_h \in \Re^{nx1}$, the *loading vector* for A

model: $A = \hat{t}_h b_h^T + E_A$

least-squares solution: $\hat{b}_h = A^T \hat{t}_h / \hat{t}_h^T \hat{t}_h$

step 6. Calculation of the residuals in A and c

A residuals: $E_A = A - \hat{t}_h \hat{b}_h^T$

c residuals: $e_c = c - \hat{v}_h \hat{t}_h$

step 7. Increment h, substitute E_A for A and e_c for c in step 2 and continue for the desired number of loading vectors (or the optimal number of PLS factors, h^o †)

† If all of the factors were retained, $h^o = m$ (for $m < n$) or $h^o = n$ (for $m \geq n$)

PLS1 Calibration Model & Prediction

* Once the optimal number of weight loading vectors, $\underline{\hat{w}}_h$, the regression coefficients (or inner relationships), $\hat{v}_h$, and the loading vectors, $\underline{\hat{p}}_h$, for $h = 1, 2, \ldots, h^o$ (where h^o is the optimal number of factors) are determined, then it's time to "build" the calibration model, $\underline{\hat{b}}_{f\text{-PLS1}}$. There are several

methods that can be used to do this, but we will study only one here.

* This method is a "batch" method and involves the following steps;

Step 1 From the PLS1 calibration phase, the following 3 matrices are formed for the optimal number of factors, i.e., h^o:

$$\hat{W}^T = [\hat{w}_1, \hat{w}_2, \ldots, \hat{w}_{h^o}] \in \mathbb{R}^{n \times h^o} \quad (22)$$

$$\hat{B}^T = [\hat{b}_1, \hat{b}_2, \ldots, \hat{b}_{h^o}] \in \mathbb{R}^{n \times h^o} \quad (23)$$

and

$$\hat{\underline{V}}^T = [\hat{V}_1, \hat{V}_2, \ldots, \hat{V}_{h^o}] \in \mathbb{R}^{1 \times h^o} \quad (24)$$

Step 2. Compute the optimal calibration model:

$$\underset{n\times 1}{\hat{\underline{b}}_{f\text{-}PLS1}} = \underset{n\times h^o}{\hat{\underline{W}}^T} \left(\underset{h^o \times n}{\hat{\underline{B}}} \underset{n \times h^o}{\hat{\underline{W}}^T} \right)^{-1} \underset{h^o \times 1}{\hat{\underline{V}}} \quad (25)$$

Note: $\rho(\hat{\underline{B}}\hat{\underline{W}}^T) = h^o$, where $h^o << m$ or n.

Step 3: Prediction

* Before we get into the business of prediction using the PLS calibration model, $\hat{\underline{b}}_{f\text{-PLS1}}$, we need to discuss the details of the data

$\{\underline{A}, \underline{c}\}$

* It's assumed that the data set can be split into two pieces that are associated with "training" and another part associated with

"testing". Therefore, for the steps previously given for building the calibration model (and those 7 steps previously given in the PLS/1 calibration Algorithm), the data set that's assumed to be available is:

$$\{A_{train}, C_{train}\}$$

and what's left is used in the prediction phase, i.e.,

$\{\underline{A}_{test}, \underline{\mathcal{C}}_{test}\}$

* Therefore, starting with the global dataset:

$\{\underline{A}, \underline{\mathcal{C}}\} \xrightarrow{①} \{\underline{A}_{train}, \underline{\mathcal{C}}_{train}\}$

②

$\{\underline{A}_{test}, \underline{\mathcal{C}}_{test}\}$

* So the prediction phase simply involves doing the following:

$$\hat{\underline{c}}_{test} = \underline{A}_{test}\,\hat{\underline{b}}_{f\text{-}PLS1} \qquad (26)$$

and then $\hat{\underline{c}}_{test}$ can be compared to $\underline{c}_{test}$ (a known vector).

Note: If the training data were mean-centered (where the mean of the dependent reference data is given as $\underline{\bar{c}}_{train}$), then

$$\hat{\underline{c}}_{test} = \underline{A}_{test}\,\hat{\underline{b}}_{f\text{-}PLS1} + \underline{\bar{c}}_{train} \qquad (27)$$

PLS1 Prediction Algorithm

step 1. From the PLS1 calibration phase, form the following matrices (for the optimal or desired number of PLS factors, $h = h^o$)

$$\hat{W}^T = [\hat{w}_1\ \hat{w}_2 \cdots \hat{w}_{h^o}] \quad (\text{where: } \hat{W} \in \Re^{h^o x n})$$

$$\hat{B}^T = [\hat{b}_1\ \hat{b}_2 \cdots \hat{b}_{h^o}] \quad (\text{where: } \hat{B} \in \Re^{h^o x n})$$

$$\hat{v}^T = [\hat{v}_1\ \hat{v}_2 \cdots \hat{v}_{h^o}] \quad (\text{where: } \hat{v} \in \Re^{h^o x 1})$$

step 2. Compute the final regression coefficients, or the optimal calibration model, $\hat{b}_{f\,PLSR}$

$$\hat{b}_{f\,PLSR} = \hat{W}^T(\hat{B}\hat{W}^T)^{-1}\hat{v} \quad (\text{Note: } \text{rank}(\hat{B}\hat{W}^T) = h^o, \text{ where } h^o << m \text{ or } n)$$

step 3. Given a set of measurements, A_{test} (not used to develop the calibration model $\hat{b}_{f\,PLSR}$), estimate the outputs (or dependent variables)

$$\hat{c}_{test} = A_{test}\hat{b}_{f\,PLSR}$$

or if the training data were mean centered (where the mean of the dependent reference data is given as $\bar{c}_{train}$)

$$\hat{c}_{test} = A_{test}\hat{b}_{f\,PLSR} + \bar{c}_{train}$$

* The only item left to cover is how to choose the optimal number of factors, h^o. Before we see how that's carried out, let's define the Standard Error of Prediction (SEP):

$$SEP \triangleq \left(\sum_{i=1}^{M_{test}} (\ell_{i\,test} - \hat{\ell}_{i\,test})^2 / M_{test} \right)^{1/2} \quad (28)$$

where $\underline{\ell}_{test} \in \mathbb{R}^{M_{test} \times 1}$

$\underline{\hat{\ell}}_{test} \in \mathbb{R}^{M_{test} \times 1}$

and ℓ_i's & $\hat{\ell}_i$'s are elements of the respective vectors given above.

* The SEP in Eq. 28 is an rms error figure of merit that can be used to assess performance, which can in turn be used to ultimately determine the optimal number of factors, h.

* So here's how one can go about determining the optimal number of PLS factors, h^o.

Using $\{\underline{A}_{train}, \underline{c}_{train}\}$:

for $h = 1, 2, \ldots, q$ (where q is some number $\leq \rho(\underline{A})$)

build q PLS1 calibration models:

$\{\hat{\underline{b}}_{f\text{-}PLS}^{(1)}$ (Calibration model using 1 factor), $\hat{\underline{b}}_{f\text{-}PLS}^{(2)}, \ldots, \hat{\underline{b}}_{f\text{-}PLS}^{(q)}\} =$

$\{\hat{\underline{b}}_{f\text{-}PLS}(h)$: for $h = 1, 2, \ldots, h^o, \ldots q\}$

however, right now we don't know what this is. (29)

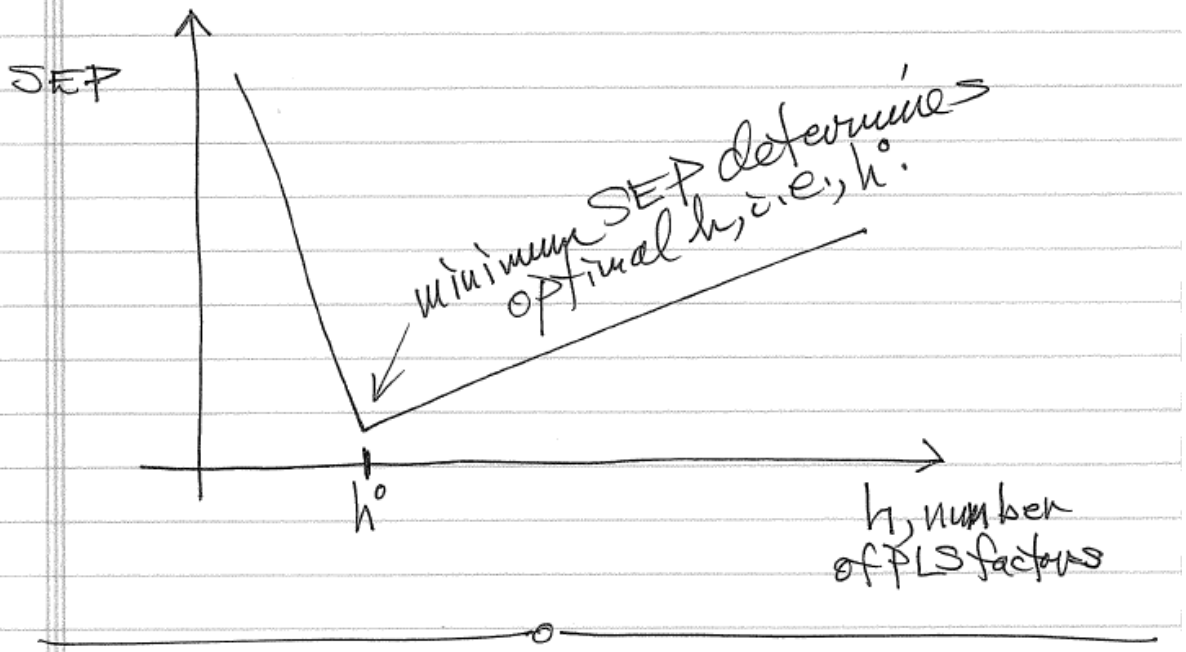
Then using $\{\underline{A}_{test}, \underline{c}_{test}\}$ and each of the calibration models from (29), form

$$\{\hat{\underline{c}}_{test\text{-}PLS}^{(1)}, \hat{\underline{c}}_{test\text{-}PLS}^{(2)}, \ldots, \hat{\underline{c}}_{test\text{-}PLS}^{(q)}\}$$

and then compute a series of SEP values $\Longrightarrow$

$\{SEP(1), SEP(2), \ldots, SEP(q)\}$

Then plot these SEP values as a function of the number of factors $\Longrightarrow$



Problem Assignment

It is desired to develop a Partial Least-Squares (PLS) statistical calibration model using spectrophotometric data (synthetic data) to predict concentrations of glucose. In the MATLAB workspace: "ECE5248_Prob_14_data_set_.mat" you will find the following:

TEST_data_1	**100x100**	**80000 double array**
TRAIN_data_1	**100x100**	**80000 double array**
test_reference_1	**100x1**	**800 double array**
train_reference_1	**100x1**	**800 double array**

Grand total is 20200 elements using 161600 bytes

TEST_data_1	=> $\boldsymbol{A}_{test}$
TRAIN_data_1	=> $\boldsymbol{A}_{train}$
test_reference_1	=> $\boldsymbol{c}_{test}$
train_reference_1	=> $\boldsymbol{c}_{train}$

Tasks (Perform <u>every</u> task):

(1) Write the MATLAB function (m-file) to carry out the Calibration Phase of PLS. This should be a single function that can take in $\{\boldsymbol{A}_{train}, \boldsymbol{c}_{train}\}$[1] and extract the weight loading vectors, $\hat{\boldsymbol{w}}_h$, loading vectors, $\hat{\boldsymbol{b}}_h$, and the regression coefficients, $\hat{v}_h$, for h=1,2, ⋯, h^o.

(2) Write a MATLAB function (m-file) that can build the statistical calibration model using the information found in Part (1).

(3) Develop the PLS statistical calibration model by determining the *optimal number* of PLS factors. How many PLS factors did you need? Plot the components of the PLS calibration model and clearly mark the graph's axes.

(4) Plot the first four weight loading vectors and clearly mark the graph's axes.

(5) Given the data $\{\boldsymbol{A}_{test}, \boldsymbol{c}_{test}\}$ use $\boldsymbol{A}_{test}$ predict the glucose concentrations, i.e., determine $\hat{\boldsymbol{c}}_{test}$.

(6) Plot the glucose concentration predictions, $\hat{\boldsymbol{c}}_{test}$ (using PLS), versus the actual concentrations, $\boldsymbol{c}_{test}$, on one graph but only the discrete points. Also plot on the same graph the "y=x" line (continuous) so the results can be easily compared. Clearly mark the graph. **What is the final SEP value using the PLS model?**

(7) Develop a *classical least-squares (CLS) calibration model* and determine the glucose concentration predictions, $\hat{\boldsymbol{c}}_{test}$ (using CLS).

(8) Plot the glucose concentration predictions, $\hat{\boldsymbol{c}}_{test}$ (using CLS), versus the actual concentrations, $\boldsymbol{c}_{test}$, on one graph but only the discrete points. Also plot on the same graph the "y=x" line (continuous) so the results can be easily compared. Clearly mark the graph. **What is the final SEP value using the CLS model?**

[1] The data matrices are assumed to have NIR (near-infrared spectrophotometric data, synthetic), and the data in the target vectors are concentration of glucose with the units mg/dl).

Florida Institute of Technology
High Tech with a Human Touch™

COLLEGE OF ENGINEERING
We Engineer the Future™

Generate Synthetic NIR Spectra Date

```
%
% Generates Synthetic Near-Infrared (NIR) Data
%

% Spectrum of the component of interest (could be NIR spectrum of
  glucose)
      Spi = .6*gaussd(30,100,15) + .3*gaussd(50,100,70);
% GAUSSD generates a Gaussian distribution
% Spectrum of obscuring component (NIR spectrum of water)
      Spo = .8*gaussd(10,100,20) + .6*gaussd(20,100,80);
      A = zeros(200,100);
      for i = 1:200
          A(i,:) = i*Spi;
      end
% Concentrations (could be glucose concentrations)
      p = ones(100,1);
      C1 = 1000*atan(.0001*A*p);
% Addition of the zero-mean Gaussian noise
      An = A + randn(200,100);
% Addition of the obscuring component
      An1 = zeros(200,100);
      for i =1:200
          An1(i:) = An(i,:) + (1000+30*randn)*Spo;
      end
% Form the Training and Test Data
% Training Spectra (each row is a NIR spectrum)
      TRAIN = An1 (1:2:200,:);
% Training Concentrations (target values)
      TRAINC = C1(1:2:200,:);
% Test Spectra (each row is a NIR spectrum)
      TEST = An1(2:2:200,:);
% Test Concentrations (target values)
      TESTC = C1(2:2:200,:);
clear p i A An An1 C1 Spi Spo
```

PLSR Calibration MATLAB Code

```
function [W,B,V]=pls1cal(A,c,maxrank)

% [W,B,V]=pls1cal(A,c,maxrank)
%
% pls1cal.m extracts the PLS weight loading
% vectors in the columns of W, the PLS
% loading vectors in the columns of B, and
% the PLS regression coefficients (or inner
% relationships) in the column vector V.
%
% A      = is an mxn matrix containing the training
%          data as rows
% c      = is an mx1 vector which contains the target
%          values (or repsonse variables)
% maxrank = number of PLS factors to be retained
%          (i.e., h)
% W      = (n,h) matrix containing the PLS weight
%          loading vectors
% B      = (n,h) matrix containing the PLS loading
%          vectors
% V      = (h,1) vector containing the PLS regression
%          coefficients (or PLS inner relationships)
%

[m,n]=size(A);

W=zeros(n,maxrank);
B=zeros(n,maxrank);
V=zeros(maxrank,1);

for h=1:maxrank
          W(:,h)=(A'*c)/(c'*c);
         W(:,h)=W(:,h)/norm(W(:,h));
       t=A*W(:,h);
     V(h,1)=(t'*c)/(t'*t);
   B(:,h)=(A'*t)/(t'*t);
  A=A-t*B(:,h)';
 c=c-V(h,1)*t;
end
```

PLSR Calibration MATLAB Code

```
function [W,B,V]=pls1cal(A,c,maxrank)
[m,n]=size(A);
W=zeros(n,maxrank);
B=zeros(n,maxrank);
V=zeros(maxrank,1);
for h=1:maxrank
            W(:,h)=(A'*c)/(c'*c);
          W(:,h)=W(:,h)/norm(W(:,h));
        t=A*W(:,h);
      V(h,1)=(t'*c)/(t'*t);
    B(:,h)=(A'*t)/(t'*t);
  A=A-t*B(:,h)';
 c=c-V(h,1)*t;
end
```

PLSR Prediction Method 1 MATLAB Code

```
function plscalmod = pls1prd1(W,B,V,maxrank)

% plscalmod = pls1prd1(W,B,V,maxrank)
%
% pls1prd1.m generates the PLS calibration model.
%
% plscalmod = is an nx1 vector, the PLS calibration
%          model
% maxrank   = number of PLS factors to use
%          (i.e., h)
% W        = (n,h) matrix containing the PLS weight
%          loading vectors
% B        = (n,h) matrix containing the PLS loading
%          vectors
% V        = (h,1) vector containing the PLS regression
%          coefficients (or PLS inner relationships)
%
% NOTE:    To obtain predictions, the calibration model
%         (plscalmod) is used as follows:
%         cpredpls(mx1)=ATEST(mxn)*plscalmod(nx1)
%

plscalmod=W(:,1:maxrank)*inv(B(:,1:maxrank)'*W(:,1:maxrank))*V(1:maxrank,1);
```

PLSR Prediction Method 1 MATLAB Code

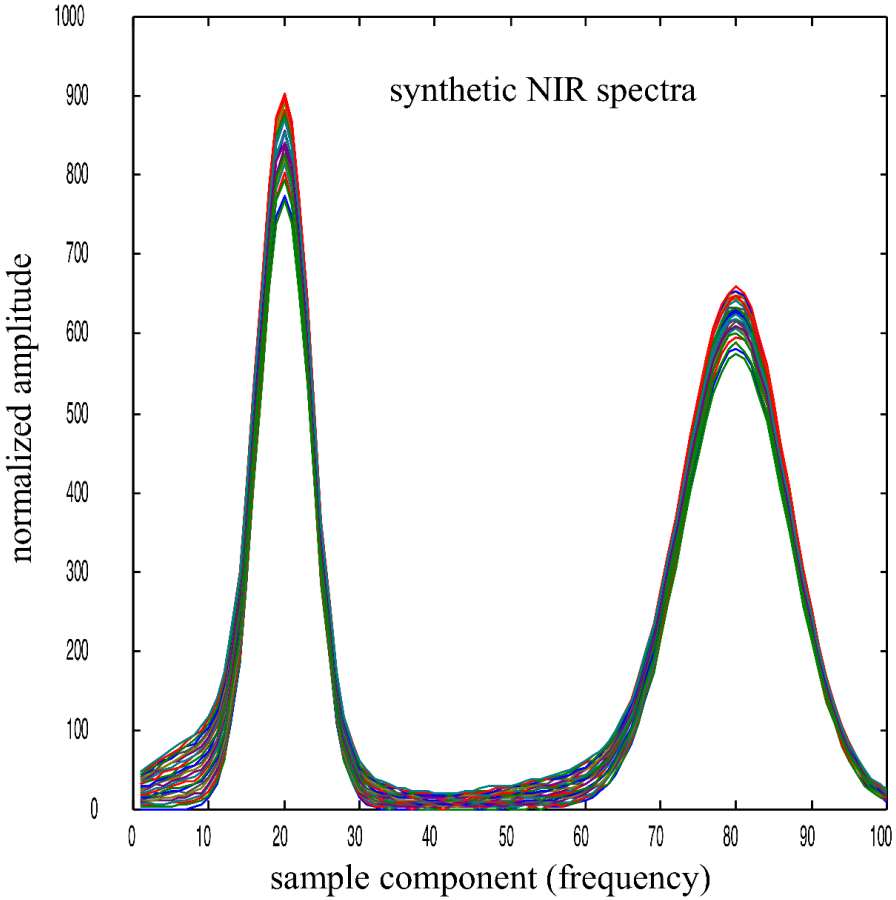
```
function plscalmod = pls1prd1(W,B,V,maxrank)
plscalmod=W(:,1:maxrank)*inv(B(:,1:maxrank)'*W(:,1:maxrank))*V(1:maxrank,1);
```

PLSR Prediction Method 2 (Recursive)

```
function [Cp] = pls1prd2(A,W,B,V);

% pls1prd2 - predict the target vector
%
% [Cp] = pls1prd2(A,W,B,V);
%
% A - absorption given rowwise
% W - vectors of direction  ( W in Matlab)
% B - spectral factors      ( P in Matlab)
% V - concentration scores  ( B in Matlab)
%
% Cp - predicted concentrations

[M,N] = size(A);
Cp = zeros(M,1);
[N,H] = size(W);
a = A;

for i = 1:M
   for h = 1:H
     Cp(i)  = Cp(i)  + a(i,:) * W(:,h) * V(h,:);
     a(i,:) = a(i,:) - a(i,:) * W(:,h) * B(:,h)';
   end
end
```

Example

Simulated NIR Spectra



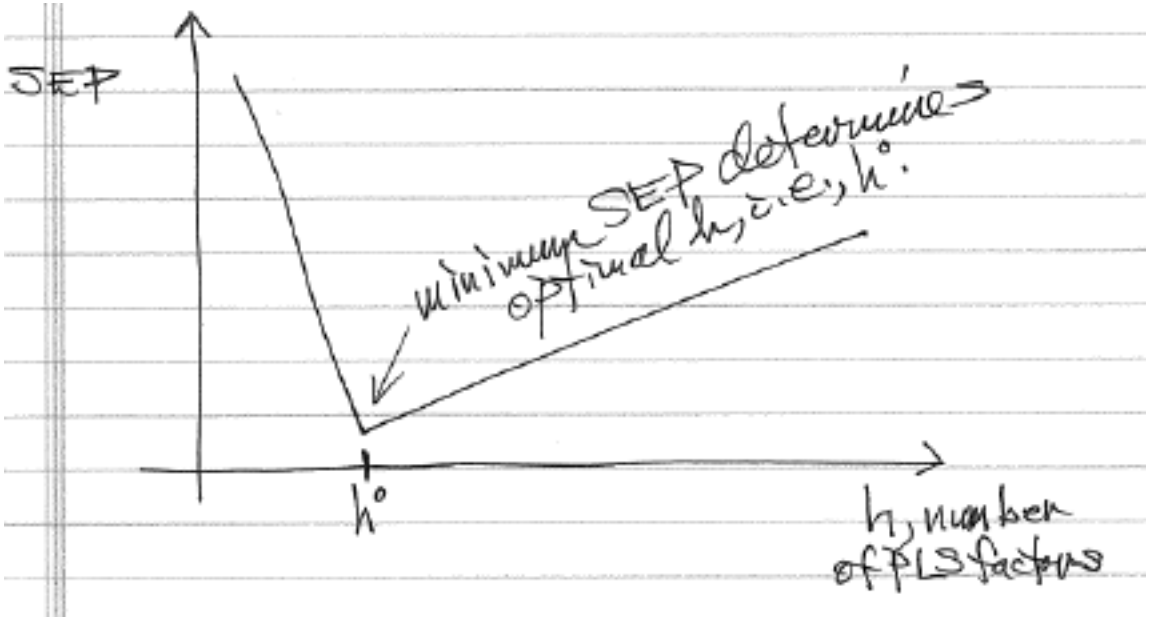
Training data for all three methods. This represents 25% of the total amount of training data (i.e., 25 spectra).

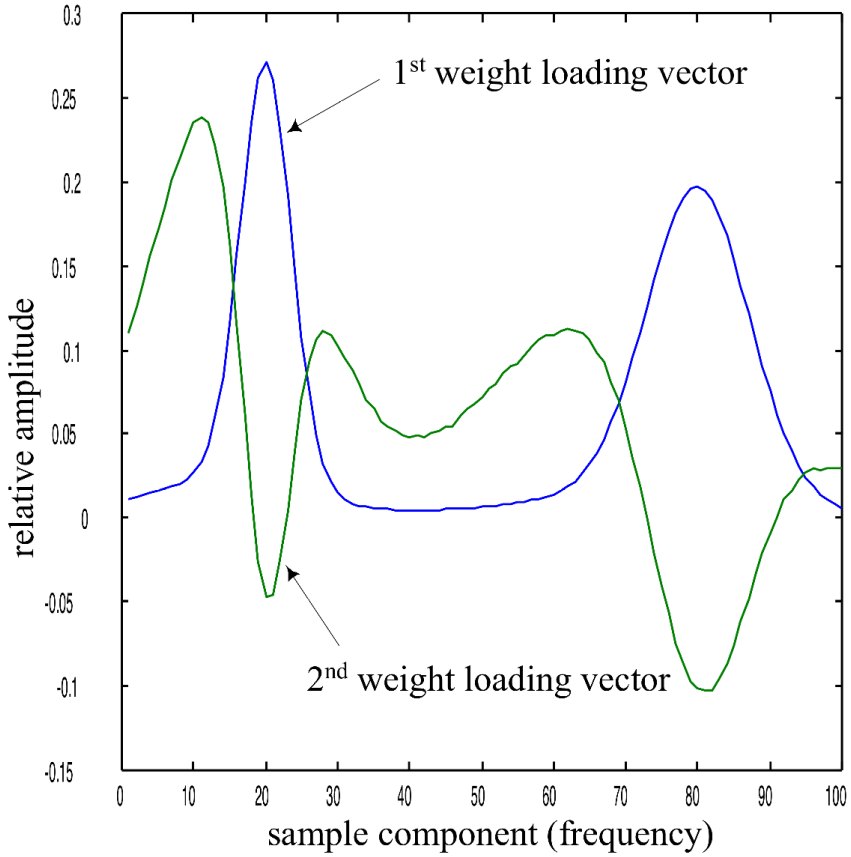
Florida Institute of Technology
High Tech with a Human Touch™

COLLEGE OF ENGINEERING
We Engineer the Future™

In this example, h° (h optimal) = 2!

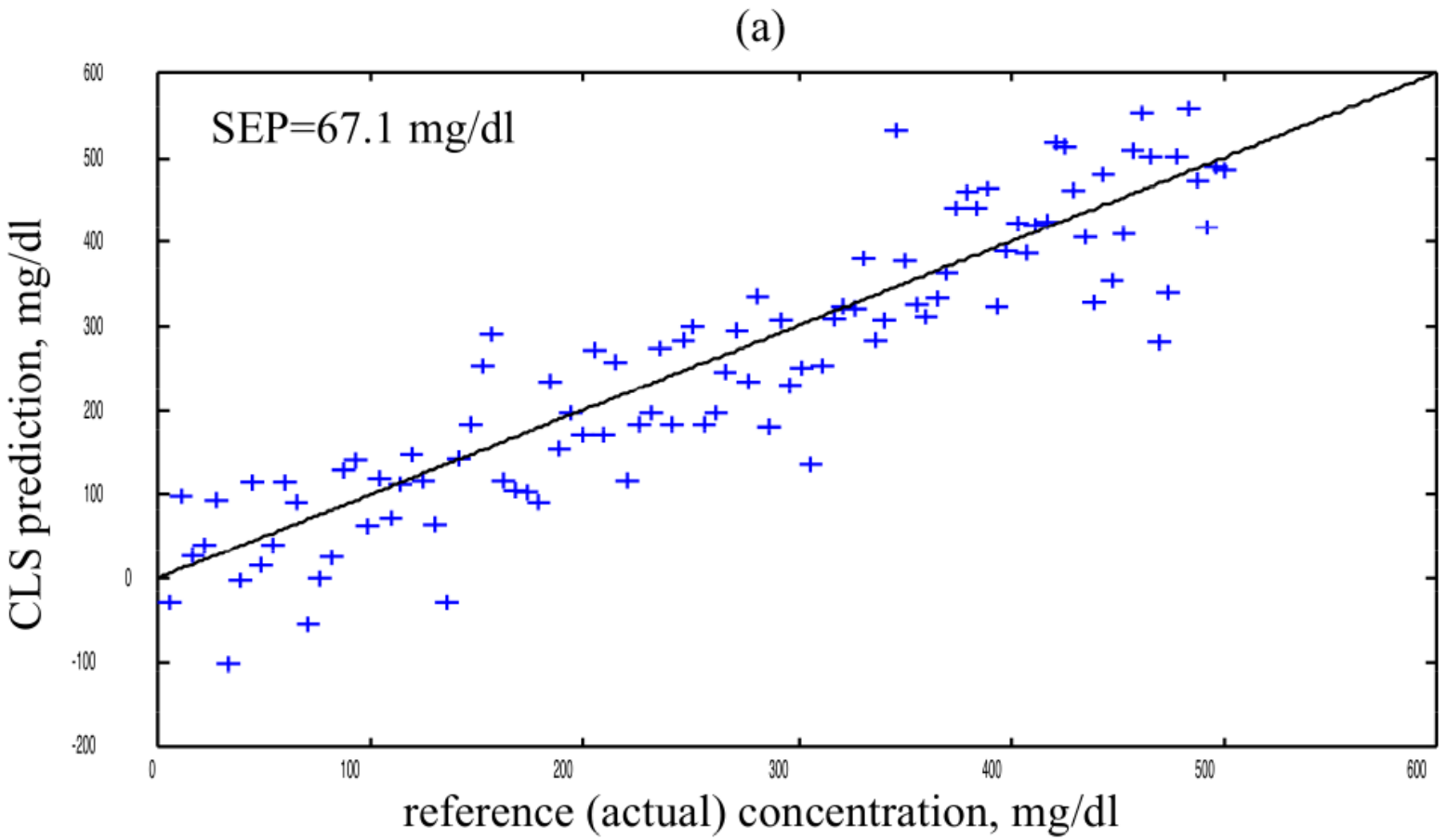
(Out of 100 factors available!)





First two weight loading vectors extracted

Simulation Results

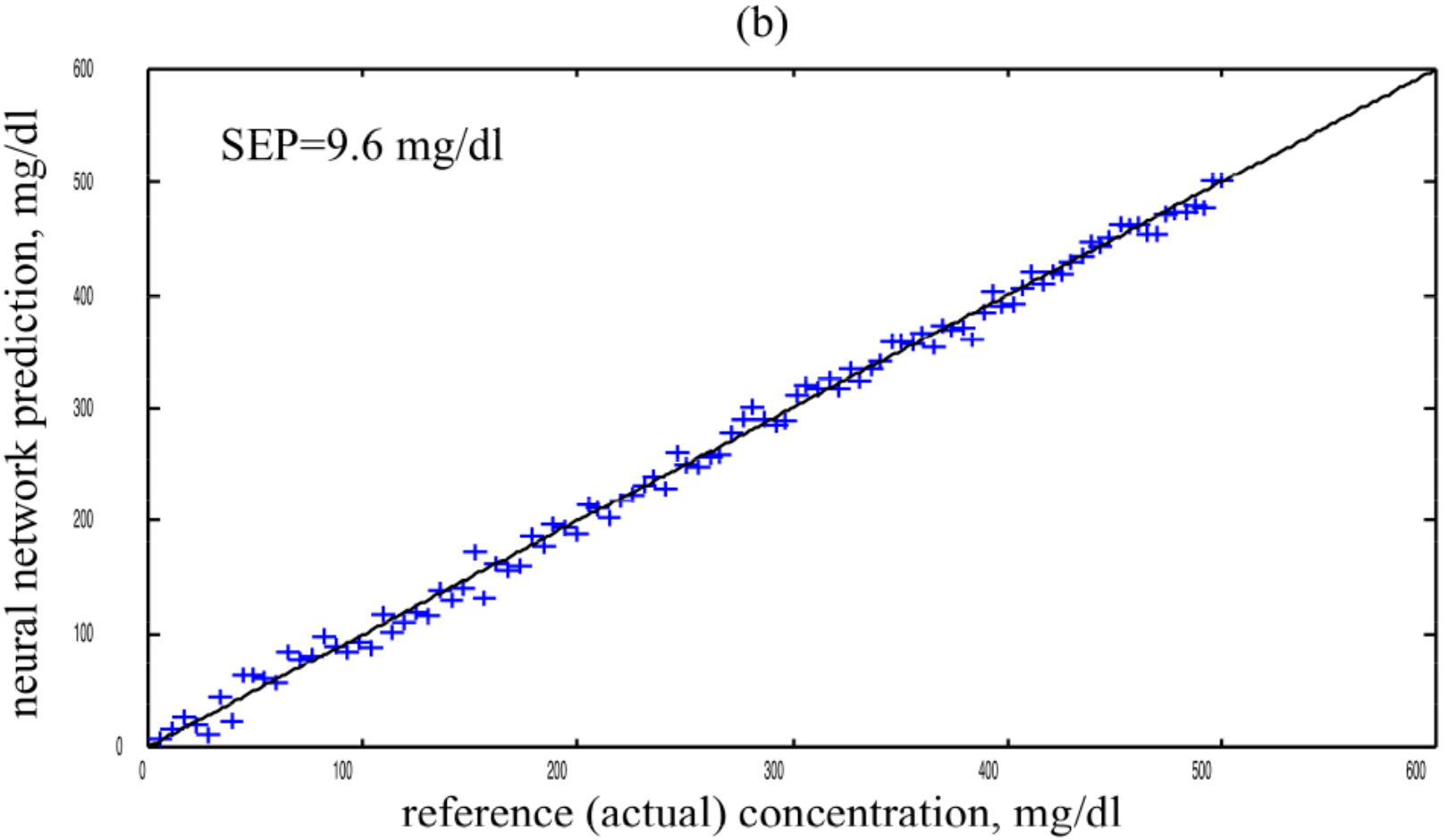


Florida Institute of Technology
High Tech with a Human Touch™

IPL Information Processing Laboratory at Florida Tech

COLLEGE OF ENGINEERING
We Engineer the Future™

Simulation Results (cont')

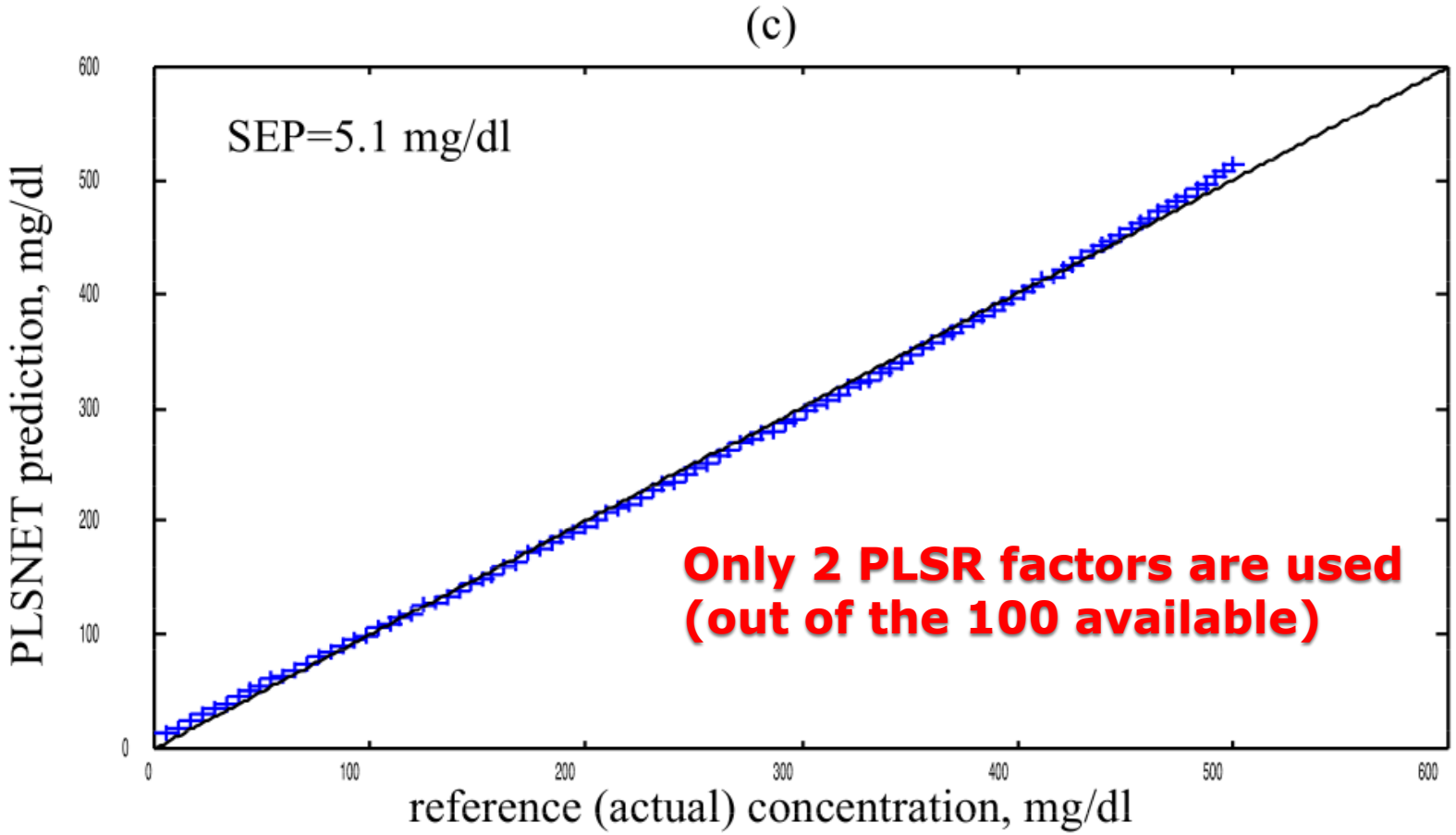


Florida Institute of Technology
High Tech with a Human Touch™

IPL Information Processing Laboratory at Florida Tech

COLLEGE OF ENGINEERING
We Engineer the Future™

Simulation Results (cont')



Only 2 PLSR factors are used (out of the 100 available)

Florida Institute of Technology
High Tech with a Human Touch™

IPL Information Processing Laboratory at Florida Tech

COLLEGE OF ENGINEERING
We Engineer the Future™

Thank You!

Fredric M. Ham, Ph.D.
fmh@fit.edu
fredric.ham@sit.kmutt.ac.th

Florida Institute of Technology
High Tech with a Human Touch™

IPL Information Processing Laboratory at Florida Tech

COLLEGE OF ENGINEERING
We Engineer the Future™