Research Article

Journal of CHEMOMETRICS

Received: 6 August 2013, Revised: 11 December 2013, Accepted: 11 December 2013, Published online in Wiley Online Library

(wileyonlinelibrary.com) DOI: 10.1002/cem.2589

The geometry of PLS1 explained properly: 10 key notes on mathematical properties of and some alternative algorithmic approaches to PLS1 modelling

Ulf G. Indahl*

The insight from, and conclusions of this paper motivate efficient and numerically robust 'new' variants of algorithms for solving the single response partial least squares regression (PLS1) problem. Prototype MATLAB code for these variants are included in the Appendix. The analysis of and conclusions regarding PLS1 modelling are based on a rich and nontrivial application of numerous key concepts from elementary linear algebra. The investigation starts with a simple analysis of the nonlinear iterative partial least squares (NIPALS) PLS1 algorithm variant computing orthonormal scores and weights.

A rigorous interpretation of the squared P-loadings as the variable-wise explained sum of squares is presented. We show that the orthonormal row-subspace basis of W-weights can be found from a recurrence equation. Consequently, the NIPALS deflation steps of the centered predictor matrix can be replaced by a corresponding sequence of Gram–Schmidt steps that compute the orthonormal column-subspace basis of T-scores from the associated non-orthogonal scores.

The transitions between the non-orthogonal and orthonormal scores and weights (illustrated by an easy-to-grasp commutative diagram), respectively, are both given by QR factorizations of the non-orthogonal matrices. The properties of singular value decomposition combined with the mappings between the alternative representations of the PLS1 'truncated' X data (including P^tW) are taken to justify an invariance principle to distinguish between the PLS1 truncation alternatives. The fundamental orthogonal truncation of PLS1 is illustrated by a Lanczos bidiagonalization type of algorithm where the predictor matrix deflation is required to be different from the standard NIPALS deflation. A mathematical argument concluding the PLS1 inconsistency debate (published in 2009 in this journal) is also presented. **Copyright © 2014 John Wiley & Sons, Ltd.**

Keywords: PLS1 algorithms; bidiagonalization; orthogonal and non-orthogonal weights; scores and projections; change of coordinates and bases; truncation; QR factorization; singular value decomposition; reorthogonalization

1. INTRODUCTION

In the chemometrics community, single response partial least squares regression (PLS1) has been a popular tool for solving regression problems with multicollinear data for more than 30 years; see [1–4]. Over the years, several papers focusing interpretations and various theoretical aspects of PLS1 have been published (see [5–14] for a selection). However, as late as 2007, Pell *et al.* [15] published a paper claiming inconsistencies in the residuals of conventional PLS1 regression due to the truncation (projection) implied by the nonlinear iterative partial least squares (NIPALS) PLS1 algorithm. The claimed inconsistency resulted in a debate (published in *Journal of Chemometrics* 2009; 23, pages 67–77, see [16–19]) between some of the most influential contributors to the field, without reaching a unanimous conclusion.

The purpose of the present paper is to clarify unnecessary misunderstandings by pinpointing some important but often overlooked mathematical properties of PLS1 with particular focus on

(a) the extraction of *orthonormal* row-subspace and column-subspace bases for computing various orthogonal projections of the observed data;
(b) its close relationship to one of the fundamental problems of numerical linear algebra: the computation of singular values by bidiagonalization of the matrix subject to investigation;
(c) interpretations related to the applications of the PLS1 method by considering the various coordinate representations of the data with respect to the orthonormal row-subspace and column-subspace bases.

* *Correspondence to: Ulf G. Indahl, Department of Mathematical Sciences and Technology, Norwegian University of Life Sciences, N-1432 Ås, Norway. E-mail: ulf.indahl@nmbu.no*

Ulf G. Indahl
Department of Mathematical Sciences and Technology, Norwegian University of Life Sciences, N-1432 Ås, Norway

 Copyright © 2014 John Wiley & Sons, Ltd.

and last but not least

(d) a mathematical conclusion of the PLS1 inconsistency debate by considering the key orthogonal and non-orthogonal projections involved in PLS1 model building.

The main source of inspiration for the work presented subsequently was found in the two publications [12,18] by Ergon.

2. THE NIPALS PLS1 ALGORITHM WITH ORTHONORMAL SCORES

The widely applied NIPALS PLS1 algorithm with orthogonal (but not normalized scores) [1–3] is usually considered as the benchmark for comparison of other algorithmic approaches to PLS1 modelling. According to [20], the NIPALS PLS1 is relatively slow but numerically stable in most practical situations. The main reason for its lack of speed is the extensive data matrix deflation that requires computation of the outer products between each extracted component and the corresponding loadings. In the typical applications of PLS, where the number of predictors is large compared to the number of observations, deflation of the predictor matrix is a computationally expensive way to extract the desired sets of orthogonal PLS1 scores (and weights).

To make the mathematics and interpretations as transparent as possible, we will focus on the version of the NIPALS PLS1 algorithm (algorithm 1) that computes *orthonormal scores* (orthogonal unit vectors). As usual, we will assume that $\mathbf{X}_0$ is the mean-centered version of the $n \times m$ predictor matrix $\mathbf{X}$ and that $\mathbf{y}$ is an n-dimensional response vector where the entries are associated with the corresponding rows of $\mathbf{X}$.

According to the conventional PLS1 terminology and properties, the matrices of *scores* ($\mathbf{T}$) and *weights* ($\mathbf{W}$) are both orthogonal, that is, $\mathbf{T^tT} = \mathbf{W^tW} = \mathbf{I}_A$ (the $A \times A$ identity matrix). Hence, the associated vectors represent orthonormal bases for the PLS1 column and row subspaces, respectively, with $\mathbf{TT}^t$ and $\mathbf{WW}^t$ as the associated orthogonal projections. The column vectors of $\mathbf{P}$ and the entries of $\mathbf{q}$ are referred to as the corresponding $\mathbf{X}$- and $\mathbf{y}$-*loadings* of the associated PLS1 model.

By application of elementary linear algebra to the various parts of the NIPALS algorithm described above, we are going to establish a sequence of fundamental PLS1 modelling properties listed as notes in the following sections.

3. PLS1 WITHOUT THE DEFLATION STEP OF NIPALS

Note 1: explained variance by P-loadings

The $\mathbf{X}$-loadings identity $\mathbf{P} = \mathbf{X}_0^t\mathbf{T}$ follows from

$$\mathbf{X}_0^t\mathbf{t}_a = \mathbf{X}_{a-1}^t\mathbf{t}_a + (\mathbf{X}_0 - \mathbf{X}_{a-1})^t\,\mathbf{t}_a = \mathbf{X}_{a-1}^t\mathbf{t}_a = \mathbf{p}_a \tag{1}$$

because the column space of the matrix $(\mathbf{X}_0 - \mathbf{X}_{a-1})$ is spanned by the orthonormal subset $\{\mathbf{t}_1, \ldots, \mathbf{t}_{a-1}\}$ of scores that are all orthogonal to $\mathbf{t}_a$.

Each row of $\mathbf{P}$ represents the coordinates of the corresponding $\mathbf{X}_0$-column w.r.t. the orthonormal column-subspace basis associated with $\mathbf{T}$. Hence, $\mathbf{P}$ can also be referred to as the *projected variables coordinate matrix*.

The projection of the i-th column $\mathbf{x}_i$ of $\mathbf{X}_0$ onto the direction of the score basis vector $\mathbf{t}_a$ equals $\mathbf{x}_{i_a} = \left(\mathbf{t}_a\mathbf{t}_a^t\right)\mathbf{x}_i = \left(\mathbf{x}_i^t\mathbf{t}_a\right)\mathbf{t}_a$, and the coordinate value $p_{a(i)} = \mathbf{x}_i^t\mathbf{t}_a$ relates directly to the amount of $\mathbf{x}_i$-variance $\left[= \frac{1}{n}\mathbf{x}_{i_a}^t\mathbf{x}_{i_a}\right]$ *accounted for* by the a-th PLS1 component (i.e. $\mathbf{t}_a$), that is,

$$\frac{1}{n}\mathbf{x}_{i_a}^t\mathbf{x}_{i_a} = \frac{1}{n}\left(\mathbf{x}_i^t\mathbf{t}_a\right)^2 = \frac{1}{n}\left(p_{a(i)}\right)^2 \tag{2}$$

Hence, with orthonormal scores, the explained sum of squares corresponding to the i-th variable ($\mathbf{x}_i$, $1 \le i \le m$) accounted for

for $a = 1 : A$, (A - the number of components to be extracted)

1. $\mathbf{v}_a = \mathbf{X}_{a-1}^t\mathbf{y}$
2. $\mathbf{w}_a = \mathbf{v}_a/\|\mathbf{v}_a\|$
3. $\boldsymbol{\tau}_a = \mathbf{X}_{a-1}\mathbf{w}_a$
4. $\mathbf{t}_a = \boldsymbol{\tau}_a/\|\boldsymbol{\tau}_a\|$
5. $\mathbf{p}_a = \mathbf{X}_{a-1}^t\mathbf{t}_a$
6. $\mathbf{X}_a = \mathbf{X}_{a-1} - \mathbf{t}_a\mathbf{p}_a^t$ (the deflation step)
7. $q_a = \mathbf{t}_a^t\mathbf{y}$

end

Organize vectors and numbers into the matrices:
$\mathbf{T} = [\mathbf{t}_1\ \mathbf{t}_2 \ldots \mathbf{t}_A]$ (the orthonormal scores),
$\mathbf{W} = [\mathbf{w}_1\ \mathbf{w}_2 \ldots \mathbf{w}_A]$ (the orthonormal weights),
$\mathbf{P} = [\mathbf{p}_1\ \mathbf{p}_2 \ldots \mathbf{p}_A]$ (the matrix of $\mathbf{X}$-loadings),
$\mathbf{q}^t = [q_1\ q_2 \ldots q_A]$ (the vector of $\mathbf{y}$-loadings).

Finally compute the regression coefficients w.r.t. the original predictors of $\mathbf{X}$:
$\mathbf{b} = \mathbf{W}(\mathbf{P}^t\mathbf{W})^{-1}\mathbf{q}$.

Algorithm 1: The NIPALS algorithm for computing an A-components PLS1 model with orthonormal scores and loadings.

 Copyright © 2014 John Wiley & Sons, Ltd.

by the PLS1 model is found by squaring and adding all (A) entries in the i-th row of $\mathbf{P}$.

Note 2: W-weights recurrence equation gives NIPALS alternative without deflation

By step 1 in algorithm 1, the vectors $\mathbf{v}_a^t = \mathbf{y}^t\mathbf{X}_{a-1}$ ($a = 1,\ldots,A$). Left multiplication by $\mathbf{y}^t$ in step 6 of algorithm 1 gives the following identities and implications:

$$\mathbf{y}^t\mathbf{X}_a = \mathbf{y}^t\mathbf{X}_{a-1} - \mathbf{y}^t\mathbf{t}_a\mathbf{p}_a^t \Rightarrow \mathbf{v}_{a+1} = \mathbf{v}_a - \frac{(\mathbf{w}_a^t\mathbf{v}_a)}{\|\boldsymbol{\tau}_a\|}\mathbf{p}_a = \mathbf{v}_a - \frac{\|\mathbf{v}_a\|}{\|\boldsymbol{\tau}_a\|}\mathbf{p}_a$$

$$\Downarrow$$

$$\mathbf{v}_{a+1} = \|\mathbf{v}_a\|\left(\mathbf{w}_a - \frac{1}{\|\boldsymbol{\tau}_a\|}\mathbf{p}_a\right) \tag{3}$$

By normalization of $\mathbf{v}_{a+1}$, we obtain $\mathbf{w}_{a+1} = \frac{1}{\|\mathbf{v}_{a+1}\|}\mathbf{v}_{a+1}$. Hence, (3) followed by normalization defines a recurrence equation for computing the orthonormal PLS1 weights. The associated nested sequence of vector equations can be entirely solved from the starting vectors $\mathbf{w}_1, \mathbf{t}_1, \mathbf{p}_1$ ($a = 1$) and norms $\|\mathbf{v}_1\|, \|\boldsymbol{\tau}_1\|$ that are all available before execution of the first deflation step in the NIPALS algorithm. Note that Equation (3) with the succeeding normalization is equivalent to the content of lemma 2 in [21].

For $a > 1$, a trivial projection argument shows that the score vector $\mathbf{t}_a = \frac{1}{\|\boldsymbol{\tau}_a\|}\boldsymbol{\tau}_a$ is obtained from a *Gram–Schmidt* (GS) orthogonalization step of the vector $\boldsymbol{\tau}_a^\star = \mathbf{X}_0\mathbf{w}_a$ with respect to the preceding orthonormal scores $\{\mathbf{t}_1,\ldots,\mathbf{t}_{a-1}\}$ that forms a basis for the column space of $(\mathbf{X}_0 - \mathbf{X}_{a-1})$. Because the corresponding loading vector can be found directly by $\mathbf{p}_a = \mathbf{X}_0^t\mathbf{t}_a$ according to note 1, $\mathbf{w}_{a+1}$ can (by induction) be found without executing the deflation of $\mathbf{X}_0$.

From notes 1 and 2, we can now establish a mathematically equivalent PLS1 algorithm where deflation of $\mathbf{X_0}$ is replaced with a GS step to obtain the orthonormal scores:

Note 3: the W-weights recurrence equation shows that $\mathbf{P}^t\mathbf{W}$ is bidiagonal

By a re-arrangement of Equation (3), the $\mathbf{X}$-loadings

$$\mathbf{p}_a = \|\boldsymbol{\tau}_a\|\left(\mathbf{w}_a - \frac{\|\mathbf{v}_{a+1}\|}{\|\mathbf{v}_a\|}\mathbf{w}_{a+1}\right),\ a = 1,\ldots,A < m \tag{4}$$

appear as linear combinations of exactly two successive weights. These vector equations of (4) are compactly expressed by the matrix product

$$\mathbf{P} = \mathbf{W}_+\mathbf{B}_2 \tag{5}$$

where the coordinate matrix

$$\mathbf{B}_2 = \begin{bmatrix} \|\boldsymbol{\tau}_1\| & & & & & 0 \\ -\|\boldsymbol{\tau}_1\|\frac{\|\mathbf{v}_2\|}{\|\mathbf{v}_1\|} & \|\boldsymbol{\tau}_2\| & & & & \\ & -\|\boldsymbol{\tau}_2\|\frac{\|\mathbf{v}_3\|}{\|\mathbf{v}_2\|} & \ddots & & & \\ & & \ddots & \|\boldsymbol{\tau}_{A-1}\| & & \\ & & & -\|\boldsymbol{\tau}_{A-1}\|\frac{\|\mathbf{v}_A\|}{\|\mathbf{v}_{A-1}\|} & \|\boldsymbol{\tau}_A\| \\ 0 & & & & -\|\boldsymbol{\tau}_A\|\frac{\|\mathbf{v}_{A+1}\|}{\|\mathbf{v}_A\|} \end{bmatrix} \tag{6}$$

of $\mathbf{P}$ is ($A + 1 \times A$) lower bidiagonal, and the corresponding orthonormal basis vectors are arranged in the augmented $p\times$ $(A+1)$ weight matrix $\mathbf{W}_+ = [\mathbf{W}\ \mathbf{w}_{A+1}]$. Consequently,

$$\mathbf{P}^t\mathbf{W} = \mathbf{B}_2^t\left(\mathbf{W}_+^t\mathbf{W}\right)$$

$$= \begin{bmatrix} \|\boldsymbol{\tau}_1\| & -\|\boldsymbol{\tau}_1\|\frac{\|\mathbf{v}_2\|}{\|\mathbf{v}_1\|} & & & 0 \\ & \|\boldsymbol{\tau}_2\| & -\|\boldsymbol{\tau}_2\|\frac{\|\mathbf{v}_3\|}{\|\mathbf{v}_2\|} & & \\ & & \ddots & \ddots & \\ & & & \|\boldsymbol{\tau}_{A-1}\| & -\|\boldsymbol{\tau}_{A-1}\|\frac{\|\mathbf{v}_A\|}{\|\mathbf{v}_{A-1}\|} \\ 0 & & & & \|\boldsymbol{\tau}_A\| \end{bmatrix} \tag{7}$$

is upper bidiagonal and of size $A \times A$. Note that $\mathbf{P}^t\mathbf{W}$ equals the transposed coordinates of $\mathbf{P}$ truncated to the A basis (column) vectors of $\mathbf{W}$. According to (7), $\mathbf{P}^t\mathbf{W}$ can therefore be found directly from the normalizing constants of the orthogonal scores ($\mathbf{T}$) and weights ($\mathbf{W}$) only.

Note 4: some computational issues in algorithms 1 and 2

The main difference between algorithms 1 and 2 is that the computationally ‘expensive’ deflation (step 6) in algorithm 1 (involving the computation and subtraction of A large vector outer products) is accounted for by a considerably less expensive GS orthogonalization (step 4) in algorithm 2 (involving the indicated computationally ‘moderate’ matrix–vector products only). Consequently, an implementation of algorithm 2 executes considerably faster than the NIPALS PLS1. The GS orthogonalization step replacing the deflation step of algorithm 1 indicates that the numerical robustness in proper implementations of the two algorithms should be similar.

The competitiveness of algorithm 2 with the fastest ‘stable’ PLS1 algorithms discussed in [20] will not be investigated in detail in the present paper, but some comments on speed and precision is given in Section 5. A prototype MATLAB implementation of algorithm 2 is given in Appendix A.1.

4. ADDITIONAL NOTES ON MATHEMATICS AND ALGORITHMIC FACETS OF PLS1

The vectors $\boldsymbol{\tau}_a^\star$ of algorithm 2 coincide with the non-orthogonal scores found by Martens' alternative PLS1 algorithm (see frame 3.5 in [3]) that was shown by Helland [7] to be mathematically equivalent to the NIPALS algorithm: both algorithms share the same set of orthogonal weights.

Note 5: alternative coordinates and interpretations by elementary linear algebra

By inspection of algorithm 2, the orthonormal scores $\mathbf{T} = [\mathbf{t}_1\ \mathbf{t}_2\ \ldots\ \mathbf{t}_A]$ are successively derived from the non-orthogonal scores $\mathbf{T}^\star = \mathbf{X}_0\mathbf{W} = \left[\boldsymbol{\tau}_1^\star\ \boldsymbol{\tau}_2^\star\ \ldots\ \boldsymbol{\tau}_A^\star\right]$ by the GS-orthonormalization steps (4 and 5) establishing an orthonormal basis for the subspace spanned by the non-orthogonal scores.

It should not be ignored that the rows of the non-orthogonal score matrix $\mathbf{T}^\star$ represent coordinates of the observations with respect to the orthonormal row-subspace basis $\mathbf{W} = [\mathbf{w}_1\ \mathbf{w}_2\ \ldots\ \mathbf{w}_A]$ of PLS1 weights (the coordinate interpretation of the $\mathbf{T}^\star$ entries is valid because we compute inner products

Copyright © 2014 John Wiley & Sons, Ltd.

Initialization:

- $\mathbf{v}_1 = \mathbf{X}_0^t\mathbf{y}, \mathbf{w}_1 = \mathbf{v}_1/\|\mathbf{v}_1\|$
- $\mathbf{w}_1 = \mathbf{v}_1/\|\mathbf{v}_1\|$
- $\boldsymbol{\tau}_1 = \mathbf{X}_0\mathbf{w}_1$
- $\mathbf{t}_1 = \boldsymbol{\tau}_1/\|\boldsymbol{\tau}_1\|$
- $\mathbf{p}_1 = \mathbf{X}_0^t\mathbf{t}_1$

for $a = 2 : A$,

1. $\mathbf{v}_a = \|\mathbf{v}_{a-1}\|(\mathbf{w}_{a-1} - \frac{1}{\|\boldsymbol{\tau}_{a-1}\|}\mathbf{p}_{a-1})$ (according to equation (3))
2. $\mathbf{w}_a = \mathbf{v}_a/\|\mathbf{v}_a\|$
3. $\boldsymbol{\tau}_a^\star = \mathbf{X}_0\mathbf{w}_a$
4. $\boldsymbol{\tau}_a = \boldsymbol{\tau}_a^\star - \mathbf{T}_{a-1}(\mathbf{T}_{a-1}^t\boldsymbol{\tau}_a^\star)$ (GS-step, $\mathbf{T}_{a-1} = [\mathbf{t}_1 \ldots \mathbf{t}_{a-1}]$)
5. $\mathbf{t}_a = \boldsymbol{\tau}_a/\|\boldsymbol{\tau}_a\|$
6. $\mathbf{p}_a = \mathbf{X}_0^t\mathbf{t}_a$
7. $q_a = \mathbf{t}_a^t\mathbf{y}$

end
Organize vectors and numbers into the matrices
$\mathbf{T} = [\mathbf{t}_1\ \mathbf{t}_2 \ldots \mathbf{t}_A], \mathbf{W} = [\mathbf{w}_1\ \mathbf{w}_2 \ldots \mathbf{w}_A],\ \mathbf{P} = [\mathbf{p}_1\ \mathbf{p}_2 \ldots \mathbf{p}_A], \mathbf{q}^t = [q_1\ q_2 \ldots q_A]$
and compute the final **X**-regression coeffs: $\mathbf{b} = \mathbf{W}(\mathbf{P}^t\mathbf{W})^{-1}\mathbf{q}$

Algorithm 2: An alternative PLS1 algorithm without $\mathbf{X_0}$-deflation.

between the $\mathbf{X}_0$ observations and the **W** basis vectors). From the latter interpretation (focusing on the rows of observations), it makes sense to refer to $\mathbf{T}^\star$ as the *Martens coordinate matrix* w.r.t the basis **W** .

The associated orthogonal projection of the centered data matrix $\mathbf{X}_0$ onto to the row subspace spanned by the orthonormal basis **W** results in the truncation $\mathbf{X}_{0tr}$ given by

$$\mathbf{X}_{0tr} = \mathbf{T}^\star\mathbf{W}^t = \mathbf{X}_0\mathbf{W}\mathbf{W}^t \tag{8}$$

with respect to the original coordinates. Note that $\mathbf{T}^\star = \mathbf{X}_0\mathbf{W} = \mathbf{X}_0\mathbf{W}\mathbf{W}^t\mathbf{W} = \mathbf{X}_{0tr}\mathbf{W}$. Hence, the GS steps for deriving the orthonormal scores from the Martens coordinate matrix imply the existence of an invertible upper triangular ($A \times A$) matrix $\mathbf{D}_2$ that when paired with **T** represents a QR factorization of $\mathbf{T}^\star$ (compare with section 3 in [18]),

$$\mathbf{T}^\star [= \mathbf{X}_0\mathbf{W}] = \mathbf{T}\mathbf{D}_2 \tag{9}$$

Left multiplication by $\mathbf{T}^t$ in Equation (9) solves for $\mathbf{D}_2$ as follows:

$$\mathbf{T}^t\mathbf{T}^\star = \mathbf{T}^t\mathbf{X}_0\mathbf{W} = \mathbf{T}^t\mathbf{T}\mathbf{D}_2 = \mathbf{D}_2 \tag{10}$$

Thus, with respect to the orthonormal column-subspace basis **T**, an alternative set of (untruncated) coordinates for the non-orthogonal scores (the columns of $\mathbf{T}^\star$) is given by the corresponding columns of

$$\mathbf{D}_2 = \left(\mathbf{T}^t\mathbf{X}_0\right)\mathbf{W} = \mathbf{P}^t\mathbf{W} \tag{11}$$

which is a bidiagonal matrix according to note 3. Consequently, in PLS1 modelling, the matrix product $\mathbf{P}^t\mathbf{W}$ has two different coordinate interpretations.

Right multiplication of $\mathbf{T}^\star$ by $\mathbf{D}_2^{-1}$ in the preceding QR factorization (9) gives

$$\mathbf{T} = \mathbf{T}^\star\mathbf{D}_2^{-1}\left[= \mathbf{T}^\star\left(\mathbf{T}^t\mathbf{T}^\star\right)^{-1}\right] = \mathbf{X}_0\mathbf{W}\mathbf{D}_2^{-1} = \mathbf{X}_0\mathbf{W}^\star \tag{12}$$

where

$$\mathbf{W}^\star = \mathbf{W}\mathbf{D}_2^{-1} = \mathbf{W}\left(\mathbf{T}^t\mathbf{T}^\star\right)^{-1} = \mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1} \tag{13}$$

is the matrix of corresponding non-orthogonal weights. $\mathbf{W}^\star$ coincides with the (non-orthogonal) weights matrix directly computed by the mathematically equivalent SIMPLS algorithm [22] (in the case of a single response vector **y**). Finally, a left multiplication of Equation (13) by $\mathbf{W}^t$ shows that

$$\mathbf{D}_2^{-1} = \mathbf{W}^t\mathbf{W}^\star \tag{14}$$

The basic algebraic properties of PLS1 (just pointed out in notes 4 and 5) are illustrated by the commutative diagram shown in Figure 1.

Note 6: QR factorization of the non-orthogonal $\mathbf{W}^\star$-weights

The inverse of an upper triangular matrix is always upper triangular. Because **W** is orthonormal ($\mathbf{W}^t\mathbf{W} = \mathbf{I}_A$), the first identity $\mathbf{W}^\star = \mathbf{W}\mathbf{D}_2^{-1}$ of Equation (13) represents a QR factorization of the non-orthogonal weight matrix $\mathbf{W}^\star$ because $\mathbf{D}_2^{-1}$ is upper triangular.

Note 7: everything can be calculated if you know W or $\mathbf{W}^\star$

Any PLS1 algorithm (mathematically equivalent to the NIPALS algorithm) computing either the orthonormal weights **W** or corresponding non-orthogonal weights $\mathbf{W}^\star$ can easily be modi-

 Copyright © 2014 John Wiley & Sons, Ltd.

fied to output the complete set of relevant matrices/vectors and regression coefficients ($\mathbf{T}, \mathbf{T}^\star, \mathbf{W}, \mathbf{W}^\star, \mathbf{P}, \mathbf{D}_2, \mathbf{q}, \mathbf{b}$) by using the appropriate QR factorization (Equation 9 or 13) followed by direct computation of the required matrix products.

Note 8: orthogonal projections, a skew projection in disguise and singular value decomposition consistency

The orthonormal bases of $\mathbf{W}$ and $\mathbf{T}$ span subspaces of the row and column spaces of $\mathbf{X}_0$, respectively. Correspondingly, a complete orthogonal truncation of $\mathbf{X}_0$ includes the projections of $\mathbf{X}_0$ onto both subspaces, that is,

$$\left(\mathbf{T}\mathbf{T}^t\right)\mathbf{X}_0\left(\mathbf{W}\mathbf{W}^t\right) = \mathbf{T}\left(\mathbf{P}^t\mathbf{W}\right)\mathbf{W}^t = \left[\mathbf{X}_0\mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\right]\left(\mathbf{P}^t\mathbf{W}\right)\mathbf{W}^t$$
$$= \mathbf{X}_0\mathbf{W}\mathbf{W}^t = \mathbf{X}_{0tr} \qquad (15)$$

Hence, the suggested bi-orthogonal projection of (15) is equivalent to applying the (row subspace) orthogonal projection (8), obtained by right multiplication of $\mathbf{X}_0$ with $\mathbf{H}_{tr} = \mathbf{W}\mathbf{W}^t$ (compare with section 2 in [12]), only. The truncation $\mathbf{X}_{0tr}$ is indicated in the center of the commutative diagram shown in Figure 1. $\mathbf{H}_{tr}$ is trivially an orthogonal projection (idempotent and symmetrical).

According to equation (15) and note 3, $\mathbf{T}^t\mathbf{X}_{0tr}\mathbf{W} = \mathbf{P}^t\mathbf{W}$ is bidiagonal with $\mathbf{T}\left(\mathbf{P}^t\mathbf{W}\right)\mathbf{W}^t$ as the corresponding bidiagonalization of $\mathbf{X}_{0tr}$. According to the fundamental relationships used for computing singular values in the famous paper by Golub and Kahan [23], the bidiagonal matrix $\mathbf{P}^t\mathbf{W}$, the Martens coordinates $\mathbf{T}^\star$ (alias the non-orthogonal scores) and the truncated data matrix $\mathbf{X}_{0tr}$ all share *the same set of nonzero singular values*. Consequently, the singular values of the bidiagonal matrix $\mathbf{D}_2 = \mathbf{P}^t\mathbf{W}$ describes (up to a scaling factor) the variances along the principal axes of both the Martens coordinates $\mathbf{T}^\star$ and the truncated data $\mathbf{X}_{0tr}$, and confirm *consistency* between these alternative simplifications of the original data.

It is a non-intuitive fact that truncation of $\mathbf{X}_0$ by the left orthogonal projection $\mathbf{T}\mathbf{T}^t$ is equivalent to an alternative double projection of $\mathbf{X}_0$ that collapses to application of the right

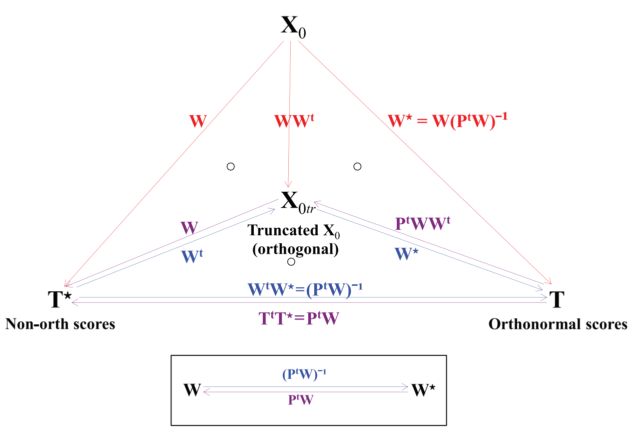

Figure 1. Commutative diagram showing the elementary linear algebra of PLS1 modelling. Arrows indicate multiplication from the right by the corresponding matrices. (All directed paths in the diagram with the same start and endpoints lead to the same result by composition.)

projection $\mathbf{H}_{str} = \mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\mathbf{P}^t \left[= \mathbf{W}^\star\mathbf{P}^t\right]$, that is,

$$\mathbf{T}\mathbf{T}^t\mathbf{X}_0 = \mathbf{T}\mathbf{P}^t = \mathbf{T}\mathbf{T}^t\left(\mathbf{X}_0\mathbf{W}^\star\right)\mathbf{P}^t = \mathbf{T}\mathbf{T}^t\left[\mathbf{X}_0\mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\right]$$
$$\mathbf{P}^t = \mathbf{X}_0\mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\mathbf{P}^t = \mathbf{X}_0\mathbf{W}^\star\mathbf{P}^t \qquad (16)$$

The fact that $\mathbf{H}_{str}$ is a projection (idempotent) is easily checked, that is,

$$\mathbf{H}_{str}^2 = \left[\mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\mathbf{P}^t\right]\left[\mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\mathbf{P}^t\right]$$
$$= \mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\left[\left(\mathbf{P}^t\mathbf{W}\right)\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\right]\mathbf{P}^t = \mathbf{H}_{str}$$

However, in general, $\mathbf{H}_{str}$ is *not* symmetrical $\left(\mathbf{H}_{str} \neq \mathbf{H}_{str}^t\right)$ and therefore not orthogonal. Consequently, the left orthogonal projection $\mathbf{X}_{0str} = \mathbf{T}\mathbf{P}^t$ of $\mathbf{X}_0$, most commonly focused in PLS1 modelling according to Pell *et al.* [15] and Wold *et al.* [16], turns out to be a non-orthogonal (skew) right projection 'in disguise'. Our intuition with respect to 'truncations' as obtained by the NIPALS algorithm must therefore be handled with some care. It should be noted that in Equation (29d) of the noteworthy paper by Phatak and de Jong [24], the projection matrix $(\mathbf{I} - \mathbf{H}_{str})$ appear and is recognized to represent a skew (oblique) projection for computing the PLS residual data matrix.

The singular values of the skew truncation $\mathbf{X}_{0str}$ will in general differ from the singular values of $\mathbf{P}^t\mathbf{W}$, $\mathbf{T}^\star$ and the orthogonal truncation $\mathbf{X}_{0tr}$. This is due to the fact that the first score vector $\boldsymbol{\tau}_{A+1}^\star$ (in the non-exhaustive case) not included in the PLS1 model is non-orthogonal to the subspace spanned by its orthogonal scores $\mathbf{T}$.

The difference between the two truncations is exactly accounted for by the fact that $\mathbf{X}_{0str}$ also includes the projection of $\boldsymbol{\tau}_{A+1}^\star$ onto the $\mathbf{T}$-subspace. Consequently, the nonzero singular values of $\mathbf{P}^t \left[= \left(\mathbf{P}^t\mathbf{W}_+\right)\mathbf{W}_+^t\right]$, $\mathbf{X}_{0str} = \mathbf{T}\mathbf{P}^t$ and $\mathbf{T}\mathbf{P}^t\mathbf{W}_+ = \left(\mathbf{T}\mathbf{T}^t\right)\mathbf{T}_+^\star$, where the last expression denotes the projection of the augmented matrix $\mathbf{T}_+^\star = \left[\mathbf{T}^\star\ \boldsymbol{\tau}_{A+1}^\star\right] = \mathbf{X}_0\mathbf{W}_+$ of non-orthogonal scores, are all identical and confirm consistency between these data representations.

Note 9: a recursion formula for upper bidiagonal inverses

According to a simple recursion formula for the inverse $\mathbf{U}^{-1} = \left[\tilde{\mathbf{u}}_1\ \tilde{\mathbf{u}}_2\ \ldots\ \tilde{\mathbf{u}}_A\right]$ of upper bidiagonal matrices of the form

$$\mathbf{U} = \begin{bmatrix} d_1 & -b_1 & & & 0 \\ & d_2 & -b_2 & & \\ & & \ddots & \ddots & \\ & & & d_{A-1} & -b_{A-1} \\ 0 & & & & d_A \end{bmatrix} \qquad (17)$$

$\tilde{\mathbf{u}}_1 = \left(\frac{1}{d_1}\ 0\ \ldots\ 0\right)^t$ and for $1 < a \leq A$,

$$\tilde{\mathbf{u}}_a = \frac{b_{a-1}}{d_a}\tilde{\mathbf{u}}_{a-1} + \left(0\ ..\ \frac{1}{d_a}\ ..\ 0\right)^t$$

that is, the a-th (and only nonzero) entry of the last vector is equal to $\frac{1}{d_a}$; see [25]. Hence, by taking $d_a = \|\boldsymbol{\tau}_a\|$ and $b_a = \|\boldsymbol{\tau}_{a-1}\|\frac{\|\mathbf{v}_a\|}{\|\mathbf{v}_{a-1}\|}$, the desired columns in the upper trian-

Copyright © 2014 John Wiley & Sons, Ltd.

gular inverse $\mathbf{D}_2^{-1} = (\mathbf{P}^t\mathbf{W})^{-1}$ can be calculated successively while computing the PLS1 components. An explicit application of this inversion strategy is demonstrated in the MATLAB code (see Appendix A.2) implementing a slightly modified version of the bidiagonalization algorithm for PLS1 modelling introduced by Manne [5].

Note 10: how to avoid GS in algorithm 2

In exact arithmetic, the Gram–Schmidt orthonormalization steps (4 and 5) of algorithm 2 can be avoided by a successive computations of the non-orthogonal $\mathbf{W}^\star$-weights from the corresponding orthogonal $\mathbf{W}$-weights and the recursion formula in note 9 for expanding $(\mathbf{P^tW})^{-1}$.

MATLAB code for an algorithm including this modification is given in Appendix A.3. The relationships to the PLS1 versions of Appendix A.1 and A.2 are evident.

It should be noted that when using the code in Appendix A.3, one runs into the same kind of trouble as was reported for the Bidiag2 in [20].

5. DISCUSSION

5.1. Notes 1–3

Extraction of orthonormal bases for the subspaces of interest simplifies later computations and interpretations, and is common practice in applied linear algebra. The advantages of normalization seems, however, to have been partly overlooked by the earliest pioneers of PLS modelling. Although mathematically equivalent, the original NIPALS PLS1 algorithm, with extraction of orthogonal but not normalized scores, disturbs the interpretation of the squared $\mathbf{P}$-loading entries as variable-wise explained sum of squares that was shown in Equation (2) of note 1.

The missing normalization is perhaps also the reason why the simple relationship between the weights and loadings as described in Equation (3) of note 2 was overlooked when formulating the original NIPALS PLS1 algorithm. Another possible explanation is that the NIPALS PLS was initially designed to also handle multivariate responses ($\mathbf{Y}$) where the possibilities of avoiding the $\mathbf{X}$-deflation were not so obvious. This problem has, however, later been solved by de Jongs SIMPLS algorithm [22]. A minor modification of the direct scores PLS1 algorithm of Andersson [20] (according to its close relationship to the SIMPLS algorithm) will also solve the problem for multivariate responses without $\mathbf{X}$-deflation.

The bidiagonal form of $\mathbf{P}^t\mathbf{W}$ is of course 'old news' in the context of PLS1 modelling. The purpose of including note 3 is to demonstrate that the bidiagonal form has a very simple and transparent interpretation as coordinates. From Equation (4), it is clear that each loading weight is a linear combination of not more than two distinct weights and hence has at most two nonzero coordinates with respect to both of the bases $\mathbf{W}_+$ and (the projection onto the subspace spanned by) $\mathbf{W}$.

5.2. Notes 4 and 5

The main challenges of numerical problem solving are related to numerical precision and computational efficiency. A brief comparison of the two algorithms indicate that their major difference is the deflation step (6 in algorithm 1) and the GS step (4 in algorithm 2). Application of MATLABs *profiling* tool to the corresponding MATLAB prototype code confirms this to be the case. Computation of the extra matrix–vector products for finding the weights (step 1) also adds significantly to the computational cost of algorithm 1. In algorithm 2, the weights are found from the loading computations (step 6) and the weighed vector differences (step 1), indicating that the computational cost of finding the weights is reduced with 50% alone. Orthogonality of the score vectors from both algorithms appear to be numerically satisfactory. However, for the weights (when the number of extracted components is large), this is not necessarily the case. For both algorithms, this deficiency may cause precision problems in the computation of the desired regression coefficients. A possible solution to the problem is to include an additional *reorthogonalization* step as indicated in Appendix 1 (line 17) of the MATLAB prototype code. Later, we will briefly explain the reasons for and the possible solutions to this problem.

For de Jongs SIMPLS algorithm [22] with a single response vector ($\mathbf{y}$), computation of the non-orthogonal $\mathbf{W}^\star$-weights (in [22], the notation $\mathbf{R}$ is used for the non-orthogonal weights) is driven by the orthogonality requirement for the scores:

$$\mathbf{I}_A = \mathbf{T}^t\mathbf{T} = \mathbf{T}^t\mathbf{X}_0\mathbf{W}^\star = \mathbf{P}^t\mathbf{W}^\star \tag{18}$$

Hence, requiring orthonormality between the scores is equivalent to requiring orthogonality between the $\mathbf{P}$-loadings and $\mathbf{W}^\star$-weights not corresponding to the same component. De Jong found this requirement to be satisfied exactly by the residual weight vector obtained after projecting $\mathbf{w} = \mathbf{X}_0^t\mathbf{y}$ onto the subspace spanned by the $\mathbf{P}$-vectors found 'so far'. Appropriate scaling of the desired residual weight vector was chosen to obtain normalization of the corresponding score vector. Thus, the SIMPLS algorithm has its focus on computation of the (non-orthogonal) weights required to obtain an orthonormal basis for the desired PLS1 column subspace that exactly corresponds to the column subspace basis found by algorithm 2.

5.3. Notes 6 and 7

Although $\mathbf{P}^t\mathbf{W}$ is bidiagonal only in the PLS1 case, the commutative diagram of Figure 1 is valid for any right projection of $\mathbf{X}_0$, where the row coordinates $\mathbf{T}^\star = \mathbf{X}_0\mathbf{W}$ with respect to an orthonormal-subspace basis $\mathbf{W}$ of the m-dimensional Euclidean space. By the QR factorization of $\mathbf{T}^\star = \mathbf{TR}$, we obtain the orthonormal column-subspace basis $\mathbf{T}$ and the upper triangular matrix $\mathbf{R} = \mathbf{P}^t\mathbf{W} = \mathbf{T}^t\mathbf{T}^\star$ describing the coordinate relationships of both $\mathbf{P}$ with respect to $\mathbf{W}$ and $\mathbf{T}^\star$ with respect to $\mathbf{T}$. In particular, this describes the situation for the multiresponse case (PLS2) as well as the various modifications of PLS1 based on extra requirements in the computation of the orthonormal weights.

For any number A of components, all algorithms mathematically equivalent to the NIPALS PLS1 must necessarily compute a set of basis vectors for the subspace spanned by the columns of $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_A]$ (or $\mathbf{W}^\star$). If $\mathbf{W}$ is not found directly, the associated orthonormal basis can always be uniquely (up to the sign of each basis vector) obtained by a normalized GS post-processing. Thereafter, the entire collection of scores, weights, loadings and regression coefficients can easily be calculated. Consequently, interpretations of the resulting model are not restricted by the particular choice of algorithm. The main concerns when choosing between the mathematically equivalent PLS1 algorithms should therefore be directed towards numerical precision and computational efficiency.

 Copyright © 2014 John Wiley & Sons, Ltd.

5.4. Note 8

As pointed out earlier, the orthonormal weights $\mathbf{W}$ together with the centered data $\mathbf{X}_0$ are sufficient to describe the entire PLS1 model. The processing of any observation $\mathbf{x}_r$ (a row vector either present in $\mathbf{X}_0$ or new) according to the PLS1 model can therefore be based on its orthogonal projection $\mathbf{x}_{r_{tr}} = \mathbf{x}_r\mathbf{H}_{tr} = \mathbf{x}_r\mathbf{W}\mathbf{W}^t$ onto the modelled row subspace and the corresponding residual $\mathbf{x}_{r_{res}} = \mathbf{x}_r - \mathbf{x}_{r_{tr}}$. The right multiplication of the skew truncation $\mathbf{T}\mathbf{P}^t$ by $\mathbf{W}\mathbf{W}^t$ in Equation (15) is exactly the post-processing solution, suggested by Ergon [18], to the alleged *inconsistency problem* of the PLS1 model space lined out by Pell *et al.* [15] (and later discussed in *Journal of Chemometrics* (2009; **23**, pages 67–77), see [16–19]). The truncation alternative (corresponding to a right multiplication by the skew projection $\mathbf{H}_{str} = \mathbf{W}^{\star}\mathbf{P}^t$) advocated by Wold *et al.* [16] is illustrated in the extended commutative diagram of Figure 2.

Consistent use of the skew truncation requires the non-orthogonal projection $\mathbf{x}_{r_{str}} = \mathbf{x}_r\mathbf{H}_{str}$ to be considered together with the residual $\mathbf{x}_{r_{sres}} = \mathbf{x}_r - \mathbf{x}_{r_{str}}$. Note that the vector of regression coefficients $\mathbf{b} = \mathbf{W}(\mathbf{P}^t\mathbf{W})^{-1}\mathbf{q}$ is an eigenvector corresponding to the eigenvalue $\lambda = 1$ for both $\mathbf{H}_{tr}$ and $\mathbf{H}_{str}$:

$$\begin{aligned}\mathbf{H}_{tr}\mathbf{b} &= \left[\mathbf{W}\mathbf{W}^t\right]\mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\\ \mathbf{q} &= \mathbf{W}\left(\mathbf{W}^t\mathbf{W}\right)\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\\ \mathbf{q} &= \mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\mathbf{q} = \mathbf{b}\end{aligned} \tag{19}$$

and

$$\begin{aligned}\mathbf{H}_{str}\mathbf{b} &= \left[\mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\mathbf{P}^t\right]\mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\\ \mathbf{q} &= \mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\left(\mathbf{P}^t\mathbf{W}\right)\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\\ \mathbf{q} &= \mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\mathbf{q} = \mathbf{b}\end{aligned} \tag{20}$$

Consequently,

$$\hat{y}_r = \mathbf{x}_r\mathbf{b} = \mathbf{x}_{r_{tr}}\mathbf{b} = \mathbf{x}_{r_{str}}\mathbf{b} \tag{21}$$

demonstrates that both projections of $\mathbf{x}_r$ are consistent with application of the regression coefficients $\mathbf{b}$.

According to [15], the minimum norm regression coefficients for the NIPALS truncation $\mathbf{T}\mathbf{P}^t$ of $\mathbf{X}_0$ are

$$\boldsymbol{\beta} = \left(\mathbf{T}\mathbf{P}^t\right)^{+}\mathbf{y} = \mathbf{P}\left(\mathbf{P}^t\mathbf{P}\right)^{-1}\mathbf{q} \tag{22}$$

and does not share the properties reflected by Equation (21), that is,

$$\mathbf{x}_{r_{tr}}\boldsymbol{\beta} \neq \hat{y}_r = \mathbf{x}_{r_{str}}\boldsymbol{\beta} \neq \mathbf{x}_r\boldsymbol{\beta} \tag{23}$$

However, from the identity $\mathbf{T}\mathbf{P}^t = \mathbf{X}_0\mathbf{H}_{str}$, it follows that the corresponding $\mathbf{X}_0$-regression coefficients $\mathbf{b}$ are obtained by applying the skew projection $\mathbf{H}_{str}$ to $\boldsymbol{\beta}$:

$$\mathbf{H}_{str}\boldsymbol{\beta} = \left[\mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\mathbf{P}^t\right]\mathbf{P}\left(\mathbf{P}^t\mathbf{P}\right)^{-1}\mathbf{q} = \mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\mathbf{q} = \mathbf{b} \tag{24}$$

By the orthogonal projection corresponding to $\mathbf{G} = \mathbf{P}\left(\mathbf{P}^t\mathbf{P}\right)^{-1}\mathbf{P}^t$ of $\mathbf{b}$, we obtain $\boldsymbol{\beta}$:

$$\mathbf{G}\mathbf{b} = \left[\mathbf{P}\left(\mathbf{P}^t\mathbf{P}\right)^{-1}\mathbf{P}^t\right]\mathbf{W}\left(\mathbf{P}^t\mathbf{W}\right)^{-1}\mathbf{q} = \boldsymbol{\beta} \tag{25}$$

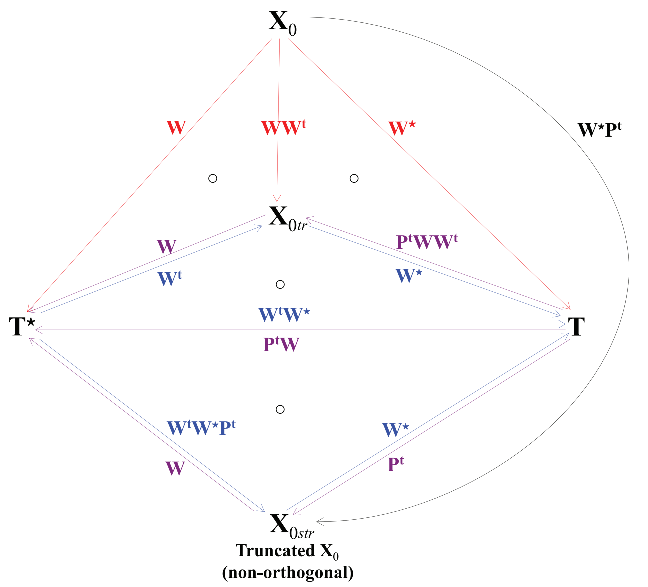

Figure 2. The extended commutative diagram showing the elementary linear algebra of PLS1 modelling with both types of truncation. Arrows indicate multiplication from the right.

Consequently, the regression vector part of the inconsistency debate can be ended by concluding that we can *consistently* navigate between the alternative vectors of regression coefficients by using the projection matrices $\mathbf{H}_{str}$ and $\mathbf{G}$ according to Equations (24) and (25).

In traditional PLS1 modelling, preference for the skew truncation $\mathbf{T}\mathbf{P}^t$ of $\mathbf{X}_0$ seems to be mainly justified by the deflation step of the NIPALS PLS1 algorithm. It is, however, important to keep in mind that the particular choice of deflation strategy is not at all theoretically critical. In particular, application of the right orthogonal projections (equivalent to double projections) in the corresponding deflations

$$\mathbf{X}_a = \mathbf{X}_{a-1} - \mathbf{X}_{a-1}\mathbf{w}_a\mathbf{w}_a^t = \mathbf{X}_{a-1} - \boldsymbol{\tau}_a\mathbf{w}_a^t \tag{26}$$

inside the Bidiag2 algorithm (demonstrated in the MATLAB code of appendix A.4) is one way of stabilizing this algorithm. By noting that (i) this type of deflation is identical to the deflation step in Martens alternative PLS1 algorithm [3] and (ii) the NIPALS type of deflation will not work correctly inside these algorithms, it is clear that sacrificing orthogonality (between the row-subspace residuals and corresponding part of the PLS model) is an unnecessary price to pay. Actions should therefore be taken to review our PLS modelling tools accordingly (in particular the latent variable model view of PLS advocated in [16]). Putting the pieces together after computing the singular value decomposition (SVD) of $\mathbf{P}^t\mathbf{W}$ (i.e. with the orthonormal $\mathbf{T}$ and $\mathbf{W}$) is all that is needed to get it right.

5.5. Notes 9 and 10

The *Lanczos bidiagonalization* algorithm suggested by Golub and Kahan [23] was originally designed for computing the SVD of a matrix. This algorithm is much referred to as *Bidiag2* according to Paige and Saunders [26] and was introduced as a PLS1 algorithm to the chemometrics community by Manne [5] under this acronym. Eldén [13] has published a related paper on the mod-

Copyright © 2014 John Wiley & Sons, Ltd.

elling properties of PLS1 regression based on the equivalence between NIPALS PLS1 and Lanczos bidiagonalization.

It should be noted that by switching signs in the off-diagonal elements of the bidiagonal matrix generated by Bidiag2, we obtain consistency with the **W**-signs of the NIPALS algorithm.

Unfortunately, direct implementations of Lanczos algorithms are known to be computationally unstable in floating point arithmetic with potentially rapid loss of orthogonality for the computed vectors (see Golub and Kahan [23], Simon [27] and Björck [28]). A desired stabilization can be obtained by introducing a so-called full reorthogonalization or a (computationally more sophisticated and efficient) partial reorthogonalization strategy; see Simon [27] and Larsen [29]. However, for the computations of a limited number of PLS components only (typical in the majority of PLS1 applications), the computational savings of a partial reorthogonalization strategy compared to the full reorthogonalization is rarely critical (in comparison to the corresponding savings by partial reorthogonalization for a full SVD).

For the sake of completeness, an efficient and numerically stabilized MATLAB prototype version of the Bidiag2 (to be compared to the unstable version and other algorithms in Andersson [20]) is given in Appendix A.5.

5.6. Final remarks

Reorthogonalization in PLS algorithms focusing on stabilization of the scores (only) has been briefly considered by Faber and Ferré [30]. To avoid problems in the subsequent computations of regression coefficients, reorthogonalization of the weights seems to be just as important.

One should note that the most delicate issues of floating point arithmetic and orthogonalization are not necessarily robustly dealt with by the MATLAB statements commented 'Included for numerical stability' in the prototype scrips presented in the Appendix.

As pointed out in several talks by Björck [31], who is a distinguished numerical analysis researcher, commercial implementations of PLS software should take more seriously the development of numerically precise algorithms (either by robust reorthogonalization strategies for the Lanczos bidiagonalization algorithm or by alternatives based on Householder transformations in the case of PLS1 regression).

In the traditional NIPALS PLS1 applications [4], the orthogonal (but not normalized) scores corresponds to multiplying the **T**-columns of algorithm 2 with the corresponding diagonal elements of $\mathbf{P}^t\mathbf{W}$. One should not confuse these scores with any kind of orthogonal projection of the original $\mathbf{X}_0$-data. de Jong [22] proved that the NIPALS PLS1 scores represent an orthogonal basis (and hence a reference system) for the A-dimensional subspace (of the space spanned by the $\mathbf{X}_0$-columns) maximizing the $\mathbf{X}_0 - \mathbf{y}$ covariances with orthogonality constraints. The corresponding theoretically important SIMPLS algorithm was designed to solve the constrained covariance maximization problem. The SIMPLS orthonormal score vectors are of course mathematically equivalent to those found by the GS step of algorithm 2 and directly in the MATLAB prototype of Appendix A.3.

REFERENCES

1. Wold S, Martens H, Wold H. The multivariate calibration method in chemistry solved by the PLS method. In *Proceedings of the Conference of Matrix Pencils*, Ruhe A, Kågström B (eds), Lecture Notes in Mathematics. Springer Verlag: Heidelberg, 1983; 286–293.
2. Wold S, Ruhe A, Wold H, Dunn WJ, III. The collinearity problem in linear regression. The partial least squares (PLS) approach to generalized inverses. *SIAM J. Scientific Comput* 1984; **5**: 735–743.
3. Martens H, Naes T. Multivariate Calibration (2nd edn). Wiley: Chichester, UK, 1989.
4. Wold S, Sjöström M, Eriksson L. PLS-regression: a basic tool of chemometrics. *Chemom. Intell. Lab. Syst* 2001; **58**: 109–130.
5. Manne R. Analysis of two partial-least-squares algorithms for multivariate calibration. *Chemom. Intell. Lab. Syst.* 1987; **2**: 187–197.
6. Höskuldsson A. PLS regression methods. *J. Chemom* 1988; **2**: 211–228.
7. Helland IS. On the structure of partial least squares regression. *Commun. Statist.(Simul. Comput.)* 1988; **17**: 581–607.
8. Frank IE, Friedman JH. A statistical view of some chemometrics regression tools. *Technometrics* 1993; **35**(2): 109–148.
9. Garthwaite PH. An interpretation of partial least squares. *J. Am. Stat. Assoc.* 1994; **89**(425): 122–127.
10. ter Braak CJF, de Jong S. The objective function of partial least squares regression. *J. Chemom.* 1998; **12**: 41–54.
11. Helland IS. Some theoretical aspects of partial least squares regression. *Chemom. Intell. Lab. Syst.* 2001; **58**: 97–107.
12. Ergon R. PLS score-loading correspondence a bi-orthogonal factorization. *J. Chemom.* 2002; **16**: 368–373.
13. Eldén L. Partial least-squares vs. Lanczos bidiagonalization-I:analysis of a projection method for multiple regression. *Comput. Stat. Data Anal.* 2004; **46**: 11–31.
14. Boulesteix AL, Strimmer K. Partial least squares: a versatile tool for the analysis of high-dimensional genomic data. *Brief. Bioinform.* 2006; **8**(1): 32–44.
15. Pell RJ, Ramos LS, Manne R. The model space in partial least squares regression. *J. Chemom.* 2007; **21**: 165–172.
16. Wold S, Høy M, Martens H, Trygg J, Westad F, MacGregor J, Wise BM. The PLS model space revisited. *J. Chemom.* 2009; **23**: 67–68.
17. Bro R, Eldén L. PLS works. *J. Chemom.* 2009; **23**: 69–71.
18. Ergon R. Re-interpretation of NIPALS results solves PLSR inconsistency problem. *J. Chemom.* 2009; **23**: 72–75.
19. Manne R, Pell RJ, Ramos LS. The PLS model space: the inconsistency persists. *J. Chemom.* 2009; **23**: 76–77.
20. Andersson M. A comparison of nine PLS1 algorithms. *J. Chemom.* 2009; **23**: 518–529.
21. Ergon R. Finding Y-relevant part of X by use of PCR and PLSR model reduction methods. *J. Chemom.* 2007; **21**: 537–546.
22. de Jong S. SIMPLS: an alternative approach to partial least squares regression. *Chemom. Intell. Lab. Syst.* 1993; **18**: 251–263.
23. Golub GH, Kahan W. Calculating the singular values and pseudoinverse of a matrix. *SIAM J. Numer. Anal.* 1965; **2**: 205–224.
24. Phatak A, de Jong S. The geometry of partial least squares. *J. Chemom.* 1997; **11**: 311–338.
25. Higham NJ. Efficient algorithms for computing the condition number of a tridiagonal matrix. *SIAM J. Sci. Stat. Comput.* 1986; **7**(1): 150–165.
26. Paige C, Saunders M. LSQR: an algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Softw.* 1982; **8**: 43–71.
27. Simon HD. The Lanczos algorithm with partial reorthogonalization. *Math. Comp.* 1984; **42**(165): 115–142.
28. Björck Å. Numerical Methods for Least Squares Problems. SIAM: Philadelphia, 1996.
29. Larsen RM. Lanczos bidiagonalization with partial reorthogonalization. *Technical Report , DAIMI PB-357*, Department of Computer Science, Aarhus University, September 1998.
30. Faber NM, Ferré J. On the numerical stability of two widely used PLS algorithms. *J. Chemom.* 2002; **22**: 101–105.
31. Björck Å. Available from: http://www.mai.liu.se/~akbjo/ (accessed Jan 7th, 2014).

 Copyright © 2014 John Wiley & Sons, Ltd.

APPENDIX A. MATLAB CODE FOR ALGORITHMS

A.1. The non-deflating NIPALS PLS1

```
function [beta,W,T,P,D2,R,q] = nondef_pls1(X,Y,A)
% -------------------------------------------------------------------------
% -------------------- Ulf Indahl 12/04-2013 -------------------------------
% -------------------------------------------------------------------------
[n,m] = size(X);
T     = zeros(n,A);              % Orthonormal scores
W     = zeros(m,A); P = W;       % Orthonormal weights, X-loadings
nw    = zeros(1,A); nt = nw;     % Norms of weights and scores before normalization
% ---------------------- Initialization ------------------------------------
w = X'*Y; nw(1) = norm(w); W(:,1) = w/nw(1);
t = X*W(:,1); nt(1) = norm(t); T(:,1)=t/nt(1);
P(:,1) = X'*T(:,1);
if A > 1
% ---------------------- Solution of the PLS1-problem ----------------------
    for a = 2:A
        w = nw(a-1)*(W(:,a-1)-P(:,a-1)/nt(a-1)); nw(a) = norm(w);
        % w = w-W(:,1:a-1)*(W(:,1:a-1)'*w);  % Include for numerical stability.
        W(:,a) = w/nw(a);
        t = X*W(:,a);
        t = t-T(:,1:a-1)*(T(:,1:a-1)'*t); nt(a) = norm(t);
        T(:,a) = t/nt(a);
        P(:,a) = X'*T(:,a);
    end
% -------------------------------------------------------------------------
end
% -- Postprocessing comp. to find regression coeffs & other key matrices --
d2   = -(nw(2:end)./nw(1:end-1)); D2 = diag(nt);
D2(A+1:A+1:end)=d2.*nt(1:end-1); % The upper bidiagonal matrix (D2 = P'*W)
R    = W/D2; % The non-orthogonal "SIMPLS weights".
q    = Y'*T; % Regression coeffs (Y-loadings) for the orthogonal scores
beta = cumsum(bsxfun(@times,R, q),2); % The X-regression coefficients
% -------------------------------------------------------------------------
% Note 1: T*D2 is the QR-factorization of X*W (the "non-orthogonal scores").
% Note 2: W/D2 is the QR-factorization of R (the "non-orthogonal SIMPLS weights").
```


Copyright © 2014 John Wiley & Sons, Ltd.

A.2. The modified Bidiag2

```
function [beta,W,T,P,D2,R,q] = bidiag2m(X,Y,A)
% ------------------------------------------------------------------------------
% ------------------ Ulf Indahl 12/04-2013 --------------------------------------
% ------------------------------------------------------------------------------
[n,m] = size(X);
T  = zeros(n,A); % Orth. scores
W  = zeros(m,A); % Orthonormal weights
D2 = zeros(A); iD2 = D2; % The bidiagonal and its inverse
% -------------------- Initialization ---------------------------------------
W(:,1) = X'*Y;
W(:,1) = W(:,1)/norm(W(:,1));
T(:,1) = X*W(:,1);
D2(1,1) = norm(T(:,1));
T(:,1) = T(:,1)/D2(1,1);
iD2(1,1) = 1/D2(1,1);
if A > 1
% --------------------- Solution of the PLS1-problem -----------------------
    for a = 2:A,
            W(:,a) = X'*T(:,a-1)-D2(a-1,a-1)*W(:,a-1);
            % W(:,a) = W(:,a)-W(:,1:a-1)*(W(:,1:a-1)'*W(:,a)); % Include for ...
                numerical stability.
            D2(a-1,a) = -norm(W(:,a)) ; % Sign switched for consistency with ...
                other algs.
            W(:,a) = W(:,a)/D2(a-1,a);
            T(:,a) = X*W(:,a)-D2(a-1,a)*T(:,a-1);
            % T(:,a) = T(:,a)-T(:,1:a-1)*(T(:,1:a-1)'*T(:,a)); % Include for ...
                numerical stability.
            D2(a,a) = norm(T(:,a));
            T(:,a) = T(:,a)/D2(a,a);
            iD2(a,a)=1/D2(a,a);     % Buliding the inverse of the bidiagonal D2 ...
                (= P'*W)
            iD2(1:a-1,a)= iD2(1:a-1,a-1)*(-D2(a-1,a)/D2(a,a));
    end
% ------------------------------------------------------------------------------
end
P = X'*T; % X-loadings.
R = W*iD2;% The non-orthogonal "SIMPLS weights"
q = Y'*T; % Regression coeffs (Y-loadings) for the orthogonal scores
beta = cumsum(bsxfun(@times,R, q),2); % The X-regression coefficients
% ------------------------------------------------------------------------------
% Note 1: T*D2 is the QR-factorization of X*W (the "non-orthogonal scores").
% Note 2: W/D2 is the QR-factorization of R (the "non-orthogonal SIMPLS weights").
```

 Copyright © 2014 John Wiley & Sons, Ltd.

A.3. Direct computation of orthonormal scores

```
function [beta,W,T,P,D2,R,q] = mod_nondef_pls1(X,Y,A)
% -------------------------------------------------------------------------
% -------------------- Ulf Indahl 12/04-2013 ------------------------------
% -------------------------------------------------------------------------
[n,m] = size(X);
T     = zeros(n,A);                 % Orth. scores
W     = zeros(m,A); P = W; R = W;   % % Orthonormal weights, X-loadings, ... SIMPLS-weights
nw    = zeros(1,A); nt = nw;        % Norms of scores & weights before normalization
% ---------------------- Initialization -----------------------------------
w = X'*Y; nw(1) = norm(w); W(:,1) = w/nw(1);
t = X*W(:,1); nt(1) = norm(t); T(:,1)=t/nt(1);
R(:,1) = W(:,1)/nt(1);
P(:,1) = X'*T(:,1);
if A > 1
% ----------------------- Solution of the PLS1-problem --------------------
    for a = 2:A
        w = nw(a-1)*(W(:,a-1)-P(:,a-1)/nt(a-1));
        % w = w-W(:,1:a-1)*(W(:,1:a-1)'*w);         % Include for numerical ... stability.
        nw(a) = norm(w);
        W(:,a) = w/nw(a);
        R(:,a) = R(:,a-1)*(nt(a-1)*nw(a))/nw(a-1)+W(:,a);
        t = X*R(:,a);
        % t = t-T(:,1:a-1)*(T(:,1:a-1)'*t);         % Include for numerical ... stability.
        nt(a)=norm(t);
        R(:,a)=R(:,a)/nt(a);
        T(:,a) = t/nt(a);
        P(:,a) = X'*T(:,a);
    end
% -------------------------------------------------------------------------
end
% -- Postprocessing comp. to find regression coeffs & other key matrices --
D2 = diag(nt); d2   = -(nw(2:end)./nw(1:end-1));
D2(A+1:A+1:end)=d2.*nt(1:end-1); % The upper bidiagonal matrix (D2 = P'*W)
q    = Y'*T; % Regression coeffs (Y-loadings) for the orthogonal scores
beta = cumsum(bsxfun(@times,R, q),2); % The X-regression coefficients
% -------------------------------------------------------------------------
% Note 1: T*D2 is the QR-factorization of X*W (the "non-orthogonal scores").
% Note 2: W/D2 is the QR-factorization of R (the "non-orthogonal SIMPLS weights").
```

 Copyright © 2014 John Wiley & Sons, Ltd.

A.4. Bidiag2 with deflation

```matlab
function [beta,W,T,P,D2,R,q] = bidiag2defl(X,Y,A)
% ------------------------------------------------------------------------
% ------------------- Ulf Indahl 30/06-2013 --------------------------------
% ------------------------------------------------------------------------
[n,m] = size(X);
T  = zeros(n,A);              % Orthonormal scores
W  = zeros(m,A); P = W;       % Orthonormal weights, X-loadings
D2 = zeros(A);                % The upper bidiagonal matrix (D2 = P'*W)
% -------------------- Initialization ---------------------------------------
W(:,1) = X'*Y; W(:,1) = W(:,1)/norm(W(:,1));
T(:,1) = X*W(:,1); D2(1,1) = norm(T(:,1)); T(:,1) = T(:,1)/D2(1,1);
P(:,1) = X'*T(:,1);
X = X - D2(1,1)*T(:,1)*W(:,1)'; % First deflation step
if A > 1
% -------------------- Solution of the PLS1-problem ------------------------
    for a = 2:A,
        W(:,a) = P(:,a-1)-D2(a-1,a-1)*W(:,a-1);
        % W(:,a) = W(:,a)-W(:,1:a-1)*(W(:,1:a-1)'*W(:,a)); % Include for ...
            numerical stability if X-deflation is skipped.
        D2(a-1,a) = -norm(W(:,a));                          % Adjustment of ...
            sign: D2(a-1,a) = norm(W(:,a)); % (Original Bidiag2)
        W(:,a) = W(:,a)/D2(a-1,a);
        t = X*W(:,a); T(:,a) = t-D2(a-1,a)*T(:,a-1);
        % T(:,a) = T(:,a)-T(:,1:a-1)*(T(:,1:a-1)'*T(:,a)); % Include for ...
            numerical stability if X-deflation is skipped.
        D2(a,a) = norm(T(:,a));
        T(:,a) = T(:,a)/D2(a,a);
        P(:,a) = X'*T(:,a);
        X = X-t*W(:,a)'; % Complete orthogonal X-deflation
        % X = X-T(:,a)*P(:,a)';       % Standard NIPALS X-deflation (is not ...
            correct for this algorithm)
    end
% ------------------------------------------------------------------------
end
R    = W/D2; % the non-orthogonal "SIMPLS weights"
q    = Y'*T; % Regression coeffs (Y-loadings) for the orthogonal scores
beta = cumsum(bsxfun(@times,R, q),2);    % The X-regression coefficients
% -----------------------------------------------------------------------
% Note 1: T*D2 is the QR-factorization of X*W (the "non-orthogonal scores").
% Note 2: W/D2 is the QR-factorization of R (the "non-orthogonal SIMPLS weights").
```


Copyright © 2014 John Wiley & Sons, Ltd.

A.5. Bidiag2 with stabilization of scores and weights

```
function [beta,W,T,P,D2,R,q] = bidiag2stab(X,Y,A)
% ---------------------------------------------------------------------------
% ------------------- Ulf Indahl 30/06-2013 ----------------------------------
% ---------------------------------------------------------------------------
[n,m] = size(X);
T  = zeros(n,A);             % Orthonormal scores
W  = zeros(m,A); P = W;      % Orthonormal weights, X-loadings
D2 = zeros(A);               % The upper bidiagonal matrix (D2 = P'*W)
% -------------------- Initialization ----------------------------------------
W(:,1) = X'*Y; W(:,1) = W(:,1)/norm(W(:,1));
T(:,1) = X*W(:,1); D2(1,1) = norm(T(:,1)); T(:,1) = T(:,1)/D2(1,1);
if A > 1
% --------------------- Solution of the PLS1-problem -------------------------
    for a = 2:A,
        W(:,a) = X'*T(:,a-1)-D2(a-1,a-1)*W(:,a-1);
        W(:,a) = W(:,a) - W(:,1:a-1)*(W(:,1:a-1)'*W(:,a)); % Included for ...
            numerical stability.
        D2(a-1,a) = -norm(W(:,a));                         % Adjustment of ...
            sign: D2(a-1,a) = norm(W(:,a)); % (Original Bidiag2)
        W(:,a) = W(:,a)/D2(a-1,a);
        T(:,a) = X*W(:,a) -D2(a-1,a)*T(:,a-1);
        T(:,a) = T(:,a) - T(:,1:a-1)*(T(:,1:a-1)'*T(:,a)); % Included for ...
            numerical stability.
        D2(a,a) = norm(T(:,a));
        T(:,a) = T(:,a)/D2(a,a);
    end
% ---------------------------------------------------------------------------
end
P    = X'*T; % X-loadings.
R    = W/D2; % the non-orthogonal "SIMPLS weights"
q    = Y'*T; % Regression coeffs (Y-loadings) for the orthogonal scores
beta = cumsum(bsxfun(@times,R, q),2);    % The X-regression coefficients
% --------------------------------------------------------------------------
% Note 1: T*D2 is the QR-factorization of X*W (the "non-orthogonal scores").
% Note 2: W/D2 is the QR-factorization of R (the "non-orthogonal SIMPLS weights").
```

 Copyright © 2014 John Wiley & Sons, Ltd.